ECS FINAL REPORT

VIT-AP UNIVERSITY

A Project Report:

ADVANCED CRASH DETECTION

ECS3002

FALL SEM (2024-25)

Submitted by:

- G.Rajasekhar – 22BCE8832
- P.Shiva Prakash – 22BCE9200
- B.D.S.Mallik– 22BCE8169
- P.Pope Nicolas – 22BCE9390
- M.Brahmagna– 22BCE8584

Under the guidance of professor: Prof.Naga Prasad.

ABSTRACT:

Advanced accident detection system is an electronic device installed in a helmet to ensure the safety of the owner or the driver. This paper proposed to design a Advanced accident detection system that works using GPS and GSM technology, which would be the cheapest source of vehicle tracking and it would work as the saftey system. It is an embedded system which is used for tracking and positioning of any vehicle by using Global Positioning System (GPS) and Global system for mobile communication (GSM). This design will continuously monitor a moving Vehicle and report the status of the Vehicle on demand. For doing so an Arduino board is interfaced serially to a GSM Modem and GPS Receiver. A GSM modem is used to send the position (Latitude and Longitude) of the vehicle from a remote place. The GPS modem will continuously give the data i.e. the latitude and longitude indicating the position of the vehicle. The same data is sent to the mobile at the other end from where the position of the vehicle is demanded. When the request by user is sent to the number at the GSM modem, the system automatically sends a return reply to that mobile indicating the position of the vehicle in terms of latitude and longitude in real time.

INTRODUCTION:

Most of the accidents happened with the motorcycle. Nowadays this problem is still increasing due to poor rider' s like speed driving, drunk driving, riding with no helmet protection, riding without sufficient sleep, etc. The numbers of death because of late assistance to people who got the accident. Therefore, the research group and major motorcycle manufacturers including Honda have developed safety devices to protect riders from accidental injuries. The good safety device for a motorcycle is difficult to implement and very expensive. Accident detection with a tracking system only. In this project black box using a MEMS accelerometer sensor and GPS location tracking system is developed for accidental monitoring. When the accident will happen at the same time GSM will send the authorized mobile phone. The Location of the vehicle sends a short message to 3 members who are well-wishers of the rider, using a GPS device to a family member. The system consists of an accelerometer sensor, Arduino Uno microcontroller, GPS device and GSM module for sending a short message. An accelerometer sensor is applied X, Y, Z direction fall detection of an accident. The speed of the motorcycle and threshold algorithm is used to decide a fall or accident in real-time. A mobile short message containing position from GPS (latitude, longitude) will be sent when a motorcycle accident is detected. The robust package design is implemented so that it is safe from water's spray and dust in the environment. This system is installed under the motorcycle seat. A high-performance microcontroller is used to process and store real-time signals from an accelerometer sensor. Thus, this device is analogous to a black box in an airplane.The device keeps a data log of track and acceleration data for 1 minute before and after an accident. Moreover, this device can be used to track a motorcycle after it was stolen but it can't operate in real-time in this case. In this case, the user can send request command with the alphabet to the device and the device will return the position with some basic information.

BACKGROUND:

Advanced accident detection systems using helmets are innovative technological solutions aimed at enhancing the safety of individuals, particularly those involved in activities like cycling, motorcycling, or any other vehicle-related pursuits. These systems utilize a combination of sensors, communication technologies, and smart algorithms to detect potential accidents or collisions and then send out alerts to appropriate parties, such as emergency responders or designated contacts. The primary focus of such systems is to minimize response time in case of accidents, thereby increasing the chances of providing timely medical assistance and reducing the severity of injuries.

PROBLEM DEFINITION:

The problem of Advanced accident detection using helmets addresses the need for enhancing the safety of individuals engaged in activities like cycling, motorcycling, and similar pursuits by developing a technological solution that can quickly detect potential accidents and notify appropriate parties. The primary challenge lies in creating a reliable system that can accurately differentiate between normal movements and actual accidents, while also ensuring timely and effective communication of alerts to emergency responders or designated contacts.

