IOT-INFUSED STRESS ANALYSIS AND RECOMMENDATION SYSTEM FOR HUMANS THROUGH SENSOR DATA MONITORING

A PROJECT REPORT

Submitted by

DEVI PRIYA Y - 410620243004

RAJA ABI C - 410620243025

in partial fulfillment for the award of the degree

of

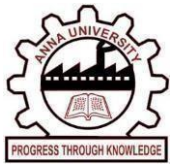
BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

DHAANISH AHMED COLLEGE OF ENGINEERING

PADAPPAI, CHENNAI - 601 301.



ANNA UNIVERSITY : CHENNAI 600 025

MAY 2024

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"IOT-INFUSED STRESS ANALYSIS AND RECOMMENDATION SYSTEM FOR HUMANS THROUGH SENSORDATA MONITORING"** is the bonafide work of **DEVI PRIYA Y (410620243004), RAJA ABI C (410620243023)** who carried out the project work under my supervision.

SIGNATURE
Mrs.R.SIVAKANI,M.E.,(Ph.D),
HEAD OF THE DEPARTMENT

Artificial Intelligence and Data Science,
Dhaanish Ahmed College of Engineering,
Padappai, Chennai- 601 301

SIGNATURE
Mrs.R.SIVAKANI,M.E.,(Ph.D),
ASSISTANT PROFESSOR
SUPERVISOR
Artificial Intelligence and Data Science,
Dhaanish Ahmed College of Engineering,
Padappai, Chennai- 601 301

Submitted for the Anna University Project Viva Voce held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We thank our almighty god and our beloved parents for their consistent support and encouragement provided from the beginning of our project work and till its completion.

We express our heart-felt gratitude to our Chairman **ALHAJ K. MOOSA,** and to our Secretary **Mr. M. KADARSHAH, B.A., M.B.A.,** for providing necessary facilities for the successful completion of our project.

We take the privilege to express our indebted sense of gratitude to our respected director sir **Dr. P. PARAMASIVAN, M.Sc., M.Phil., Ph.D.,** and our beloved Principal, **Dr. G. UMA GOWRI, M. Tech., Ph.D.,** Dhaanish Ahmed College of Engineering for granting permission to undertake the project in our college.

We would like to express our gratitude to the Dean **Prof. S. SUMAN RAJEST, Ph.D.,** (R & D and International Student Affairs) for giving the valuable suggestion to complete the project.

We express our thanks to our Academic Coordinator **Mr. G. RAJASEKARAN, M.E.,** for his valuable suggestion to complete this project.

We would express our sincere thanks to our Head of the Department and Project Coordinator **Mrs. R. SIVAKANI, M.E., (Ph.D.,)** for her kind permission to carry out the project and encouragement given to complete the project.

We would like to extend our thanks to our Project Supervisor **Mrs. K. SIVAKANI, M.E., (Ph.D.,)** Assistant Professor, for her excellent guidance to complete the project.

We are extremely thankful to our Faculty Members for their valuable support throughout this project. We also thank our Family and Friends for their moral support.

ABSTRACT

In response to the escalating stress levels witnessed among workers and students during the COVID-19 pandemic, this research undertakes a comprehensive investigation into the physiological responses to stress, particularly within the context of remote work and virtual learning environments. Recognizing the critical importance of understanding these responses, the study meticulously examines various parameters influencing stress levels among remote workers and students alike. Through a multidisciplinary approach, incorporating insights from psychology, physiology, and technology, the research identifies key factors contributing to heightened stress in these populations. The proposed system holds significant implications for occupational health, offering remote workers access to tools and resources designed to create healthier work environments. Likewise, within the realm of educational technology, the system stands to enhance student well-being in virtual learning settings, providing vital support amid the challenges of remote education. Moreover, the application of such technology extends beyond these domains, finding relevance in telemedicine and contributing to the development of technology-driven solutions for stress management and overall health. By leveraging advancements in both psychological understanding and technological innovation, we endeavor to empower individuals to better cope with the stresses of remote work and virtual learning, ultimately fostering resilience and well-being in the face of unprecedented challenges.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ECG - ELECTRO CARDIO GRAM
GPS - GLOBAL POSITIONING SYSTEM
HRV - HEART RATE VARIABILITY
HVAC - HEATING, VENTILATION, AIR CONDITION
LDO - LOW DROP OUT
MCU - MICRO CONTROLLER UNIT
MRI - MAGNETIC RESONANCE IMAGING
PWM - PULSE WIDTH MODULATION
RISC - REDUCED INSTRUCTION SET COMPUTING
RTOS - REAL TIME OPERATING SYSTEM
USB - UNIVERSAL SERIAL BUS
VCR - VOLTAGE CONTROLLER RESISTER
VIN - VOLTAGE INPUT PIN

CHAPTER 1

INTRODUCTION

The COVID-19 pandemic has brought about unprecedented challenges for individuals worldwide, with workers and students particularly susceptible to elevated stress levels due to the abrupt shift to remote work and virtual learning. Recognizing the profound impact of this global crisis on mental health, this research embarks on a critical exploration of stress management strategies tailored to address the unique needs of these populations. Amid the upheaval caused by the pandemic, understanding the physiological responses to stress becomes paramount. Remote workers and students alike face novel stressors, ranging from increased isolation and blurred work-life boundaries to technological challenges and academic pressures. To effectively address these challenges, it is imperative to delve deeply into the mechanisms underlying stress and identify actionable interventions that promote well-being in the face of adversity. This research aims to bridge the gap between psychological insights and technological innovation, offering a holistic approach to stress management for remote workers and students. By leveraging advancements in both psychology and technology, the study seeks to develop a sophisticated system capable of providing personalized recommendations and guidance tailored to individual needs and preferences. The implications of this research extend far beyond the realms of occupational health and educational technology. By contributing to the development of technology-driven solutions for stress management, the study holds promise for broader applications in telemedicine and the promotion of overall health and well-being.

1.1 EMBEDDED SYSTEMS

An embedded system is a controller programmed and controlled by a real-time operating system (RTOS) with a dedicated function within a larger mechanical or electrical system, often with real-time consumption of embedded systems computing constraints.

It is embedded as part of a complete device often including hardware and mechanical parts. Embedded systems control many devices in common use today. Ninety-eight percent of all microprocessors are manufactured to serve as embedded system component.

Examples of properties of typical embedded computers when compared with general-purpose counterparts are low power consumption, small size, rugged operating ranges, and low per-unit cost. This comes at the price of limited processing resources, which make them significantly more difficult to program and to interact with.

However, by building intelligence mechanisms on top of the hardware, taking advantage of possible existing sensors and the existence of a network of embedded units, one can both optimally manage available resources at the unit and network levels as well as provide augmented functions, well beyond those available. For example, intelligent techniques can be designed to manage power.

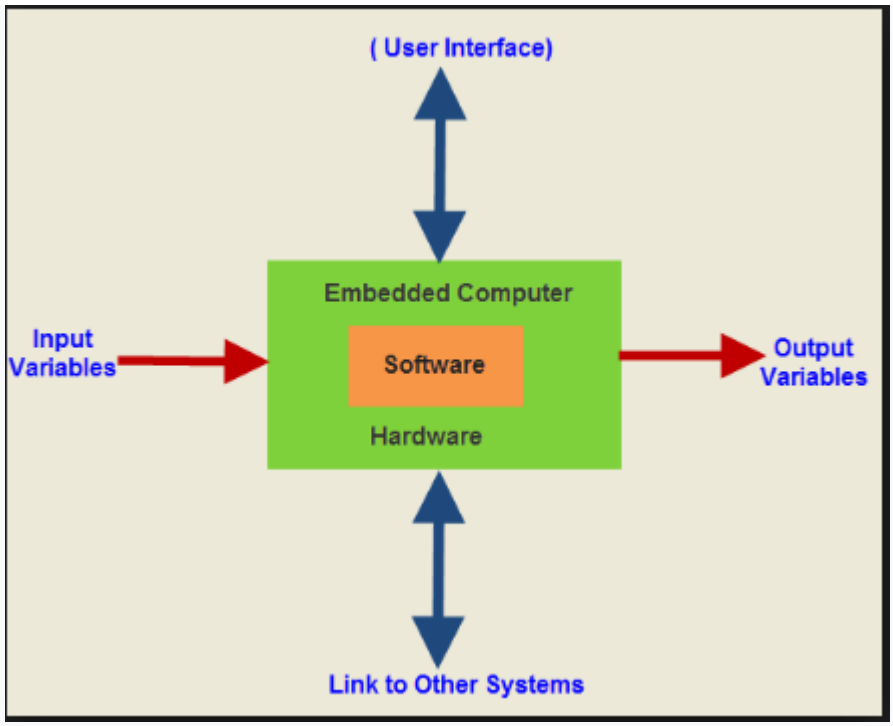


Fig. 1.1.1 Embedded Computer Sub-Assembly for Electronic Voting Machine

Embedded systems are commonly found in consumer, industrial, automotive, medical, commercial and military applications.

Telecommunications systems employ numerous embedded systems from telephone switches for the network to cell phones at the end user. Computer networking uses dedicated routers and network bridges to route data.

Consumer electronics include MP3 players, mobile phones, video game consoles, digital cameras, GPS receivers, and printers. Household appliances, such as microwave ovens, washing machines and dishwashers, include embedded systems to provide flexibility, efficiency and features.

CLASSIFICATIONS OF EMBDDED SYSTEMS

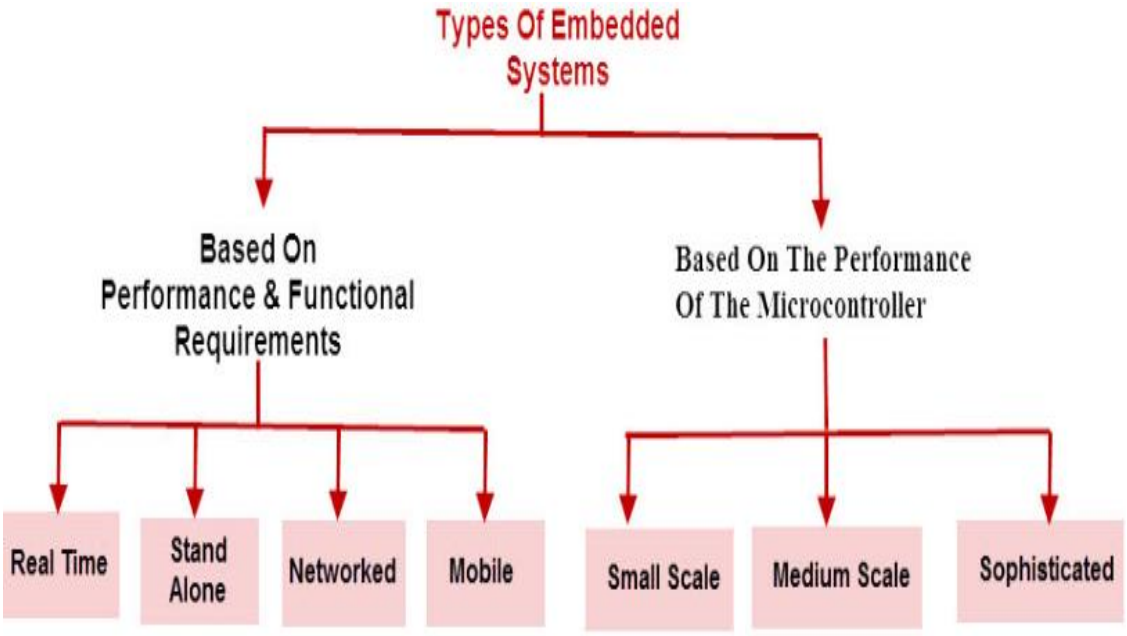


Fig. 1.1.2 Types of Embedded Systems

Advanced HVAC systems use networked thermostats to more accurately and efficiently control temperature that can change by time of day and season. Home automation uses wired- and wireless-networking that can be used to control lights, climate, security, audio/visual, surveillance, etc., all of which use embedded devices for sensing and controlling.

Embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations. Complexity varies from low, with a single microcontroller.

Block diagram of an embedded system

An embedded system usually contains an embedded processor. Many appliances that have a digital interface microwaves, VCRs, cars utilize embedded systems. Some embedded systems include an operating system. Others are very specialized resulting in the entire logic being implemented as a single program. These systems are embedded into some device for some specific purpose other than to provide general purpose computing.

