DROWSINESS DETECTION SYSTEM

DESCRIPTION: Develop a simple Microcontroller based system that can detect the drowsiness of drivers and can alert the driver

PROBLEM STATEMENT: Driving while fatigued is a serious safety hazard that puts both drivers and pedestrians at danger. Drowsiness or fatigue while driving can hamper a driver's focus and quick reflexes, increasing the risk of collisions and fatalities. This project aims to create a dependable and effective microcontroller-based system that can recognize indicators of driver drowsiness and send out timely alerts to reduce the possible risks related to driving when sleepy. For the safety of all users of the road, the system must correctly track the condition of the driver and issue warnings in real-time.

SCOPE OF SOLUTION: This project's scope includes the design, development, and deployment of a microcontroller-based system to detect driver weariness and send out prompt alerts to avert accidents. The following essential elements and capabilities will be part of the solution:

1.Sensor Integration: Adaptive sensors should be used to track drowsiness-related physiological signals and driver behavior, such as facial recognition, eye tracking, or other physiological sensors.

2. Data analysis and processing: Analyze the sensor data to spot sleepiness symptoms by looking for patterns. Use algorithms to differentiate between regular driving behavior and drowsiness-related behaviors.

3. Real-time Monitoring: Enable the driver's condition to be monitored in real-time while continuously analyzing sensor data to spot early indicators of fatigue.

4. Alert generation: Implement an alarm system to warn the driver when drowsiness is detected, such as aural, visual, or haptic signals. The alerts ought to be created so that they draw the driver's attention without becoming distracting.

5. Integration with Vehicle Systems: Connect the solution to the car's systems so that when drowsiness is identified, certain actions, such as seat vibrations, flashing lights, or alarms, are triggered.

6. Power Efficiency: Create the system with minimal impact on the electrical system and battery life by designing it to run with low power consumption.

7. User Interface: Create a user interface that allows users to configure the system, change the alert levels, and check the status of the system. For convenience of usage, this could include a display or a mobile application.

8. Safety and Reliability: To protect the safety of the driver and other road users, ensure the system's dependability and accuracy in detecting drowsiness by minimizing false positives and false negatives.

9. Validation and Testing: To assure the system's effectiveness and dependability in real-world applications, carry out thorough testing to confirm the system's performance under a variety of driving circumstances and scenarios.

10. Documentation and User Instructions: To make the drowsiness detection system easier to comprehend, install, and operate, create thorough documentation that includes user manuals and instructions.

REQUIRED COMPONENETS TO DEVELOP SOLUTIONS (Including IDE name, software and hardware)

Hardware Components:

1. Microcontroller: Arduino UNO R3

2. Sensors: IR Sensor

3. Alert Mechanisms: LED, Buzzer

4. Power Supply: Battery or appropriate power supply to run the system in a vehicle.

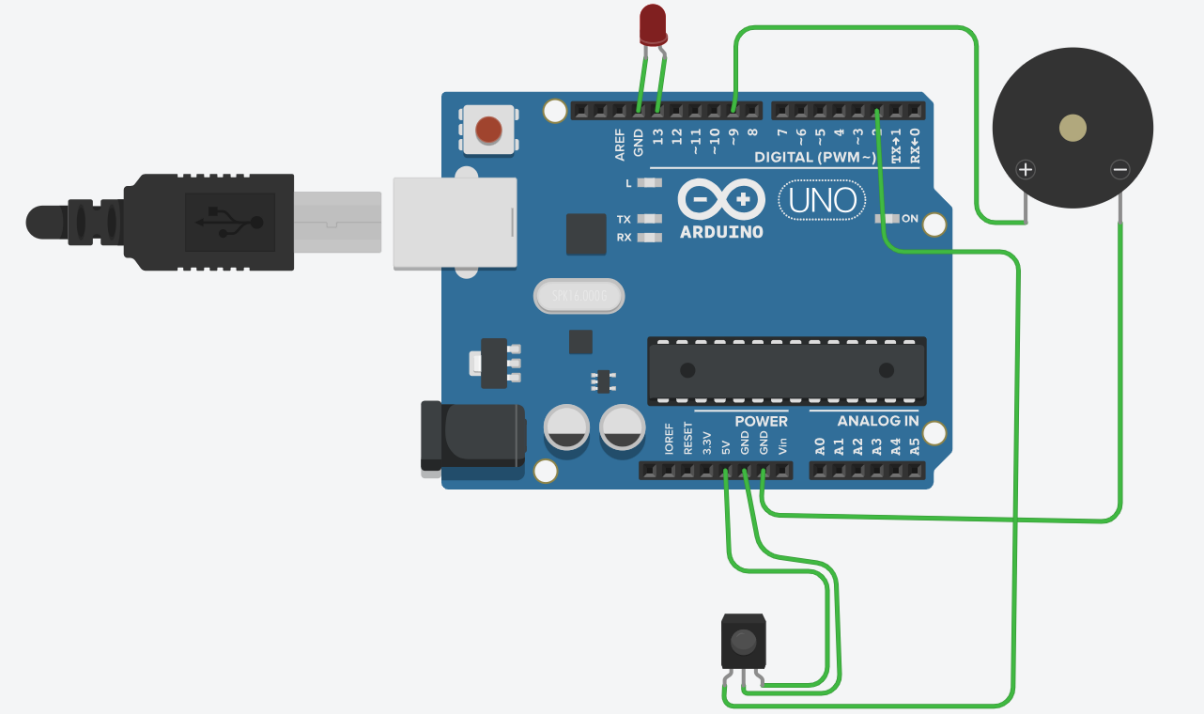
Software Components:

1. Integrated Development Environment (IDE): Arduino IDE

2. Programming Languages: C/C++ for programming the microcontroller

3.Simulation and Modeling Tools: TinkerCAD

SIMULATED CIRCUIT (TinkerCAD/ Fritzing)



CODE FOR THE SOLUTION

```
const int irPin = 2;  // IR sensor output pin
const int ledPin = 13;  // LED connected to pin 13
const int buzzerPin = 9;  // Buzzer connected to pin 9
void setup() {
  pinMode(irPin, INPUT);
  pinMode(ledPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
}
void loop() {
  int irValue = digitalRead(irPin);
  // IR sensor detects an object (simulating drowsiness)
  if (irValue == HIGH) {
    digitalWrite(ledPin, HIGH);  // Turn on LED
    tone(buzzerPin, 1000);  // Play a tone on the buzzer
    delay(500);  // Delay for a brief period
    digitalWrite(ledPin, LOW);  // Turn off LED
    noTone(buzzerPin);  // Stop the buzzer sound
  } else {
    digitalWrite(ledPin, LOW);  // No drowsiness detected, turn off LED
    noTone(buzzerPin);  // No drowsiness detected, stop the buzzer sound
  }
}
```