OBJECTIVES OF THE PROPOSED WORK:

- Accurate Accident Detection: Develop algorithms and sensor configurations that can accurately distinguish between normal movements and potential accidents. The system should be capable of recognizing sudden impacts, decelerations, or falls that indicate a real accident.

- Rapid Alert Generation: Design the system to promptly generate alerts when an accident is detected. Alerts should be triggered within milliseconds of an accident occurrence to ensure timely assistance.

- Quick Communication: Establish a reliable communication mechanism to transmit alerts to three of the emergency contacts, first responders, or designated parties. Ensure that alerts are delivered swiftly, allowing for immediate action to be taken.

- Real-time Monitoring: Enable continuous real-time monitoring of sensor data to provide instant updates on the user's condition. This includes monitoring vital signs

and providing data that can aid medical responders in assessing the severity of the situation.

- User-friendly Design: Design the helmet and its integrated technology to be comfortable, ergonomic, and aesthetically pleasing. The system should not impede the user's experience or hinder their normal activities.

- Privacy and Data Security: Implement robust data privacy measures to ensure that personal information and location data are securely managed. User consent should be obtained for data collection and sharing.

- Emergency Contact Management: Develop a user-friendly mechanism for users to manage their emergency contacts. Allow them to easily update, add, or remove contacts who will receive alerts in case of an accident.

- Compatibility and Integration: Ensure compatibility with smartphones or other devices for receiving alerts. Provide options for integration with existing emergency response systems or services.

- Battery Efficiency: Optimize the power consumption of the helmet's components to extend battery life. Users should be able to rely on the system during their outdoor activities without frequent recharging.

- User Awareness and Training: Educate users about the system's capabilities, its limitations, and proper usage. This can help users make the most of the technology and reduce false alarms caused by misunderstanding.

- Feedback Mechanism: Incorporate a feedback mechanism for users to report false positives, system glitches, or improvements. This feedback loop can contribute to the ongoing refinement of the system.

- Regulatory Compliance: Ensure that the system adheres to relevant safety standards and regulatory requirements in the regions where it is deployed.

- Scalability: Design the system architecture in a way that allows for easy scaling to accommodate a larger user base. This is especially important if the system gains popularity and widespread adoption.

Components and supplies:

- Arduino
- Accelerometer MPU6050
- Neo 6M GPS Module
- LCD 16x2
- GSM Modem SIM 800A
- Jumper wires
- Helmet

METHODOLOGY/PROCEDURE:

- **Sensor Data Collection:**

Integrated sensors like accelerometers, gyroscopes, and GPS within the helmet continuously gather data about the wearer's movements, speed, acceleration, and orientation.

- **Real-time Data Processing:**

Algorithms process the sensor data in real-time, analysing it for patterns that could indicate an accident, such as sudden deceleration, sharp changes in orientation, or significant impacts.

- **Accident Identification:**

The algorithms determine if the collected data matches the criteria for an accident, considering predefined thresholds and patterns.

- **Alert Generation within Helmet:**

If the system identifies a potential accident, it activates an alert within the helmet itself. This alert could be a sound, vibration, or visual indicator.

- **Alert Communication Activation:**

After the confirmation window (if applicable), the system activates its communication modules, such as Bluetooth, Wi-Fi, or cellular connectivity.

- **Emergency Contact Notification:**

The system sends an alert notification to three designated emergency contacts or a pre-defined list of contacts. This notification includes critical information such as the wearer's GPS coordinates, detected impact, and other relevant data.

- **Recipient Response:**

The designated emergency contact or response centre receives the alert notification and assesses the situation.
Depending on the severity of the situation, the recipient can take appropriate action, such as contacting emergency services, dispatching help, or reaching out to the wearer to gather more information.

- **Assistance Dispatch:**

If necessary, emergency services are dispatched to the location using the GPS coordinates provided by the system.

RESULTS AND DISSCUSSION:

Results: Auto alert accident detection using helmets streamlines response times by swiftly identifying potential accidents, activating alerts, and notifying relevant parties, ultimately enhancing safety during outdoor activities.