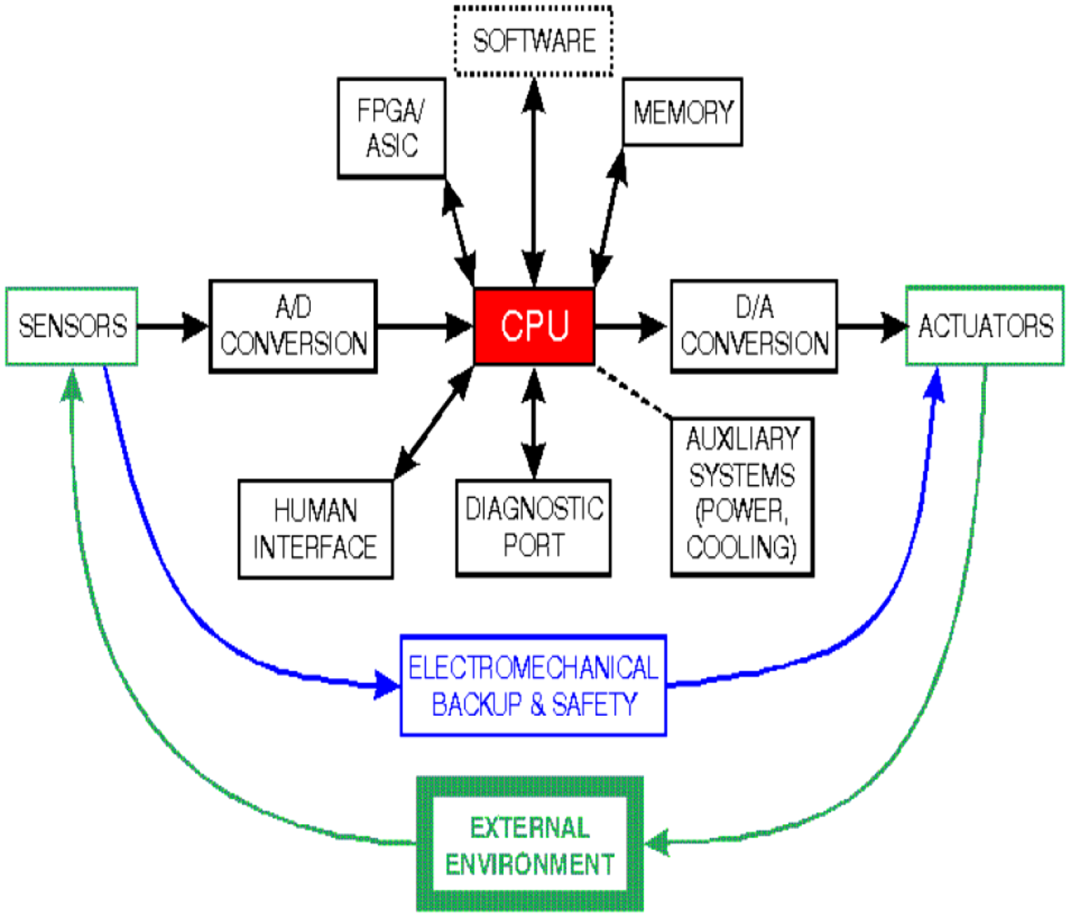


Fig .1.1.3 Block diagram of a typical embedded system

1.1.1 EMBEDDED SYSTEMS APPLICATIONS

Embedded systems in automobiles include motor control, cruise control, body safety, engine safety, robotics in an assembly line, car multimedia, car entertainment, E-com access, mobiles etc.

- Embedded systems in telecommunications include networking, mobile computing, and wireless communications, etc.
- Embedded systems in smart cards include banking, telephone and security systems.
- Embedded Systems in satellites and missiles include defense, communication, and aerospace
- Embedded systems in computer networking & peripherals include image processing, networking systems, printers, network cards, monitors and displays
- Embedded Systems in digital consumer electronics include set-top boxes, DVDs, high-definition TVs and digital cameras

1.2 INTERNET OF THINGS

The term Internet of Things generally refers to scenarios where network connectivity and computing capability extends to objects, sensors and everyday items not normally considered computers, allowing these devices to generate, exchange and consume data with minimal human intervention. There is, however, no single, universal definition.

Enabling Technologies: The concept of combining computers, sensors, and networks to monitor and control devices has existed for decades. The recent confluence of several technology market trends, however, is bringing the Internet of Things closer to widespread reality.

These include Ubiquitous Connectivity, Widespread Adoption of IP-based Networking, Computing Economics, Miniaturization, Advances in Data.

Connectivity Models: IoT implementations use different technical communications models, each with its own characteristics. Four common communications models described by the Internet Architecture Board include: Device-to-Device, Device-to-Cloud, Device-to-Gateway, and Back-End Data-Sharing. These models highlight the flexibility in the ways that IoT devices can connect and provide value to the user.

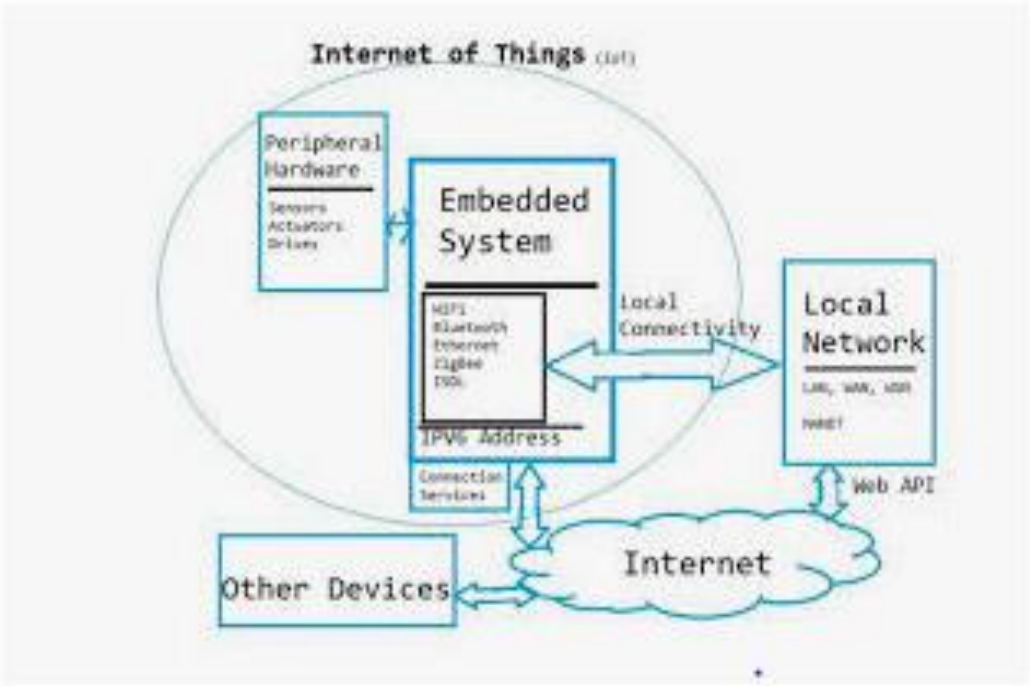


Fig. 1.2.1 IOT Block Diagram

IoT devices are implemented using both hardware and software components. Dedicated hardware components are used to implement the interface with the physical world, and to perform tasks which are more computationally complex. Microcontrollers are used to execute software that interprets inputs and controls the system. This module discusses the roles of both the hardware and software components in the system.

The functions of common hardware components are described and the interface between the software and hardware through the microcontroller is explained.

IoT devices often use an operating system to support the interaction between the software and the microcontroller. We will define the role of an operating system in an IoT device and how an IoT operating system differs from a standard one.

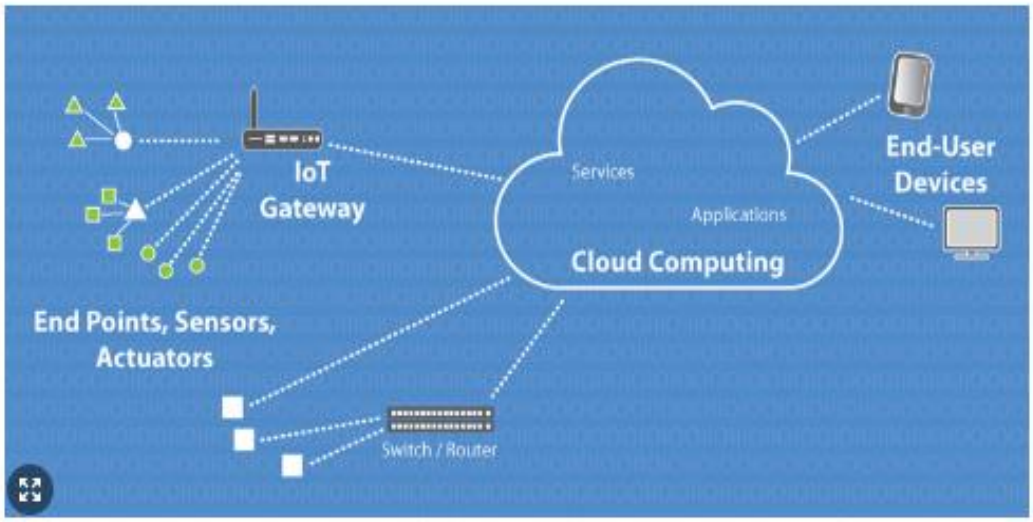


Fig.1.2.2 Cloud Computing

How IoT works

An IoT ecosystem consists of web-enabled smart devices that use embedded processors, sensors and communication hardware to collect, send and act on data they acquire from their environments. IoT devices share the sensor data they collect by connecting to an IoT gateway or other edge device where data is either sent to the cloud to be analyzed or analyzed locally. Sometimes, these devices communicate with other related devices and act on the information they get from one another.

The devices do most of the work without human intervention, although people can interact with the devices for instance, to set them up, give them instructions or access the data.

The connectivity, networking and communication protocols used with these web-enabled devices largely depend on the specific IoT applications deployed.

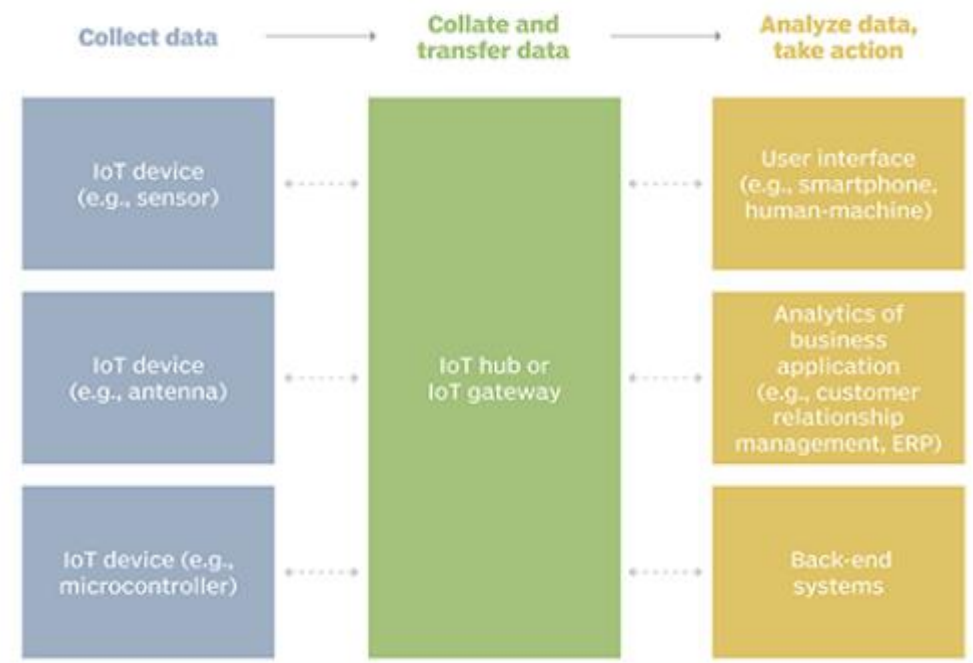


Fig .1.2.3 Example of an IOT system

Benefits of IoT

The internet of things offers a number of benefits to organizations, enabling them to:

- improve the customer experience
- enhance employee productivity
- integrate and adapt business models
- make better business decision.

IoT encourages companies to rethink the ways they approach their businesses, industries and markets and gives them the tools to improve their business strategies.

Consumer and enterprise IoT applications

There are numerous real-world applications of the internet of things, ranging from consumer IoT and enterprise IoT to manufacturing and industrial IoT (IoT). IoT applications span numerous verticals, including automotive, telco, energy and more.

In the consumer segment, for example, smart homes that are equipped with smart thermostats, smart appliances and connected heating, lighting and electronic devices can be controlled remotely via computers, smartphones or other mobile devices.

Wearable devices with sensors and software can collect and analyze user data, sending messages to other technologies about the users with the aim of making users' lives easier and more comfortable. Wearable devices are also used for public safety for example, improving first responders' response times during emergencies by providing optimized routes to a location or by tracking construction workers' or firefighters vital signs at life-threatening sites.

