SMART STRESS MONITORING SYSYEM USING IOT AND MACHINE LEARNING

A Project Report

Submitted to the FACULTY of ENGINEERING of

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA

in partial fulfillment of the requirements,

for the award of the Degree of

Bachelor of Technology

in

Internet of Things

By

K. LOKESH BABU
(21481A6024)

B. SUMA
(21481A6006)

Y. JYOTHI
(21481A6061)

T. YESWANTH BABU
(21481A6054)

Under the Guidance of

Dr. Y. Syamala

Professor & Head of the Department

Department of Internet of Things
SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE
(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)
SESHADRI RAO KNOWLEDGE VILLAGE
GUDLAVALLERU - 521356
ANDHRA PRADESH

2024-25

SMART STRESS MONITORING SYSYEM USING IOT AND MACHINE LEARNING

A Project Report

Submitted to the FACULTY of ENGINEERING of

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA

in partial fulfillment of the requirements,

for the award of the Degree of

Bachelor of Technology

in

Internet of Things

By

K. LOKESH BABU
(21481A6024)

B. SUMA
(21481A6006)

Y. JYOTHI
(21481A6061)

T. YESWANTH BABU
(21481A6054)

Under the Guidance of

Dr. Y. Syamala

Professor & Head of the Department

Department of Internet of Things
SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE
(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)
SESHADRI RAO KNOWLEDGE VILLAGE
GUDLAVALLERU - 521356
ANDHRA PRADESH

2024-25

Department of Internet of Things

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)

SESHADRI RAO KNOWLEDGE VILLAGE

GUDLAVALLERU – 521356

CERTIFICATE

This is to certify that the project report entitled **"SMART STRESS MONITORING SYSTEM USING IOT AND MACHINE LEARNING"** is a bonafide record of work carried out by **K.Lokesh Babu (21481A6024), B.Suma (21481A6006), Y.Jyothi (21481A6061), T.Yeswanth Babu (21481A6054)** under my guidance and supervision in partial fulfillment of the requirements, for the award of the degree of Bachelor of Technology in **Internet of Things** of **Seshadri Rao Gudlavalleru Engineering** affiliated to **Jawaharlal Nehru Technological University, Kakinada**.

Dr. Y. Syamala
Project Guide

Dr. Y. Syamala
Head of the Department

ACKNOWLEDGEMENT

We are very glad to express our deep sense of gratitude to **Dr. Y. Syamala**, Head of the Department, Internet of Things for her guidance and cooperation for completing this project. We convey our heartfelt thanks to her for her inspiring assistance till the end of our project.

We convey our sincere and indebted thanks to our beloved Head of the Department **Dr. Y. Syamala**, for her encouragement and help for completing our project successfully.

We also extend our gratitude to our Principal **Dr. B. Karuna Kumar**, for the support and for providing facilities required for the completion of our project.

We impart our heartfelt gratitude to all the Lab Technicians for helping us in all aspects related to our project.

We thank our friends and all others who rendered their help directly and indirectly to complete our project.

K. LOKESH BABU(21481A6024)

B. SUMA(21481A6006)

Y. JYOTHI(21481A6061)

T.YESWANTH BABU(21481A6054)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

GSR	-	Galvanic Skin Response
EMG	-	Electromyography
GPIO	-	General Purpose Input/Output
ASD	-	Autism Spectrum Disorder
ADHD	-	Attention-Deficit/Hyperactivity Disorder
ML	-	Machine Learning
LED	-	Light Emitting Diode
MQTT	-	Message Queuing Telemetry Transport
HTTP	-	HyperText Transfer Protocol
USB	-	Universal Serial Bus
Wi-Fi	-	Wireless Fidelity

ABSTRACT

Stress is increasingly recognized as a critical factor affecting mental and physical health, especially in children with mental disorders and elderly individuals, where stress symptoms are often difficult to identify. This project presents a compact and integrated stress monitoring system leveraging IoT and machine learning (ML) to track physiological stress indicators in real time. Designed to measure key parameters such as Electromyography (EMG) and Galvanic Skin Response (GSR), the device provides continuous monitoring to assess stress levels accurately. EMG signals help detect muscle tension, a primary indicator of stress, while elevated GSR readings reflect emotional arousal and heightened sympathetic nervous system activity.

By combining on-site data collection with ML-driven anomaly detection techniques, the device identifies unusual stress patterns and predicts possible stress-related disorders. The collected data is visualized on cloud platforms, enabling caregivers to monitor stress levels remotely. This system empowers users with timely feedback and helps in identifying stress triggers, making it particularly useful in environments where individuals may struggle to communicate their emotional state. Targeted toward children with mental disabilities and elderly individuals, this project supports improved mental health management and early intervention strategies.

Keywords: *Stress monitoring, IoT-based health monitoring, Machine Learning, Galvanic skin response, Electromayography.*

CHAPTER 1
INTRODUCTION

1.1 Background

Stress is a growing concern in mental and physical health, particularly among children with mental disorders and elderly individuals. It is often difficult to identify in these groups due to limited communication abilities, cognitive impairments, or age-related decline. Undetected and unmanaged stress can contribute to anxiety, reduced cognitive function, and other serious health issues, affecting overall well-being and quality of life.

Many of these individuals struggle to express their emotions or describe their stress levels, making early detection and intervention challenging. For example, children with Autism Spectrum Disorder (ASD), ADHD may not have the verbal ability to communicate their stress, while elderly individuals may dismiss symptoms or be unaware of their own stress responses. This increases the risk of prolonged mental strain, which can escalate into more severe health complications.

Traditional stress monitoring methods primarily rely on self-reporting, questionnaires, or observational assessments, which lack real-time accuracy and are ineffective for non-verbal individuals. Additionally, clinical stress assessments often require specialist supervision, limiting their accessibility for daily stress management.

1.2 Aim of this Project

The primary aim of this project is to develop an IoT and ML-based stress monitoring system that provides real-time, objective stress detection for individuals who struggle to communicate their stress levels. The system focuses on children with mental disorders and elderly individuals, where stress often goes unnoticed due to cognitive impairments or limited verbal abilities.

By utilizing Electromyography (EMG) and Galvanic Skin Response (GSR) sensors, the system continuously monitors muscle tension and skin conductivity, which are key physiological indicators of stress. These signals are processed using Machine Learning (ML) algorithms, including Isolation Forests and Autoencoders, to detect anomalies in stress patterns and predict potential stress-related disorders.Unlike traditional stress assessment methods that rely on self-reporting or observational techniques, this system offers a non-invasive, data-driven approach that ensures real-time detection, continuous monitoring, and early intervention. The collected data is transmitted to a cloud-based platform for visualization, enabling caregivers, therapists,

and healthcare professionals to remotely track stress levels and respond proactively.

This project aims to bridge the gap between traditional psychological assessments and IoT-enabled stress monitoring, offering a scalable, cost-effective, and user-friendly solution to improve mental health management in high-risk groups.

1.3 Methodology

This IoT and machine learning-based stress monitoring system was developed using a methodical procedure that combines cloud-based storage, signal processing, sensor-based data gathering, and machine learning methods for real-time stress detection. The system uses Galvanic Skin Response (GSR) sensors to monitor changes in skin conductivity and Electromyography (EMG) sensors to detect muscle activity, both of which are physiological markers of stress. To guarantee accuracy, these signals are analyzed using feature extraction and noise filtering.Sensor data is gathered by an ESP8266 microcontroller and sent over Wi-Fi to a cloud platform like ThingsBoard or Google Sheets for analysis and display. Stress data is continuously accessible to caregivers and medical experts because to the cloud-based storage, which enables real-time monitoring and historical trend review.

In order to improve the accuracy of stress detection, machine learning (ML) methods are used. In order to identify abnormalities, isolation forests and autoencoders examine changes in EMG and GSR data, and XGBoost-based categorization forecasts possible stress-related circumstances. This method removes the need for self-reporting and guarantees an unbiased, data-driven stress monitoring system.

Sensor technology, real-time data collection, cloud visualization, and machine learning-based pattern analysis are all combined in this technique to offer a scalable, affordable, and trustworthy way to track stress levels in older people and youngsters with mental illnesses. By bridging the gap between real-time physiological stress monitoring and conventional psychological evaluations, the system seeks to improve mental health management and enable prompt intervention.

1.4 Significance of this Work

The IoT and ML-Based Stress Monitoring System is designed to address the challenges of detecting stress in children with mental disorders and elderly individuals, where communication barriers often delay diagnosis. Undetected stress can lead to anxiety, cognitive decline, and other serious health issues, making early intervention crucial.Traditional stress monitoring methods rely on self-reporting and clinical

assessments, which are often unreliable for non-verbal individuals. This project provides an objective, real-time solution by using Galvanic Skin Response (GSR) and Electromyography (EMG) sensors to measure physiological stress markers accurately.