Discussion: The implementation of auto alert accident detection systems using helmets signifies a remarkable stride towards safeguarding individuals engaged in outdoor activities. By seamlessly integrating advanced sensor technology, such systems enable rapid detection of accidents through real-time data analysis. The immediate activation of alerts within the helmet, followed by swift communication to emergency contacts or responders, underscores the potential to mitigate injury severity and save lives. However, this technology's effectiveness hinges on factors like algorithm accuracy, user comfort, and efficient communication channels. As these systems evolve, they hold the promise of revolutionizing safety norms, fostering a proactive culture of accident prevention, and solidifying helmets as a crucial tool not just for physical protection, but for active response during critical moments.

CONCLUSION AND FUTURE SCOPE:

Conclusion:

In conclusion, the integration of auto alert accident detection systems into helmets marks a significant leap forward in personal safety during outdoor pursuits. By capitalizing on cutting-edge sensor technology and real-time data analysis, these systems have the potential to substantially reduce response times in the event of accidents. The immediate activation of alerts within the helmet, followed by prompt communication to designated contacts or responders, offers a tangible means to minimize the impact of accidents and ensure timely assistance. As this technology continues to evolve, it holds the promise of not only enhancing individual safety but also reshaping the way we approach outdoor activities, setting a new standard for proactive accident prevention and emergency response.

Future Scope:

1. Enhanced Sensor Integration: As sensor technology continues to evolve, future systems could incorporate more advanced and precise sensors, allowing for even more accurate accident detection. This could include additional sensors that measure vital signs, such as heart rate and oxygen levels, providing responders with a more comprehensive picture of the wearer's condition.

2. Artificial Intelligence (AI) and Machine Learning: Implementing AI and machine learning algorithms can enhance the system's ability to differentiate between actual accidents and false positives. Machine learning models could be trained on a wide range of scenarios to continuously improve accuracy.

3. Predictive Analytics: By analysing historical data and patterns, these systems could predict high-risk areas or potential accident scenarios, enabling users to be more cautious in those situations and contributing to overall road safety.

4. Multi-modal Communication: Future systems might incorporate multiple communication methods, such as direct communication with emergency services, automated notifications to nearby vehicles, or integration with smart city infrastructure for a more coordinated response.

5. Wearable Integration: Auto alert systems could extend beyond helmets, integrating with other wearable devices like smartwatches or fitness trackers. This would offer a more comprehensive safety net, especially for individuals who engage in various activities.

6. Cloud Connectivity: Cloud-based solutions could enhance data storage and analysis capabilities, allowing for real-time monitoring by loved ones and offering insights for traffic management and urban planning.

7. Vehicle Integration: In the case of accidents involving vehicles, these systems could integrate with vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication systems, alerting drivers and nearby vehicles to potential accidents.

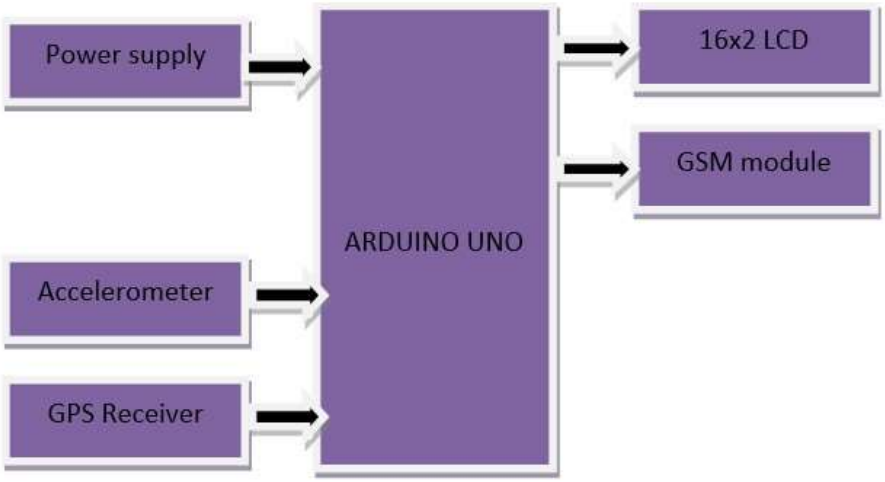
8. Personalized Alerts and Feedback: Future systems could provide personalized feedback to users, offering suggestions to improve their riding behaviour based on data analysis and accident patterns.

