The Mini Project entitled

Simulation of Driver Assistance System

Submitted in partial fulfilment of academic requirements for the award of the degree of Bachelor of Engineering (Computer Science and Engineering)

By

PADALA ABHIJITH	**2451-15-733-005**
MEDURI SURYAA PRANAV	**2451-15-733-009**
GANESH THANU	**2451-15-733-010**

Department of Computer Science and Engineering

M.V.S.R. ENGINEERING COLLEGE

(Affiliated to Osmania University & Recognized by AICTE)

Nadergul, Saroor Nagar Mandal, Hyderabad – 501 510

2017-18.

Department of Computer Science and Engineering

M.V.S.R. ENGINEERING COLLEGE

(Sponsored by Matrusri Education Society, Estd. 1980)

(Affiliated to Osmania University & Recognized by AICTE)

Nadergul, Saroor Nagar Mandal, Hyderabad – 501 510

CERTIFICATE

This is to certify that the Mini Project entitled **"Simulation of Driver Assistance System",** is being submitted by Mr. **PADALA ABHIJITH** bearing H.T No **2451-15-733-005,** Mr. **MEDURI SURYAA PRANAV** bearing H.T No **2451-15-733-009,** Mr. **GANESH THANU** bearing H.T No **2451-15-733-010** in partial fulfilment of academic requirements for the award of the degree of BACHELOR OF ENGINEERING in COMPUTER SCIENCE AND ENGINEERING from MVSR Engineering College, affiliated to OSMANIA UNIVERSITY, is a record of bonafide work carried out by him under the guidance and supervision of the faculty (CSED). The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of my knowledge and belief.

Project Coordinator

T.Sujanavan

Assistant Professor

Dept. of CSE.

MVSR Engineering College

ACKNOWLEDGMENT

We take this opportunity to express our profound and sincere gratitude to all those who helped us in carrying out this project successfully.

At the very outset, we are thankful to our Principal **Dr. G.Kanaka Durga** and **Dr. Akhil Khare**, Professor and Head, Department of Computer Science and Engineering, MVSR Engineering College, Hyderabad for their consent to do the project work as a part of our B.E Degree (CSE). We thank them for their valuable suggestions and advice during our project work.

We would like to thank our Internal Project Guide **Mr. T.Sujanavan** , Assistant Professor, Department of Computer Science and Engineering, MVSR Engineering College, for his useful suggestions, guidance and encouragement.

We also thank our mentors **Mr. T.Sujanavan** and **Mrs. D.Sirisha** Assistant Professors, Department of Computer Science and Engineering, MVSR Engineering College, for their constant support and feedback.

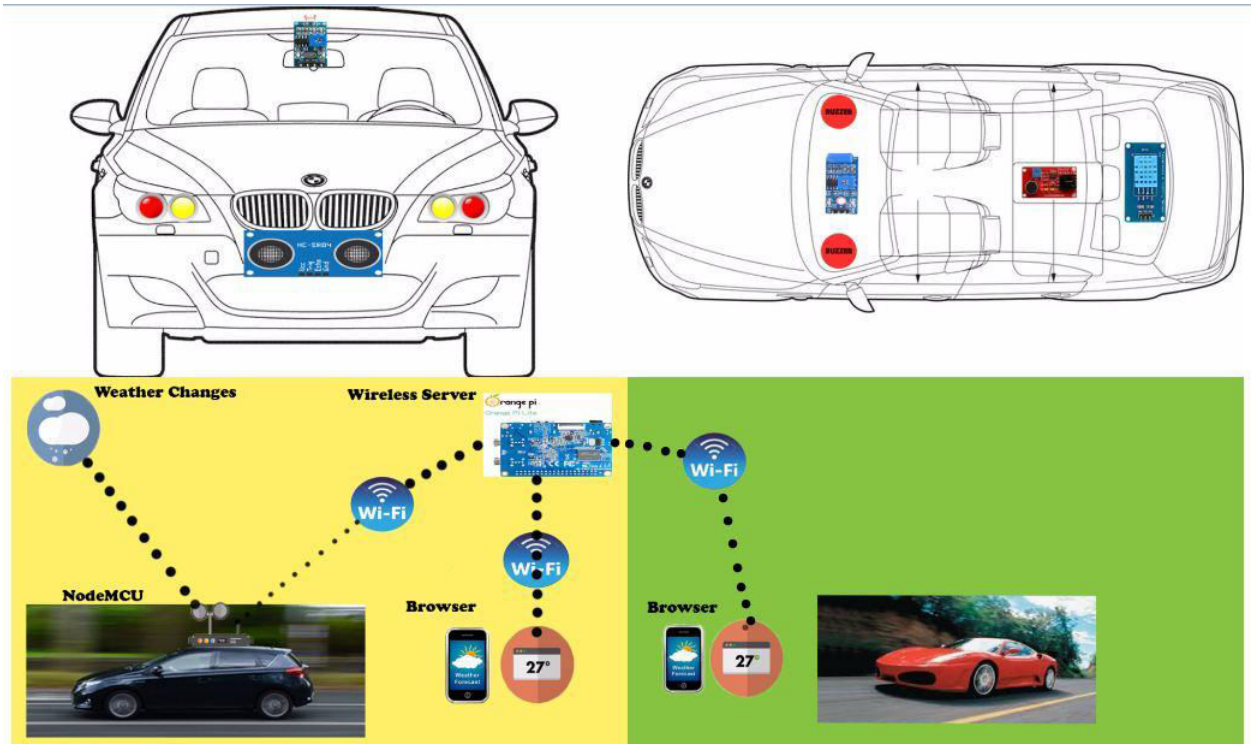
We thank the teaching and non-teaching staff of CSE for extending their support. Finally we are thankful to our parents for their cooperation and support throughout all endeavours in our life.

PADALA ABHIJITH (2451-15-733-005)

MEDURI SURYAA PRANAV (2451-15-733-009)

GANESH THANU (2451-15-733-010)

Simulation of Driver Assistance System



ABSTRACT

Driver assistance system, or **DAS**, is a system to help the driver in the driving process. DAS is developed to automate/adapt/enhance vehicle systems for safety and better driving. Safety features are designed to avoid collisions and accidents by offering technologies that alert the driver to potential problems, or to avoid collisions by implementing safeguards and taking over control of the vehicle. Adaptive features include automated lighting, theft alarm, warning during over speeding, temperature and humidity information, connection to smartphones, alert driver to other cars or dangers and indicate obstacles in blind spots while backing. In addition to the above functionalities, We have also simulated digital weather system which provides us with the information of the weather in our neighbouring environment. For example it can provide us with details about the surrounding temperature, barometric pressure, humidity, etc. By knowing the weather ahead of the journey we know what precautions are to be taken and go well prepared.

