DROWSINESS DETECTION GLASS WITH IOT

ABSTRACT

Drowsiness detection glasses utilizing eyeblink sensor technology represent a proactive approach to enhancing safety and preventing accidents, particularly during activities such as driving or operating heavy machinery. This project aims to integrate an eyeblink sensor seamlessly into the glasses, allowing real-time processing and analysis of blink data to identify patterns indicative of drowsiness. The system is designed to provide timely alerts to users upon detecting signs of drowsiness, enhancing their awareness and reducing the risk of accidents. Key components include developing efficient algorithms for blink pattern recognition, designing effective alert mechanisms, creating a user-friendly interface for configuration and monitoring, and optimizing power efficiency for prolonged usage. Through rigorous testing and validation, the drowsiness detection glasses aim to offer a reliable and user-friendly solution for promoting alertness and safety in diverse environments. In developing this technology, we aim to help people stay safe and avoid accidents by giving them a heads-up when they need to take a break or rest.

INTRODUCTION

- To create a simple IoT based system to detect drowsiness.
- Making the wheels of the truck to stop when the driver is not in control.
- The project aim is to save the life of the driver by preventing the accidents.
- This project is used to reduce accidents in rode during night time.

CHAPTER 2
EXISTING SYSTEM

2.1 EXISTING SYSTEM

Air bag is the only safety equipment used in the truck for the protection of the driver. Other than there are no other safety concerns for the driver. These systems are really good at helping people stay safe, especially when driving or working, by alerting them when they need to pay more attention or take a break to avoid accidents caused by drowsiness.

DISADVANTAGE

- Limitations of Drowsiness Detection System False Alarms
- Some drowsiness detection systems may generate false alarms due to factors such as bright lights, sudden movements, or changes in the driver's posture.
- Reliance on Technology: Drowsiness detection systems rely on technology, which can malfunction or fail

PROPOSED SYSTEM

The main aim of this project is to save the life of the driver from big accidents by stopping the truck when the driver feels sleepy. This system uses eye blink sensor to detect the driver activities during drive. This system uses sensors and slow down the speed of the truck and stop the truck to save the driver.

ADVANTAGES

- Save truck from accident.
- Less cost to install.
- Low maintenance and high safety.

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

Processor	-	CORE i5-processor10 or higher
RAM	-	8GB or higher
SSD	-	512 GB
Operating System	-	Windows 11 or higher

3.2 SOFTWARE REQUIREMENTS

ARDIUNO IDE

CHAPTER 4

PROJECT DESCRIOTION

4.1PROJECT DESCRIPOTION

For any vehicle accidents driver's faults are the most accountable aspect to cause dangerous problems to the society. Many drivers cannot control the vehicles due to different reasons it may cause accidents and sometime death. For vehicle accidents various factors involved such as drunk driving, over speeding, many distractions like texting while driving, talking with others, playing with children etc. The Government of India, Ministry of Road Transport and Highway Government of India prepare a strategy to diminish the amount of motorway accidents and losses by 50 % by 2023.

4.2MODULES

There are three modules in our project:

- Drowsiness detection.
- Buzzer sound.
- stopping truck**.**

4.2.1 Drowsiness detection

When the driver closes his eye for over 20 seconds the eye blink sensor detect that the driver is in sleep. The signal from the sensor is transfer to the Arduino nano.

4.2.2 Buzzer sound:

If the Drowsiness is detected the micro controller will send signal to the buzzer. It makes the buzzer to make buzz sound. It may casus the driver woke up.