9. Global Adoption and Standards: Widespread adoption could lead to the establishment of global standards for these systems, ensuring compatibility, effectiveness, and regulatory compliance across different regions.

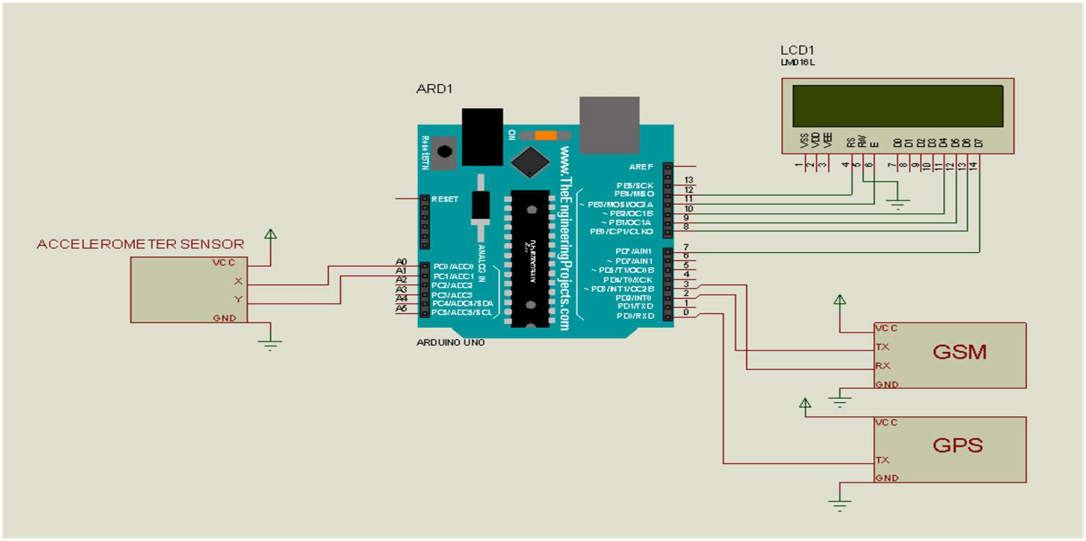
10. Partnerships with Insurance Companies: Auto alert systems could partner with insurance companies to offer incentives and discounts for users who adopt the technology, leading to increased adoption rates.
11. Integration with Emergency Medical Services (EMS): Seamless integration with EMS systems could provide paramedics with critical information about the user's condition before they arrive on the scene, enabling more targeted and efficient treatment.

How Our Project Works:

The helmet is equipped with various sensors, such as accelerometers, gyroscopes, and GPS. These sensors continuously gather data about the wearer's movements, speed, orientation, and acceleration. The collected sensor data is processed in real-time by sophisticated algorithms within the helmet's integrated system. These algorithms analyse the data patterns to identify sudden and unusual changes in movement or acceleration. The algorithms are designed to recognize patterns associated with potential accidents, such as sudden deceleration or abrupt changes in orientation. These patterns indicate situations like collisions, falls, or impacts that might lead to accidents. When the algorithms detect a pattern consistent with a potential accident, they trigger an alert within the helmet. This alert can take the form of a sound, vibration, or visual indication to alert the wearer. To prevent false alarms, the system might provide a brief window for the user to confirm whether an accident has occurred. If the user confirms, the system proceeds to the next step. After the user confirmation period (if applicable), the system activates its communication modules. These modules can include Bluetooth, Wi-Fi, or cellular connectivity. The system sends an alert notification to three designated emergency contacts, a predefined list of contacts, or an emergency response center. This notification includes crucial information such as the user's GPS coordinates, detected impact, and other relevant data. The designated emergency contact or response center receives the alert notification. They can assess the situation based on the provided information and initiate appropriate actions, such as contacting emergency services or reaching out to the user. If necessary, emergency services are dispatched to the location using the GPS coordinates provided by the system. Some systems might continue to monitor the user's condition and provide updates to responders as they approach the scene. Users might have the option to provide feedback on the accuracy and effectiveness of the system. This feedback can contribute to ongoing improvements and refinements.