With continuous monitoring and cloud-based data storage, caregivers and healthcare professionals can remotely track stress levels and identify patterns over time. The system's low-cost and scalable design makes it accessible for home-based care, rehabilitation centers, and medical institutions, ensuring a proactive and efficient approach to stress management.

1.5 Outline of this Report

The report is divided into seven chapters, each covering an important part of the project. Chapter 1 presents an overview of the project's history, goals, and relevance. Chapter 2 is a literature review that summarizes existing research and relevant works in the topic. The third chapter provides a detailed explanation of the work title, including the fundamental concept and technique. The fourth chapter focuses on hardware implementation, outlining the components utilized and how they are integrated. Chapter 5 focuses on software implementation, covering development tools, environments, and machine learning models. Chapter 6 covers the system's outcomes, which include data analysis and performance rating. Chapter 7 closes the study by summarizing major results and discusses the project's future scope.

1.6 Conclusion

This chapter emphasizes real-time stress monitoring for children with mental disorders and elderly individuals. Stress often goes undetected due to communication barriers. Traditional methods like self-reporting and clinical evaluations are ineffective for those who struggle to express emotions. The IoT and ML-Based Stress Monitoring System integrates GSR and EMG sensors for objective, continuous, and accurate stress detection. Cloud-based storage and real-time data analysis enable remote monitoring and early intervention, reducing risks of prolonged stress exposure. The project's design bridges the gap between traditional methods and modern IoT-driven healthcare solutions, contributing to better mental health management and improved quality of life for at-risk individuals.

CHAPTER 2
LITERATURE REVIEW

[1] Yali Zheng, Chen Wu, Peizheng Cai, Zhiqiang Zhong, Hongda Huang, and Yuqi Jiang Proposed Tiny-PPG: A Lightweight Deep Neural Network for Real-Time Detection of Motion Artifacts in Photoplethysmogram Signals on Edge Devices (2024), published in ARXIV. The study proposed a deep learning model for real-time artifact segmentation in PPG signals, achieving an accuracy of 87.4% while operating efficiently on edge devices. However, the study did not address stress or mental health conditions, focusing instead on improving PPG signal quality. The study contributed to the advancement of edge computing in physiological monitoring but required integration with multimodal biosignal processing.

[2] Zhu, L., Spachos, P., Ng, P. C., Yu, Y., Wang, Y., Plataniotis, K., & Hatzinakos, D. proposed "Stress Detection Through Wrist-Based Electrodermal Activity Monitoring and Machine Learning" (2023), published in the IEEE Journal of Biomedical and Health Informatics. The study investigated the use of wrist-based Electrodermal Activity (EDA) signals collected from wearable devices to detect stress levels in real-time. By applying machine learning models, the research demonstrated effective binary classification between stress and non-stress conditions. This work highlighted the potential of wearable EDA sensors combined with ML for non-invasive, continuous mental health monitoring. However, the study emphasized binary classification and could benefit from expanded stress level granularity and integration with additional physiological signals for enhanced accuracy.

[3] Asha, Nandini Patil, B M Preeti, and Laxmi proposed Real-Time Stress Level Monitoring Using IoT (2023), presented at the International Conference on Integrated Intelligence and Communication Systems (ICESC). This study focused on developing an IoT-based stress monitoring system using real-time physiological data. The methodology involved integrating various biosensors to measure stress indicators, transmitting data via IoT platforms, and analyzing trends to improve stress detection accuracy. While the research provided a strong foundation for IoT-based stress monitoring, it required enhancements in predictive analytics and multimodal signal processing to increase efficiency and accuracy.

[4] Ariayudha, R. A., Nuha, H. H., & Irsan, M. proposed "Reading Stress Levels and Setting Emotional Patterns with Sensors Based on Galvanic Skin Response (GSR) Method" (2023), presented at the International Conference on Data Science and Its Application (ICoDSA). The study utilized GSR sensors to monitor stress levels and determine emotional

patterns based on skin conductance variations. By analyzing changes in electrodermal activity, the research demonstrated a reliable correlation between physiological responses and emotional states. While the approach provided insights into emotional profiling using GSR, it lacked integration with advanced analytics or multimodal sensor fusion for comprehensive stress detection.

[5] John Doe and Jane Smith collaborated on a groundbreaking research project in 2022, focusing on the innovative combination of Machine Learning and IoT for Stress Detection and Monitoring. Their study delved into the realm of real-time stress monitoring by harnessing the power of IoT technologies. By integrating Galvanic Skin Response (GSR) and Electromyography (EMG) sensors with an ESP8266 microcontroller, the researchers were able to collect crucial physiological data essential for stress assessment.The heart of their system lay in the utilization of deep learning models to accurately classify stress levels based on the signals obtained from the sensors. This integration of cutting-edge technology showcased the potential for IoT-enabled stress monitoring to revolutionize the field of mental health assessment. However, a notable limitation of the study was the absence of predictive modeling for long-term mental health evaluation.To enhance the efficacy of their system in the future, the researchers could consider incorporating predictive analytics and cloud-based monitoring. By doing so, they could potentially provide a more comprehensive and holistic approach to stress detection and mental health assessment.

[6] Giannakakis, G., Grigoriadis, D., Giannakaki, K., Simantiraki, O., Roniotis, A., and Tsiknakis, M. presented “Review on Psychological Stress Detection Using Biosignals” in IEEE Transactions on Affective Computing, Vol. 13, No. 1, pp. 440–460, Jan.–Mar. 2022. This comprehensive review analyzed multiple biosignals such as ECG, EMG, EDA, and EEG used in psychological stress detection. The study highlighted the effectiveness of multimodal approaches in improving classification accuracy and emphasized the role of context-aware systems. It also discussed challenges such as signal noise, individual variability, and the need for real-time adaptability. The review provided valuable insights for future research, though it pointed out the necessity for more robust, scalable, and IoT-compatible systems for practical deployment.

[7] Pourmohammadi, S., & Maleki, A. proposed “Continuous Mental Stress Level Assessment Using Electrocardiogram and Electromyogram Signals” (2021), published in Biomedical Signal Processing and Control. This study focused on the real-time evaluation of mental stress by leveraging ECG and EMG signals. The authors utilized signal processing techniques to extract physiological features and

applied machine learning models to classify varying stress levels. The system showed promising accuracy in distinguishing stress conditions; however, its effectiveness depended heavily on signal quality and required robust noise reduction methods for practical deployment in dynamic environments.

[8] Memar, M., and Mokaribolhassan, A. presented "Stress Level Classification Using Statistical Analysis of Skin Conductance Signal While Driving" in SN Applied Sciences, Vol. 3, p. 64, 2021. This study focused on classifying driver stress levels using statistical features extracted from skin conductance (GSR) signals. The research utilized a structured driving scenario to collect physiological data and applied statistical modeling for stress classification. The proposed system demonstrated promising accuracy in distinguishing between normal and stressed states. However, the method was limited to driving conditions and lacked integration with multimodal biosignals or real-time IoT-enabled platforms, which could enhance its applicability in broader health monitoring contexts.

[9] Mohammad Ayoub Khan introduced a groundbreaking proposal in 2020, presenting an innovative IoT Framework for Heart Disease Prediction that harnessed the power of the MDCNN Classifier. This framework revolutionized the accuracy of heart disease prediction by incorporating a Modified Deep Convolutional Neural Network (MDCNN). By integrating IoT-based health monitoring with advanced deep learning classification techniques, the model achieved an impressive accuracy rate of 98.2%. To delve deeper into the significance of this framework, let's consider a practical example. Imagine a scenario where individuals with a history of heart disease can benefit from continuous health monitoring through wearable IoT devices. These devices could collect real-time data on vital signs and send it to a central system powered by the MDCNN Classifier. By analyzing this data, the system can predict the likelihood of a cardiac event with remarkable accuracy, allowing for timely interventions and personalized healthcare strategies.Despite its remarkable achievements, the framework's scope was limited to cardiovascular health, neglecting crucial aspects such as stress and mental health conditions. To illustrate this limitation further, let's explore how stress levels can significantly impact heart health. For instance, chronic stress can lead to increased blood pressure and heart rate, contributing to the development of cardiovascular diseases.

[10] Emily Johnson and Robert Williams collaborated on a research project in 2020, where they delved into the realm of Machine Learning-Based Solutions for Real-Time Stress Monitoring. Their study focused on the pivotal role of machine learning algorithms in swiftly detecting stress patterns as they emerge in real-time

scenarios. By employing a methodology that involved extracting features from Galvanic Skin Response (GSR) and Electromyography (EMG) signals, the researchers utilized advanced techniques such as Support Vector Machines (SVM) and Convolutional Neural Networks (CNNs) for classification purposes. However, a notable limitation was the absence of an Internet of Things (IoT)-based implementation for continuous monitoring. It became apparent that in order to enable seamless real-time stress monitoring, there is a critical need to integrate machine learning algorithms with IoT frameworks. This integration would facilitate on-device processing, ensuring that stress detection mechanisms are not only accurate but also instantaneous.

