A MINI PROJECT REPORT

ON

SLEEPWALKING DETECTOR AND PREVENTOR

Submitted

in partial fulfilment requirements for the credit of the

Course

On

Internet Of Things

CC2002-1

by

Dhanya T Kunder (NNM22CC015)

K.P Bhoomika Rai (NNM22CC026)

Thrisha J Shetty (NNM22CC064)

Under the guidance

Of

Mr. Krishna Prasad D S

Assistant Professor

Department of Computer and Communication Engineering

2024-25

DEPARTMENT OF COMPUTER AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the following students **DHANYA T KUNDER(NNM22CC015), K.P BHOOMIKA.RAI(NNM22CC026)** and **THRISHA J SHETTY(NNM22CC064)** have successfully completed the mini project work on 'Sleepwalking Detector and Preventor' and submitted in partial fulfilment of the requirements of the Course on **Internet Of Things** prescribed by the **NMAMIT, Nitte** during the academic year **2024-2025**.

Course Instructor **Professor and Head**

Mr. Sandeep Kumar S **Dr. Radhakrishna**

Assistant Professor

TABLE OF CONTENTS

CHAPTER 1

ABSTRACT

The Sleepwalking Detector and Preventor is an IoT-based wearable device designed to aid those who experience sleepwalking. Shaped like a wristwatch, it monitors the wearer's movements and delivers a mild electrical shock to awaken them if significant motion indicating sleepwalking is detected. Key features include a manual pause button to disable the device temporarily, such as when the user wakes at night, and a morning alarm function utilizing the same shock mechanism. The device consists of a motion sensor, a microcontroller to process sensor data, and an output module for controlled shock delivery. This compact, battery-powered solution offers a proactive and user-friendly approach to preventing potentially harmful sleepwalking incidents, making it both practical and beneficial for improving safety and quality of life.

CHAPTER 2

INTRODUCTION

2.1 Introduction to Mini Project

People with sleepwalking, also known as somnambulism, engage in activities like walking while still feeling sleepy. This can result in risky circumstances like falling, straying into dangerous locations, or inadvertently leaving the house. There is a need for an intelligent system that can identify sleepwalking and stop accidents without causing needless agony, even while conventional remedies like medicine or physical barriers might not always work.

The goal of this project is to develop a wearable, Internet of Things-based device that can identify and stop sleepwalking. The sleepwalker's wearable gadget and the guardian's monitoring system are the two main parts of the system. The wearable gadget has a push button, a shock module, a pedometer or distance sensor, and an ESP32 microprocessor. The pedometer tracks the sleepwalker's movement, and the device wakes them up with a little electric shock if they go over a certain threshold (for example, five meters). The sleepwalker can halt the shock and reset the system for ten minutes by pressing the push button.

When the sleepwalker goes beyond of the designated range, the guardian's monitoring system—which is also based on the ESP32—notifies them via Bluetooth and sounds a wired buzzer. This offers a responsive, non-invasive method of preventing sleepwalking that minimises interruption while guaranteeing safety.

2.2 Introduction to ESP32 & Step Counter

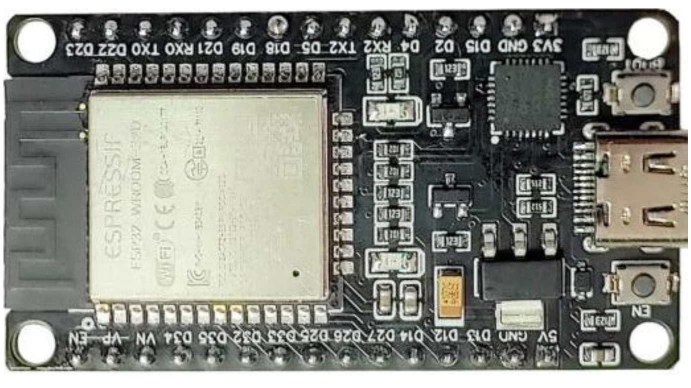


Fig 2.1 ESP32-WROOM-32D

A small, multipurpose Wi-Fi and Bluetooth module, the ESP32-WROOM-32D is made for smart applications and the Internet of Things. Its dual-core processor provides excellent performance, low power consumption, and sophisticated communication features. Numerous applications, including smart wearables, automation, and wireless sensor networks, are made possible by the module's support for 2.4 GHz Wi-Fi, Bluetooth, and BLE. The ESP32-WROOM-32D's GPIOs, PWM, ADC, and DAC enable flexible interface with sensors, actuators, and other peripherals. It is perfect for real-time, energy-sensitive Internet of Things projects because of its sturdy architecture, strong encryption capabilities, and effective power management.