CIRCUIT DIAGRAM:



REFRENCES:

https://justdoelectronics.com/smart-bike-helmet-safety-alert-system-usingarduino/

http://www.ijareeie.com/upload/2020/july/29_Smart_NC.PDF

https://www.ripublication.com/ijaerspl2019/ijaerv14n6spl_64.pdf

CODE IN MPU6050:

```
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
Adafruit_MPU6050 mpu;
void SensorSetup()
   {
     mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
  Serial.print("Accelerometer range set to: ");
  switch (mpu.getAccelerometerRange()) {
  case MPU6050_RANGE_2_G:
    Serial.println("+-2G");
    break;
  case MPU6050_RANGE_4_G:
    Serial.println("+-4G");
    break;
  case MPU6050_RANGE_8_G:
    Serial.println("+-8G");
    break;
  case MPU6050_RANGE_16_G:
    Serial.println("+-16G");
    break;
  }
  mpu.setGyroRange(MPU6050_RANGE_500_DEG);
  Serial.print("Gyro range set to: ");
  switch (mpu.getGyroRange()) {
  case MPU6050_RANGE_250_DEG:
    Serial.println("+- 250 deg/s");
    break;
  case MPU6050_RANGE_500_DEG:
    Serial.println("+- 500 deg/s");
    break;
  case MPU6050_RANGE_1000_DEG:
    Serial.println("+- 1000 deg/s");
    break;
  case MPU6050_RANGE_2000_DEG:
    Serial.println("+- 2000 deg/s");
    break;
  }

  mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);
  Serial.print("Filter bandwidth set to: ");
  switch (mpu.getFilterBandwidth()) {
  case MPU6050_BAND_260_HZ:
    Serial.println("260 Hz");
    break;
  case MPU6050_BAND_184_HZ:
    Serial.println("184 Hz");
    break;
  case MPU6050_BAND_94_HZ:
    Serial.println("94 Hz");
    break;
  case MPU6050_BAND_44_HZ:
    Serial.println("44 Hz");
    break;
```

```
  case MPU6050_BAND_21_HZ:
    Serial.println("21 Hz");
    break;
  case MPU6050_BAND_10_HZ:
    Serial.println("10 Hz");
    break;
  case MPU6050_BAND_5_HZ:
    Serial.println("5 Hz");
    break;
  }
   }
   void MPU6050_Init()
   {
     if (!mpu.begin()) {
    lcd.clear();
    lcd.print("MPU Not Found");
    while (1) {
      delay(10);
    }
  }
  lcd.clear();
  lcd.print("MPU6050 Found!");
  delay(1000);
  SensorSetup();
   }
```