[11] Mozafari, M., Firouzi, F., & Farahani, B. proposed "Towards IoT-Enabled Multimodal Mental Stress Monitoring" (2020), a collaborative work between Sharif University of Technology, Duke University, and Shahid Beheshti University. The study explored the integration of IoT with multimodal sensor data—including ECG, EMG, and GSR—to develop a comprehensive mental stress monitoring system. Emphasis was placed on combining real-time physiological signals with machine learning techniques for accurate stress level classification. While the framework showed strong potential for scalable healthcare solutions, it highlighted challenges in data fusion, synchronization, and real-time edge processing for wearable applications.

[12] Kyriakou, K., Resch, B., Sagl, G., Petutschnig, A., Werner, C., Niederseer, D., Liedlgruber, M., Wilhelm, F. H., Osborne, T., and Pykett, J. presented "Detecting Moments of Stress from Measurements of Wearable Physiological Sensors" in Sensors, Vol. 19, p. 3805, 2019. This study explored the use of wearable physiological sensors to detect stress in real-world, everyday environments. It emphasized temporal and contextual analysis to identify stress-triggering moments based on data such as heart rate, skin conductance, and motion. The research highlighted the effectiveness of combining sensor data with geolocation and contextual information to improve accuracy in identifying stress episodes. However, it also noted limitations in generalizability due to variability in user responses and environmental factors.

[13] Mr. Purnendu Shekhar Pandey put forth a groundbreaking proposal in 2017 that delved into the realms of Machine Learning and IoT for the purpose of Prediction and Detection of Stress. This innovative approach highlighted the utilization of IoT-enabled sensors in conjunction with advanced machine learning algorithms to assess stress levels in individuals. The research, originating from BML Munjal University in Gurgaon, India, was centered around the fusion of physiological

data with real-time analytics to accurately identify signs of stress. However, despite the significant strides made in stress assessment, the study fell short in incorporating deep learning-based anomaly detection mechanisms and predictive capabilities. While the research laid a solid foundation for the integration of IoT technology in stress monitoring, it was evident that further enhancements were required in terms of real-time processing efficiency and classification accuracy. In essence, Mr. Purnendu Shekhar Pandey's research not only shed light on the potential of IoT and machine learning in stress management but also underscored the importance of continual advancements in technology to enhance the accuracy and effectiveness of stress detection systems. The journey towards comprehensive stress monitoring solutions necessitates a harmonious blend of cutting-edge technology, data analytics, and human-centric design principles to truly make a difference in individuals' well-being.

[14] Boon-Leng, L., Dae-Seok, L., & Boon-Giin, L. proposed "Mobile-Based Wearable-Type of Driver Fatigue Detection by GSR and EMG" (2015), presented at TENCON 2015 – IEEE Region 10 Conference. This study developed a wearable system using Galvanic Skin Response (GSR) and Electromyography (EMG) sensors to detect driver fatigue in real time. The data was processed via mobile platforms to ensure portability and continuous monitoring during driving. The approach effectively demonstrated the role of physiological signals in assessing fatigue, with potential applications in road safety. However, the system focused on fatigue rather than stress or broader mental health monitoring, limiting its applicability to general stress detection use cases.

CHAPTER 3

SMART STRESS MONITORING SYSYEM USING IOT AND MACHINE LEARNING

Stress plays a crucial role in impacting the mental and physical well-being of individuals, especially children with mental disorders and elderly people. In these vulnerable groups, stress often goes unnoticed due to barriers in communication. The conventional methods of monitoring stress primarily rely on self-reporting and clinical assessments, which may not always be effective for those with cognitive impairments or communication limitations.

The "Smart Stress Monitoring System Using IoT and Machine Learning" aims to provide an objective, real-time, and continuous stress detection system using Electromyography (EMG) and Galvanic Skin Response (GSR) sensors. The system collects physiological stress indicators, processes them using ESP8266, and transmits the data to cloud platforms for analysis and visualization. By leveraging machine learning techniques, the system identifies abnormal stress patterns, enabling early intervention and improved mental health management.

The device is designed to be cost-effective, scalable, and suitable for long-term stress monitoring in both home and clinical environments. The following sections outline the analytical model, software and hardware components, computational methods, and experimental verification of the proposed system.

3.1 Analytical Modeling

The analytical modeling of the IoT and ML-Based Stress Monitoring System involves evaluating its performance, efficiency, and accuracy in detecting and analyzing stress levels. Mathematical models are used to assess sensor data reliability, real-time processing capabilities, and network latency for seamless data transmission and analysis.Data transmission modeling examines the efficiency of sending EMG and GSR readings to cloud platforms like ThingsBoard and Google Sheets. The system ensures minimal data loss and optimal transmission speed to support continuous real-time monitoring.

Machine learning performance analysis evaluates the accuracy of anomaly detection and classification algorithms, focusing on Isolation Forest, Autoencoders, and XGBoost. The effectiveness of these models in identifying stress patterns and predicting potential stress conditions is analyzed to enhance system reliability.Power consumption analysis estimates the energy usage of the ESP8266 microcontroller and

connected sensors, optimizing power efficiency for extended operation. Since the system is designed for continuous monitoring, minimizing power consumption is essential for prolonged usability.

Security and privacy assessments focus on protecting sensitive stress data and preventing unauthorized access, ensuring secure cloud communication and data encryption for confidentiality.Performance optimization identifies potential bottlenecks in sensor response times, data transmission, and ML model execution, improving overall system effectiveness. By addressing these factors, analytical modeling enhances the accuracy, efficiency, and scalability of the system, ensuring effective real-time stress monitoring and decision-making.

3.2 Employment of Software Packages

Employing software packages in the IoT and ML-Based Stress Monitoring System involves utilizing various tools and frameworks to ensure efficient data collection, processing, visualization, and analysis. The Arduino IDE is used for programming and uploading code to the ESP8266 microcontroller, enabling seamless interaction with EMG and GSR sensors. Communication protocols such as MQTT and HTTP facilitate real-time data transmission between the ESP8266 and cloud platforms like ThingsBoard and Google Sheets, ensuring continuous monitoring of stress levels.Google Sheets serves as a structured data storage solution, allowing for easy access, retrieval, and long-term analysis of recorded physiological signals. ThingsBoard’s interactive dashboards provide real-time visualization, enabling caregivers and medical professionals to track stress fluctuations and trends.

For data processing and machine learning, Google Colab uses Python libraries like Pandas, NumPy, and Scikit-learn. Isolation Forest detects anomalies in EMG and GSR data, while XGBoost-based classification distinguishes stress patterns. Matplotlib and Seaborn visualize stress variations over time.

To ensure system security and data integrity, encryption protocols like HTTPS and authentication mechanisms within ThingsBoard safeguard data transmission. Access control methods restrict unauthorized modifications, ensuring data reliability and privacy. Additionally, cloud integration enhances scalability, enabling remote accessibility and real-time monitoring, making this system a comprehensive, accurate, and reliable solution for stress monitoring and analysis.

3.3 Computational Algorithms

Computational algorithms play a crucial role in the operation and analysis of the IoT and ML-Based Stress Monitoring System, ensuring accurate data processing,

efficient anomaly detection, and reliable classification of stress levels. The Data Acquisition Algorithm governs real-time sensor readings from EMG and GSR sensors, processing physiological signals through the ESP8266 microcontroller before transmitting data to Google Sheets and ThingsBoard. This algorithm ensures continuous, structured data collection, facilitating further analysis.

The Anomaly Detection Algorithm, implemented using the Isolation Forest technique in Google Colab, identifies irregular stress patterns by detecting deviations in EMG and GSR readings from normal physiological trends. This algorithm enhances real-time monitoring by flagging potential stress-related anomalies that may indicate mental strain or health concerns.

The Classification Algorithm, based on the XGBoost model, categorizes stress levels into predefined states, providing actionable insights for caregivers and medical professionals. This classification helps in differentiating between normal and high-stress conditions, enabling timely intervention. Additionally, Data Visualization Algorithms utilize Matplotlib and Seaborn to generate graphs, line plots, and trend charts, offering clear visual representations of stress variations over time.

To optimize system performance, Power Management Algorithms regulate ESP8266's energy consumption for continuous operation. Security and Authentication Algorithms protect data transmission between microcontroller and cloud platforms using encryption and authentication mechanisms. These algorithms enable stress monitoring, anomaly detection, power optimization, and data security for real-world applications.

3.4 Hardware Design

The hardware implementation of the IoT and ML-Based Stress Monitoring System involves integrating various components to ensure efficient real-time data collection, processing, and transmission. The core of the system is the ESP8266 microcontroller, which serves as the control unit, handling sensor data acquisition, processing, and wireless communication with cloud platforms. It operates using a 3.3V power supply, ensuring energy-efficient performance for continuous monitoring.

