NANDHA ENGINEERING COLLEGE

ERODE–638052 (Autonomous)

(Affiliated to Anna University, Chennai)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

22AIC14 – INTERNET OF THINGS AND ITS APPLICATIONS

MINI PROJECT REPORT ON

TOPIC – DRIVER ANTI-SLEEP DEVICE USING ESP32-CAM

Submitted by

REGISTER NUMBER	NAME
22AI002	ARAVINNDHAN A
22AI017	HARISH S.S
22AI056	VELUSAMY G

NANDHA ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

This is to certify that the project work entitled "DRIVER ANTI SLEEP DEVICE USING ESP32-CAM" is the Bonafide work of ARAVINDHAN A(22AI002), HARISH S.S(22AI017), VELUSAMY G(22AI056) who carried out the work under my supervision.

Signature of the Supervisor
Dr. K. Lalitha,
Professor,
Department of AI & DS,
Nandha Engineering College,
Erode – 638052.

Signature of the HOD
Dr. P. Karunakaran,
Head of the Department,
Department of AI & DS,
Nandha Engineering College,
Erode – 638052.

Submitted for End semester PBL review held on _____

DRIVER ANTI-SLEEP DEVICE USING ESP32 CAM

AIM:

To design and implement a real-time drowsiness detection system using the ESP32-CAM module. This system monitors the driver's eye movements and alerts them in case of prolonged eye closure, aiming to reduce the risk of accidents caused by driver fatigue.

SCOPE:

The **driver anti-sleep device using ESP32-CAM** has a wide-ranging scope due to its potential applications in ensuring road safety, its affordability, and its ability to be integrated into various environments.

BRIEF HISTORY:

The development of **driver anti-sleep device using the ESP32-CAM** reflects evolution of driver safety technologies, embedded systems, and real-time monitoring solutions.

PROPOSED METHODOLOGY:

Proposed Methodology for Driver Anti-Sleep Device Using ESP32 CAM

1. **Hardware Setup**
    - Assemble the ESP32 CAM module, MQ3 alcohol sensor, LCD display, SIM800C GSM module, buzzer, and relay with the required power supply.
    - Integrate the ESP32 controller with all peripherals for centralized control and data processing.

2. **Eye Status Detection**
    - Use the ESP32 CAM to capture real-time video of the driver.
    - Apply machine learning algorithms in MATLAB to analyze eye status (open or closed) and determine drowsiness.
    - Transmit processing results back to the ESP32 for further actions.
3. **Alcohol Detection**
    - Use the MQ3 alcohol sensor to monitor alcohol levels in the driver's breath.
    - Trigger alerts and ignition lock if alcohol consumption exceeds the threshold.
4. **Alert Mechanisms**
    - Activate the buzzer to alert the driver upon detecting drowsiness or alcohol consumption.
    - Display alerts and system status on the 16x4 LCD screen for real-time feedback.
5. **Vehicle Control**
    - Use the relay module to deactivate the vehicle ignition system if drowsiness persists or alcohol is detected.
6. **Emergency Notifications**
    - Utilize the SIM800C GSM module to send SMS notifications to predefined emergency contacts with location details.
7. **IoT and Cloud Integration** *(Optional)*
    - Use MQTT or HTTP protocols to upload data to IoT platforms for remote monitoring and analysis.
8. **Testing and Validation**
    - Test the system under various conditions, including different lighting, alcohol levels, and drowsiness scenarios.
    - Calibrate the system for optimal performance and reliability.

This methodology ensures a comprehensive approach to detecting and mitigating drowsiness and alcohol-related risks, enhancing road safety.