Consumer enterprises develop IoT-enabled healthcare devices and telemedicine platforms to monitor patients remotely, collect health data, and facilitate virtual consultations with healthcare providers. IoT solutions improve access to healthcare services, enable early detection of health issues, and support chronic disease management.

CHAPTER 2

LITERATURE SURVEY

1. Stress Analysis and Care Prediction System for Online Workers

AUTHOR: A. A. S. M Amarasinghe; I. M. S. Malassri; K. C. N.

YEAR : 2021

DESCRIPTION

Working from home (WFH) online during the covid-19 pandemic has caused increased stress level. Online workers/students have been affecting by the crisis according to new researches. Natural response of body, to external and internal stimuli is stress. Even though stress is a natural occurrence, prolonged exposure while working Online to stressors can lead to serious health problems if any action will not be applied to control it. Our research has been conducted deeply to identify the best parameters, which have connection with stress level of online workers. As a result of our research, a desktop application has been created to identify the users stress level in real time. According to the results, our overall system was able to provide outputs with more than 70% accuracy. It will give best predictions to avoid the health problems. Our main goal is to provide best solution for the online workers to have healthy lifestyles. Updates for the users will be provided according to the feedback we will have in the future from the users. Our System will be a most valuable application in the future among online workers.

2. Influence of Online Meditational Aid on Emotional Health and Job Stress during Pandemic

AUTHOR : Sandhya Java; Badriya Abdul Jaffar; Aradhana Balodi Bhardwaj; Smitha Prabhakar

YEAR : 2021

DESCRIPTION

Experts and youths for hundreds of years have practiced the art of mindfulness meditation. A reliable contemplation practice can help quiet your mind and do substantially more for your general wellbeing. Emotional intelligence or EQ is one's ability to understand and manage emotions appropriately. Research indicates that EQ is a superior indicator of welfare & progress than IQ. Job stress occurs when the prerequisites of the employees do not coordinate with the abilities, assets, or necessities of the individuals. Research suggests that "thought decrease" or "mental quietness" may have explicit impacts applicable to work pressure and emotional welfare. Much of the population ventured into online mediums to continue their meditation practice due to its ease of accessibility and self-benefits specially during the pandemic. The two main aims of this paper were, 1) To study the difference between the experimental and control group on the dimensions of Emotional Intelligence. 2) To study the difference between the experimental and control group on the dimensions of Work Stress. An empirical- analytical approach was applied through an extensive review of existing literature and two predetermined questionnaires to study the Emotional Intelligence and Work stress level of individuals engaging in online meditation. For this study, an experimental research design was implemented for collection of data. Emotional health and stress are linked in their impact on an individual's well-being.

3. Online Non-Collocated Estimation of Payload and Articular Stress for Real-Time Human Ergonomic Assessment

AUTHOR: Yeshasvi Tirupachuri; Prashanth Ramadoss; Lorenzo Rapetti; Claudia Latella; Kourosh Darvish

YEAR : 2021

DESCRIPTION

Improving the quality of work for human beings is receiving a lot of attention from multiple research communities. In particular, digital transformation in human factors and ergonomics is going to empower the next generation of the socio-technical workforce. The use of wearable sensors, collaborative robots, and exoskeletons, coupled with novel technologies for the real-time assessment of human ergonomic forms the crux of this digital transformation. In this direction, this paper focuses on the open problem of estimating the interaction wrench experienced at the human extremities (such as hands), where the feasibility of direct sensor measurements is not practical. We refer to our approach as non-collocated wrench estimation, as we aim to estimate the wrench at known contact locations but without using any direct force-torque sensor measurements at these known locations. We achieve this by extending the formulation of stochastic inverse dynamics for humans by considering a centroidal dynamics constraint to perform a reliable non-collocated estimation of interaction wrench and the joint torques (articular stress) experienced as a direct consequence of the interaction. Our approach of non-collocated estimation is thoroughly validated in terms of payload estimation and articular stress estimation through validation and experimental scenarios involving dynamic human motions like walking.

4. Heart Rate Variability Measurement to Assess Acute Work-Content-Related Stress of Workers in Industrial Manufacturing Environment-A Systematic Scoping Review

AUTHOR : Tuan-Anh Tran; Márta Péntek; Hossein Motahari-Nezhad

YEAR : 2023

DESCRIPTION

Human workers are indispensable in the human–cyber-physical system in the forthcoming Industry 5.0. As inappropriate work content induces stress and harmful effects on human performance, engineering applications search for a physiological indicator for monitoring the well-being state of workers during work; thus, the work content can be modified accordingly. The primary aim of this study is to assess whether heart rate variability (HRV) can be a valid and reliable indicator of acute work-content-related stress (AWCRS) in real time during industrial work. Second, we aim to provide a broader scope of HRV usage as a stress indicator in this context. Methods: A search was conducted in Scopus, IEEE Xplore, PubMed, and Web of Science between 1 January 2000 and 1 June 2022. Eligible articles are analyzed regarding study design, population, assessment of AWCRS, and its association with HRV. Results: A total of 14 studies met the inclusion criteria. No randomized control trial (RCT) was conducted to assess the association between AWCRS and HRV. Five observational studies were performed. Both AWCRS and HRV were measured in nine further studies, but their associations were not analyzed. Results suggest that HRV does not fully reflect the AWCRS during work, and it is problematic to measure the effect of AWCRS on HRV in the real manufacturing environment.

5. A Survey of AI-Based Facial Emotion Recognition: Features, ML & DL Techniques, Age-Wise Datasets and Future Directions

AUTHOR : Chirag Dalvi; Manish Rathod; Shruti Patil; Shilpa Gite; Ketan Kotecha

YEAR : 2021

DESCRIPTION

Facial expressions are mirrors of human thoughts and feelings. It provides a wealth of social cues to the viewer, including the focus of attention, intention, motivation, and emotion. It is regarded as a potent tool of silent communication. Analysis of these expressions gives a significantly more profound insight into human behavior. AI-based Facial Expression Recognition (FER) has become one of the crucial research topics in recent years, with applications in dynamic analysis, pattern recognition, interpersonal interaction, mental health monitoring, and many more. However, with the global push towards online platforms due to the Covid-19 pandemic, there has been a pressing need to innovate and offer a new FER analysis framework with the increasing visual data generated by videos and photographs. Furthermore, the emotion-wise facial expressions of kids, adults, and senior citizens vary, which must also be considered in the FER research. Lots of research work has been done in this area. However, it lacks a comprehensive overview of the literature that showcases the past work done and provides the aligned future directions. In this paper, the authors have provided a comprehensive evaluation of AI-based FER methodologies, including datasets, feature extraction techniques, algorithms, and the recent breakthroughs with their applications in facial expression identification. this is the only review paper stating all aspects of FER for various age brackets and would significantly impact the research community in the coming years.

CHAPTER 3

PROJECT DISCRIPTION

3.1 EXISTING SYSTEM

The existing digital stress management system represents a significant departure from traditional methods, offering a comprehensive approach to identifying, addressing, and tracking stressors in individuals' lives. Through its user-friendly interface accessible across multiple devices, the system employs advanced sensors and data analytics to continuously monitor physiological indicators of stress, such as heart rate variability and sleep patterns. This real-time data collection allows for accurate identification of stress triggers, facilitating timely interventions to alleviate stress. Personalized recommendations for stress mitigation techniques, including guided meditation sessions, breathing exercises, and physical activities, are tailored to each user's preferences and stress profile. Additionally, users can set specific stress management goals and track their progress over time, gaining insights into their stress patterns through visualizations and reports provided by the system. Timely reminders and notifications prompt users to engage in stress management activities, while integration with external resources such as online wellness programs and mental health resources enhances overall well-being and resilience. In sum, the digital stress management system offers a proactive, personalized, and sustainable approach to stress relief, empowering individuals to take control of their mental health and well-being in today's fast-paced world.

The system also includes features that promote regular engagement with stress management activities. Timely reminders and notifications prompt users to engage in stress management activities, ensuring that they are consistently working towards their goals. Additionally, the system integrates with external resources such as online wellness programs and mental health resources, providing users with access to a wealth of information and support.

In conclusion, the digital stress management system offers a proactive, personalized, and sustainable approach to stress relief. By continuously monitoring physiological indicators of stress, providing personalized recommendations, and tracking progress towards goals, the system empowers individuals to take control of their mental health and well-being in today's fast-paced world.

3.2 PROPOSED SYSTEM

The proposed system aims to overcome limitations by offering a dedicated desktop application. Integrated with ECG sensors, control modes, and personalized interventions, it caters to effective stress management. Tailored for the distinct challenges of remote work during the pandemic, it ensures comprehensive support for individuals' well-being and productivity. The analysis of ECG data in our project is executed through a Convolutional Neural Network (CNN) algorithm. Leveraging deep learning, CNN efficiently extracts intricate patterns from ECG signals, enhancing the accuracy and effectiveness of stress assessment within our dedicated stress management system.

3.3 BLOCK DIAGRAM

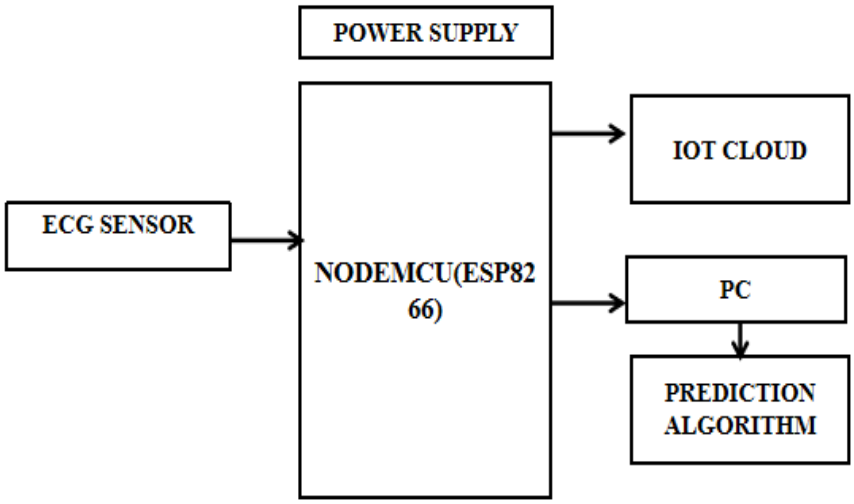


Fig. 3.3.1 Block Diagram

3.4 HARDWARE REQUIREMENT

- NODEMCU ESP8266
- ECG Sensor
- Power supply

3.5 SOFTWARE REQUIREMENT

- Vs code
- CNN Algorithm

CHAPTER 4

HARDWARE REQUIREMENT

4.1 NODEMCU ESP8266

Node MCU is an open-source Lua based firmware and development board specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Express if Systems, and hardware which is based on the ESP-12 module.

Node MCU ESP8266 Specifications & Features

- Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 16
- Analog Input Pins (ADC): 1
- UARTs: 1
- SPIs: 1
- I2Cs: 1
- Flash Memory: 4 MB
- SRAM: 64 KB
- Clock Speed: 80 MHz
- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- PCB Antenna
- Small Sized module to fit smartly inside your IoT projects

Brief About Node MCU ESP8266

The Node MCU ESP8266 development board comes with the ESP-12E module containing ESP8266 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor. This microprocessor supports RTOS and operates at 80MHz to 160 MHz adjustable clock frequency. Node MCU has 128 KB RAM and 4MB of Flash memory to store data and programs. Its high processing power with in-built Wi-Fi / Bluetooth and Deep Sleep Operating features make it ideal for IoT projects.

Node MCU can be powered using Micro USB jack and VIN pin (External Supply Pin). It supports UART, SPI, and I2C interface.

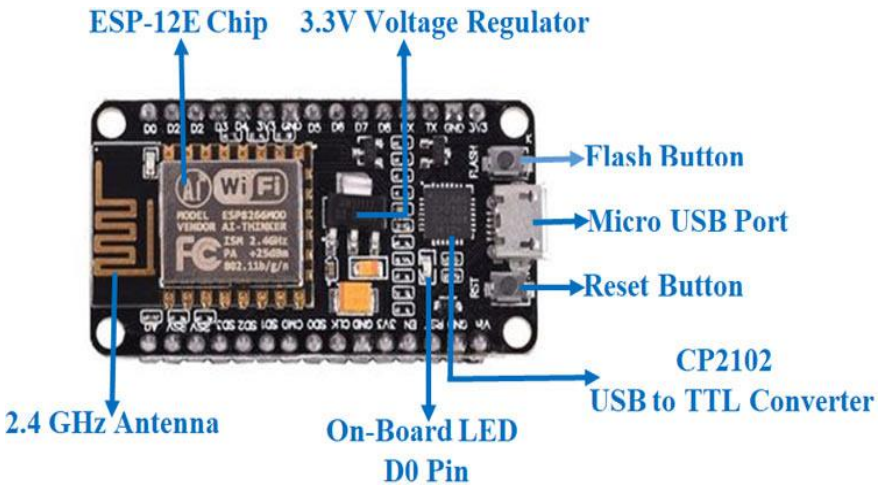


Fig. 4.1.1 Node-MCU diagram

Power Requirement

As the operating voltage range of ESP8266 is 3V to 3.6V, the board comes with a LDO voltage regulator to keep the voltage steady at 3.3V. It can reliably supply up to 600mA, which should be more than enough when ESP8266 pulls as much as 80mA during RF transmissions. The output of the regulator is also broken out to one of the sides of the board and labeled as 3V3. This pin can be used to supply power to external components.