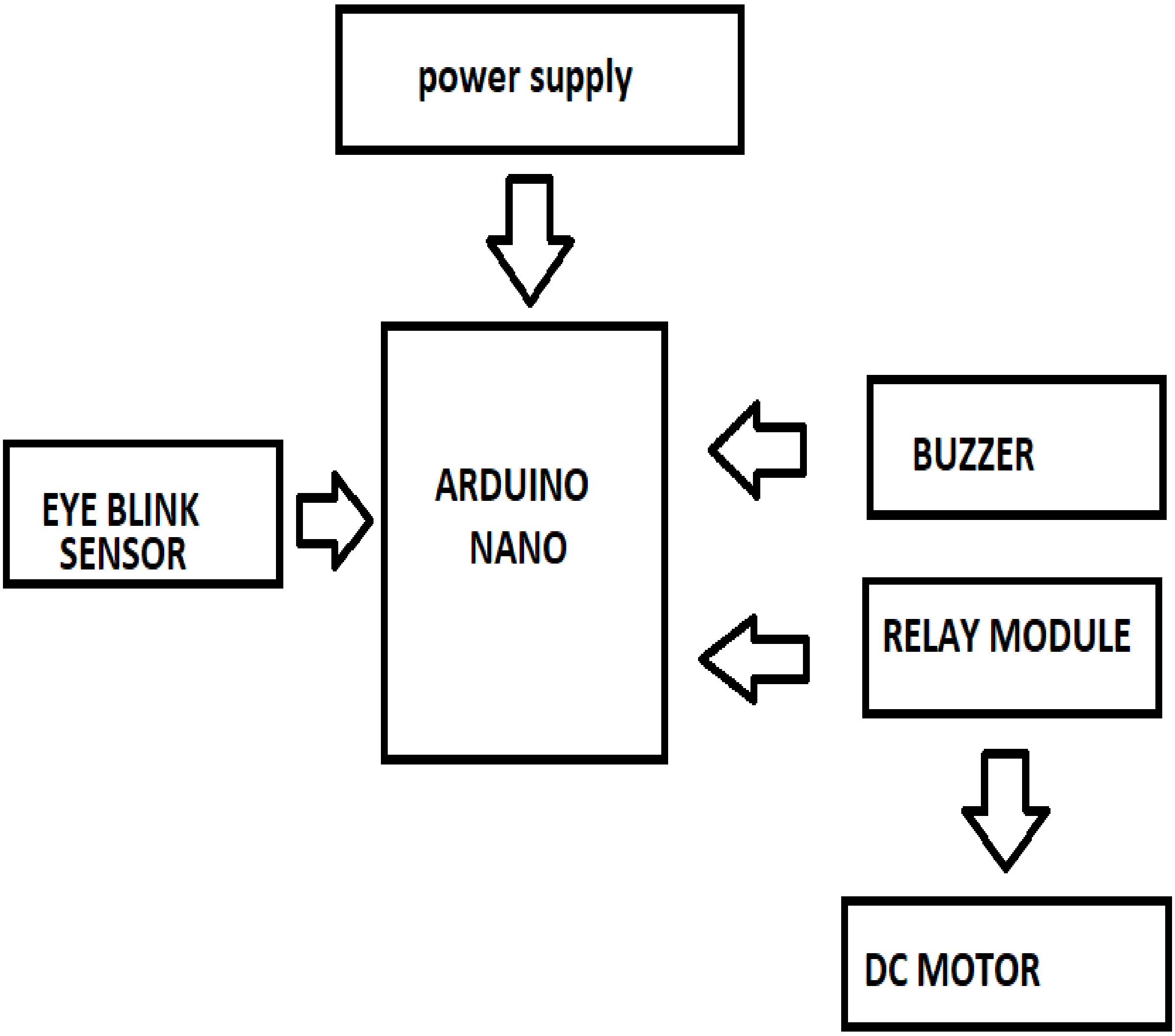
4.2.3 Stopping truck:

Later the break of the truck will apply then the truck will be stopped. All this event will be held between 15-20 seconds. By this the truck will not get into big accidents.

CHAPTER 5

SYSTEM DESIGN

5.1 WORKING FLOW DIAGRAM



CHAPTER 6

SOFTWARE & HARDWARE DESCRIPTION

6.1 SOFTWARE DESCRIPTION

6.1.1 ARDIUNO IDE:

Arduino IDE is an **open-source software**, designed by Arduino.cc and mainly used for writing, compiling & uploading code to almost all Arduino Modules. It is an official Arduino software, making code compilation too easy that even a common person with no prior technical knowledge can get their feet wet with the learning process**.**

6.2 HARDWARE DESCRIPTION:

6.2.1 CORE I5 PROCESSOR:

Developed and manufactured by intel, and first introduced and released at 2009, the core 15 processor is dual core computer processor, available for use in both desktop and laptop computers. It is one of the processors in the “i” series processor introduced by intel. The core i5 processor also called as intel core family of processor.

6.2.2 SSD DESCRIPTION:

Capacity:512gb (Gigabytes) type: solid state drive (SSD) interface: serial ATA for faster speed in some cases.

6.2.3RAM DESCRIPTION:

Capacity:16GB (Gigabytes), Type: DD4 (Double data rate), Form factor: DIMM (DUAL IN-LINE MEMORY MODULE), Speed: the speed of the ram module is measured in MHz, indicates how quickly data can be transferred to the memory.

CHAPTER 7
SYSTEM TESTING

SYSTEM TESTING:

The purpose of testing is to discover errors in testing and discovering conceivable fault or weakness in a working product if provides a way to check the functionality of components, subassemblies or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of testing Each test type addresses a specific testing requirement.

7.1 TYPES OF TESTS:

- Unit Testing
- Integration Testing
- Performance Testing
- Security Testing

7.1.1-unit testing:

Unit testing is an important part of ensuring that the individual learning system for malware detection correctly and producing the expected outputs. By testing each component in isolation. Developers can identify and address issues early on, before they larger problems in the overall systems.

7.1.2 INTERGRATION TESTING

Integration testing is an important part of ensuring that the different components of the machine learning systems for malware detection are working together as expected. By testing the integration of different components, developers can identify address issues early on, before they become big problems in the overall system.

7.1.3 PERFOEMANCE TESTING

- Testing the system's processing speed
- Testing the system's accuracy
- Testing the system's scalability
- Testing the system's resource usage

7.1.4 SECURITY TESTING

Security testing is an important part of ensuring that the machine learning system for malware detection is secure and that it is protecting user data and system resources from different types of attacks. By testing the system's resilience to different types of attacks and itsdata protection mechanisms. developers can identify and address any security vulnerabilities early on, before they become larger problems in the overall system.