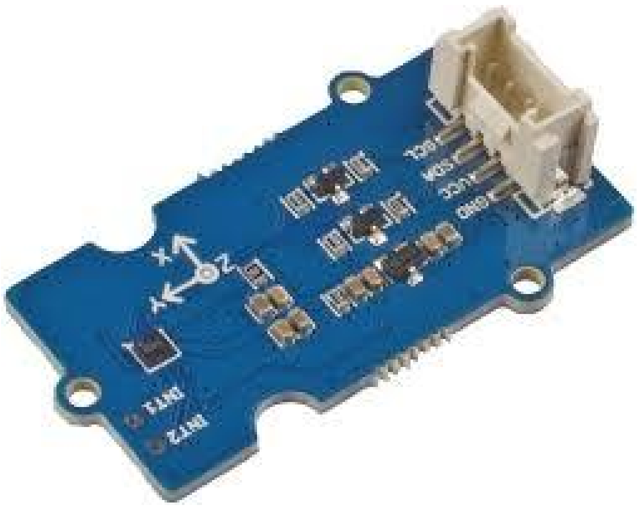


Fig 2.2 Step counter BMA456

2.3 Introduction to Arduino IDE

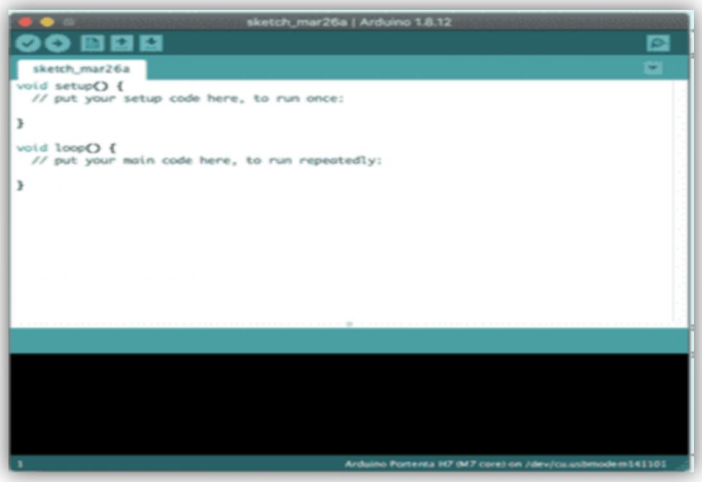


Fig 2.3 Arduino IDE interface

The Arduino IDE, which stands for Integrated Development Environment, is an open-source program created by Arduino.cc that allows users to write, compile, and upload code to the majority of Arduino modules. The availability of numerous built-in codes and their ease of compilation make it simple for someone without technical expertise to run. The Arduino IDE, which runs on the Java Platform and has built-in tools and instructions for debugging, editing, and generating code, is compatible with operating systems like Windows, Linux, MAC, and others. Both C and C++ are supported by the Arduino IDE. It includes a compiler as well as an editor.