LIST OF CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

Vehicles and drivers are gaining a new level of safety from DAS. Safety systems were once only backup cameras and parking assistance. Now, they are being fused with other subsystems and integrated with new technology to provide life-saving features like emergency braking. As these systems are used in more and more safety-critical applications, the testing to ensure they function properly needs to become more rigorous and simultaneously support the rapid innovation that is happening.

1.1. PROBLEM STATEMENT

Demand for Automated Driver Assistance Systems (ADAS) is caused by desire to build safer vehicles and roads in order to reduce the number of road fatalities and by legislation in the leading countries. ADAS is made of the following physical sensors: radar, LIDAR, ultrasonic, photonic mixer device (PMD), cameras, and night vision devices—that allow a vehicle to monitor near and far fields in every direction and of evolving and improving sensor fusion algorithms that ensure vehicle, driver, passenger's, and pedestrian's safety based on factors such as traffic, weather, dangerous conditions etc.

1.2. SCOPE

Our project is Digital Driver Assistance System through which the driver can monitor all the safety measures of the vehicle through his mobile. As an enhancement to the project we have also added a Digital Weather Assistance system which shows the weather conditions of the destination. This is done without the help of the internet. It takes the details of the weather of the destination and stores it in the weather station (Orangepi). These details can be accessed by other users by connecting their wi fi to this station.

1.3. OBJECTIVES

One of the main objectives of our project is to establish a Digital Driver Assistance System which is of mutual use to both the driver and the vehicle and to make the life of the driver safe and easy. It also gives an opportunity to work with IoT devices to simulate the entire environment. Working on this project gives us a better experience that can be used in developing real time projects in coming future.

CHAPTER-2

LITERATURE SURVEY

2.1. MICROCONTROLLER

A microcontroller (or MCU for microcontroller unit) is a small computer on a single integrated circuit. In modern terminology, it is similar to, but less sophisticated than a system on a chip or SoC; an SoC may include a microcontroller as one of its components. A microcontroller contains one or more CPUs along with memory and programmable input/output peripherals. Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, toys and other embedded systems. Here we use ESP8266 NodeMCU. NodeMCU is an eLua based firmware for the ESP8266 WiFi SOC from Espressif. The firmware is based on the Espressif NON-OS SDK and uses a file system based on spiffs. The code repository consists of 98.1% C-code that glues the thin Lua veneer to the SDK. The NodeMCU firmware is a companion project to the popular NodeMCU dev kits, ready-made open source development boards with ESP8266-12E chips.

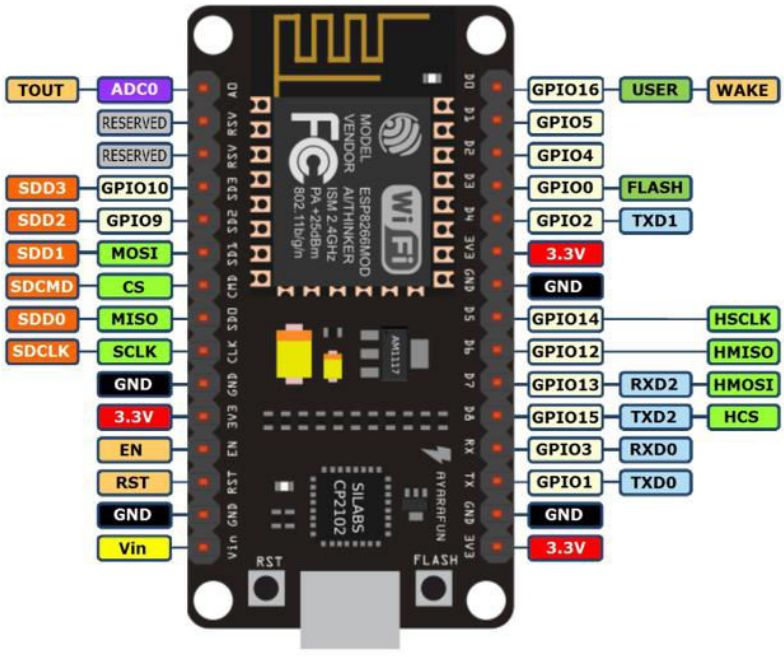


Fig.2.1.Microcontroller

2.2. SENSORS

Sensor is a device, module, or subsystem whose purpose is to detect events or changes in its environment and send the information to other electronics, frequently a computer processor. A sensor is always used with other electronics, whether as simple as a light or as complex as a computer.

Sensors are used in everyday objects such as touch-sensitive elevator buttons (tactile sensor) and lamps which dim or brighten by touching the base, besides innumerable applications of which most people are never aware. With advances in micro machinery and easy-to-use microcontroller platforms, the uses of sensors have expanded beyond the traditional fields of temperature, pressure or flow measurement

2.2.1. LDR Module (Light Dependent Resistor)

LDR sensor module is used to detect the intensity of light. It is associated with both analog output pin and digital output pin labelled as AO and DO respectively on the board. When there is light, the resistance of LDR will become low according to the intensity of light. The greater the intensity of light, the lower the resistance of LDR. The sensor has a potentiometer knob that can be adjusted to change the sensitivity of LDR towards light.

Fig.2.2.1. LDR module

2.2.2. Ultrasonic Sensor (HC-SR04)

The ultrasonic sensor also known as distance measuring sensor provides 2cm to 400cm of non-contact measurement functionality with a ranging accuracy that can reach up to 3mm.Each HC-SR04 module includes an ultrasonic transmitter, a receiver and a control circuit. There are only four pins that you need to worry about on the HC-SR04: VCC (Power), Trig (Trigger), Echo (Receive), and GND (Ground).

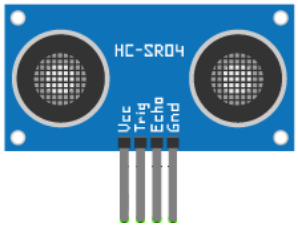


Fig.2.2.2. Ultrasonic Sensor

2.2.3. Humidity and Temperature Sensor (DHT11)

The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). Its fairly simple to use, but requires careful timing to grab data.

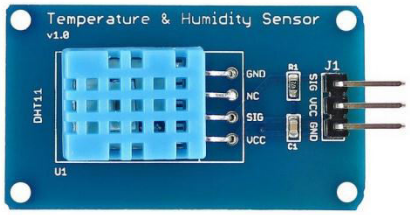


Fig.2.2.3. Humidity and Temperature Sensor

2.2.4. Sound Sensor

The Sound Sensor can detect the sound strength of the environment. The main component of the module is a simple microphone and LM393 level convertor chip. The sensor can provide both digital as well as analog output.