Power to the ESP8266 Node MCU is supplied via the on-board Micro B USB connector. Alternatively, if you have a regulated 5V voltage source, the VIN pin can be used to directly supply the ESP8266 and its peripherals.

Peripherals and I/O

The ESP8266 Node MCU has total 17 GPIO pins broken out to the pin headers on both sides of the development board. These pins can be assigned to all sorts of peripheral duties, including:

- ADC channel – A 10-bit ADC channel.
- UART interface – UART interface is used to load code serially.
- PWM outputs – PWM pins for dimming LEDs or controlling motors.
- SPI, I2C & I2S interface – SPI and I2C interface to hook up all sorts of sensors and peripherals.
- I2S interface – I2S interface if you want to add sound to your project.

On-board Switches & LED Indicator

The ESP8266 Node MCU features two buttons. One marked as RST located on the top left corner is the Reset button, used of course to reset the ESP8266 chip. The other FLASH button on the bottom left corner is the download button used while upgrading firmware.

The board also has a LED indicator which is user programmable and is connected to the D0 pin of the board.

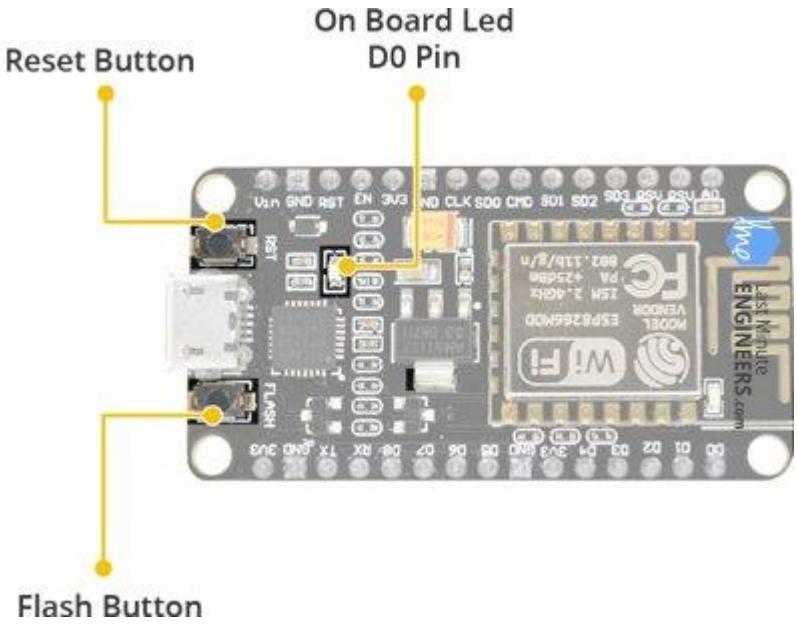


Fig. 4.1.2 Switch diagram

Serial Communication

The board includes CP2102 USB-to-UART Bridge Controller from Silicon Labs, which converts USB signal to serial and allows your computer to program and communicate with the ESP8266 chip.

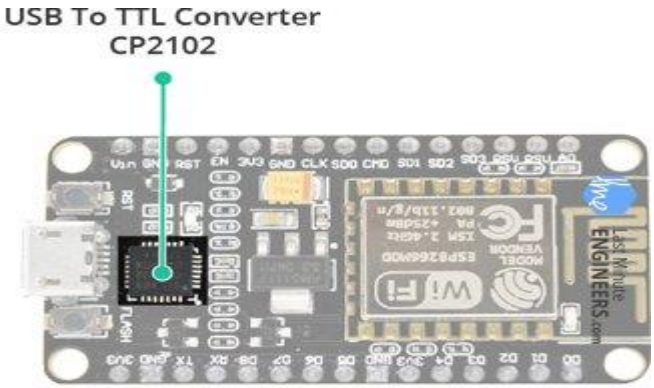


Fig. 4.1.3 Serial communication diagram

If you have an older version of CP2102 driver installed on your PC, we recommend upgrading now.

ESP8266 Node MCU Pinout

The ESP8266 Node MCU has total 30 pins that interface it to the outside world. The connections are as follows:

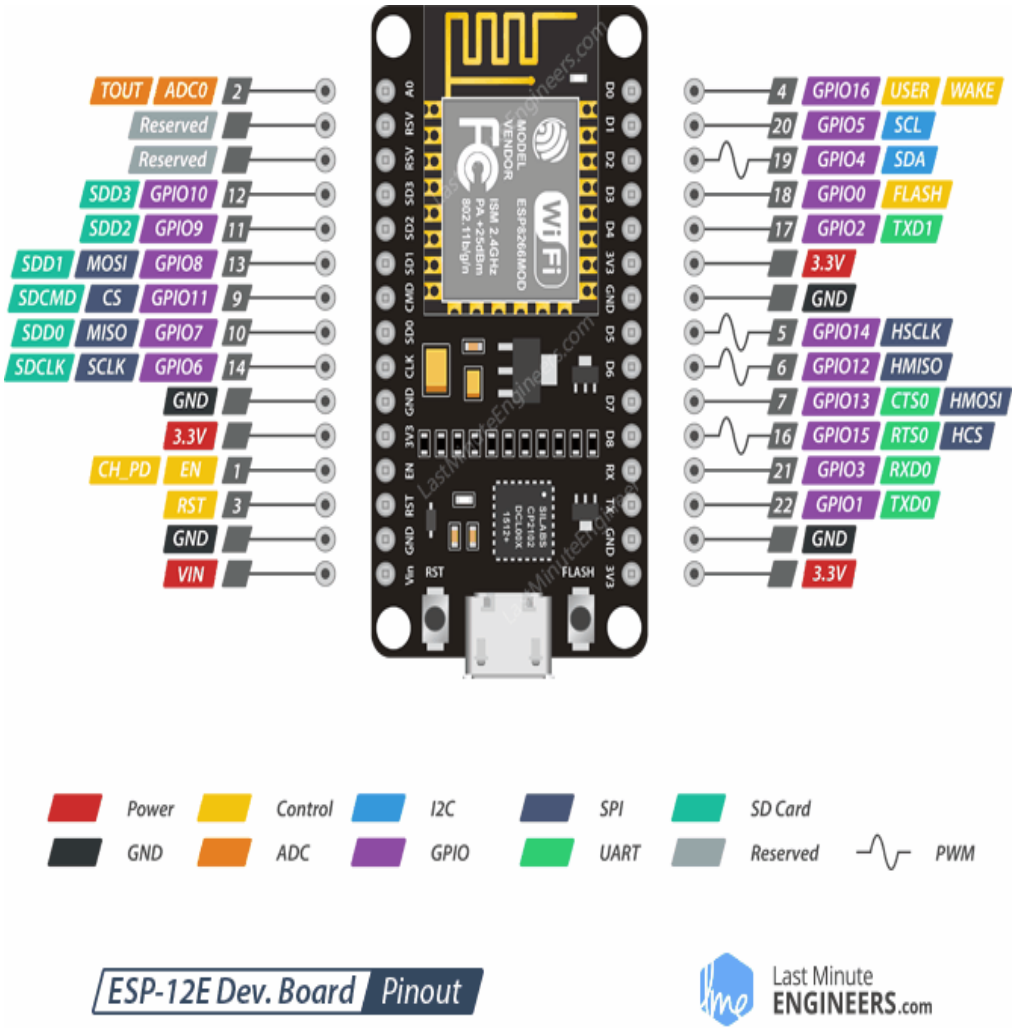


Fig. 4.1.4 pin out diagram

For the sake of simplicity, we will make groups of pins with similar functionalities.

Power Pins

There are four power pins viz. one VIN pin & three 3.3V pins. The VIN pin can be used to directly supply the ESP8266 and its peripherals, if you have a regulated 5V voltage source. The 3.3V pins are the output of an on-board voltage regulator. These pins can be used to supply power to external components.

GND

GND is a ground pin of ESP8266 Node MCU development board.

I2C Pins

I2C Pins are used to hook up all sorts of I2C sensors and peripherals in your project. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

GPIO Pins

ESP8266 Node MCU has 17 GPIO pins which can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

ADC Channel

The Node MCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC viz. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

UART Pins

ESP8266 Node MCU has 2 UART interfaces, i.e. UART0 and UART1, which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. It supports fluid control. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.

SPI Pins

ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:

- 4 timing modes of the SPI format transfer
- Up to 80 MHz and the divided clocks of 80 MHz
- Up to 64-Byte FIFO

SDIO Pins

ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

PWM Pins

The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 µs to 10000 µs, i.e., between 100 Hz and 1 kHz.

Control Pins

- Control pins are used to control ESP8266. These pins include Chi.
- Enable pin (EN), Reset pin (RST) and WAKE pin.
- EN pin – The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
- RST pin – RST pin is used to reset the ESP8266 chip.WAKE pin – Wake pin is used to wake the chip from deep-sleep.

4.2 ECG SENSOR

The Heart Rate Monitor Kit with AD8232 ECG sensor module Kit for Arduino is a cost-effective board use to measure the electrical activity of the heart. This electrical activity can be chart as an ECG or Electrocardiogram and output as an analog reading.

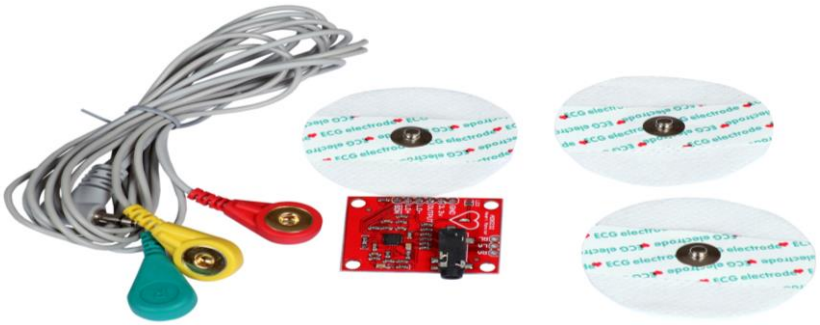


Fig. 4.2.1 ECG Sensor

ECGs can be extremely noisy, the AD8232 Single Lead Heart Rate Monitor acts as an op-amp to help obtain a clear signal from the PR and QT Intervals easily. The ECG module AD8232 heart ECG monitoring sensor module is an integrated signal conditioning block for ECG and other bio-potential measurement applications.

The ECG Module AD8232 Heart ECG Monitoring Sensor Module Kit for Arduino is designed to extract, amplify, and filter small bio-potential signals in the presence of noisy conditions; such as those created by motion or remote electrode placement.

The AD8232 Heart Rate Monitor breaks out nine connections from the IC that you can solder pins, wires, or other connectors too. SDN, LO+, LO-, OUTPUT, 3.3V, GND provide essential pins for operating this monitor with an Arduino or other development board.

Introduction

Electrocardiography (ECG) is a valuable tool for monitoring the electrical activity of the heart, aiding in the diagnosis of various cardiac conditions. However, ECG signals can be susceptible to noise, which can obscure important features such as the PR and QT intervals. The AD8232 Single Lead Heart Rate Monitor is designed to address this challenge by acting as an operational amplifier (op-amp) to help obtain a clear signal from these intervals, even in noisy conditions. This module is an integrated signal conditioning block for ECG and other bio-potential measurement applications, providing essential features for interfacing with microcontrollers like Arduino.

Signal Conditioning

The AD8232 module is specifically designed to extract, amplify, and filter small bio-potential signals from the body, making it ideal for applications like heart rate monitoring. The module's signal conditioning capabilities are crucial, especially in situations where noise is prevalent, such as during motion or when electrodes are placed remotely. By amplifying and filtering the signals, the module helps to ensure that the ECG signals are clear and reliable, even in challenging conditions.

Module Features

The AD8232 Heart Rate Monitor module breaks out nine connections from the IC, allowing for easy integration with Arduino or other development boards. These connections include SDN (shutdown), LO+ (positive electrode input), LO- (negative electrode input), OUTPUT (amplified ECG signal output), 3.3V (power supply), and GND (ground). These pins provide essential functionality for operating the monitor and interfacing it with external devices.