CHAPTER 3

LITERATURE REVIEW

The study of sleepwalking and its management has evolved significantly with the advent of wearable technologies and IoT-based systems. Research has identified various factors contributing to sleepwalking, including genetic predisposition and environmental influences, with lifestyle changes and medications often serving as primary treatments [1]. Wearable devices, such as those equipped with accelerometers and other sensors, have shown potential in monitoring sleepwalking episodes and alerting caregivers or the individuals themselves to prevent injury [2]. The use of ultrasonic sensors, for example, has proven effective in non-invasively detecting movement and providing real-time data to wearable devices [3]. Furthermore, wearable shock modules integrated into sleepwalking prevention systems aim to mitigate the risks of injury by delivering mild stimuli when episodes occur [4]. IoT-based systems that offer real-time monitoring and instant alerts to caregivers have become central to the detection and management of sleepwalking, ensuring timely intervention [5]. The consequences of sleepwalking, which can include physical injuries and psychological distress, underscore the importance of these technological interventions [6]. Bluetooth technology plays a crucial role in enabling communication between devices, allowing for remote alerts and facilitating immediate responses from caregivers [7]. However, the ethical implications of using shock modules in sleepwalking treatment have raised concerns regarding their safety and long-term effects, prompting further exploration of non-invasive alternatives [8]. Additionally, the integration of alarm systems and buzzers within wearable devices provides a reliable method for alerting individuals and caregivers when a sleepwalking event is detected [9]. Together, these advancements demonstrate the growing role of technology in enhancing the safety and management of sleepwalking, improving the quality of life for those affected.

CHAPTER 4

DESIGN

4.1 System Architecture

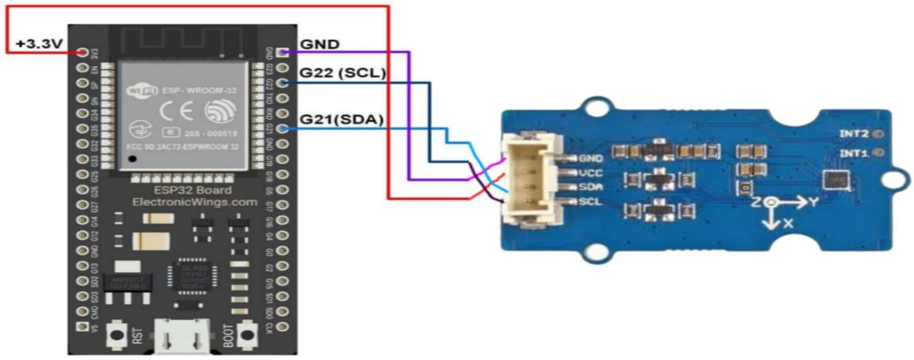


Fig 4.1 ESP32 connected to step counter

The Sleepwalking Detection and Prevention System's hardware consists of an ESP32, a push button, a wired buzzer, a shock module, a pedometer or distance sensor, and a rechargeable battery. The ESP32 microcontroller manages Bluetooth connectivity, sensor data, and shock module control. The shock module gives a gentle shock when the threshold is crossed, and the distance sensor tracks the sleepwalker's movements. The wired buzzer notifies the guardian when the sleepwalker travels past the designated distance, and the push button enables physical intervention to stop the shock. Firmware for the ESP32, which manages sensor data processing, Bluetooth communication, and shock module control, is one of the software components. A straightforward browser-based program or a mobile application for monitoring that provides real-time status updates can serve as the user interface software.

4.2 Behavioural model

The first step after hardware setup is to use a laptop or PC and a USB cable to upload the code to the ESP32. In order to detect movement, the ESP32 then interacts with the wearable device's shock module and pedometer. The ESP32 activates the shock module to provide a moderate shock, shown by an LED, when the sleepwalker crosses a predetermined threshold (for example, five meters). The shock ceases and the device resets for a predetermined amount of time if the sleepwalker presses the push button. Additionally, the ESP32 uses

Bluetooth to connect to the guardian's system, setting off a wired buzzer to notify the guardian.

4.3 Requirement Specification

Hardware Requirements

- ESP32 Microcontroller
- Pedometer or Step Counter
- Shock Module
- Push Button
- Wired Buzzer
- Power Supply
- Jumper Wires