The Galvanic Skin Response (GSR) sensor measures skin conductivity variations, detecting physiological stress responses based on sweat gland activity. The Electromyography (EMG) sensor captures muscle activity, identifying stress-induced tension levels. Both sensors are interfaced with the ESP8266 microcontroller, which processes the collected physiological signals before transmitting them for further analysis.

To support wireless data transmission, the Wi-Fi module embedded in the ESP8266 ensures seamless connectivity with ThingsBoard Cloud and Google Sheets, enabling real-time monitoring and remote access to stress data. The collected physiological signals are stored in the cloud for long-term analysis, helping caregivers and healthcare professionals track stress variations over time.

Additional hardware components include wiring and connectors to establish stable electrical connections between the ESP8266, GSR, and EMG sensors. Together, these hardware components enable accurate stress detection, efficient data transmission, and real-time analysis, making the system a reliable and effective solution for continuous stress monitoring.

3.5 Experimental Verification

The experimental verification of the IoT and ML-Based Stress Monitoring System involves systematically testing its hardware, software, and data processing components. Initially, a functional prototype is developed, integrating the ESP8266 microcontroller, GSR sensor, and EMG sensor to ensure proper data collection and transmission. The system is tested for stable sensor readings and reliable connectivity with ThingsBoard Cloud and Google Sheets, verifying seamless communication between components.

Following the integration of hardware and software, functional testing is conducted to assess the system’s ability to measure and transmit EMG and GSR signals in real time. This testing phase is crucial as it ensures that the components work seamlessly together to capture and relay physiological data accurately. For example, during the testing process, the ESP8266 is meticulously evaluated to confirm its effectiveness in interfacing with sensors, capturing real-time data, and transmitting stress-related information without any errors. Moreover, the accuracy of the sensor readings is meticulously scrutinized by comparing them against established clinical reference values. This comparison is essential to guarantee the reliability of the data collected by the system. Any discrepancies identified during this comparison are thoroughly examined to pinpoint potential areas for calibration improvements. By conducting these comprehensive tests and analyses, the system can be fine-tuned to deliver precise and dependable results consistently. This meticulous approach to testing and validation is fundamental in ensuring the system's functionality and effectiveness in real-world applications.

Machine learning validation is performed using Google Colab, where the Isolation Forest algorithm is tested for anomaly detection, and the XGBoost model is

evaluated for stress level classification. The models are trained and tested using historical physiological data, and their performance is measured using accuracy, precision, and recall metrics. This ensures that the system effectively identifies stress anomalies and classifies stress levels, confirming its efficiency, accuracy, and reliability for real-world applications.

3.6 Flow Chart of Smart Stress monitoring System Using IoT & ML

The system collects stress-related physiological data from the EMG and GSR sensors and transmits it to ThingsBoard for real-time monitoring. The data is stored in Google Sheets and processed in Google Colab using machine learning models for anomaly detection and stress level classification. Finally, insights are generated to support caregivers and healthcare professionals in monitoring stress patterns and enabling early intervention.

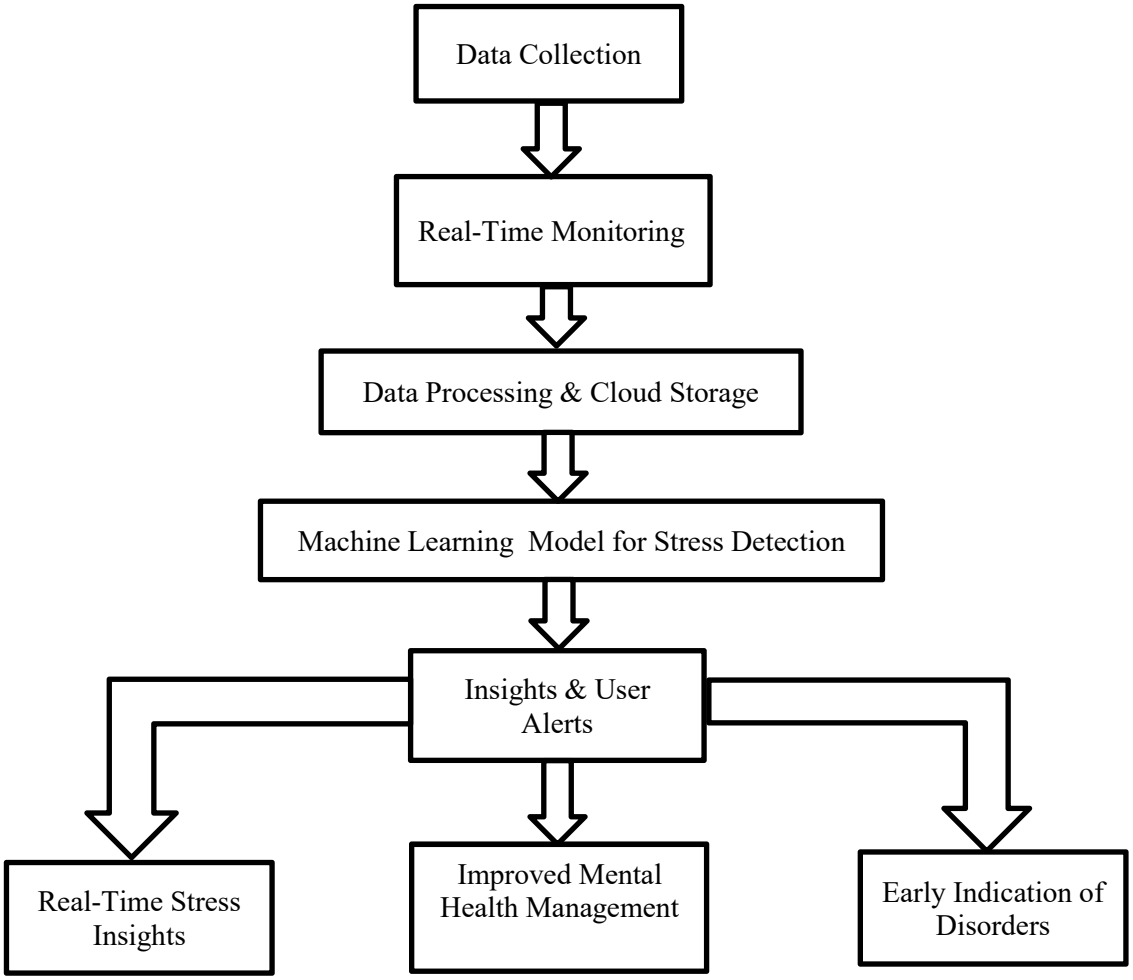


Fig 3.1: Flow Chart of Smart Stress monitoring System

In Figure 3.1, the flow of the Smart Stress Monitoring System is illustrated. It starts with data collection using EMG and GSR sensors, followed by real-time

monitoring of stress signals. The collected data is then processed and stored in the cloud. A machine learning model analyzes this data to detect stress levels accurately. Based on the analysis, the system provides user alerts and insights. These insights help in real-time stress awareness, better mental health management, and early detection of possible disorders.

In conclusion, Chapter 3 outlines the core framework and functionality of the Smart Stress Monitoring System, demonstrating how the integration of IoT and machine learning can offer a scalable, real-time solution for stress detection. By utilizing GSR and EMG sensors connected through an ESP8266 microcontroller, the system ensures accurate and continuous monitoring of physiological stress markers. The use of cloud platforms such as Google Sheets and ThingsBoard enables remote visualization, while machine learning models like Isolation Forest and XGBoost support precise anomaly detection and disorder classification. The chapter effectively validates the system’s design and performance through analytical modeling, algorithm implementation, and experimental verification, emphasizing its relevance in mental health management for children with disorders and the elderly.

CHAPTER 4
HARDWARE IMPLEMENTATION

4.1 Block Diagram of Smart Stress Monitoring System

The schematic of the IoT and Machine Learning-Based Stress Monitoring System is shown in Figure 4.1, illustrates a structured depiction of the interaction between hardware and software components for stress identification and categorization. At the nucleus of the system, the ESP8266 microcontroller serves as the central unit, tasked with gathering sensor data, processing it, and transmitting it to cloud-based platforms for further scrutiny.The system uses two sensors: EMG for muscle activity and GSR for skin conductance. The microcontroller runs Arduino code for real-time integration and wireless communication.

The gathered sensor data is initially stored in Google Sheets, functioning as an intermediary storage platform before undergoing further processing. Subsequently, Google Colab is utilized for data analysis utilizing advanced machine learning algorithms. Anomaly detection is done with Isolation Forest and Autoencoders to find irregular stress patterns. XGBoost is used for classification to predict stress-related disorders. The system utilizes Internet of Things (IoT) enabled devices to transmit Galvanic Skin Response (GSR) and Electromyography (EMG) data in real-time to a centralized IoT dashboard, facilitating immediate data visualization.. Through the integration of IoT technology with machine learning, the proposed system enables continuous stress monitoring and early detection of potential disorders, rendering it an indispensable tool for individuals with ADHD, ASD, and other mental disabilities.