CODE IN GSMGPS:

```
#include <Wire.h>
#include<LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>
#include <TinyGPS.h>
LiquidCrystal_I2C lcd(0x27,16,2);
SoftwareSerial gsm(3, 2);
TinyGPS gps;
char number1[13];
char number2[13];
char number3[13];
char rcv,rcv1,rcv2,rcv3;
int ii,i;
int NumberCount=0;
float lat, lon;
 void gpsread()
{
  for (unsigned long start = millis(); millis() - start < 1000;)
  {
 while(Serial.available())
 {
    if(gps.encode(Serial.read()))// encode gps data
  {
  gps.f_get_position(&lat,&lon); // get latitude and longitude
 lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("LAT:");
```

```
  lcd.setCursor(4,0);
 lcd.print(lat,6);
 lcd.setCursor(0,1);
 lcd.print("LON:");
  lcd.setCursor(4,1);
  lcd.print(lon,6);
  delay(1000);
  }
 }
  }
  }

void SerialInit()
{
  Serial.begin(9600);
  gsm.begin(9600);
}
void LCDInit()
{
  lcd.begin();
  lcd.backlight();
}
void ok()
{
  do{rcv = gsm.read();}while(rcv != 'K');
}
void GSMInit()
{
  gsm.println("AT");ok();
  gsm.println("AT+CLIP=1");ok();
  gsm.println("AT+CMGF=1");ok();
  gsm.println("AT+CNMI=1,2,0,0");ok();
  //gsm.println("AT+CIPSHUT");ok();
  //gsm.println("AT+CIPMUX=0");ok();
  //gsm.println("AT+CGATT=1");ok();
  //gsm.println("AT+CSTT=\"APN\",\"USER NAME\",\"PASSWORD\"");ok();
  //gsm.println("AT+CIICR");ok();
  //gsm.println("AT+CIFSR");
  //delay(3000);
}

void storenumber1()
{
  do
    {
        do
        {
            while ( !gsm.available() );
        } while ( '+' != gsm.read() );

        while(!gsm.available());
      } while ( 'C' != gsm.read() );
    do
    {
        while ( !gsm.available() );
```

```
    } while ( ':' != gsm.read() );

    do
    {
        while ( !gsm.available() );
    } while ( '"' != gsm.read() );

    ii = 0;
    do
    {
        while ( !gsm.available() );
        while ( !gsm.available() );
        rcv1 = gsm.read();
       number1 [ ii ] = rcv1;
       ii ++;
       } while ( ii!= 13 );
lcd.clear();
i=0;
do{
      lcd.write(number1[i]);
      i++;
    }while(i!=13);
    delay(1000);
gsm.println("ATH");
delay(1000);

}
void storenumber2()
{
  do
    {
        do
        {
            while ( !gsm.available() );
        } while ( '+' != gsm.read() );

        while(!gsm.available());
      } while ( 'C' != gsm.read() );
    do
    {
        while ( !gsm.available() );
    } while ( ':' != gsm.read() );

    do
    {
        while ( !gsm.available() );
    } while ( '"' != gsm.read() );

    ii = 0;
    do
    {
        while ( !gsm.available() );
        while ( !gsm.available() );
        rcv2 = gsm.read();
       number2 [ ii ] = rcv2;
```

```
       ii ++;
       } while ( ii!= 13 );
lcd.clear();
i=0;
do{
      lcd.write(number2[i]);
      i++;
    }while(i!=13);
    delay(1000);
gsm.println("ATH");
delay(1000);

}
void storenumber3()
{
  do
    {
        do
        {
            while ( !gsm.available() );
        } while ( '+' != gsm.read() );

        while(!gsm.available());
      } while ( 'C' != gsm.read() );
    do
    {
        while ( !gsm.available() );
    } while ( ':' != gsm.read() );

    do
    {
        while ( !gsm.available() );
    } while ( '"' != gsm.read() );

    ii = 0;
    do
    {
        while ( !gsm.available() );
        while ( !gsm.available() );
        rcv3 = gsm.read();
       number3 [ ii ] = rcv3;
       ii ++;
       } while ( ii!= 13 );
lcd.clear();
i=0;
do{
      lcd.write(number3[i]);
      i++;
    }while(i!=13);
    delay(1000);
gsm.println("ATH");
delay(1000);

}
void sendregsms1()
```

```
{
/*gsm.print("AT+CMGS=");
gsm.write('"');
i=0;
do{
     gsm.write(number1[i]);
     i++;
    }while(i!=13);
    gsm.write('"');
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.print("YOUR MOBILE NUMBER REGISTERED");
    gsm.write(26);
    ok();*/
    lcd.clear();
    lcd.print("Your Mobile NUM");
    lcd.setCursor(0,1);
    lcd.print("Registered");
    delay(1000);
}
void sendregsms2()
{
/*gsm.print("AT+CMGS=");
gsm.write('"');
i=0;
do{
     gsm.write(number2[i]);
     i++;
    }while(i!=13);
    gsm.write('"');
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.print("YOUR MOBILE NUMBER REGISTERED");
    gsm.write(26);
    ok();*/
    lcd.clear();
    lcd.print("Your Mobile NUM");
    lcd.setCursor(0,1);
    lcd.print("Registered");
    delay(1000);
}
void sendregsms3()
{
/*gsm.print("AT+CMGS=");