Software Requirements

- Arduino IDE
- Libraries
- ESP32 Firmware

CHAPTER 5

METHODOLOGY

5.1 Circuit setup

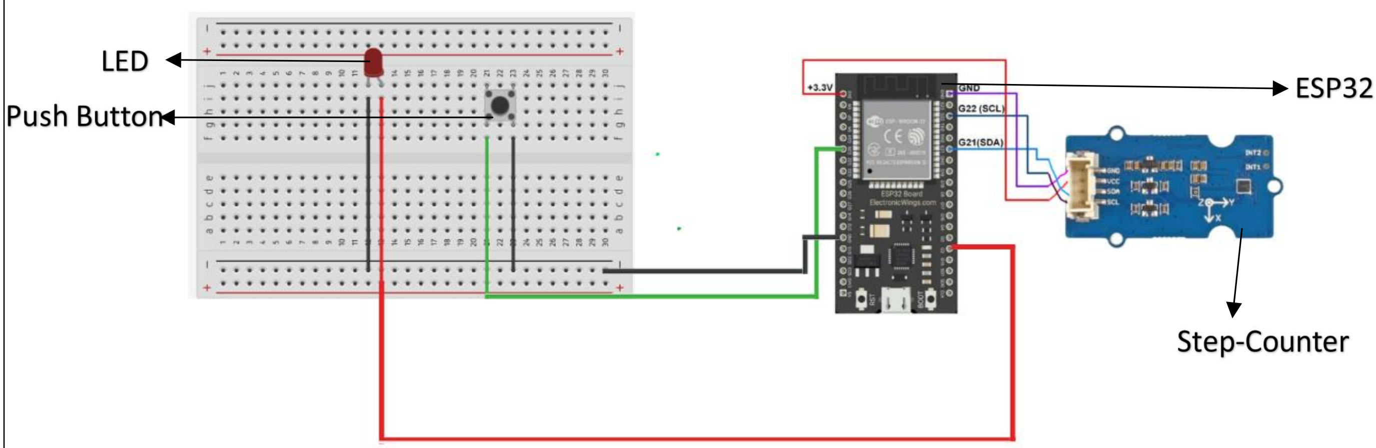


Fig 5.1 Sleepwalker's device

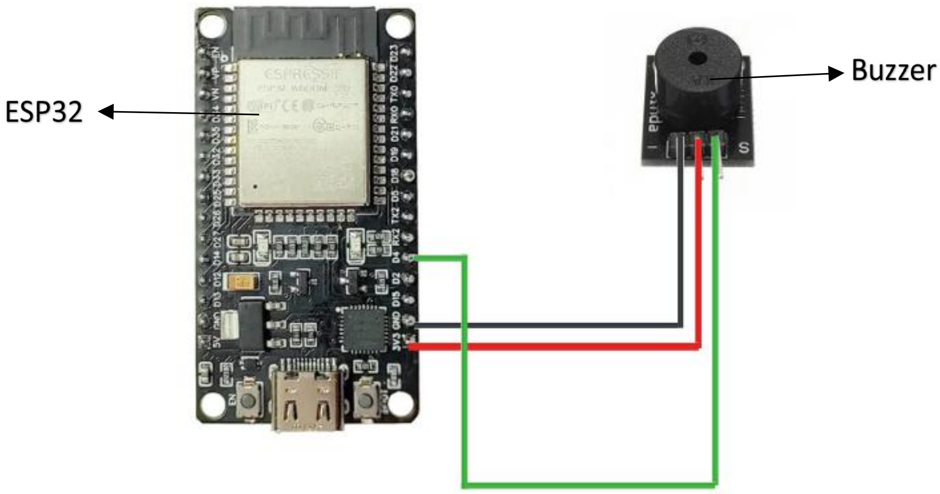


Fig 5.2 Guardian's device

5.2 Code

Code for Walker’s module:

```
#include "arduino_bma456.h"

// Define pins

#define LED_PIN 2

#define BUTTON_PIN 15

uint32_t step = 0; // variable to store step count

uint32_t stepoffset = 0; // Offset to simulate step reset

bool ledon = false; // Track LED state

bool buttonPressed = false;

void setup() {

  Serial.begin(115200);

  Serial.println("BMA456 Step Counter");

  // Initialize BMA456 sensor

  bma456.initialize(RANGE_4G, ODR_100_HZ, NORMAL_AVG4, CONTINUOUS);

  bma456.stepCounterEnable();

  // Initialize pins

  pinMode(LED_PIN, OUTPUT);

  pinMode(BUTTON_PIN, INPUT_PULLUP); // Use internal pull-up resistor

  // Ensure LED is off at the start

  digitalWrite(LED_PIN, LOW);

}

// Custom function to reset step count by setting an offset

void resetStepCount() {
```

```
    stepoffset = bma456.getStepCounterOutput(); // Store the current step count as offset
}
void loop() {
    // Get the current step count from the sensor
    step = bma456.getStepCounterOutput() - stepoffset; // Subtract offset to reset the count
    Serial.print("Step: ");
    Serial.println(step);
    // Check step count and LED state to control LED
    if (step > 20 && !ledon) {
        digitalWrite(LED_PIN, HIGH);
        ledon = true;
        Serial.println("LED turned on due to step count exceeding 20.");
        resetstepCount(); // Reset step count
    }
    // Check button state to turn off LED
    if (digitalRead(BUTTON_PIN) == LOW) { // Button pressed (assuming active LOW)
        digitalWrite(LED_PIN, LOW); // Turn off LED
        ledon = false;
        Serial.println("Button pressed! LED turned off.");
        delay(500); // Debounce delay
    }
    delay(500); // Delay to reduce print frequency
}
```