Fig.2.2.4. Sound Sensor

2.2.5. Vibration Sensor (SW-420)

The Vibration module based on the vibration sensor SW-420 and Comparator LM393 to detect if there is any vibration that beyond the threshold. The threshold can be adjusted by the on-board potentiometer. When this no vibration, this module output logic LOW the signal indicate LED light, And vice versa.

Fig.2.2.5. Vibration Sensor

2.2.6. Temperature Sensor (LM35)

LM35 is a precision IC temperature sensor with its output proportional to the temperature (in ^{o}C). The sensor circuitry is sealed and therefore it is not subjected to oxidation and other processes. With LM35, temperature can be measured more accurately than with a thermistor. It also possess low self heating and does not cause more than 0.1 ^{o}C temperature rise in still air.The operating temperature range is from -55°C to 150°C.

Fig.2.2.6. Temperature Sensor

2.3. MICROPROCESSOR

A **microprocessor** is a computer processor which incorporates the functions of a computer's central processing unit (CPU) on a single integrated circuit (IC), or at most a few integrated circuits. The microprocessor is a multipurpose, clock driven, register based, digital-integrated circuit which accepts binary data as input, processes it according to instructions stored in its memory, and provides results as output. The integration of a whole CPU onto a single chip or on a few chips greatly reduced the cost of processing power, increasing efficiency. Integrated circuit processors are produced in large numbers by highly automated processes resulting in a low per unit cost. Here we use Orange Pi Lite as the Microprocessor.

Orange Pi Lite is an open-source single-board computer. It can run Android 4.4, Ubuntu, Debian, Raspbian Image. It uses the AllWinner H3 SoC, and has 512MB DDR3 SDRAM. We can use it to build as a computer, wireless server etc.

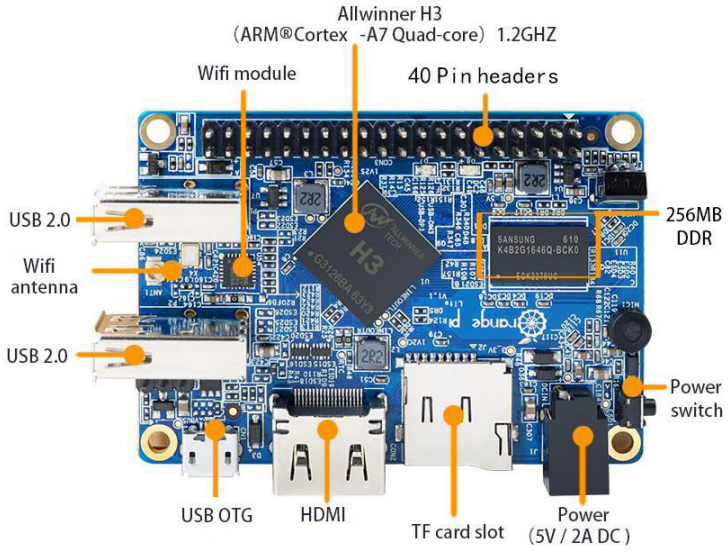


Fig.2.3.Microprocessor

2.4. IDE (Integrated Development Environment)

IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. In this we use Arduino IDE for iot development. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software. The IDE is a text editor-like program that allows you to write Arduino code. The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. 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.