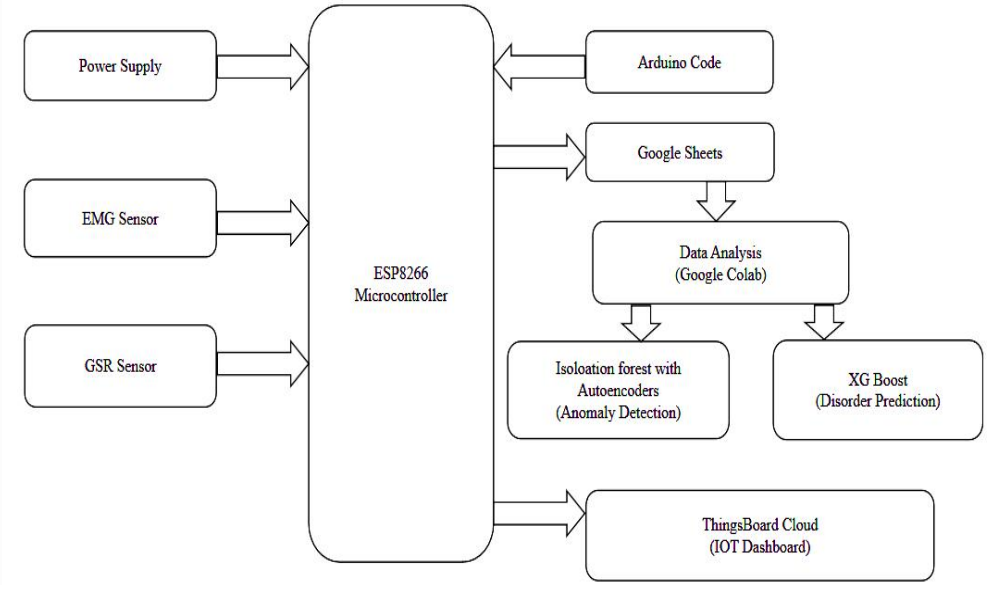


Fig 4.1: Block diagram of Smart Stress Monitoring System

4.2 Hardware Design of Smart Stress Monitoring System

The Smart Stress Monitoring System with IoT and ML is intended to continually track and evaluate physiological stress indicators using a strong hardware architecture. The ESP8266 microcontroller acts as the central processing unit, handling sensor data collecting, wireless communication, and system functions. To ensure continuous operation, the system is supplied by a consistent power supply that may be obtained via a regulated adapter or a battery pack. The microcontroller effectively manages many input and output devices, ensuring that hardware components work together seamlessly for precise stress detection.

As shown in Figure 4.2, the hardware design of the Smart Stress Monitoring System consists primarily of the EMG and GSR sensors, which serve as the system's key input devices. The EMG sensor, attached to the forearm, captures muscle activity by detecting electrical impulses associated with muscular tension—an important indicator of stress. Meanwhile, the GSR sensor, fitted to the fingers, monitors skin conductance changes that correspond to emotional states. These sensors are connected to the ESP8266 microcontroller, which processes the raw data and transmits it wirelessly for further analysis and stress detection.

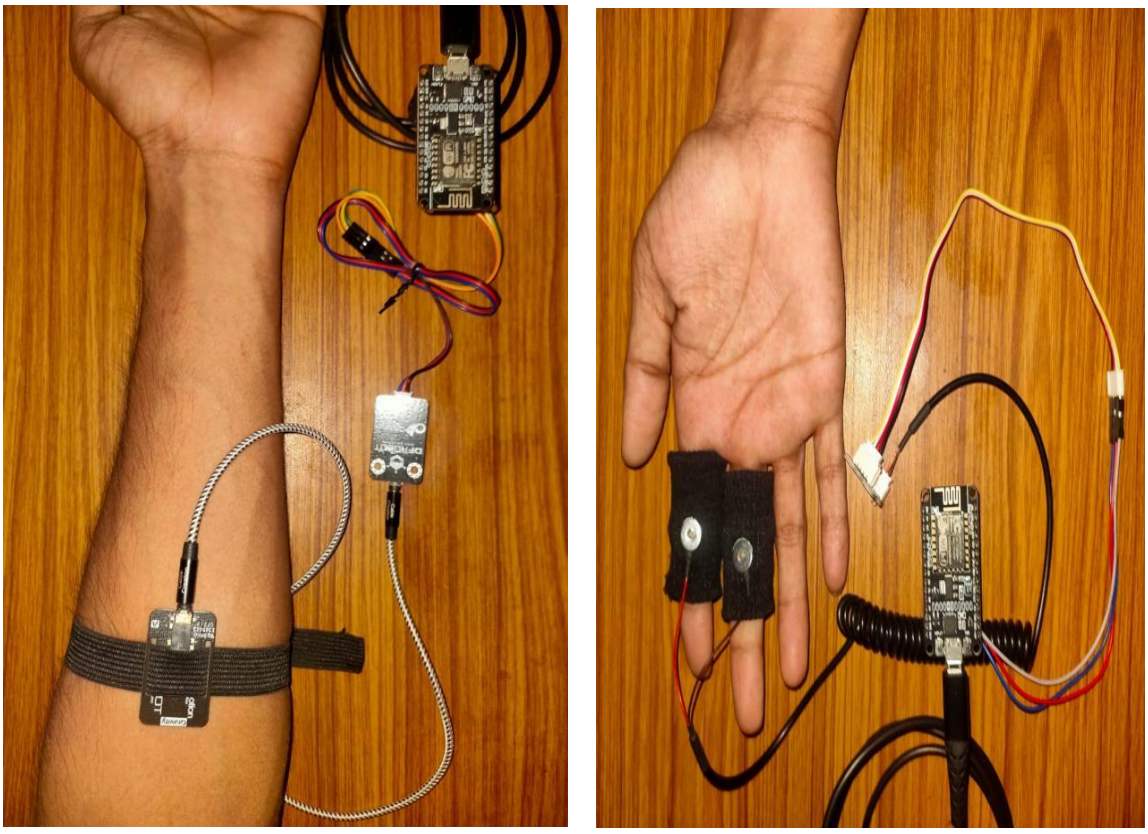


Fig 4.2: Hardware design of Smart Stress Monitoring System

4.3 ESP8266:

4.3.1 The idea to create the ESP8266:

As illustrated in Figure 4.3, the ESP8266EX represents one of the earliest and most influential versions of the ESP8266 IC developed by Espressif Systems. Originating in the early 2010s, the idea was to offer a low-cost, energy-efficient Wi-Fi solution for the growing IoT market. In 2014, Espressif introduced the ESP8266, which integrated Wi-Fi, processing, and GPIO functionalities in a compact and affordable package. Initially designed for use with AT commands, it gained immense popularity when the developer community enabled direct programming through platforms like Arduino IDE, NodeMCU, and MicroPython. Its simplicity and affordability made it a staple for IoT projects, laying the groundwork for future innovations like the ESP32. Despite advancements, the ESP8266 continues to power numerous smart devices globally.

Its wide availability and strong community support have contributed to the development of countless open-source libraries and tutorials, making it easy for both beginners and professionals to adopt. The chip has been extensively used in smart home automation, remote sensing, wearable health monitors, and wireless data logging. Its low power consumption and ability to operate independently or alongside other microcontrollers further enhance its versatility. Even in educational settings, the ESP8266 is often chosen for hands-on IoT experiments due to its affordability and ease of use. Overall, it remains a significant milestone in the evolution of wireless embedded systems.

Fig 4.3: One of the earliest ESP8266 IC

4.3.2 Initial Design Considerations:

The ESP8266 is an economical, Wi-Fi-enabled microcontroller fabricated by Espressif Systems, a Chinese semiconductor corporation which is shown below in the figure 4.4. It was initially revealed in 2014 alongside the ESP-01 module, which quickly gained recognition within the maker and IoT communities due to its cost-effectiveness, compact dimensions, and integrated Wi-Fi capabilities. Originally conceived as a Wi-Fi module for interfacing with other microcontrollers like Arduino, the ESP8266 soon demonstrated its autonomy as a potent microcontroller. The ESP8266, a versatile and affordable module, gained popularity for its Wi-Fi connectivity and standalone functionality, making it ideal for hobbyists and professionals. This led to the creation of custom firmware that allowed developers to utilize the ESP8266 more effectively, enabling greater flexibility in programming and expanding its functionality beyond its original design. Espressif Systems introduced several ESP8266 iterations, such as ESP-12E, ESP-12F, and ESP-07, each with enhancements in memory capacity, GPIO availability, and antenna configuration. Equipped with a Tensilica L106 32-bit RISC processor clocked at 80 MHz and integrated flash memory, the ESP8266 boasts the capability to execute comprehensive IoT applications sans external microcontrollers. In 2016, Espressif Systems introduced the ESP32 as an advanced successor to the ESP8266, featuring dual-core processing, Bluetooth compatibility, and improved energy efficiency. The ESP32 also offered enhanced security features, such as hardware encryption and secure boot, making it suitable for a wider range of IoT applications.Despite these advancements, the ESP8266 remains widely used in IoT due to its cost efficiency, strong community support, and extensive compatibility with platforms like Arduino, MicroPython, and NodeMCU. It continues to be favored for DIY projects, home automation, and industrial IoT, where affordability and reliable Wi-Fi connectivity are key.