Code for Guardian's module:

```
#include <esp_now.h>

#include <WiFi.h>

// Define pins

#define BUZZER_PIN 13 // Buzzer connected to GPIO 13

// Structure to receive data

typedef struct {

  bool ledon;

} struct_message;

struct_message receivedData;

void setup() {

  Serial.begin(115200);

  // Initialize pins

  pinMode(BUZZER_PIN, OUTPUT);

  digitalWrite(BUZZER_PIN, LOW); // Ensure buzzer is off at start

  // Set device as a Wi-Fi Station

  WiFi.mode(WIFI_STA);

  // Initialize ESP-NOW

  if (esp_now_init() != ESP_OK) {

    Serial.println("Error initializing ESP-NOW");

    return;

  }

  // Register receive callback

  esp_now_register_recv_cb(onDataRecv);
```

```
}
void onDataRecv(const uint8_t *mac, const uint8_t *incomingData, int len) {
  memcpy(&receivedData, incomingData, sizeof(receivedData));
  // Turn on the buzzer if LED is on, otherwise turn it off
  if (receivedData.ledon) {
    digitalWrite(BUZZER_PIN, HIGH); // Turn on buzzer
    Serial.println("Buzzer turned on due to LED on remote ESP32.");
  } else {
    digitalWrite(BUZZER_PIN, LOW); // Turn off buzzer
    Serial.println("Buzzer turned off due to LED off on remote ESP32.");
  }
}
void loop() {
  // Nothing needed here, as the onDataRecv callback handles the buzzer control
}
```

5.3 Steps

Step 1:

Set up the wearable gadget by integrating the BMA456 pedometer sensor for movement tracking and configuring the ESP32 microcontroller.

Step 2:

Set motion thresholds to track sleepwalking and calibrate the BMA456 sensor to precisely detect step counts.

Step 3:

To enable communication between the guardian's ESP32 module and the wearable ESP32 device, set up Wi-Fi in station mode for the ESP32 and initialise the ESP-NOW protocol.

Step 4:

Use the BMA456 pedometer to track movement. The step count is continuously monitored, and an LED attached to GPIO 2 is activated to signal possible sleepwalking when steps surpass a predetermined threshold.

Step 5:

Send the guardian's ESP32 module the current LED status. The wearable gadget immediately alerts the guardian when it detects movement by sending the LED status via ESP-NOW.

Step 6:

Turn on the guardian's warning system. The ESP32 module in the guardian's room activates a buzzer attached to GPIO 13 to notify the guardian that the LED is on after receiving the data.

Step 7:

Manage the alarm. By pressing a button linked to GPIO 15, the wearable's LED can be turned off. This also signals the guardian's ESP32 module to turn off its buzzer, enabling the guardian to reset the alarm.

Step 8:

Make sure the guardian's module is informed of any changes to the wearable's LED and buzzer status by sending ESP-NOW LED status updates whenever the wearable's button is pressed.

Step 9: Use the wearable ESP32 for real-time monitoring, where sensor data is continuously processed to evaluate movement patterns associated with sleepwalking.