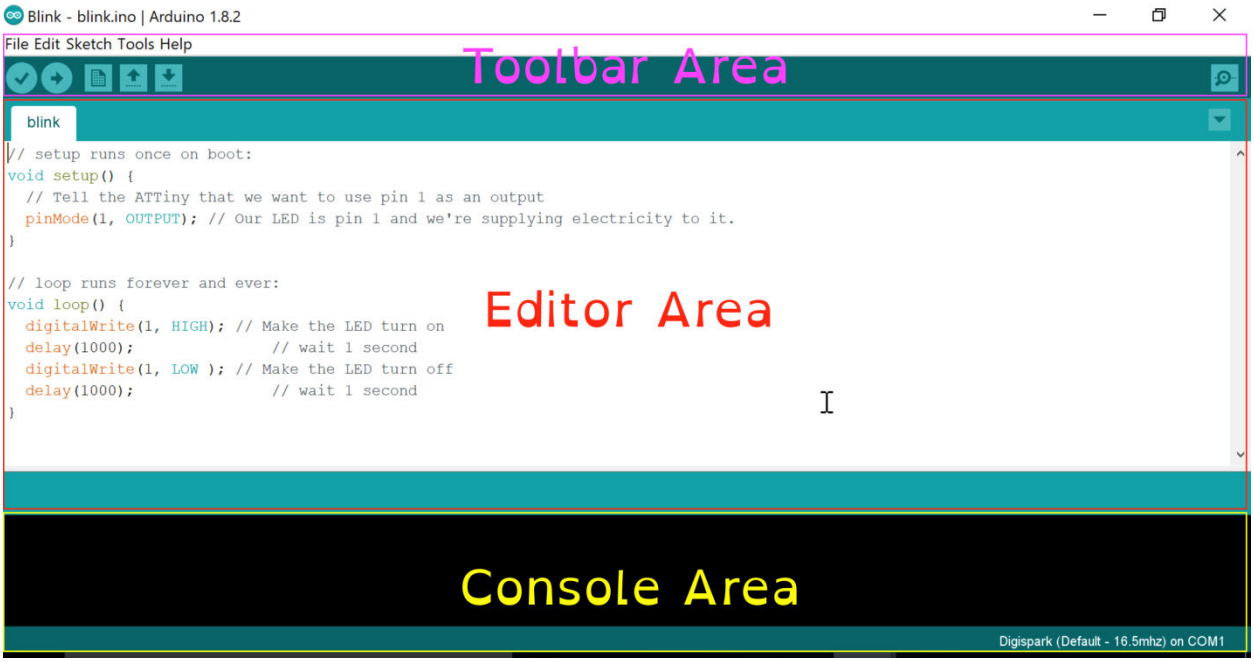


Fig.2.4. Arduino IDE

CHAPTER-3

SYSTEM ARCHITECTURE

3.1. Layout of Driver Assistance System

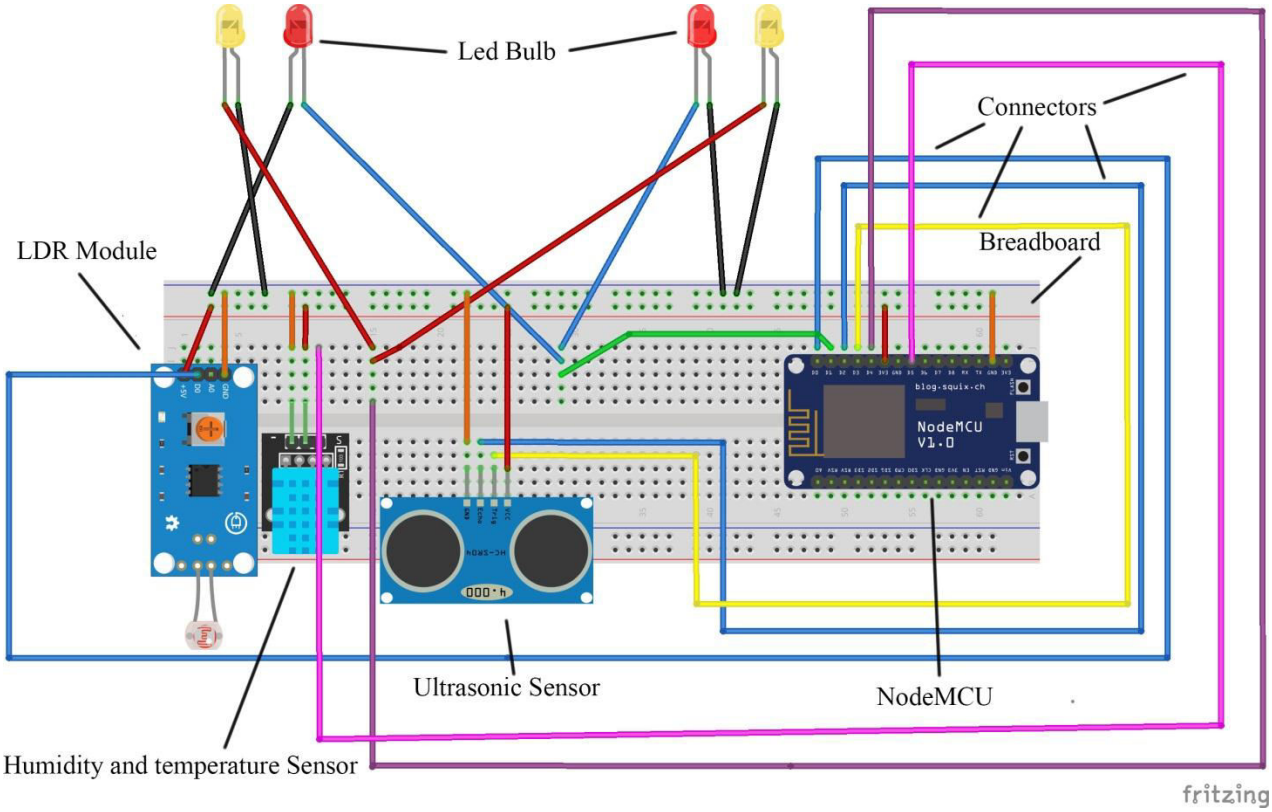


Fig.3.1.1 Layout-1: Driver Assistance System

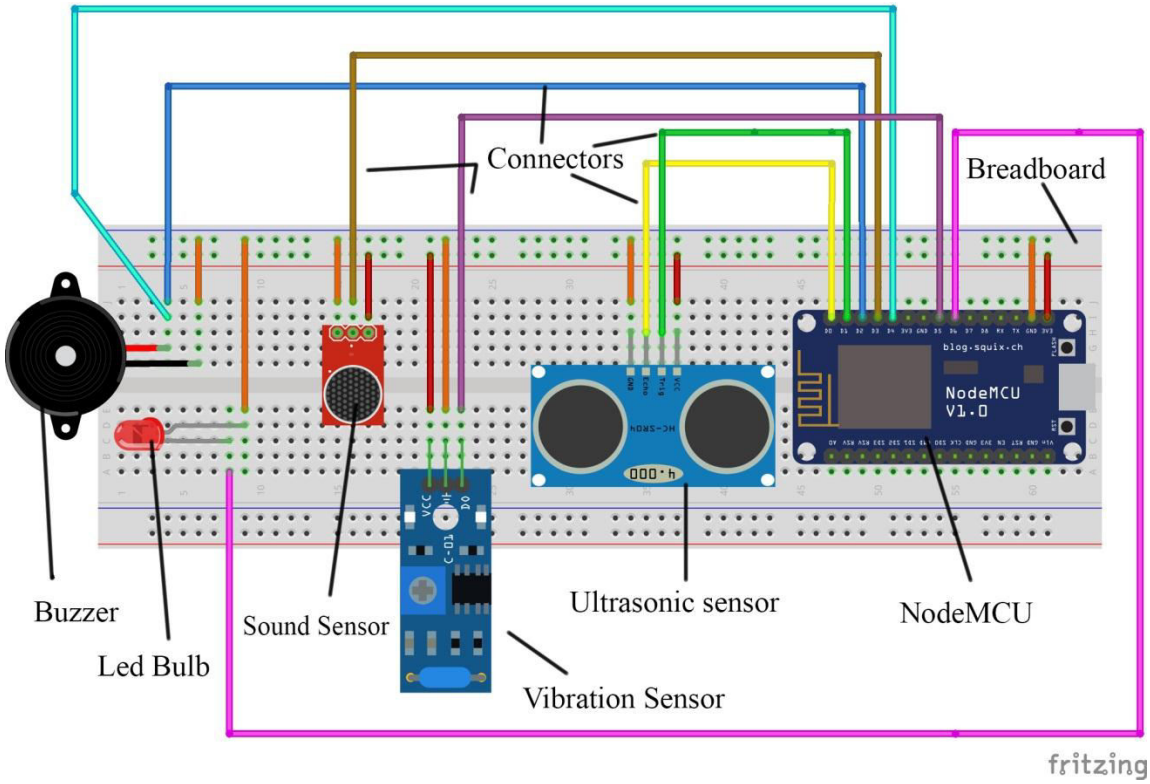


Fig.3.1.2 Layout-2: Driver Assistance System

3.2. Layout of Digital Weather System

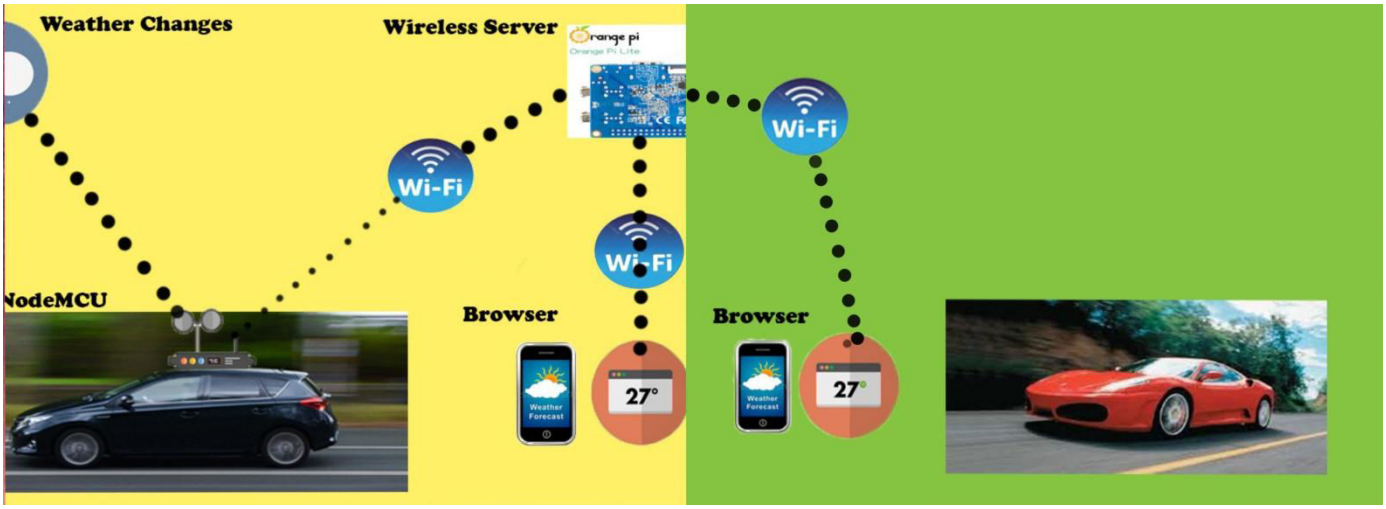


Fig.3.2. Layout of Digital Weather System

CHAPTER-4

IMPLEMENTATION

4.1. Setting up the Environment

We have simulated our driver assistance system by using a small toy car. We have attached two breadboards each of which have three sensors and these sensors collect the data depending upon the car's behaviour when it is put through some tests. We have also simulated the digital weather station which provides us the weather details of the destination place through our mobile. We store these weather details into a server (orange pi) on which data more operations can be done. The sensors that are attached to the breadboard do the following functions:

1. An Ultrasonic sensor gives out an alarm if there is any obstacle while backing the car.

2. A theft alarm goes off as a security measure, incase anyone tries to break the glass of the car.

3. There is a sound alarm to indicate that the car is overspeeding ,by using a vibration cum tilt sensor.

4. A light sensor automatically switches on the head lights of the car when dark.

5. When any vehicle or obstacle is too close to the front of the car additional headlight are automatically switched on.

6. A temperature and humidity sensor tells us the temperature and humidity surrounding the car at particular place, which can be viewed in the digital dashboard.

7. In addition, using a server (Orange Pi), we can know the temperatures at various places in our route. These temperatures can be viewed well ahead through wireless lan in our mobile.

4.2. Source code

```
#include<dht.h>
dht DHT;
#define DHT11_PIN D7
const int trigPin = D2;
const int echoPin = D3;
int buzz= D4;
long duration;
int distance;
void setup() {
  Serial.begin(9600);
  pinMode(D0, INPUT);
  pinMode(D1, OUTPUT);
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
```

```
  pinMode(buzz,OUTPUT);
}
// the loop routine runs over and over again forever:
void loop() {
  // Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2;
 // Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
 if (distance<9)
    {
    digitalWrite(D4, HIGH);
    }
    else
    {
      digitalWrite(D4, LOW);
    }
 delay(1000);
 int a = digitalRead(D0);
  Serial.println(a);
    if (a==1)
    {
    digitalWrite(D1, HIGH);
    }
    else
    {
      digitalWrite(D1, LOW);
    }
    delay(500);
int chk = DHT.read11(DHT11_PIN);