COMPONENTS REQUIRED:

S.NO	COMPONENTS	NO'S
1	ESP32 microcontroller	1
2	ESP32 cam	1
3	MQ3 Sensor	1
4	12V Relay	1
5	16*4 LCD display	1
6	GSM module	1
7	5V buzzer	1
8	DC motor	1

DESCRIPTION:

The Drowsiness Detection System using ESP32-CAM is an innovative and cost-effective solution designed to enhance road safety by preventing accidents caused by driver fatigue or alcohol impairment. This system employs the ESP32-CAM module for real-time video capture, enabling continuous monitoring of the driver's facial features, particularly the eyes, to detect signs of drowsiness. By utilizing computer vision techniques and machine learning algorithms, the system analyzes the driver's eye status (open or closed) over successive frames to identify prolonged closure, a key indicator of drowsiness.

Additionally, the system integrates an MQ3 alcohol sensor to measure the alcohol concentration in the driver's breath, ensuring the vehicle is operated safely. Alerts are triggered in case of drowsiness or alcohol detection, with multiple mechanisms in place: a buzzer sounds an alarm, a 16x4 LCD display shows warning messages, and a SIM800C GSM module sends SMS notifications to emergency contacts. For added safety, the system uses a 12V DC relay to control the vehicle's ignition, effectively stopping the vehicle if unsafe conditions are detected.

The system also leverages IoT technology via the ESP32's Wi-Fi module to log and transmit real-time data to an online platform, enabling continuous monitoring and analysis. Compact, efficient, and easy to install, the Drowsiness Detection System provides a robust safeguard for drivers and passengers, combining advanced technology with practical application to reduce road accidents.

CODING:

```
#include <WiFi.h>
#include <WebServer.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#define ssid      "Driver"       // WiFi SSID
#define password  "project123"  // WiFi password
#include "MAX30100_PulseOximeter.h"
long GPS;
long GPS1;
int AcX, AcY, AcZ, fl, y, x, f2, ge, te, hb, hb1, sdnn, rmssd, sp, al;
//SoftwareSerial mySerial(15, 13);
LiquidCrystal_I2C lcd(0x27, 16, 2);
String   t = "NULL";
String   t1 = "NULL";
String   t2 = "NULL";
String   t3 = "NULL";
char   p ;
String  etatLed = "";
#define button1 16
WebServer server ( 80 );

String getPage() {
  String page = "<html lang=fr-FR><head><meta http-equiv='refresh' content='2'/>";
  page += "<title>DRIVER SLEEP MONITORING SYSTEM</title>";
  page += "<style> body { background-color: #9AFEFF; font-family: Arial, Helvetica, Sans-
Serif; Color: #000088; }</style>";
  page += "</head><body><h1>DRIVER SLEEP MONITORING SYSTEM</h1>";
page += "<ul><li>DRIVER STATUS: ";
page += t1;
 page += "</li></ul>";
page += "<ul><li>ALCOHOL STATUS: ";
 page += t2;
  page += "</li></ul>";

return page;

}
void handleRoot() {
  if ( server.hasArg("LED") ||  server.hasArg("LED1")) {
    handleSubmit();
  } else {
    server.send ( 200, "text/html", getPage() );
  }
}
void handleSubmit() {
  // Actualise le GPIO / Update GPIO
  String LEDValue, LEDValue1;
  LEDValue = server.arg("LED");
```

```
  LEDValue1 = server.arg("LED1");

  //Serial.println("Set GPIO "); Serial.print(LEDValue);
  if ( LEDValue == "0" ) {
    //  Serial.println ("B");
    digitalWrite(14, LOW);
    t = "MOTOR OFF";
 server.handleClient();
    delay(2000);
    etatLed = "OFF";
    server.send ( 200, "text/html", getPage() );
  } else if ( LEDValue == "1" ) {
    // Serial.println ("A");
    digitalWrite(14, HIGH);
    etatLed = "ON";
    t = "MOTOR ON";
    //     hb = random(3,5);
    //lcd.setCursor(0,10);
    //  lcd.print("I:");
    //  lcd.print(hb);
    server.handleClient();
    delay(2000);
    server.send ( 200, "text/html", getPage() );

  } else {
    // Serial.println("Err Led Value");
  }
  }
void setup() {

  Serial.begin(9600);
  server.handleClient();
    WiFi.softAP(ssid, password);
    IPAddress myIP = WiFi.softAPIP();
    // On branche la fonction qui gère la premiere page / link to the function that manage
launch page
    server.on ( "/", handleRoot );
    server.begin();
serialEvent();

  lcd.init();
  // turn on LCD backlight
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("DRIVER");
  lcd.setCursor(0, 1);
  lcd.print("DROWSINESS SYSTM");
 delay(200); // Pause for 2 seconds
init_modem();
  // mySerial.begin(9600);
```