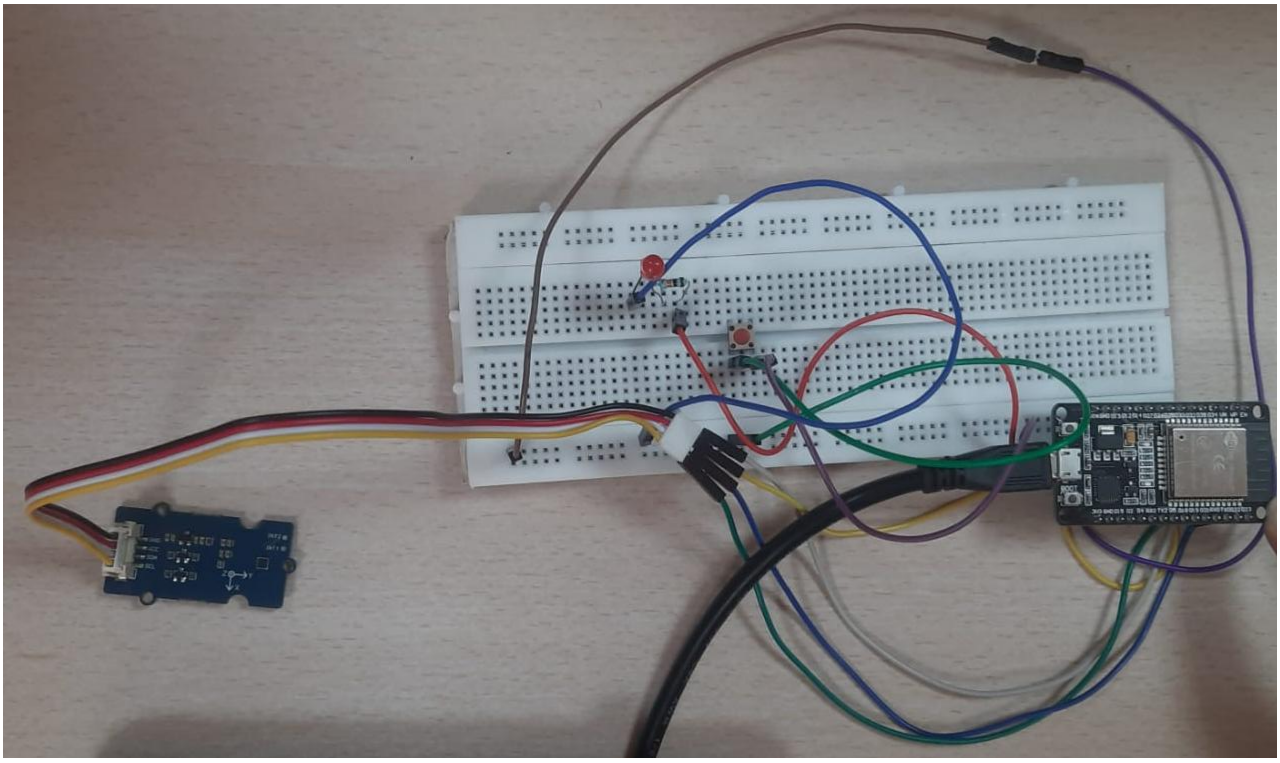
Step 10:

Make use of inconspicuous alarms. To prevent over-intervention, the guardian's ESP32 buzzer acts as the main alert system.

Step 11:

Permit manual control. To reduce the sleepwalker's tension, the alarm system can be reset or controlled by both the wearable gadget and the guardian's module.