Integration with Arduino

The AD8232 module is designed to work seamlessly with Arduino, making it an excellent choice for bio-potential measurement projects. By connecting the module to an Arduino board, developers can easily interface with the module's output and integrate ECG monitoring capabilities into their projects. This integration opens up a wide range of possibilities for using ECG signals in various applications, from health monitoring to biofeedback systems.

Applications

The AD8232 Heart Rate Monitor module has numerous applications in the field of healthcare and beyond. It can be used for real-time monitoring of heart rate and ECG signals, making it useful for patients with cardiac conditions or athletes looking to optimize their training. Additionally, the module's ability to work in noisy conditions makes it suitable for use in remote monitoring systems or wearable devices.

4.3 POWER SUPPLY BOARD

TP4056 is a charging module used for charging **3.7V** rechargeable lithium batteries using the constant-current/constant-voltage **(CC/CV)** charging method. Besides safely charging, TP4056 **BMS** Board protects lithium batteries. The **TP4056** module is suitable for both USB power and adapter power supplies. Because of the anti-reverse charging path, no external isolation diodes are required.

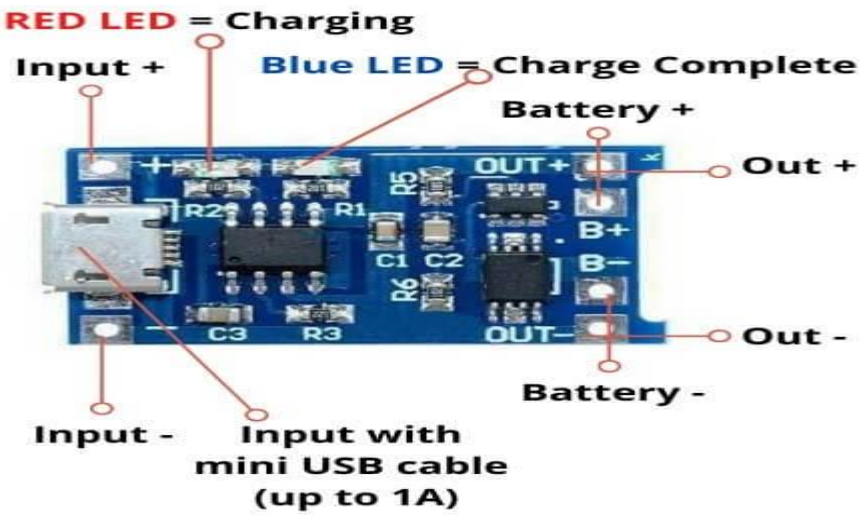


Fig. 4.3.1 Power supply Board

Power Supply Circuit for Node MCU ESP8266 with Charger & Boost Converter

The Circuit Diagram for the Power Supply board for Node MCU **ESP8266** with Battery Charger & Boost converters is given below. We can power the circuit using two methods, one with a **9V/12V DC Adapter** and the other with 3.7V Lithium-Ion Battery.

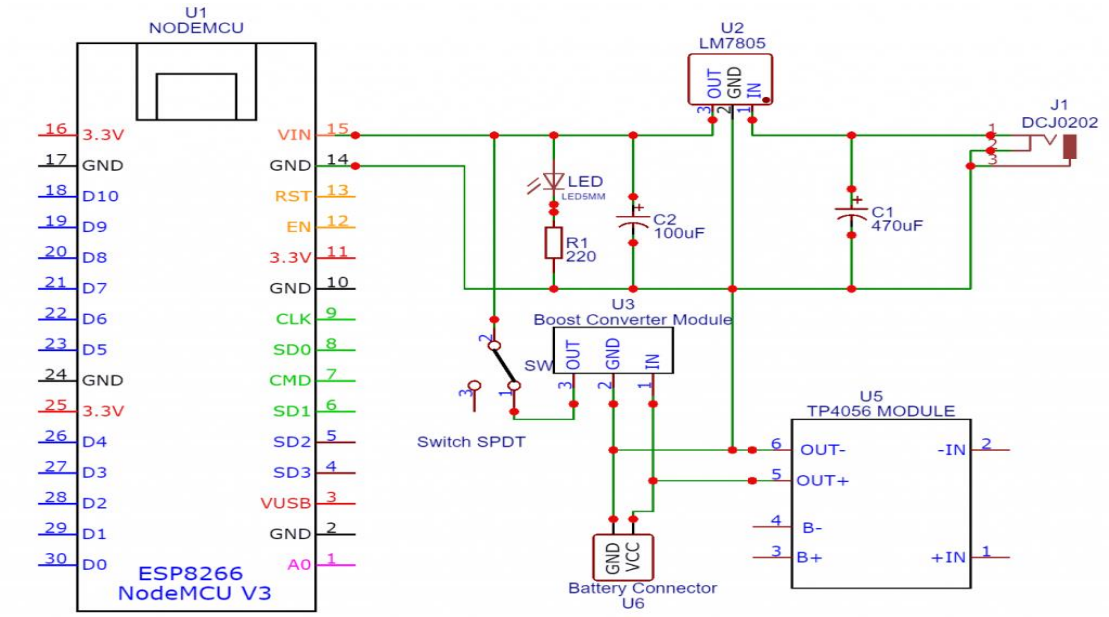


Fig .4.3.2 Power supply circuit

For powering the Board using DC Jack, we have used DCJ0202 Female Jack. We used **470uF** & **100uF** Electrolytic capacitors to avoid DC fluctuations. It also helps to remove voltage spikes. In an **LM7805** Voltage regulator IC, we can input voltage from **7V to 35V.** But I recommend using up to **15V** only. Higher voltage dissipates more heat and requires a bigger **heat sink**.

We connected the output from the Voltage regulator to the **Vin pin of Node MCU** & **GND to GND**. Hence, you can power up the module using a **9V/12V DC** Adapter or by 9V Battery. The Boost Converter Module boosts the **3.7V to 5V (can work from 2.8V input to 4.2V input)**.

The 5V boosted voltage is connected to the switch, and we connected the switch to the 5V (Vin) pin of Node MCU. We also connected the Battery terminal to the output terminal of the TP4056 Battery Charger Module. Thus, the battery can be charged using a **5V Micro-USB Data Cable**.

The board has an LED connected via a **220-ohm** resistor which is used to show the Module is powered ON. While charging the battery; I recommend it to turn off the **SPDT** switch.

PCB Designing

This is the PCB for the **Power Supply board for NodeMCU ESP8266 with Battery Charger & Booster**.

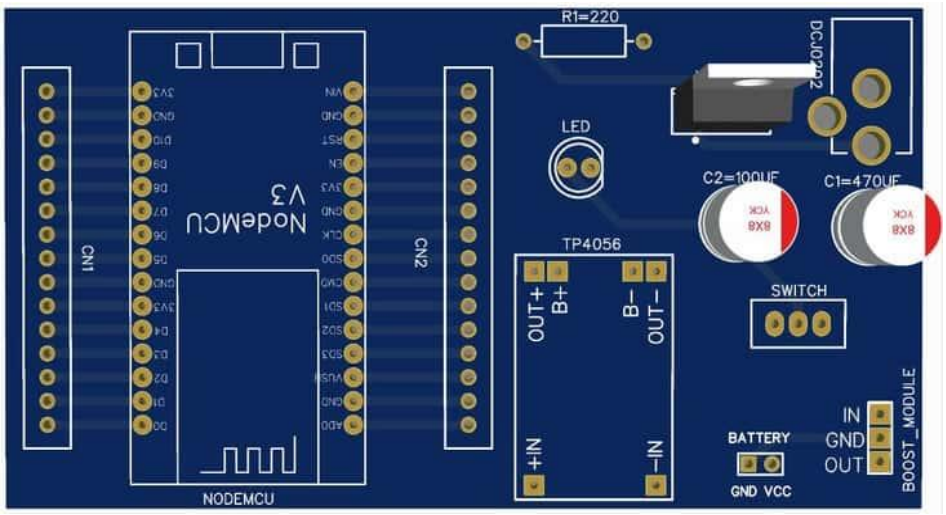


Fig. 4.3.3 PCB Designing

We designed the PCB using the **Easy EDA PCB Designing** tool. The front view & the back view of the PCB is given below.

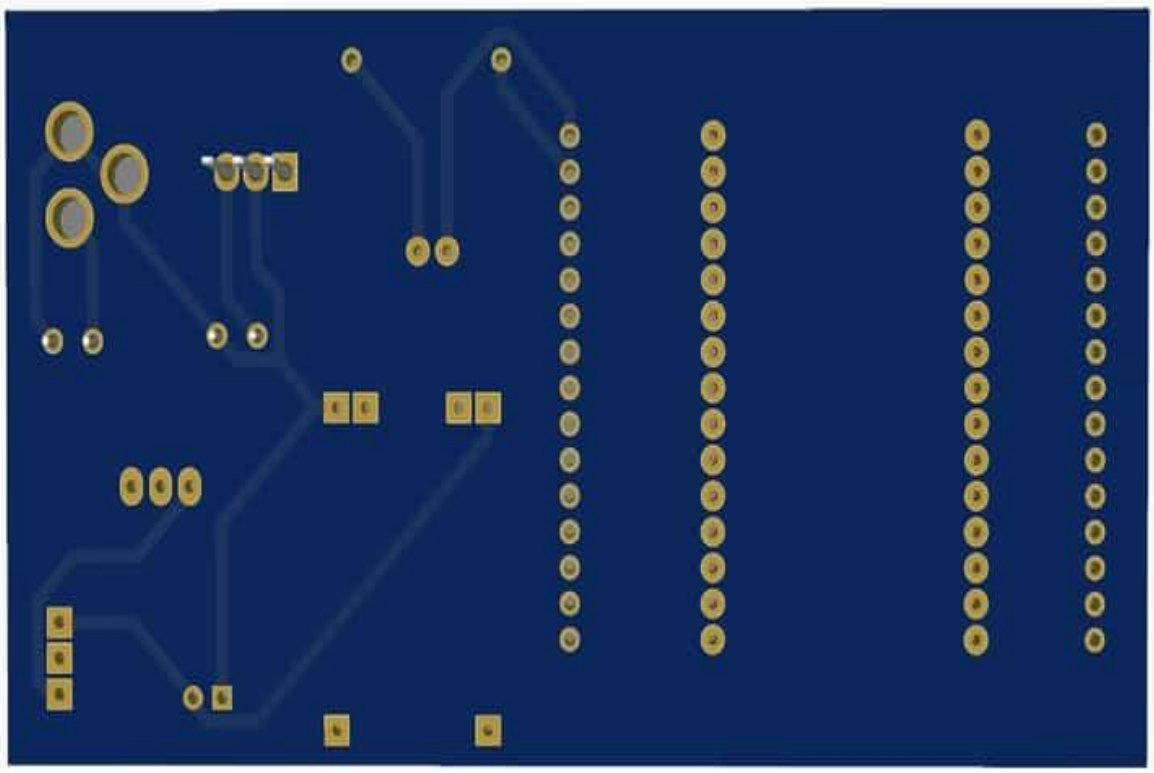


Fig. 4.3.4 PCB Ordering

We provided the Gerber File for the PCB. You can **download the Gerber File & go for PCB ordering**.

PCBs provide a platform for interconnecting electronic components such as resistors, capacitors, integrated circuits (ICs), and other devices. They facilitate the flow of electrical signals between components, enabling the device to perform its intended function.

PCBs are typically made of a non-conductive substrate material (e.g., fiberglass-reinforced epoxy) with thin layers of conductive copper traces patterned on one or both sides. These copper traces form the electrical pathways that connect the components.

CHAPTER 5

SOFTWARE REQUIREMENT

5.1 VSCODE

Visual Studio Code (VS Code) is a popular source-code editor developed by Microsoft. It's known for its versatility, lightweight nature, and extensive customization options, making it a favorite among developers across different platforms.