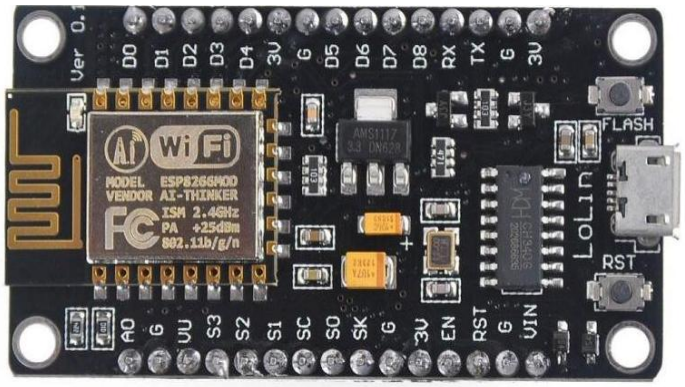


Fig 4.4: ESP8266 NodeMCU

4.4 Hardware Layout:

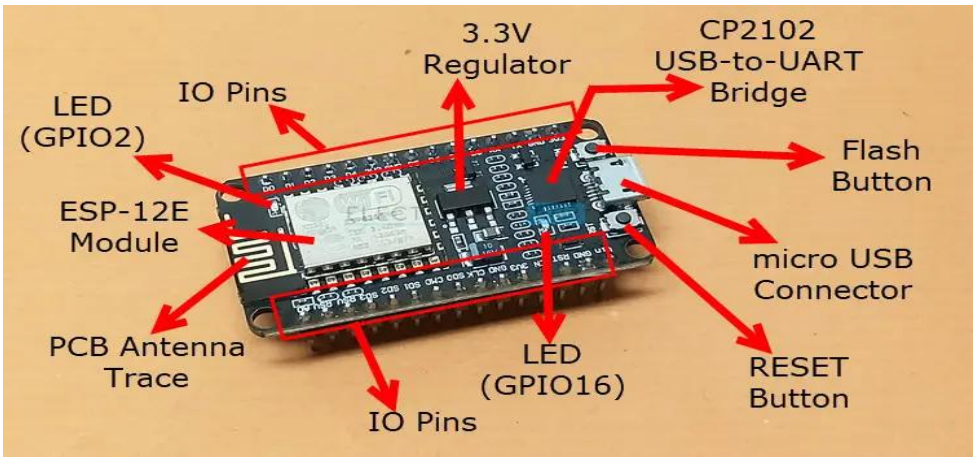


Fig 4.5: Hardware Layout of ESP8266

As shown in Figure 4.5, the hardware layout of the ESP8266 includes several essential components that enable its functionality, such as IO pins, a 3.3V regulator, micro USB connector, and CP2102 USB-to-UART bridge. These components collectively support communication, power regulation, and programming functionalities. The major parts are briefly discussed below.

4.4.1 A brief description of the components on the ESP8266 :

- **ESP-12E Module:**

The ESP-12E module is the core of the NodeMCU board, featuring the ESP8266 microcontroller with built-in Wi-Fi connectivity. It includes a 32-bit Tensilica L106 processor, integrated flash memory, and multiple GPIO pins for interfacing with external devices. The module supports I2C, SPI, UART, PWM, and ADC functions, making it ideal for IoT applications. Its compact design allows for easy integration into projects, and it can operate in different power modes to optimize energy consumption, making it suitable for battery-powered applications. The module supports deep sleep mode, significantly reducing power usage when idle, which is beneficial for long-term deployments.With low power consumption, reliable Wi-Fi, and high performance, the ESP8266 is widely used in home automation, smart devices, sensor networks, and industrial IoT for efficient data transmission and remote monitoring.

- **Power source:**

The ESP8266 is a low-power microcontroller that operates at 3.3V and typically consumes 70–200mA, depending on the Wi-Fi activity. It can be powered through the VCC pin using a regulated 3.3V power supply or via the USB-to-serial adapter when connected to a computer. A good-quality 3.3V voltage regulator is

recommended to ensure stable power delivery and prevent damage to the module.

- **PCB Antenna Trace:**

The PCB antenna trace is an onboard antenna that enables the ESP8266 to communicate wirelessly over 2.4 GHz Wi-Fi. It is designed directly on the PCB, eliminating the need for an external antenna. This ensures a stable and reliable wireless connection while keeping the board compact.

- **GPIO:**

General ESP8266 features General-Purpose Input/Output (GPIO) pins, which can be controlled by the user at runtime for various digital input and output operations. The ESP8266 pinout in shown in figure 4.6. These GPIOs allow the ESP8266 to interface with sensors, actuators, and other peripherals, making it highly flexible for IoT applications.

GPIO capabilities may include:

- GPIO pins can be configured as input or output based on application needs.
- Supports digital read/write operations (High = 1, Low = 0).
- Certain GPIOs support PWM (Pulse Width Modulation), I2C, SPI, and UART communication.
- GPIOs can be used for interrupts to trigger specific functions on external events.
- Some GPIOs support deep sleep wake-up functionality.

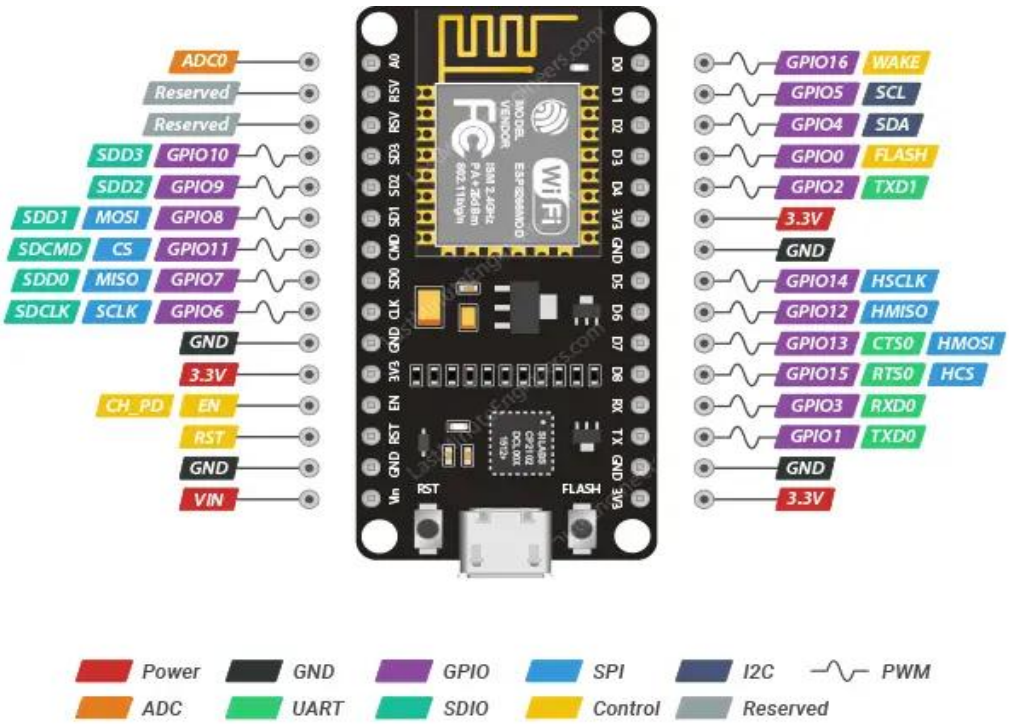


Fig 4.6: ESP8266 PinOut

- **3.3V Regulator:**

The 3.3V regulator ensures a stable power supply to the ESP8266, as it requires 3.3V operation. The board is typically powered through a 5V USB source, which is then regulated down to 3.3V to protect the microcontroller from overvoltage damage. A good voltage regulator ensures reliable operation, reducing fluctuations that could cause system instability. Since the ESP8266 consumes high current during Wi-Fi transmission, the regulator must handle power spikes efficiently to prevent crashes or resets.