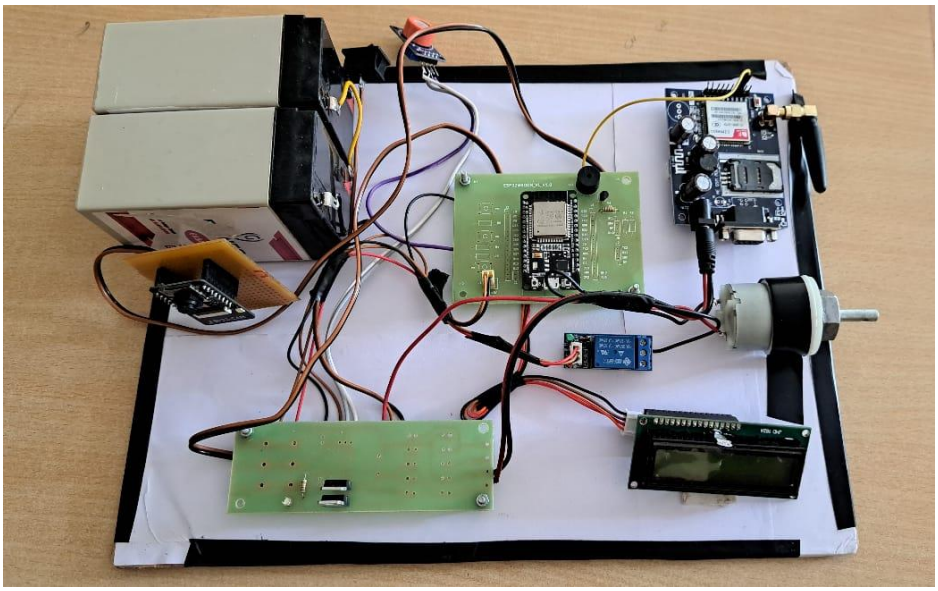
```
 // pinMode(2, OUTPUT);
  pinMode(12, INPUT_PULLUP);
  pinMode(23, OUTPUT);
  pinMode(14, OUTPUT);
 digitalWrite(23, LOW);
  digitalWrite(14, LOW);
  randomSeed(analogRead(0));
 //digitalWrite(2, LOW);
  delay(1000);
  //digitalWrite(12, LOW);
  lcd.clear();

}
void init_modem()
{
     Serial.print("AT\n\r");
    delay(300);
    Serial.print("ATE0\n\r");
    delay(300);
    Serial.print("AT+CREG=1\n\r");
    delay(300);
    Serial.print("AT+CSMS=1\n\r");
    delay(300);
    Serial.print("AT+CSCS=\"GSM\"\n\r");
    delay(300);
    Serial.print("AT+CMGF=1\n\r");
    delay(300);
    Serial.print("AT+CNMI=2,2,0,0,0\n\r");
       delay(300);
      Serial.print("AT+CNMI=1,2,0,0,0");
    delay(300);

}
void send_sms1()
{

     init_modem();
  Serial.println("AT");
    delay(300);
    Serial.println("ATE0");
    delay(300);

    Serial.println("AT+CMGF=1");
    delay(300);
    Serial.println("AT+CMGS=\"9025752023\"\r"); // Replace x with mobile number
    delay(300);
    Serial.print("DRIVER SLEPT ALERT");
//     Serial.print("11.");

//      Serial.println(GPS);
```

```
//      Serial.print("N");
//      Serial.print("78.");
//      Serial.println(GPS1);
//      Serial.print("E");
//      Serial.print("Temp.");
//      Serial.println(c1);
//      Serial.print("HB.");
//      Serial.println(c2);
      delay(500);

  Serial.write((byte)0x1A);
    delay(3000);
    lcd.clear();
    lcd.print("sms sent1");
     delay(2000);
 }

void send_sms2()
{
   init_modem();
  Serial.println("AT");
    delay(300);
    Serial.println("ATE0");
    delay(300);
     Serial.println("AT+CMGF=1");
    delay(300);
    Serial.println("AT+CMGS=\"9025752023\"\r"); // Replace x with mobile number
    delay(300);
    Serial.print("ALCOHOL DRUKEN ALERT");
//     Serial.print("11.");
//      Serial.println(GPS);
//      Serial.print("N");
//      Serial.print("78.");
//      Serial.println(GPS1);
//      Serial.print("E");

//      Serial.print("Temp.");
//      Serial.println(c1);
//      Serial.print("HB.");
//      Serial.println(c2);
      delay(500);
 Serial.write((byte)0x1A);
    delay(3000);
    lcd.clear();
    lcd.print("sms sent2");
     delay(2000);
 }
void serialEvent()
{
  while (Serial.available())
```

```
  {
    char inChar = (char)Serial.read();
    //Serial.print(inChar);

    if (inChar == 'A')
    {
      t1 = "SLEPT-->> EMERGENCY ALERT";
      t2 = "NORMAL";
      digitalWrite(23, HIGH);
      lcd.setCursor(0, 0);
      lcd.print("DRIVER ");
      lcd.setCursor(0, 1);
      lcd.print("SLEPT.");
      //server.handleClient();
      delay(10000);
       send_sms1();
      delay(10000);
      delay(10000);
      lcd.clear();
    }
    else
    {
       t1 = "SLEPT-->> EMERGENCY ALERT";

      t2 = "NORMAL";
      digitalWrite(23, HIGH);
      lcd.setCursor(0, 0);
      lcd.print("DRIVER ");
      lcd.setCursor(0, 1);
      lcd.print("SLEPT");
      //server.handleClient();
      delay(10000);
       send_sms1();
      delay(10000);
      delay(10000);
      lcd.clear();
    }
  }
}
void loop()
{
 al = analogRead(34);
   serialEvent();
   if (al<200)
   {
    serialEvent();
     t2 = "ALCOHOL DRUNKEN-->> EMERGENCY ALERT";
      t1 = "NORMAL";
  lcd.clear();
    lcd.setCursor(0, 0);
```

```
 lcd.print("ALCOHOL");
 lcd.setCursor(0, 1);
 lcd.print("DRIVING");
 server.handleClient();
 delay(500);
 digitalWrite(23, HIGH);
 send_sms2();
 delay(10000);
 delay(10000);
 }
 Else

 {serialEvent();
  t2 = "NORMAL";
   t1 = "NORMAL";
 lcd.clear();
 lcd.setCursor(0, 0);
 lcd.print("NORMAL");
 server.handleClient();
 digitalWrite(23, LOW);
 delay(500);
 }
}
```