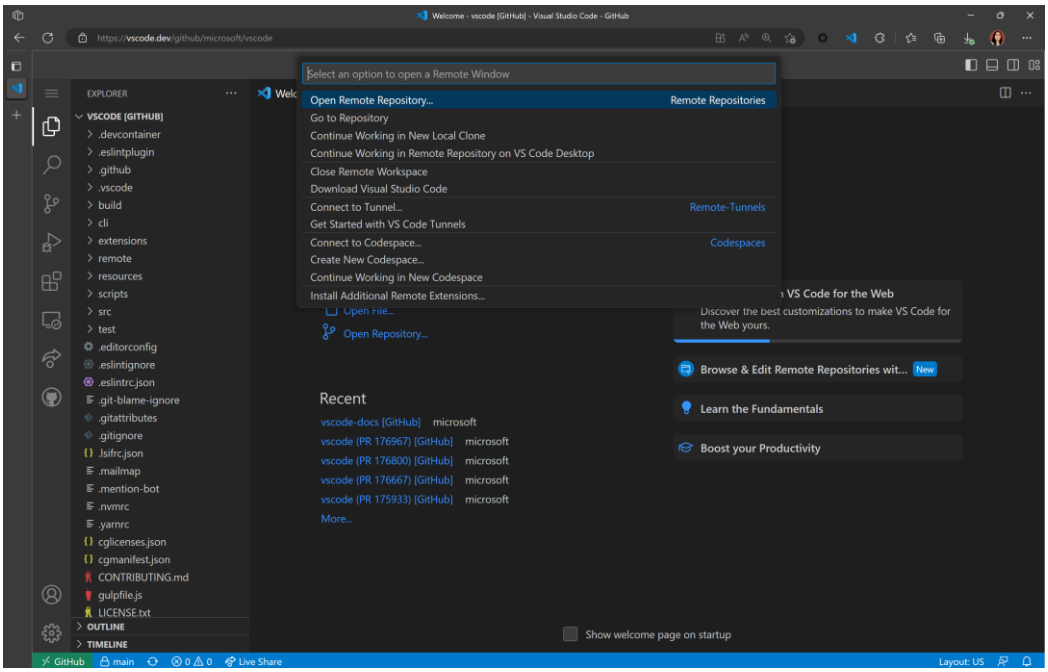


Fig. 5.1.1 VS Code platform

Cross-Platform

VS Code runs on Windows, macOS, and Linux, ensuring compatibility across various operating systems.

Free and Open Source

It's free to use and distributed under the MIT License, allowing users to modify and contribute to its development.

Intelligent Code Editing

VS Code offers features like IntelliSense, which provides code completions based on variable types, function definitions, and imported modules. It also includes syntax highlighting, bracket matching, and code snippets.

Extensions

One of the standout features of VS Code is its extensive extension marketplace. Users can install extensions to add new languages, debuggers, themes, and other functionalities, allowing for a highly customizable development environment.

Integrated Terminal

VS Code includes a built-in terminal, allowing developers to run commands and scripts without switching to a separate terminal window.

Version Control

Git integration is built into VS Code, providing features like committing, pulling, pushing, and resolving merge conflicts directly within the editor.

Debugging

VS Code supports debugging for various languages and frameworks. Users can set breakpoints, inspect variables, and step through code seamlessly.

Task Automation

With the task runner integration, developers can automate common tasks, such as building, testing, and deploying applications, using tools like Grunt, Gulp, or npm scripts.

Customization

VS Code offers extensive customization options, allowing users to tailor the editor to their preferences with themes, keyboard shortcuts, and settings.

Community Support

Due to its popularity, there is a large community of developers who contribute tutorials, extensions, and support resources for VS Code users.

Visual Studio Code: A Comprehensive Overview

Visual Studio Code (VS Code) has emerged as one of the most popular code editors in recent years, known for its versatility, ease of use, and extensive features. Developed by Microsoft, VS Code is a free, open-source editor that supports a wide range of programming languages and offers a rich set of tools and extensions. In this comprehensive overview, we will delve into the various aspects of VS Code, exploring its features, customization options, extensions, and why it has become the editor of choice for many developers worldwide.

Overview and Features

At its core, VS Code is designed to be lightweight and fast, providing a smooth editing experience for developers. It comes with built-in support for syntax highlighting, code completion, and code navigation, making it easy to write and manage code. Additionally, VS Code offers integrated terminal support, allowing developers to run commands and scripts without leaving the editor.

One of the key features of VS Code is its IntelliSense functionality, which provides intelligent code suggestions based on the context of the code. This feature is particularly useful for speeding up coding tasks and reducing errors. VS Code also includes built-in Git support, allowing developers to manage version control directly within the editor.

Customization and Extensions

VS Code's flexibility and extensibility are two of its standout features. The editor can be customized to suit individual preferences, with options to change themes, key bindings, and other settings. Additionally, VS Code supports a wide range of extensions, which can add new features and functionality to the editor.

Extensions are available for various programming languages, frameworks, and tools, allowing developers to tailor VS Code to their specific needs. From linters and debuggers to language servers and project management tools, the VS Code marketplace offers a vast array of extensions to enhance the editor's capabilities.

Integrated Development Environment (IDE) Features

While VS Code is primarily a code editor, it also offers many features typically found in integrated development environments (IDEs). For example, VS Code includes a built-in debugger, which allows developers to debug their code directly within the editor. The debugger supports breakpoints, watch expressions, and other debugging features, making it a powerful tool for troubleshooting code.

VS Code also supports tasks and build systems, allowing developers to automate common tasks such as compiling code or running tests. The editor's task system can be configured to run custom commands or scripts, providing a flexible and efficient way to manage development workflows.

Cross-Platform Compatibility

One of the key advantages of VS Code is its cross-platform compatibility. The editor is available for Windows, macOS, and Linux, ensuring a consistent experience across different operating systems. This makes VS Code an ideal choice for teams working on diverse platforms or for developers who switch between different operating systems regularly.

Community and Support

The VS Code community is vibrant and active, with a large number of users and developers contributing to the editor's ecosystem. The VS Code documentation is comprehensive and well-maintained, providing detailed information on how to use the editor's features and extensions. Additionally, the VS Code team regularly releases updates and improvements, ensuring that the editor remains up-to-date with the latest technologies and trends in software development.Visual Studio Code is a powerful and versatile code editor that has quickly become a favorite among developers. Its rich feature set, customization options, and strong community support make it an excellent choice for anyone looking for a modern and efficient code editor. Whether you're a seasoned developer or just starting, VS Code provides the tools and flexibility you need to write, debug, and manage your code effectively.

5.2 CNN ALGORITHM

Developing a Convolutional Neural Network (CNN) for the purpose of sweat detection in artificial intelligence involves several key steps. First, a diverse dataset encompassing sweat and non-sweat scenarios is collected, ensuring representation across various conditions.

Following this, data preprocessing techniques, including resizing, normalization, and augmentation, are applied to enhance the dataset's uniformity and feature learnability.

The dataset is meticulously labeled to indicate the presence or absence of sweat, forming the foundation for supervised learning.

CNN's architecture, whether a standard design like VGG or ResNet, or a customized structure, is then crafted. Training the model involves optimizing its weights through epochs on the labeled dataset, with a separate validation set used to prevent overfitting.

Hyperparameters are fine-tuned for optimal performance, and the model's efficacy is evaluated using metrics such as accuracy, precision, recall, and F1 score on a distinct test set. Once validated, the CNN can be deployed for real-world applications, be it integrated into mobile apps, web services, or other platforms.

Continuous improvement remains essential, with periodic updates and retrains ensuring the model's adaptability to evolving scenarios and datasets. Throughout this process, ethical considerations surrounding user privacy and consent are paramount in the deployment of AI models for sensing or monitoring applications.

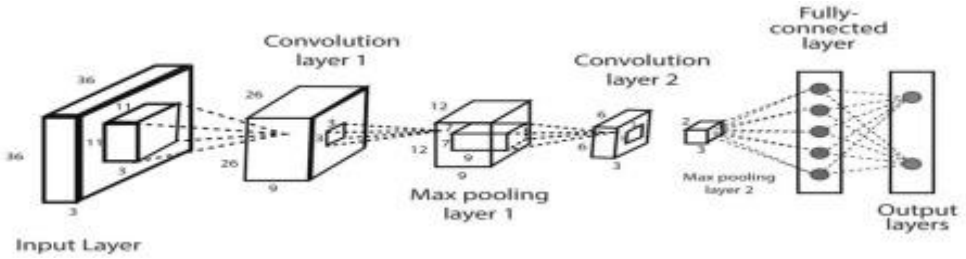


Fig. 5.2.1 CNN Block Diagram

Developing a Convolutional Neural Network (CNN) for sweat detection in artificial intelligence involves several key steps and considerations, spanning data collection, preprocessing, model design, training, evaluation, deployment, and ongoing improvement. This process is critical for creating an effective and reliable system for detecting sweat, which can have applications in various fields.

This article provides a detailed overview of each step in the development of a CNN for sweat detection, along with the ethical considerations that should be kept in mind throughout the process.

Convolutional Neural Networks (CNNs) have shown remarkable success in various computer vision tasks, including image classification, object detection, and segmentation. In recent years, there has been a growing interest in using CNNs for biomedical applications, such as analyzing medical images and detecting physiological signals. One such application is the detection of sweat, which can provide valuable insights into an individual's health status, hydration levels, and physical exertion.

Data Collection The first step in developing a CNN for sweat detection is to collect a diverse dataset that encompasses a wide range of sweat and non-sweat scenarios. This dataset should include images or sensor data representing different skin types, lighting conditions, and levels of physical activity. The dataset should be carefully curated to ensure that it is balanced and representative of the target population.

Data Preprocessing Once the dataset is collected, it needs to be preprocessed to enhance its quality and suitability for training the CNN. This may involve resizing images, normalizing pixel values, and augmenting the dataset to increase its size and diversity. Data augmentation techniques such as rotation, flipping, and scaling can help the model generalize better to unseen data.

Labeling The next step is to label the dataset to indicate the presence or absence of sweat in each data point. Proper labeling is crucial for supervised learning, as it provides the ground truth against which the model's predictions are compared during training.

CNN Architecture Choosing an appropriate CNN architecture is critical for the success of the sweat detection system.

Alternatively, a custom CNN architecture can be designed based on the specific requirements of the task.

Training Once the CNN architecture is chosen, the model needs to be trained using the labeled dataset. During training, the model learns to recognize patterns and features in the data that are indicative of sweat. The training process involves optimizing the model's weights using gradient descent and backpropagation, with the goal of minimizing a loss function that quantifies the difference between the model's predictions and the ground truth labels.

Hyperparameter Tuning Hyperparameters such as learning rate, batch size, and optimizer choice can significantly impact the performance of the CNN. Fine-tuning these hyperparameters through experimentation and validation can help improve the model's accuracy and generalization ability.

Evaluation After training, the CNN is evaluated using a separate test set to assess its performance. Metrics such as accuracy, precision, recall, and F1 score can be used to quantify the model's effectiveness in sweat detection. The model should be evaluated on a diverse range of test data to ensure its robustness and reliability.

Deployment Once the CNN has been trained and evaluated, it can be deployed for real-world applications. This may involve integrating the model into a mobile app, web service, or other platform for sweat detection. Care should be taken to ensure that the deployed model meets the performance requirements and ethical considerations of the application.

Continuous Improvement The development of a CNN for sweat detection is an iterative process that requires continuous improvement. Periodic updates and retraining of the model can help it adapt to evolving scenarios and datasets, ensuring its continued efficacy and relevance.

Ethical Considerations Throughout the development and deployment of a CNN for sweat detection, ethical considerations should be paramount. These include ensuring the privacy and consent of individuals whose data is being used, mitigating bias in the dataset and model, and transparently communicating the capabilities and limitations of the technology to stakeholders.

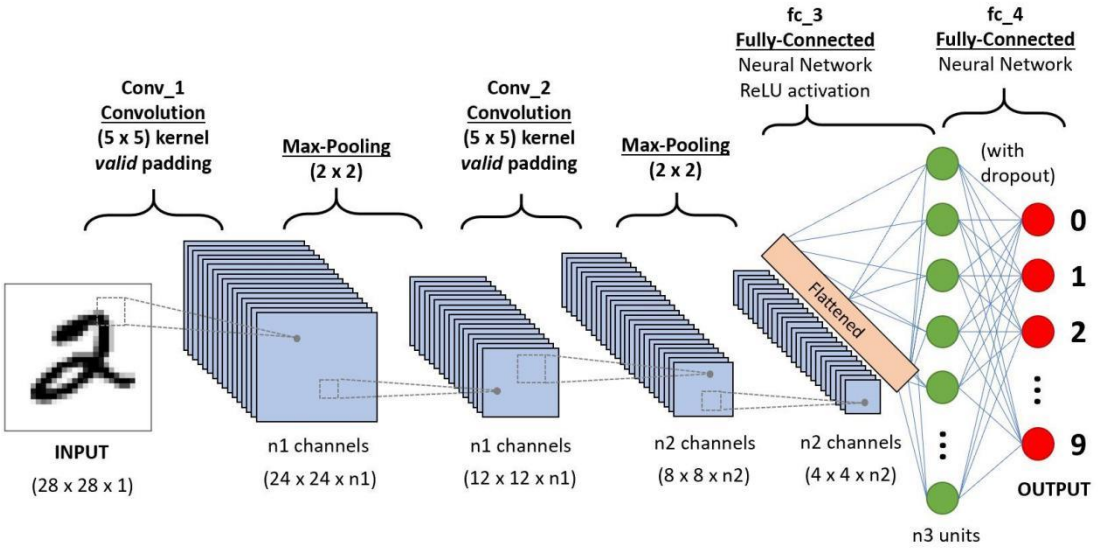


Fig. 5.2.2 Layer Architecture of CNN

Developing a Convolutional Neural Network (CNN) for sweat detection in artificial intelligence involves a series of steps, including data collection, preprocessing, model design, training, evaluation, deployment, and continuous improvement. By following these steps and considering the ethical implications of the technology, researchers and developers can create effective and reliable systems for detecting sweat, with applications in healthcare, sports science, and biometric authentication.