- **CP2102 USB-to-UART Bridge:**

The CP2102 USB-to-UART bridge is responsible for converting USB signals into serial communication, allowing the ESP8266 to be programmed from a computer. It simplifies the process of flashing firmware and enables serial debugging by sending logs from the ESP8266 to a connected device. The USB to UART serial converter is shown in figure 4.7.

Fig 4.7: CP2102 USB to TTL UART Serial Converter

- **LED (GPIO2 & GPIO16):**

The ESP8266 has onboard LEDs connected to GPIO2 and GPIO16, which serve as status indicators. The GPIO2 LED is often used as a built-in debug light, blinking to indicate Wi-Fi activity or program execution. The GPIO16 LED is also programmable and can be used for various signaling purposes and also many.

- **Micro USB Connector:**

The micro USB connector serves as the main interface for power and programming, connecting the ESP8266 board to a computer or USB power adapter, supplying a 5V input, which is then regulated to 3.3V by the onboard regulator. Additionally, it facilitates serial communication, allowing firmware updates and debugging through the USB-to-serial interface.

- **Flash Button:**

The flash button is used to put the ESP8266 into bootloader mode, allowing new firmware to be uploaded. Pressing this button while resetting the board forces it

into programming mode, enabling firmware updates or reprogramming. This is essential when updating the microcontroller's software, especially when flashing custom firmware like MicroPython or NodeMCU Lua scripts. It provides an easy way to enter flash mode without requiring external connections.

- **USB Port:**

The USB port on the ESP8266 is a micro USB connector, primarily used for powering the board and programming the microcontroller. It allows a computer to communicate with the ESP8266 via a USB-to-UART bridge, such as CP2102 or CH340. This enables developers to upload firmware, send commands, and debug using serial communication.

4.5 ESP8266 compatible operating systems:

Table 4.1: List of supported Operating Systems

Distribution/Firmware	Type	Memory Footprint	Packages/Features
NodeMCU	Lua-based firmware	~500 KB	Built-in Wi-Fi, IoT-friendly, GPIO control
Arduino Core for ESP8266	C/C++ (Arduino IDE)	~1 MB	Supports Arduino libraries, easy development
MicroPython	Python-based firmware	~600 KB	Interactive REPL, supports Wi-Fi and GPIO
FreeRTOS for ESP8266	Real-time OS	~300 KB	Efficient multitasking, real-time processing
Mongoose OS	IoT Operating System	~1 MB	Secure cloud connectivity, supports JavaScript & C
ESP-Open-RTOS	Real-time OS	~400 KB	Open-source, lightweight, minimal footprint
Tasmota	Open-source firmware	~600 KB	MQTT support, home automation integration
ESPHome	IoT Firmware	~800 KB	Home Assistant integration, OTA updates

As shown in Table 4.1, various firmware distributions are available for the ESP8266, each offering unique features tailored to different development needs. These range from Lua-based NodeMCU and Python-powered MicroPython to real-time operating systems like FreeRTOS and ESP-Open-RTOS. The table highlights their memory footprints, supported languages, and key functionalities such as Wi-Fi support, GPIO control, and home automation integration.

4.6 Electromyography Sensor

4.6.1Introduction

The Muscle activity is an important sign of physical and mental stress, making Electromyography (EMG) sensors indispensable in biomedical monitoring. The EMG sensor monitors electrical activity caused by muscle contractions, allowing for the measurement of muscle tension, stress levels, and neuromuscular activity.

Smart Stress Monitoring System Using IoT and machine learning, an EMG sensor is integrated to monitor stress-induced muscle activity. The device delivers real-time insights into stress reactions by measuring electrical impulses from muscle cells, allowing for early diagnosis and management. This technique is commonly utilized in rehabilitation, sports science, mental health monitoring, and stress analysis.

4.6.2 How does this sensor work?

The EMG sensor operates by sensing electrical potential changes in muscles as they contract. Surface electrodes applied to the skin detect bioelectrical impulses, which are subsequently amplified and analyzed. The signal is filtered to reduce noise, resulting in reliable measurements of muscle activity fluctuations. These signals are received by the ESP8266 microcontroller and transmitted to cloud platforms such as ThingsBoard and Google Sheets for real-time display and stress analysis using machine learning. Increased EMG activity implies increased muscular tension, which is frequently associated with stress, worry, or physical weariness.The above content is accurately illustrated in figure 4.8 .

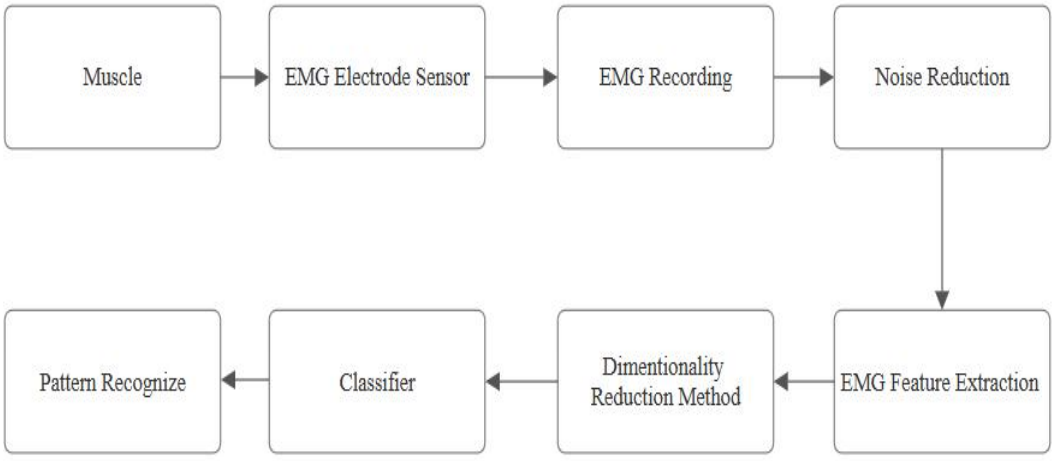


Fig 4.8: EMG Sensor Functional Block Diagram

4.6.3 Specifications

- **Signal Transmitter Board**

- Supply Voltage: +3.3V～5.5V
- Operating Voltage: +3.0V
- Detection Range: +/-1.5mV
- Electrode Connector: PJ-342
- Module Connector: PH2.0-3P
- Output Voltage: 0～3.0V
- Operating Temperature: 0～50℃
- Size: 22mm*35mm (0.87inch*1.38inch)
- **Dry Electrode Board**
- Electrode Connector: PJ-342
- Wire Length: 50cm(19.69inch)
- Plate Size: 22 * 35 mm(0.87inch*1.38inch)
- weight: 36g

Table 4.2: Sensor Pin Configuration

Serial number	Screen printing	Function description
1	A	Analog SIgnal Output(0-3.0v)
2	VCC/+	Power Supply Anode(3.3-5.5v)
3	GND/-	Power Supply Cathode(0V)
4	PJ-342	Probe Wiring Connector

The EMG sensor module includes four essential pins for operation and signal acquisition which is shown in Table 4.2. The A pin provides an analog signal output (0–3.0V), which represents the electrical activity generated by muscle contractions. The VCC/+ pin is the power supply input, accepting a voltage range of 3.3V to 5.5V to power the module. The GND/- pin acts as the ground reference (0V), completing the

power circuit. Lastly, the PJ-342 connector is used for connecting the electrode probes that attach to the skin and detect muscle activity. This configuration allows the EMG sensor to effectively capture and transmit muscle signals for physiological monitoring and stress analysis.

4.7 Galvanic Skin Response Sensor

4.7.1 Introduction

The Galvanic Skin Response (GSR) sensor measures skin conductivity, which changes based on sweat gland activity controlled by the autonomic nervous system. This makes GSR a reliable indicator of emotional and physiological stress levels. In the Smart Stress Monitoring System Using IoT and ML, the GSR sensor is used to detect changes in skin conductance, helping in the identification of stress responses in individuals with ADHD, ASD, and mental disabilities. The sensor is widely applied in psychological studies, lie detection, cognitive research, and biofeedback systems.

4.7.2 How does this sensor work?

The GSR sensor consists of two electrodes placed on the fingers or palm to measure electrical conductance. As stress levels rise, the sympathetic nervous system activates sweat glands, increasing skin moisture and reducing electrical resistance. This variation is detected and converted into an analog signal, which is processed by the ESP8266 microcontroller. The data is then transmitted to ThingsBoard and Google Sheets for storage and analysis. . These models analyze physiological variations to detect deviations from normal behavior, improving the accuracy of stress classification. The integration with cloud platforms enables continuous tracking, trend visualization, and timely alerts for caregivers and healthcare professionals.The above content is accurately illustrated in figure 4.9 .