Serial.println(" Humidity " );
Serial.println(DHT.humidity, 1);
Serial.println(" Temparature ");
Serial.println(DHT.temperature, 1);
delay(2000);
 }
```

```
// defines pins numbers
const int trigPin = D0;
const int echoPin = D1;
int soundSensor=A0;
int LED=D4;
boolean LEDStatus=false;
int vibr_pin=D3;
int LED_Pin=D8;
int green=D5;
int yellow=D6;
int buzz= D2;// for buzzer
// defines variables
long duration;
int distance;
void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
pinMode(buzz,OUTPUT);
pinMode(soundSensor,INPUT);
 pinMode(LED,OUTPUT);
 pinMode(vibr_pin,INPUT);
 pinMode(LED_Pin,OUTPUT);
 pinMode(green,OUTPUT);
 pinMode(yellow,OUTPUT);
Serial.begin(9600); // Starts the serial communication
}
void loop() {
   int SensorData=analogRead(soundSensor);
  Serial.println(SensorData);
  if(SensorData>548){
    if(LEDStatus==false){
        LEDStatus=true;
        digitalWrite(LED,HIGH);
    }
    else{
        LEDStatus=false;
        digitalWrite(LED,LOW);
    }
  }
  delay(500);
int val;
 val=digitalRead(vibr_pin);
 Serial.println(val);
 if(val==1)
 { digitalWrite(LED_Pin,HIGH); delay(500);// digitalWrite(LED_Pin,LOW); delay(1000);
 }
```

```
 else
 digitalWrite(LED_Pin,LOW);
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2;
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
 if (distance<=30)
    {
    digitalWrite(D2, HIGH);
    digitalWrite(green,LOW);
    digitalWrite(yellow,LOW);
    }
    else
    {
      digitalWrite(green,HIGH);
      digitalWrite(yellow,LOW);
      digitalWrite(D2, LOW);
    }
 if(distance>30&&distance<80)
 {
digitalWrite(green,LOW);
digitalWrite(D2, LOW);
digitalWrite(yellow,HIGH);
 }
delay(1000);
}


#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
 const char* ssid = "M.S";
const char* password = "msmsmsms";
int n=0;
int outputpin= A0;
void setup () {
 //  pinMode(LED_BUILTIN, OUTPUT);
 // pinMode(0, OUTPUT);
```

```
//int outputpin= A0;
  Serial.begin(9600);
  WiFi.begin(ssid, password);
   while (WiFi.status() != WL_CONNECTED) {
     delay(1000);
    Serial.print("Connecting..");
   }
 }
 void loop() {
  int analogValue = analogRead(outputpin);
float millivolts = (analogValue/1024.0) * 3300; //3300 is the voltage provided by NodeMCU
float celsius = millivolts/10;
Serial.println(celsius);
delay(40000);
String tem="http://192.168.43.243/temp.php?temp=" ;
tem.concat(celsius);
  if (WiFi.status() == WL_CONNECTED) { //Check WiFi connection status
     HTTPClient http;  //Declare an object of class HTTPClient
     http.begin(tem);  //Specify request destination
    int httpCode = http.GET();                                                  //Send the request
     if (httpCode > 0)
    {
      Serial.print("datasent");
    }
        }
}
```