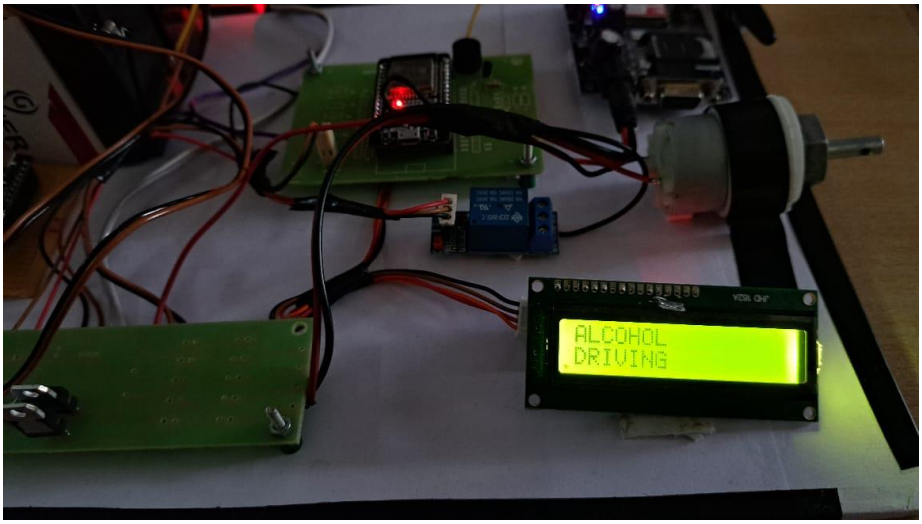
SCREENSHOTS:



OUTPUTS:

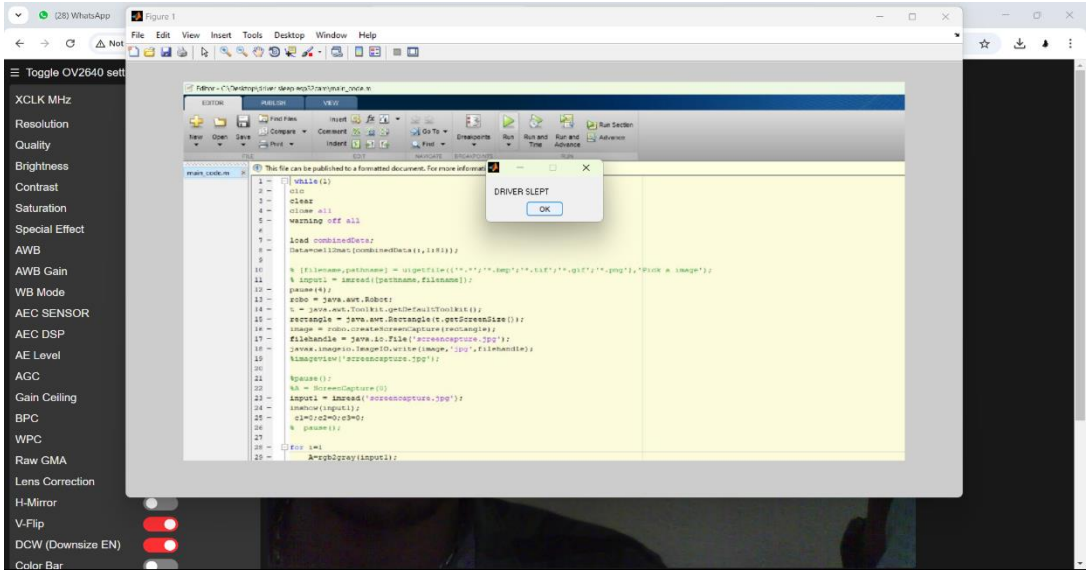
Alcohol Detection

- **Detects alcohol levels using the MQ3 sensor and triggers alerts if levels exceed the threshold.**



Drowsiness Detection

- **Real-time detection of closed eyes or signs of drowsiness.**
- **Alerts the driver when drowsiness is detected.**



Alerts and Notifications

- **Audible alerts (buzzer) to wake up the driver.**
- **SMS notifications sent to emergency contacts using the GSM module.**

PROTOCOLS:

Protocols for Driver Anti-Sleep Device Using ESP32 CAM

1. **HTTP/HTTPS**
    - For sending data to web servers or IoT platforms and enabling remote monitoring through web-based interfaces.
2. **MQTT (Message Queuing Telemetry Transport)**
    - A lightweight protocol for efficient communication with IoT platforms, enabling real-time data exchange.
3. **UART (Universal Asynchronous Receiver-Transmitter)**
    - For serial communication between the ESP32 controller, GSM module, and other components like the LCD.
4. **I2C (Inter-Integrated Circuit)**
    - Used for interfacing peripherals such as LCD displays or sensors requiring low-speed communication.
5. **SPI (Serial Peripheral Interface)**
    - For high-speed communication, often used with camera modules or SD card storage.