gsm.write('"');
i=0;
do{
     gsm.write(number3[i]);
     i++;
    }while(i!=13);
    gsm.write('"');
```

```
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.print("YOUR MOBILE NUMBER REGISTERED");
    gsm.write(26);
    ok();*/
    lcd.clear();
    lcd.print("Your Mobile NUM");
    lcd.setCursor(0,1);
    lcd.print("Registered");
    delay(1000);
}

void Number_1()
{
  lcd.clear();
  lcd.print("GIVE MIS CALL TO");
  lcd.setCursor(0,1);
  lcd.print("STORE NUMBER1");
  storenumber1();
  delay(1000);
  lcd.clear();
  lcd.print("SMS SENDING");
  sendregsms1();
  delay(100);
  lcd.clear();
  lcd.print("SMS SENT");
}
void Number_2()
{
  lcd.clear();
  lcd.print("GIVE MIS CALL TO");
  lcd.setCursor(0,1);
  lcd.print("STORE NUMBER2");
  storenumber2();
  delay(1000);
  lcd.clear();
  lcd.print("SMS SENDING");
  sendregsms2();
  delay(100);
  lcd.clear();
  lcd.print("SMS SENT");

}
void Number_3()
{
  lcd.clear();
  lcd.print("GIVE MIS CALL TO");
  lcd.setCursor(0,1);
  lcd.print("STORE NUMBER3");
  storenumber3();
  delay(1000);
  lcd.clear();
  lcd.print("SMS SENDING");
```

```
  sendregsms3();
  delay(100);
  lcd.clear();
  lcd.print("SMS SENT");
}
void StoreMobileNumbers()
{
  lcd.clear();
  lcd.print("GSM INITILIZING");
  delay(1000);
  GSMInit();
  lcd.clear();
  lcd.print("GSM INIT COMPLE");
  delay(1000);
  Number_1();
  Number_2();
  Number_3();
}
void GPSInit()
{
  lcd.clear();
  lcd.print("GPS Initilizing");
  delay(1000);
  gpsread();
  lcd.clear();
  lcd.print("GPS Init Complete");
  delay(1000);
}
void ShowSerialData()
{
  while(gsm.available()!=0)
  Serial.write(gsm.read());
  delay(5000);
}
void Accident1()
{
gsm.print("AT+CMGS=");
gsm.write('"');
i=0;
do{
      gsm.write(number1[i]);
      i++;
    }while(i!=13);
    gsm.write('"');
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.println("ACCIDENT DETECTED");
    gsm.print("VEHICLE LOCATION");
    gsm.println();
    gsm.print("https://www.google.co.in/maps/search/");
    gsm.print(lat,6);
    gsm.write(',');
    gsm.print(lon,6);
```

```
    gsm.print("\r\n");
    gsm.write(26);
    ok();
   }
   void Accident2()
{
gsm.print("AT+CMGS=");
gsm.write('"');
i=0;
do{
      gsm.write(number2[i]);
      i++;
    }while(i!=13);
    gsm.write('"');
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.println("ACCIDENT DETECTED");
    gsm.print("VEHICLE LOCATION");
    gsm.println();
    gsm.print("https://www.google.co.in/maps/search/");
    gsm.print(lat,6);
    gsm.write(',');
    gsm.print(lon,6);
    gsm.print("\r\n");
    gsm.write(26);
    ok();
   }
void Accident3()
{
gsm.print("AT+CMGS=");
gsm.write('"');
i=0;
do{
      gsm.write(number3[i]);
      i++;
    }while(i!=13);
    gsm.write('"');
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.println("ACCIDENT DETECTED");
    gsm.print("VEHICLE LOCATION");
    gsm.println();
    gsm.print("https://www.google.co.in/maps/search/");
    gsm.print(lat,6);
    gsm.write(',');
    gsm.print(lon,6);
    gsm.print("\r\n");
    gsm.write(26);
    ok();
   }
```

```
void help1()
{

gsm.print("AT+CMGS=");
gsm.write('"');
i=0;
do{
      gsm.write(number1[i]);
      i++;
    }while(i!=13);
    gsm.write('"');
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.print("VEHICLE LOCATION");
    gsm.println();
    gsm.print("https://www.google.co.in/maps/search/");
    gsm.print(lat,6);
    gsm.write(',');
    gsm.print(lon,6);
    gsm.print("\r\n");
    gsm.write(26);
    ok();
   }
   void help2()
{

gsm.print("AT+CMGS=");
gsm.write('"');
i=0;
do{
      gsm.write(number2[i]);
      i++;
    }while(i!=13);
    gsm.write('"');
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.print("VEHICLE LOCATION");
    gsm.println();
    gsm.print("https://www.google.co.in/maps/search/");
    gsm.print(lat,6);
    gsm.write(',');
    gsm.print(lon,6);
    gsm.print("\r\n");
    gsm.write(26);
    ok();
   }
   void help3()
{

gsm.print("AT+CMGS=");
gsm.write('"');
```

```
i=0;
do{
      gsm.write(number3[i]);
      i++;
    }while(i!=13);
    gsm.write('"');
    gsm.print("\r\n");
    do{
     rcv = gsm.read();
    }while(rcv != '>');
    gsm.print("VEHICLE LOCATION");
    gsm.println();
    gsm.print("https://www.google.co.in/maps/search/");
    gsm.print(lat,6);
    gsm.write(',');
    gsm.print(lon,6);
    gsm.print("\r\n");
    gsm.write(26);
    ok();
   }
```