CHAPTER 6



RESULTS

Fig 6.1 Sleepwalker's device

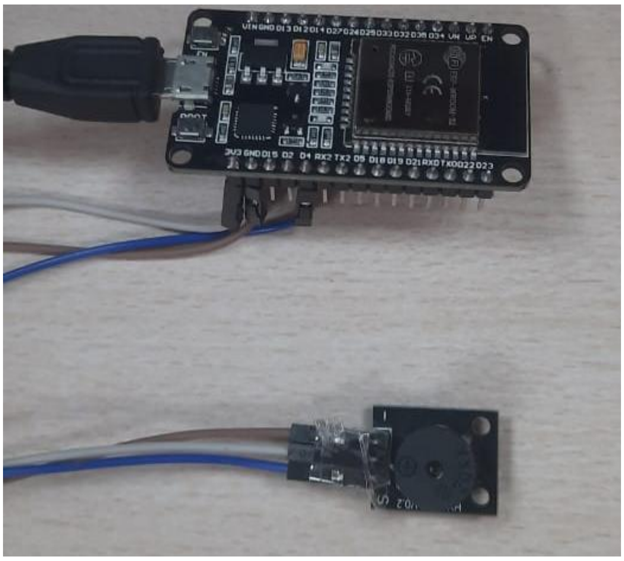


Fig 6.2 Guardian's device

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

The ESP32 microcontroller and the BMA456 pedometer sensor are used in the wearable sleepwalking detection system, which offers a practical way to keep an eye on sleepwalking behaviour. The solution facilitates smooth communication between the guardian's module and the wearable device by combining real-time movement tracking with the ESP-NOW communication protocol. Both the sleepwalker and the guardian have control over the intervention process because to the system's effective and user-friendly features, which include a manual reset feature and a non-intrusive buzzer alarm. This technique provides a dependable means of keeping an eye on sleepwalking and notifying the guardian when required, protecting the person and reducing needless intervention.

Future Enhancements:

Better Movement Detection: Incorporate sensors like gyroscopes and accelerometers to better distinguish between normal and sleepwalking by identifying movement patterns precisely.

Mobile App Integration: Create a mobile application that will enhance responsiveness and convenience by giving guardians remote control over the alarm system and real-time notifications.

Advanced connection Protocols: For longer-range connection between the wearable and guardian's module, consider alternate wireless protocols like BLE or LoRa.

Data Logging and Analysis: Put in place a system that records data over time in order to examine trends of sleepwalking behaviour and offer insights into potential triggers, enabling more individualised monitoring.

Automated Sleepwalking Detection: This technique may eliminate the need for thresholds by using machine learning to forecast sleepwalking episodes based on past movement data.

Better Power Management: Use more sophisticated power-saving techniques to prolong the wearable's battery life while keeping an eye on things constantly.

CHAPTER 8

REFERENCES

[1] Mume, Celestine. (2010). Prevalence of Sleepwalking in an Adult Population. The Libyan journal of medicine. 5. 10.4176/090907.

[2] Damkliang, K., Andritsch, J. ., Khamkom, K. . and Thongthep, N. . (2019) “A System for Sleepwalking Accident Prevention Utilizing the Remote Sensor of Wearable Device”, *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, 13(2), pp. 160–169. doi: 10.37936/ecti-cit.2019132.184442.

[3] Smith, John Damien; Schroeder, Cynthia Ann. Assessing Pedometer Accuracy while Walking, Skipping, Galloping, Sliding, and Hopping. Journal of Strength and Conditioning Research 22(1):p 276-282, January 2008. | DOI: 10.1519/JSC.0b013e31815f2f42

[4] Alberto, M.C., 2017. *Non-invasive wearable sensing system for sleep disorder monitoring* (Master's thesis, Universidade do Algarve (Portugal)).

[5] Muhammad Shoaib Akhtar Tao Feng Year: 2022 Detection of Sleep Paralysis by using IoT Based Device and Its Relationship Between Sleep Paralysis And Sleep Quality 10.4108/eetiot.v8i30.2688 DOI: 10.4108/eetiot.v8i30.2688

[6] Perogamvros L, Aberg K, Gex-Fabry M, Perrig S, Cloninger CR, Schwartz S. Increased Reward-Related Behaviors during Sleep and Wakefulness in Sleepwalking and Idiopathic Nightmares. PLoS One. 2015 Aug 19;10(8):e0134504. doi: 10.1371/journal.pone.0134504. PMID: 26287974; PMCID: PMC4546110.Zhang, Jinxiao. (2023). The application of bluetooth technology in the internet of things. Applied and Computational Engineering. 12. 177-183. 10.54254/2755-2721/12/20230334.

[7] Zhang,J. (2023).The application of bluetooth technology in the internet of things.Applied and Computational Engineering,12,177-183.

[8] Ruby Dhar, Kumar, A. and Subhradip Karmakar (2023) “Smart wearable devices for real-time health monitoring”, *Asian Journal of Medical Sciences*, 14(12), pp. 1–3. doi: 10.3126/ajms.v14i12.58664.

[9] November 2023 BMC Medical Informatics and Decision Making 23(1) DOI:10.1186/s12911-023-02350-w License CC BY 4.0.