6. **GSM Protocol**
    - For SMS transmission via the SIM800C module, sending alerts to emergency contacts.
7. **TCP/IP**
    - For reliable data transmission over Wi-Fi, supporting IoT connectivity.

These protocols collectively ensure efficient communication and seamless integration of hardware and software components.

LIMITATIONS:

1. **Limited Processing Power :** Restricted computational capacity limits the use of advanced AI models.
2. **Lighting Dependency :** Struggles in low-light conditions without additional IR components.
3. **Network Dependence :** IoT features require consistent internet, which may not always be available.
4. **False Positives/Negatives :** Potential for misclassification, leading to incorrect alerts.
5. **Single Sensor Reliance :** Limited to visual data, missing other fatigue indicators like posture or vitals.
6. **Limited Range :** Only monitors the driver directly in front of the camera.
7. **Power Consumption :** High energy usage from camera and Wi-Fi can drain batteries quickly.
8. **Privacy Concerns :** Capturing and processing video data may raise privacy and security issues.
9. **Hardware Sensitivity :** Performance can be affected by vehicle vibrations or extreme conditions.
10. **Cost of Additional Features :** Adding GPS, IR cameras, or advanced sensors increases system cost.
11. **Driver Compliance :** Drivers may ignore or disable alerts, reducing effectiveness.
12. **Environmental Limitations :** Poor performance in extreme weather like fog, rain, or bright sunlight.
13. **Scalability Issues :** System may require significant upgrades for fleet-level deployment.
14. **Frequent Calibration :** Requires recalibration for different drivers or environments to remain accurate.
15. **Limited Health Monitoring :** Does not track critical health indicators like heart rate or stress.

FUTURE ENHANCEMENTS:

1. **AI-Based Analytics**: Use advanced machine learning models like TensorFlow Lite for improved accuracy.
2. **Yawning Detection**: Add facial landmark tracking to detect yawning and other fatigue signs.
3. **Night Vision**: Integrate IR cameras for effective detection in low-light conditions.
4. **Driver Identity Recognition**: Implement facial recognition for driver verification and personalized settings.
5. **Emergency Location Tracking**: Add a GPS module to send the vehicle's location during emergencies.
6. **Automatic Vehicle Control**: Enable slowing or stopping the vehicle if the driver remains unresponsive.
7. **IoT Connectivity**: Use cloud platforms like Firebase for real-time data storage and monitoring.
8. **Enhanced Alerts**: Include vibrating seats, steering wheels, or Heads-Up Displays (HUD) for better warnings.
9. **Health Monitoring**: Add sensors to measure heart rate or temperature for health tracking.
10. **Multi-Driver Profiles**: Support personalized settings for different drivers.
11. **Mobile App Integration**: Develop an Android/iOS app for alerts, video feeds, and system controls.
12. **Energy Optimization**: Enhance power efficiency for longer system uptime.
13. **Alcohol Detection Upgrade**: Use advanced gas sensors for accurate alcohol detection.
14. **Affordable Design**: Reduce costs to make the system more accessible for wider use.
15. **Data Security**: Strengthen encryption and secure communication for privacy protection.

CONCLUSION:

The Drowsiness Detection System using ESP32-CAM is an effective and affordable solution for improving road safety. By monitoring the driver's eyes and alcohol levels, it can quickly detect drowsiness or intoxication and take action to prevent accidents. The system provides alerts through a buzzer, LCD display, SMS notifications, and can even stop the vehicle if needed.

With its simple design, real-time monitoring, and potential for future upgrades like yawning or head posture detection, this system is a reliable and practical tool for reducing road accidents and keeping drivers and passengers safe.