CODE IN IOT:

```
#include "GSMGPS.h"
#include "MPU6050_Sensor.h"
//#include "ThingSpeakServer.h"
//long writingTimer = 20;
//long startTime = 0;
//long waitTime = 0;
/*void writeThingSpeak()
{

gsm.println("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",\"80\"");//s
tart up the connection
  delay(1000);
  ShowSerialData();
  gsm.println("AT+CIPSEND");//begin send data to remote server
  delay(1000);
  ShowSerialData();
  String str="GET
https://api.thingspeak.com/update?api_key=XYAQE37CLRVDKP6I&field1="
+
  String(random(0,100)) +"&field2="+
  String(random(100,200));/* +"&field3="+
  String(hh) +"&field4="+
  String(tt)+"&field5="+
  String(WaterTemperature);//+"&field6="+*/
 /*// String(turbidity);
  Serial.println(str);
  gsm.println(str);//begin send data to remote server
  delay(1000);
  ShowSerialData();
  gsm.println((char)26);//sending
  delay(1000);//waitting for reply, important! the time is base on
the condition of internet
```

```
  gsm.println();
  ShowSerialData();
}*/
void setup()
{

  SerialInit();
  LCDInit();
  StoreMobileNumbers();
  GPSInit();
  MPU6050_Init();
  lcd.clear();
  lcd.print("BIKE ACCIDENT");
  lcd.setCursor(0,1);
  lcd.print("MONITORING SYS");
  delay(2000);

 lp:
 lcd.clear();
 lcd.print("SEND SMS TO TRACK");
 lcd.setCursor(0,1);
 lcd.print("VEHICLE POSITION");
 while(1)
  {
    do{
      rcv=gsm.read();
      sensors_event_t a, g, temp;
      mpu.getEvent(&a, &g, &temp);
      if(a.acceleration.y>6.0)
      {
          delay(5000);
          if(a.acceleration.y>6.0)
          {
            lcd.clear();
            lcd.print("Accident Detected");
            delay(1000);
            lcd.clear();
            lcd.print("Sending SMS");
            Accident1();
            Accident2();
            Accident3();
            lcd.clear();
            lcd.print("SMS Sent");
            goto lp;

          }
      }
      }while(rcv!='*');
      if(rcv=='*')
    {
      lcd.clear();
      lcd.print("COMAND RECEIVE");
      delay(1000);
      lcd.clear();
      lcd.print("VEHICLE LOCATION");
```

```
      delay(1000);
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("LAT:");
      lcd.setCursor(4,0);
      lcd.print(lat,6);
      lcd.setCursor(0,1);
      lcd.print("LON:");
  lcd.setCursor(4,1);
  lcd.print(lon,6);
  delay(2000);
  lcd.clear();
  lcd.print("SENDING SMS");
  help1();
  help2();
  help3();
  delay(1000);
  lcd.clear();
  lcd.print("SMS SENT");
  delay(1000);
  goto lp;
    }
   }
}
void loop()
{
  ;;;
}
```