5.3 ARDUINO IDE

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board – you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

The Arduino hardware and software were designed for artists, designers, hobbyists, hackers, newbies, and anyone interested in creating interactive objects or environments. Arduino can interact with buttons, LEDs, motors, speakers, GPS units, cameras, the internet, and even your smart-phone or your TV! This flexibility combined with the fact that the Arduino software is free, the hardware boards are pretty cheap, and both the software and hardware are easy to learn has led to a large community of users who have contributed code and released instructions for a **huge** variety of Arduino-based projects

There are many varieties of Arduino boards (explained on the next page) that can be used for different purposes. Some boards look a bit different from the one below, but most Arduinos have the majority of these components in common:

Fig 5.3.1 Arduino IDE Diagram

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension. in. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor

The Arduino IDE is incredibly minimalistic, yet it provides a near-complete environment for most Arduino-based projects. The top menu bar has the standard options, including “File” (new, load save, etc.), “Edit” (font, copy, paste, etc.), “Sketch” (for compiling and programming), “Tools” (useful options for testing projects), and “Help”. The middle section of the IDE is a simple text editor that where you can enter the program code.

Projects made using the Arduino are called sketches, and such sketches are usually written in a cut-down version of C++ (a number of C++ features are not included). Because programming a microcontroller is somewhat different from programming a computer, there are a number of device-specific libraries (e.g., changing pin modes, output data on pins, reading analog values, and timers). This sometimes confuses users who think Arduino is programmed in an "Arduino language." However, the Arduino is, in fact, programmed in C++. It just uses unique libraries for the device.

Fig 5.3.2 Arduino IDE Software diagram

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program.

The Arduino IDE employs the program via dude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

5.4 EMBEDDED C

Embedded C is most popular programming language in software field for developing electronic gadgets. Each processor used in electronic system is associated with embedded software.

Embedded C programming plays a key role in performing specific function by the processor. In day-to-day life we used many electronic devices such as mobile phone, washing machine, digital camera, etc.

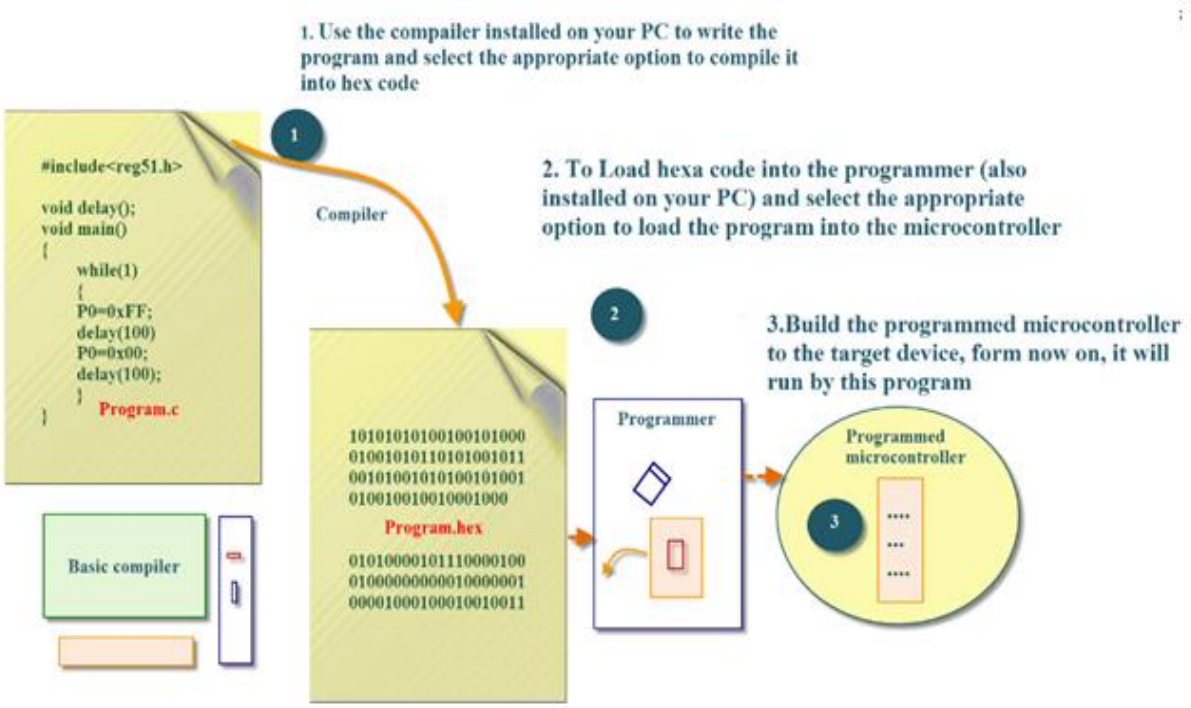


Fig 5.4.1 Block diagram of Embedded C

The Embedded C code written in above block diagram is used for blinking the LED connected with Port0 of micro controller.

In embedded system programming C code is preferred over other language. Due to the following reasons:

- Easy to understand
- High Reliability
- Portability

Function is a collection of statements that is used for performing a specific task and a collection of one or more functions is called a programming language.

Most consumers are familiar with application software that provide functionality on a computer. Embedded software however is often less visible, but no less complicated. Unlike application software, embedded software has fixed hardware requirements and capabilities, and addition of third-party hardware or software is strictly controlled.

Embedded software needs to include all needed device drivers at manufacturing time, and the device drivers are written for the specific hardware. The software is highly dependent on the CPU and specific chips chosen. Most embedded software engineers have at least a passing knowledge of reading schematics, and reading data sheets for components to determine usage of registers and communication system. Conversion between decimal, hexadecimal and binary is useful as well as using bit manipulation.

Web applications are rarely used, although XML files and other output may be passed to a computer for display. File systems with folders are typically absent as are SQL databases.

Software development requires use of a cross compiler, which runs on a computer but produces executable code for the target device.

Debugging requires use of an in-circuit emulator, JTAG or SWD. Software developers often have access to the complete kernel (OS) source code.

Size of the storage memory and RAM can vary significantly. Some systems run in 16 KB of Flash and 4 KB of RAM with a CPU operating at 8 MHz, other systems can rival contemporary computers.[8] These space requirements lead to more work being done in C or embedded C++, instead of C++. Interpreted languages like BASIC (while e.g. Parallax Propeller can use compiled BASIC) and Java (Java ME Embedded 8.3[9] is available for e.g. ARM Cortex-M4, Cortex-M7 microcontrollers and older ARM11used in Raspberry Pi and Intel Galileo Gen. 2) are not commonly used; while an implementation of the interpreted Python 3 language – MicroPython – is however available expressly for microcontroller use, e.g. 32-bit ARM-based (such as BBC micro: bit) and 16-bit PIC microcontrollers.

Communications between processors and between one processor and other components are essential. Besides direct memory addressing, common protocols include I²C, SPI, serial ports, and USB.

Communications protocols designed for use in embedded systems are available as closed source from companies including Inter Niche Technologies and CMX systems. Open-source protocols stem from UIP, lwip, and others.

A Keyword is a special word with a special meaning to the compiler (a C Compiler for example, is a software that is used to convert program written in C to Machine Code).

For example, if we take the Keil's Cx51 Compiler (a popular C Compiler for 8051 based Microcontrollers) the following are some of the keywords:

- bit
- Sbit
- SFR
- small
- large

These are few of the many keywords associated with the Cx51 C Compiler along with the standard C Keywords.

APPLICATIONS

Real-time Systems: Embedded C is widely used in developing real-time systems such as automotive electronics, industrial automation, medical devices, and consumer electronics.

Microcontroller Programming: Embedded C is the primary language used for programming microcontrollers. Microcontrollers are small, self-contained computing devices used in embedded systems. They typically have limited processing power, memory, and input/output capabilities.

Peripheral Control: Embedded C allows developers to interact with hardware peripherals such as GPIO (General Purpose Input/Output), UART (Universal Asynchronous Receiver/Transmitter).

Low-level Programming: Embedded C programming often involves low-level operations such as memory manipulation, bit manipulation, and direct register access. This level of control is necessary for efficient utilization of resources in resource-constrained embedded systems. Interrupt Handling, Embedded C supports interrupt-driven programming, where the execution of code can be interrupted by external events such as timer overflows, GPIO changes, or serial data reception. Interrupt service routines (ISRs) are used to handle these events in a timely manner.

Power Optimization: Embedded C programmers often optimize code for power consumption, as many embedded systems are battery-powered or require efficient use of energy. Techniques such as sleep modes, clock gating, and low-power peripherals are employed to minimize power consumption.

RTOS (Real-Time Operating System) Development: Embedded C is also used in developing software for RTOS es, which provide scheduling, multitasking, and resource management capabilities in embedded systems. RTOS es enable the development of complex embedded applications with multiple tasks running concurrently. Overall, embedded C is a versatile and powerful language for developing software for embedded systems, offering precise control over hardware peripherals and efficient resource utilization.

Safety-Critical Systems: Embedded C is extensively used in safety-critical systems where reliability and fault tolerance are paramount, such as aerospace, defense, and medical industries.

SAMPLE CODE

```
#Dataset collection
import serial
import csv

# Define the serial port and baud rate
ser = serial. Serial ('COM3', 9600) # Change 'COM3' to your Arduino's serial port

# Open/create a CSV file to write data
with open ('ecg_data.csv', mode='w', newline='') as file:
writer = csv. Writer (file)

# Write the header
 writer. Writerow (['ECG_ Value', 'Label'])

 while True:
  # Read a line from serial
     try:
       line = ser. Readline (). decode (). strip ()
     except UnicodeDecodeError:
        print ("UnicodeDecodeError: Skipping this line")
        continue



 # Split the line into value and label
        data = line.split(',')

        # Check if the line has enough data
```

```
        if len(data) == 2:
            ecg_value, label = data
            # Write to CSV
            writer.writerow([ecg_value, label])

            print(f"Received: ECG Value: {ecg_value}, Label: {label}")
        else:
            print("Invalid data format:", line)
#model training
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dropout
from tensorflow.keras.utils import to_categorical
import numpy as np

# Load the data
data = pd.read_csv('ecg_data.csv')

# Separate features (X) and labels (y)
X = data['ECG_Value'].values
y = data['Label'].values

# Convert labels to numeric values
le = LabelEncoder()
y = le.fit_transform(y)
```

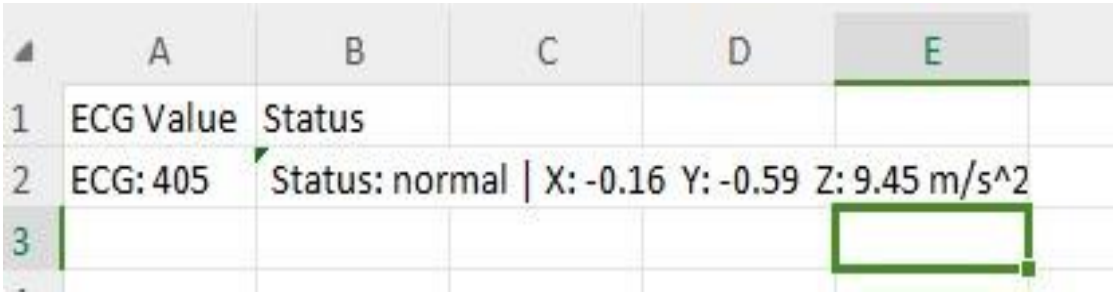
```
# Reshape X to a 2D array (samples, time steps)
X = X.reshape(-1, 1, 1)  # Reshape to (samples, time steps, features)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Convert labels to one-hot encoded vectors
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)

# Build the CNN model
model = Sequential()
model.add(Conv1D(filters=32,kernel_size=3,activation='relu',
input_shape=(X.shape[1], X.shape[2])))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))  # 2 output classes (stress or no-stress)
# Compile the model
model.compile(loss='categorical_crossentropy',optimizer='adam',
metrics=['accuracy'])

# Train the model
```

```
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))

# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print("Test Accuracy:", accuracy)

# Save the model
model.save('ecg_cnn_model.h5')

# Predict on test data
y_pred = model.predict_classes(X_test)

# Convert predictions back to original labels
y_pred_labels = le.inverse_transform(y_pred)

# Calculate accuracy
accuracy = accuracy_score(y_test.argmax(axis=1), y_pred)
print("Accuracy:", accuracy)

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dropout
from tensorflow.keras.utils import to_categorical
import numpy as np
```