CHAPTER 8

SYSTEM IMPEMENTATION

- Circuit Design and Assembly
- Programming the Microcontroller
- Integration with drivers
- Deployment and Monitoring

Circuit Design and Assembly:

Design the circuit layout based on the selected components. Connect the microcontroller to the eye blink sensor. Ensure proper wiring and connections to avoid any short circuits or malfunctions.

Programming the Microcontroller:

Write the firmware code for the microcontroller. Implement algorithms to read data from the eye blink sensor. Program the microcontroller to control the braking system and alarm system.

Integration with drivers:

Ensure proper electrical connections and compatibility with existing infrastructure. Mount the sensors and microcontroller in suitable enclosures for driver use.

Deployment and Monitoring:

Deploy the Drowsiness detection glass with IoT in targeted areas. Monitor system performance and functionality over time. Address any technical issues or maintenance requirements as they arise.

CHAPTER 9

SYSTEM MAINTANCE

REGULAR UPDATES:

Keep the chatbot's software up-to-date with the latest patches, bug fixes, and improvements. This includes updating the underlying AI models, natural language processing (NLP) algorithms, and any integrations with external systems.

Performance Monitoring:

Implement monitoring tools to track the chatbot's performance metrics, such as response time, accuracy, user satisfaction, and error rates. Identify any issues or bottlenecks and take proactive measures to address them.

User Feedback Analysis:

Collect feedback from users regarding their interactions with the chatbot and analyze it toidentify areas for improvement. Use this feedback to fine-tune the chatbot's responses, improve its usability, and enhance the overall user experience.

Compliance Management:

Ensure that the chatbot complies with relevant healthcare regulations and standards, such as HIPAA, GDPR (General Data Protection Regulation), and any industry-specific regulations. Implement measures to protect patient data, maintain confidentiality, and ensure data security.

Training and Retraining:

Continuously train and retrain the chatbot's AI models using new data to improve itsaccuracy and relevance. Incorporate feedback from healthcare professionals to refine the chatbot's understanding of medical terminology, context and patient needs.

CHAPTER 10

CONCLUSION & FUTERE ENHANCEMENT

10.1 CONCLUSION

Drowsiness detection glasses with an eyeblink sensor offer a promising solution to enhance safety, especially during tasks like driving where staying alert is crucial. By using advanced technology to monitor eyeblink patterns and detect signs of drowsiness, these glasses can alert wearers in time to prevent accidents. While there's room for improvement in accuracy and usability, these glasses represent a significant step forward in proactive safety measures and hold great potential for making our daily activities safer and more secure.

10.2 FUTURE ENHANCEMENT

- ➢ Sensor calibration test
- ➢ Power consumption
- ➢ functionality

- ❖ **Sensor calibration:** The sensor calibration is for find the eyeblink sensor is work correctly or not.
- ❖ **power consumption**: This for the microcontroller and eye blink sensor is taking the correct amount of current.
- ❖ **Functionality:** This is to find the hardware and software are working correctly or not

CHAPTER 11

APPENDIX

```
#define Relay 13
  #define buzzer A0
  static const int sensorPin = 10;
  int SensorStatePrevious = LOW;
  unsigned long minSensorDuration = 3000;
  unsigned long minSensorDuration2 = 6000;
  unsigned long SensorLongMillis;
bool SensorStateLongTime = false;
const int intervalSensor = 50;
unsigned long previousSensorMillis;
 unsigned long SensorOutDuration;
unsigned long currentMillis;
void setup() {
    Serial.begin(9600);
   pinMode(sensorPin, INPUT);
    Serial.println("Press button");
    pinMode(Relay,OUTPUT);
    pinMode(buzzer,OUTPUT);
  }
void readSensorState() {
if(currentMillis - previousSensorMillis > intervalSensor) {
int SensorState = digitalRead(sensorPin);
if (SensorState == LOW && SensorStatePrevious == HIGH &&
!SensorStateLongTime) {
       SensorLongMillis = currentMillis;
```

```
        SensorStatePrevious = LOW;
Serial.println("Button pressed");}
 SensorOutDuration = currentMillis - SensorLongMillis;
 if (SensorState == LOW && !SensorStateLongTime && SensorOutDuration >=
minSensorDuration) {
       SensorStateLongTime = true;
       digitalWrite(Relay,HIGH);
       Serial.println("Button long pressed");
     }
     if (SensorState == LOW && SensorStateLongTime && SensorOutDuration
>= minSensorDuration2) {
if (SensorState == HIGH && SensorStatePrevious == LOW) {
       SensorStatePrevious = HIGH;
       SensorStateLongTime = false;
       digitalWrite(Relay,LOW);
       digitalWrite(buzzer,LOW);
       Serial.println("Button released");}
previousSensorMillis = currentMillis;}  }
void loop() {
 currentMillis = millis();
readSensorState(); }
```