CHAPTER-5

TESTING

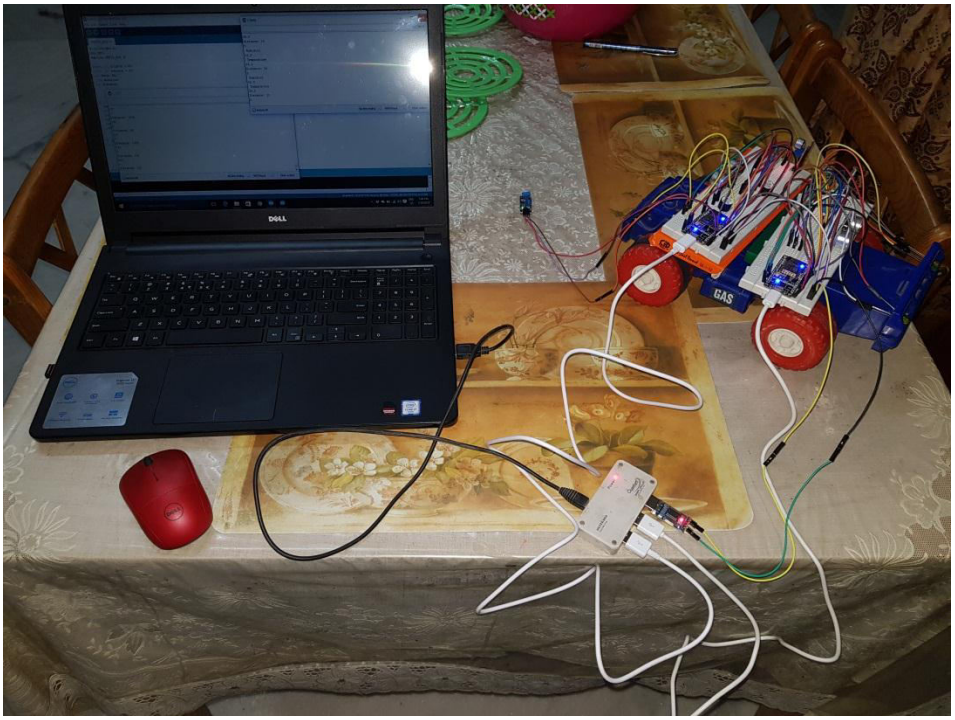


Fig.4.1. Setup of all the equipment

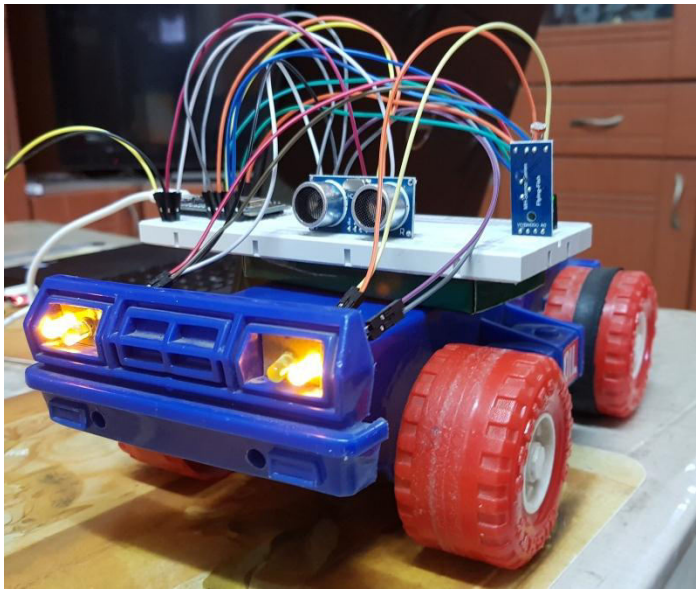


Fig.4.2.Car with headlights switched on

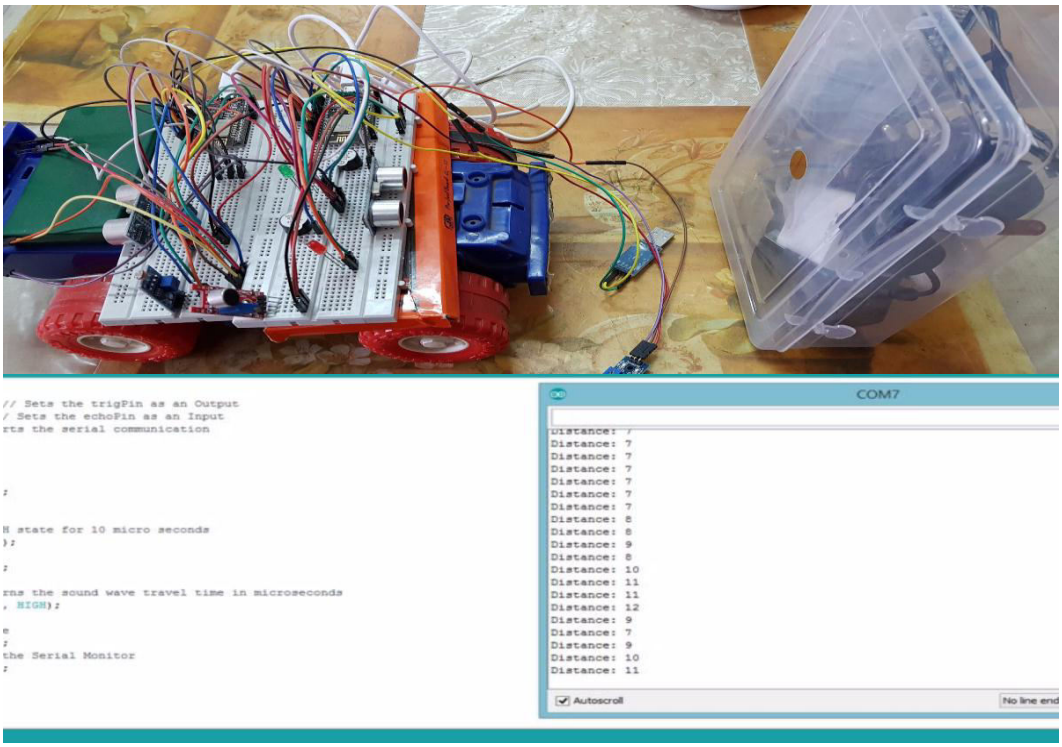


Fig.4.3. Distance between the rear of the vehicle and the obstacle is recorded

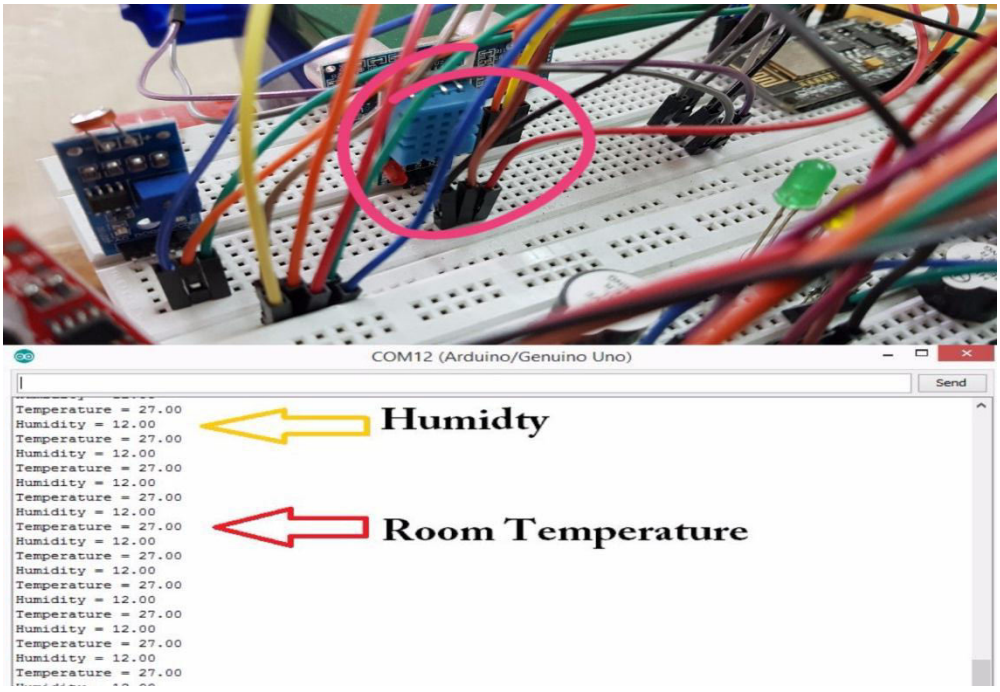


Fig.4.4.Humidity and temperature around the vehicle is shown

Fig.4.5. Mobile View

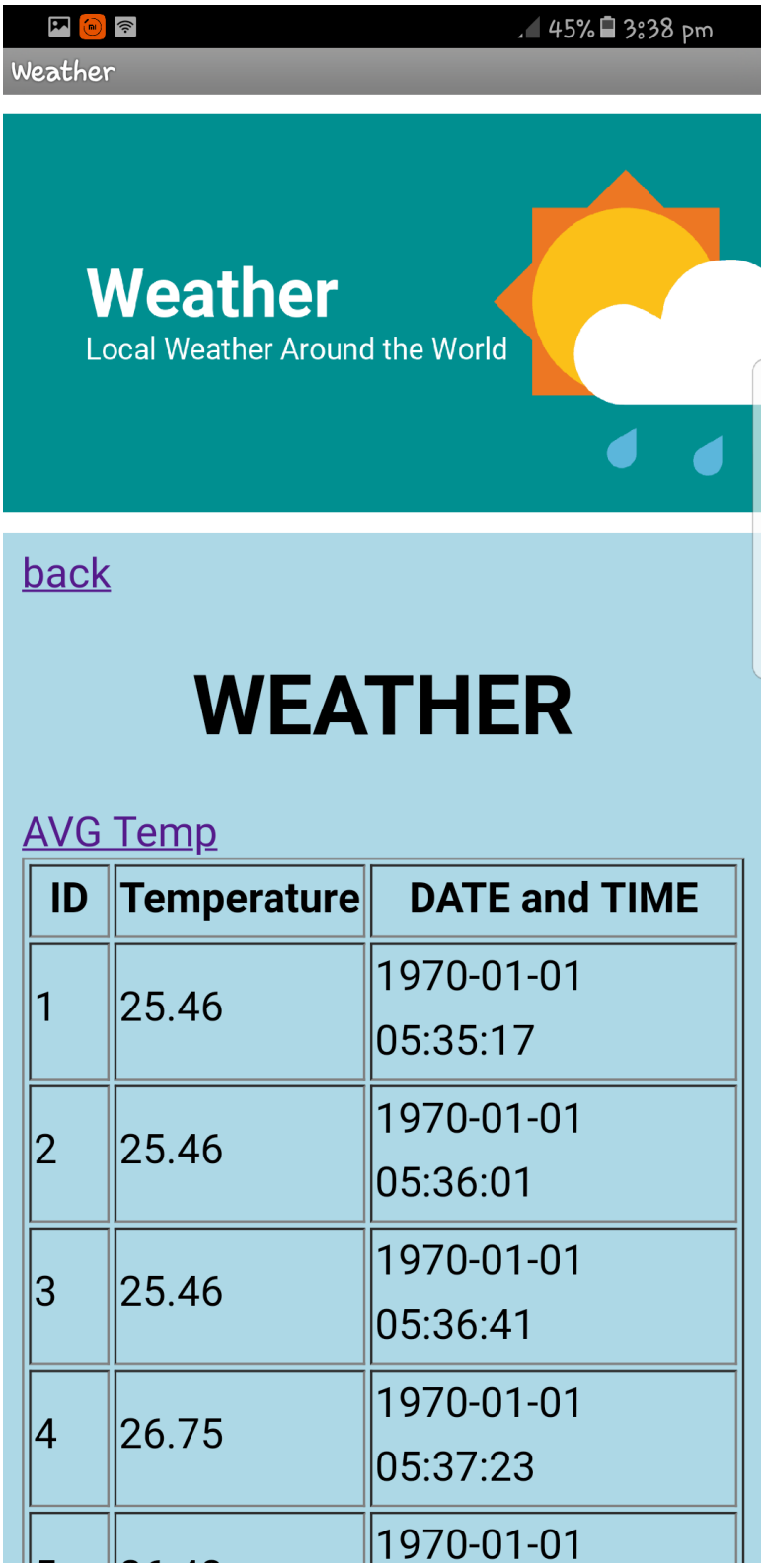


Fig.4.6. Past stored temperatures

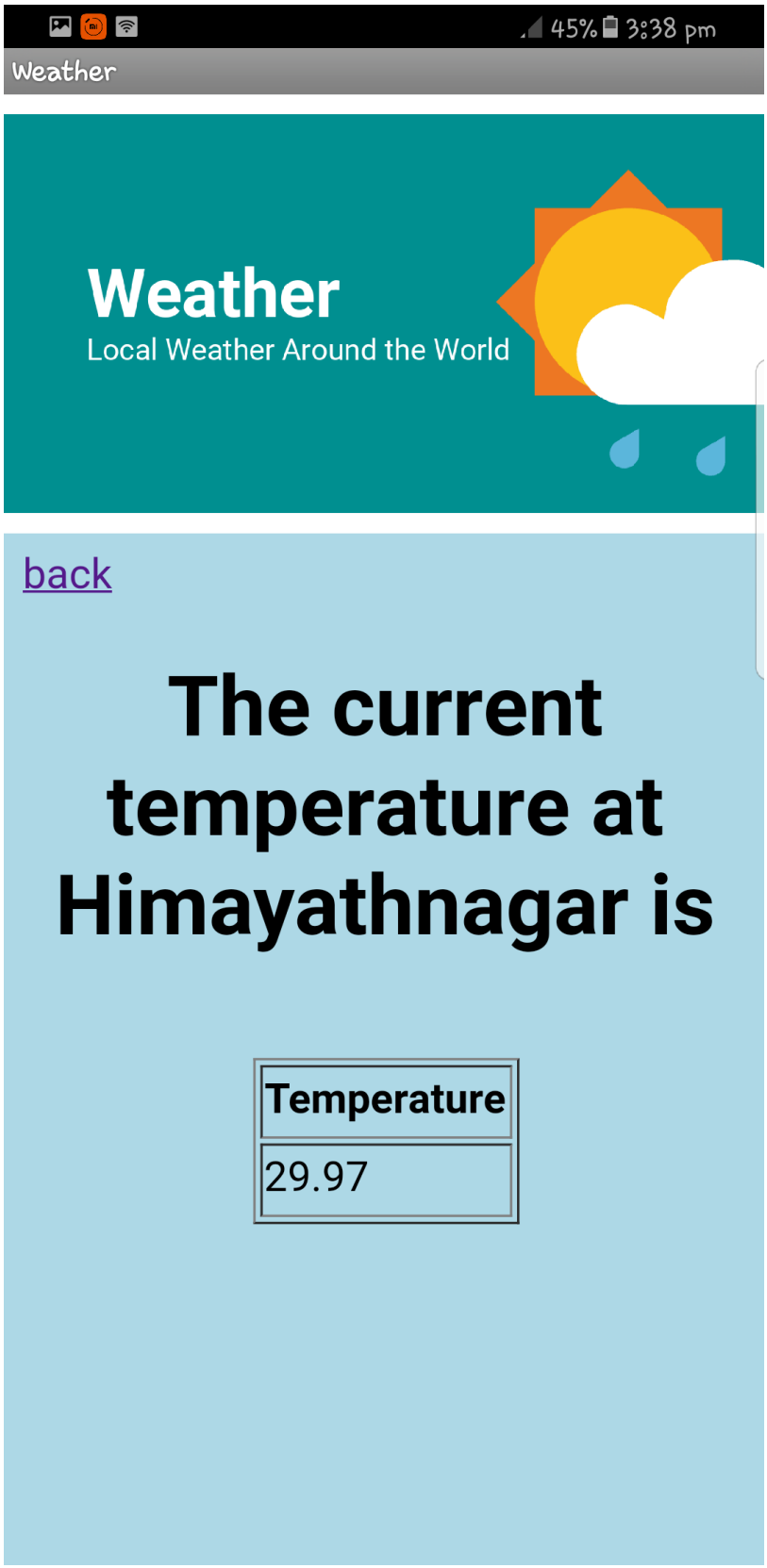


Fig.4.7. Current temperature of that area

CHAPTER-6

CONCLUSION & FUTURE ENHANCEMENTS

6.1. Conclusion

Automatic driver assistance system (ADAS) is one of the fastest growing segments in automobile industry. In ADAS, sensors and algorithms are combined to understand the vehicle environment so that the driver can receive assistance or be warned of potential hazards. (ADAS) will help to improve the driver safety and will making driving safer. It is different than the passive safety systems and other traditional safety features like Anti-lock Braking System (ABS) and