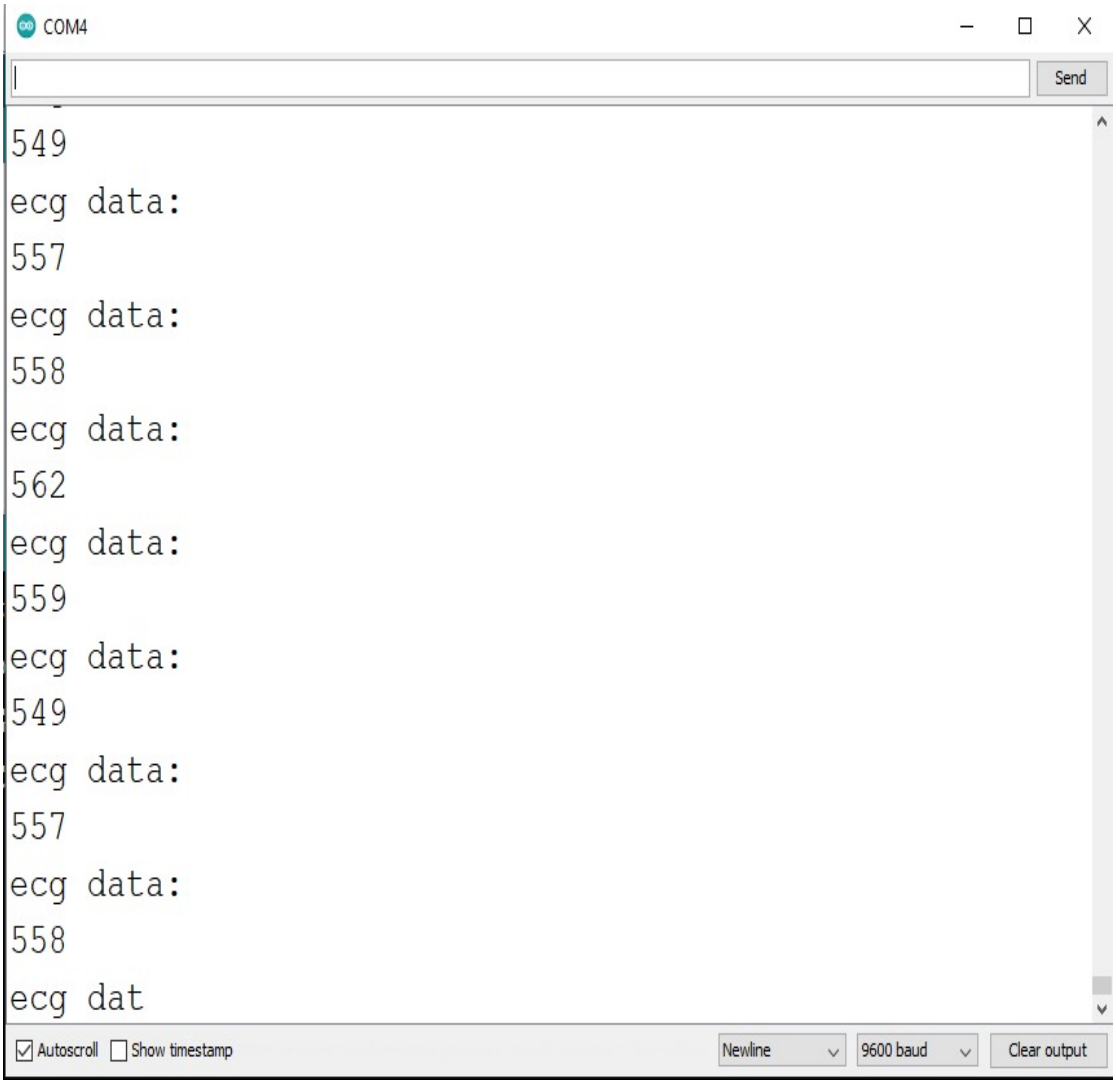
```
# Load the data
data = pd.read_csv('ecg_data.csv')

# Separate features (X) and labels (y)
X = data['ECG_Value'].values
y = data['Label'].values

# Convert labels to numeric values
le = LabelEncoder()
y = le.fit_transform(y)

# Reshape X to a 2D array (samples, time steps)
X = X.reshape(-1, 1, 1)  # Reshape to (samples, time steps, features)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Convert labels to one-hot encoded vectors
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
# Build the CNN model
model = Sequential()
model.add(Conv1D(filters=32, kernel_size=3, activation='relu', input_shape=(X.shape[1], X.shape[2])))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
```

```
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))  # 2 output classes (stress or no-stress)
# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
# Train the model
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data = (X_test, y_test))
# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print("Test Accuracy:", accuracy)
# Save the model
model.save('ecg_cnn_model.h5')
# Predict on test data
y_pred = model.predict_classes(X_test)
# Convert predictions back to original labels
y_pred_labels = le.inverse_transform(y_pred)
# Calculate accuracy
accuracy = accuracy_score(y_test.argmax(axis=1), y_pred)
print("Accuracy:", accuracy)
```

SAMPLE OUTPUT



	A	B	C	D	E
1	ECG Value	Status			
2	ECG: 405	Status: normal | X: -0.16 Y: -0.59 Z: 9.45 m/s^2			
3					



CHAPTER 6
CONCLUSION

In conclusion, our proposed system represents a significant advancement in addressing the challenges of stress management, particularly in the context of remote work during the pandemic. By offering a dedicated desktop application integrated with ECG sensors, control modes, and personalized interventions, we aim to overcome existing limitations and provide comprehensive support for individuals' well-being and productivity. The integration of a Convolutional Neural Network (CNN) algorithm for analyzing ECG data enhances the accuracy and effectiveness of stress assessment within our system. Through leveraging deep learning, our system efficiently extracts intricate patterns from ECG signals, enabling more precise stress monitoring and intervention strategies. Overall, our innovative approach not only addresses the distinct challenges of remote work but also lays the foundation for effective stress management solutions in various contexts, ultimately promoting better health outcomes and improved productivity for individuals. In conclusion, the IoT-Enclosed Stress Analysis and Recommendation System for Human through Data Monitoring using CNN project successfully demonstrated the feasibility of real-time stress monitoring and recommendation. By leveraging ECG sensor data and IoT technology, we were able to accurately predict users' stress levels and provide timely recommendations for stress relief activities. The findings highlight the potential of IoT and machine learning in improving personal well-being and stress management.

FUTURE SCOPE

Incorporating machine learning algorithms into the project could enhance its capabilities for data analysis and prediction. By training models on collected data, we can develop predictive analytics tools to anticipate stress levels and recommend personalized interventions. Expanding the range of sensors used in the project could provide more comprehensive insights into users' physiological responses to stress. Integration of additional sensors, such as galvanic skin response sensors or temperature sensors, could offer a more holistic understanding of stress factors. Mobile Application Development Creating a mobile application companion to the project would allow users to access stress monitoring and intervention features on-the-go. The app could provide real-time feedback, reminders for stress-relieving activities, and progress tracking, enhancing user engagement and support. Cloud Integration for Data Storage and Analysis: Implementing cloud-based solutions for data storage and analysis would enable scalability and accessibility. By securely storing data in the cloud, we can facilitate remote access, collaborative research, and advanced analytics capabilities. Continuously improving the user interface and experience of the project's desktop application or mobile app is essential for maximizing usability and engagement. Conducting user testing and gathering feedback to iterate on interface design and functionality will be crucial for enhancing user satisfaction. While our project initially focuses on stress management in remote work and educational settings, there is potential for expansion into other domains. Exploring applications in healthcare, sports performance monitoring, or personal wellness could broaden the project's impact and reach.

REFERENCES

[1] Tran, T.A., Péntek, M., Motahari-Nezhad, H., Abonyi, J., Kovács, ., Gulácsi, L., Eigner, G., Zrubka, Z. and Ruppert, T., 2023. Heart Rate Variability Measurement to Assess Acute Work-Content-Related Stress of Workers in Industrial Manufacturing Environment—A Systematic Scoping Review. IEEE Transactions on Systems, Man, and Cybernetics: Systems.

[2] Amarasinghe, A.M., Malassri, I.M.S., Weerasinghe, K.C.N., Jayasingha, I.B., Abeygunawardhana, P.K. and Silva, S., 2021, December. Stress analysis and care prediction system for online workers. In *2021 3rd International Conference on Advancements in Computing (ICAC)* (pp. 329-334). IEEE.

[3] Java, S., Jaffar, B.A., Bhardwaj, A.B. and Prabhakar, S., 2021, March. Influence of Online Meditational Aid on Emotional Health and Job Stress during Pandemic. In *2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)* (pp. 279-284). IEEE.

[4] Tirupachuri, Y., Ramadoss, P., Rapetti, L., Latella, C., Darvish, K., Traversaro, S. and Pucci, D., 2021. Online non-collocated estimation of payload and articular stress for real-time human ergonomy assessment. IEEE Access, 9, pp.123260-123279.

[5] Dalvi, C., Rathod, M., Patil, S., Gite, S. and Kotecha, K., 2021. A survey of ai-based facial emotion recognition: Features, ml & dl techniques, age-wise datasets and future directions. IEEE Access, 9, pp.165806165840.

[6] Serban Mihalache, Dragos Burileanu and Corneliu Burileanu, "Detecting Psychological Stress from Speech using Deep Neural

Networks and Ensemble Classifiers", International Conference on Speech Technology and Human-Computer Dialogue, 2021.

[7] K Radhika and V Ramana Murthy Oruganti, "Deep Multimodal Fusion for Subject-Independent Stress Detection", 11th International Conference on Cloud Computing Data Science & Engineering, 2021.

[8] Shruti Gedam and Sanchita Paul, "A Review on Mental Stress Detection Using Wearable Sensors and Machine Learning Techniques", IEEE Access, 2021

[9] Anubha Gupta, Deepankar Kansal, Vandana Gupta, Manu Kumar Shetty, M. P. Girish and Mohit D. Gupta, "X-ECGNet: An Interpretable DL model for Stress Detection using ECG in COVID-19 Healthcare Workers", 4th International Conference on Bio-Engineering for Smart Technologies, 2021.

[10] Manuel Gil-Martin, Ruben San-Segundo, Ana Mateos and Javier Ferreiros-Lopez, "Human Stress Detection With Wearable Sensors Using Convolutional Neural Networks", IEEE Aerospace and Electronic Systems Magazine, 2021.

[11] P Manimeghalai, J Sree Sankar, Jayalakshmi P.K, Ranjeesh R Chandran, Sreedeep Krishnan and Selshia Shiny, "ECG Based Stress Detection Using Machine Learning", Second International Conference on Advances in Electrical Computing Communication and Sustainable Technologies, 2022.

[12] Jiho Choi, Jun Seong Lee, Moonwook Ryu, Gyutae Hwang, Gyeongyeon Hwang and Sang Jun Lee, "Attention-LRCN: Long-term Recurrent Convolutional Network for Stress Detection from Photoplethysmography", IEEE International Symposium on Medical Measurements and Applications, 2022.

[13] Suresh Kumar Kanaparthi, P Surekha, Lakshmi Priya Bellamkonda, Bhavya Kadiam and Beulah Mungara, "Detection of Stress in IT

Employees using Machine Learning Technique", International Conference on Applied Artificial Intelligence and Computing, 2022.

[14] Muhammad Amin, Khalil Ullah, Muhammad Asif, Abdul Waheed, Sana Ul Haq, Mahdi Zareei, et al., "ECG-Based Driver's Stress Detection Using Deep Transfer Learning and Fuzzy Logic Approaches", IEEE Access, 2022.

[15] Eda Eren and Tuğba Selcen Navruz, "Stress Detection with Deep Learning Using BVP and EDA Signals", International Congress on Human-Computer Interaction Optimization and Robotic Applications.

[16] S. Rosaline et al., "Predicting Melancholy risk among IT professionals using Modified Deep Learning Neural Network (MDLNN)", 2022 IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT), pp. 385-391, 2022.

[17] G. R et al., "An Effectual Underwater Image Enhancement using Deep Learning Algorithm", 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 1507-1511, 2021.

[18] G. Sarason and B. R. Sarason, "Test anxiety", Handbook of social and evaluation anxiety, pp. 475-495, 1990.

[19] S. H. Tan and J. S. Pang, "Test Anxiety: An Integration of the Test Anxiety and Achievement Motivation Research Traditions", Educational Psychology Review, vol. 35, no. 1, pp. 13, 2022.

[20] S. Caviola, E. Toffalini, D. Giofrè, J. M. Ruiz, D. Szűcs and I. C. Mammarella, "Math performance and academic anxiety forms from sociodemographic to cognitive aspects: A meta-analysis on 906311 participants", Educational Psychology Review, 2022.