Electronic Stability Control. The acceptance of this system will vary from user to user. Further research in this field is led towards autonomous driving vehicles. For this reason comfort enhancing features stand a better chance than safety enhancement properties. Most drivers consider themselves at least better drivers with respect to safe behaviour than average.

6.2. Future Enhancements

The development of various ADAS systems will help to advance piloted driving. The fusion of sensors plays an increasingly important role here. It is, for instance, already possible for ultrasonic, radar and optical sensors to interconnect, so that they can work together to get a better picture of their surroundings. Added value that will come to bear in a wide variety of vehicle features. Features which will shift more and more responsibility away from people and towards technology, with the aim of making driving a safer and more relaxed experience.

The concept of Simulation of DAS can be incorporated not only to automate vehicles but also to meet societal needs. These features can be of great help to the physically impaired to aid the control of motion of a wheelchair and also when embedded in the walking sticks of the blind. Similarly, they can automate a home and can be used for various other applications.

REFERENCES

[1] https://www.arduino.cc/en/Main/Software/download/ARDUINO 1.8.5

[2] http://www.instructables.com/id/NodeMCU-With-LDR/

[3] http://www.instructables.com/id/Connected-Noise-Sensor/

[4] http://www.instructables.com/id/Distance-Measurement-Using-HC-SR04-Via-NodeMCU/

[5] http://www.instructables.com/id/Interface-LM35-With-NodeMCU/

[6] http://www.instructables.com/id/Interface-DHT11-Humidity-Sensor-Using-NodeMCU/

[7] https://www.hackster.io/andriy-baranov/high-sensitive-wifi-fish-bite-alarm-with-nodemcu-esp8266-df161c

[8] https://www.w3schools.com/php/php_mysql_connect.asp

[9] https://github.com/sujanavan/IDIoT/blob/master/device.ino

[10] https://www.phpjabbers.com/free-website-templates.php

[11] https://stackoverflow.com/questions/43433595/nodemcu-sends-data-to-raspberry-pi-server-phpmyadmin

[12] http://www.orangepi.org/downloadresources/orangepiLite/2016-12-12/oragepilite_3177c185ee304f53d881ddb0.html