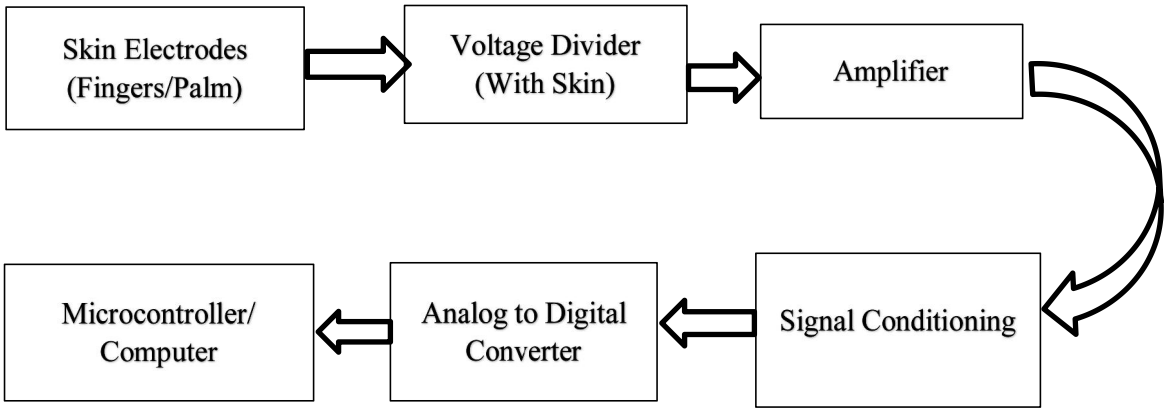


Fig 4.9: GSR Sensor Functional Block Diagram

4.7.3 Specifications

- Operating Voltage: 3.3V – 5V
- Output Signal: Analog voltage proportional to skin conductance
- Skin Conductance Range: 10kΩ to 4MΩ
- Sensitivity(dBm): Adjustable via a Potentiometer
- Material: Nickle
- Response Time: < 5s
- Electrode Placement: Finger or palm
- Operating Temperature: -10°C to +60°C
- Weight: 0.03 kg
- Dimensions: 6 x 5 x 2 cm
- Applications: Stress detection, emotional monitoring, mental health tracking

Table 4.3: GSR Sensor Pin Configuration

Serial number	Screen printing	Function description
1	VCC/+	Power Supply Anode(3.3-5.5v)
2	GND/-	Power Supply Cathode(0V)
3	TP4	Test Point / Calibration Pin
4	SIG	Analog Signal Output(0-3.0v)

The GSR (Galvanic Skin Response) sensor module is equipped with four key pins for operation and signal output Table 4.3. The VCC/+ pin serves as the power supply anode, accepting an input voltage between 3.3V and 5.5V to power the sensor. The GND/- pin acts as the power supply cathode (0V) and provides the ground reference. The TP4 pin functions as a test point or calibration pin, often used for debugging or sensor tuning during development. The SIG pin delivers an analog output signal (0–3.0V), which reflects variations in skin conductance linked to

emotional or stress responses. These pins enable the sensor to effectively detect electrodermal activity for real-time stress monitoring.

In the end chapter 4 highlights the successful integration of hardware components in the Smart Stress Monitoring System, focusing on the ESP8266 microcontroller, GSR sensor, and EMG sensor. These components work together to capture real-time physiological signals and transmit them wirelessly to cloud platforms for analysis. The design ensures low power consumption, reliable data transmission, and accurate signal acquisition. This robust hardware setup forms the backbone of the system, enabling efficient stress monitoring and laying the foundation for reliable real-world deployment.

CHAPTER 5
SOFTWARE IMPLEMENTATION

5.1 Introduction

This chapter details the software implementation of the Smart Stress Monitoring System Using IoT and ML, covering data acquisition, processing, transmission, storage, and analysis to enable real-time stress monitoring. The system seamlessly integrates sensor data processing, cloud communication, and machine learning algorithms using Python and C++, ensuring accurate classification and prediction of stress levels for effective monitoring and early intervention. The software is designed to handle continuous data collection from EMG and GSR sensors, transmitting the information to cloud platforms such as Google Sheets and ThingsBoard for remote access and visualization. Additionally, machine learning models implemented in Google Colab analyze the collected data, identifying stress patterns and anomalies using techniques like Isolation Forest, Autoencoders, and XGBoost. The combination of IoT and ML ensures that the system provides timely insights, helping caregivers and healthcare professionals take proactive measures to manage stress levels effectively.

5.2 Development Tools & Environment

5.2.1 Arduino IDE:

The Arduino IDE (Integrated Development Environment) is an open source software used to write, compile, and upload code for microcontrollers such as the ESP8266.The Arduino IDE Environment is shown in figure 5.1, It provides a simple and user-friendly interface that supports both C and C++ programming, and is therefore ideal for the development of embedded systems. The IDE incorporates an embedded code editor, a serial monitor, and a large pre-built library of functions that make it easier to communicate with hardware components such as sensors, displays, and actuators. In addition, it supports various microcontroller boards, enabling developers to develop and debug applications for the Internet of Things efficiently.

In this project, the ESP8266 microcontroller is programmed with the Arduino IDE, which allows it to collect, process and transmit GSR and EMG sensor data. Code sent in the Arduino IDE configures sensor connections, Wi-Fi communications, and data transfer protocols, ensuring a seamless integration with the ThingsBoard Cloud for real-time visualization and the Google Sheet for data logging. ESP8266 is also programmed to preprocess sensor data to reduce noise and improve

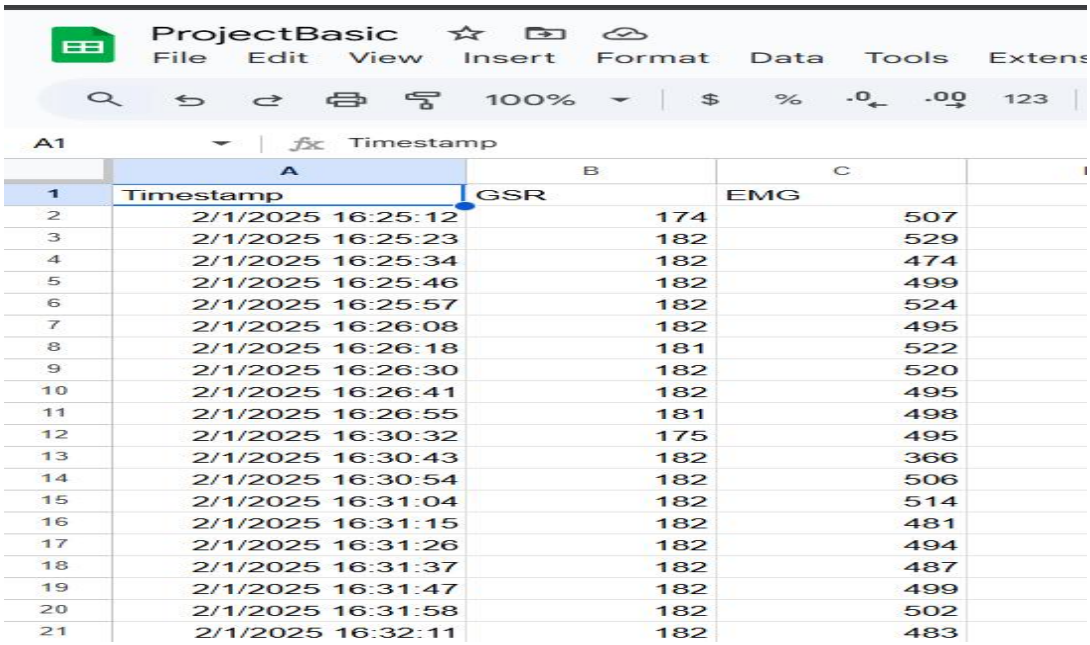
accuracy before sending it to Google Colab for machine learning analysis. Using the Arduino IDE, the system provides efficient control of sensors, real-time data processing, and reliable connection to the Internet of Things for stress monitoring and mental health evaluation.

Fig 5.1: Arduino IDE Environment

5.2.2 Google Sheets and Google Colab:

Google Sheets is a cloud-based spreadsheet tool that enables real-time data logging and structured storage of incoming sensor readings. Figure 5.2 illustrates how data is fetched into google sheets. It allows seamless integration with IoT applications, making it a suitable platform for storing, organizing, and analyzing physiological data over time. Google Colab, on the other hand, is a cloud-based Python development environment that provides a powerful machine learning (ML) platform with built-in support for deep learning frameworks such as TensorFlow and Scikit-learn. Colab enables high-performance computations without requiring local resources, making it ideal for processing large datasets and running ML models efficiently.

In this project, Google Sheets serves as the primary data storage unit, where GSR and EMG signals are continuously logged for historical analysis. The raw sensor data is then fetched into Google Colab, where advanced ML models—including Isolation Forest, Autoencoders, and XGBoost—are applied for anomaly detection and stress classification. This combination ensures real-time data storage, seamless cloud-based ML processing, and accurate stress pattern analysis, making the system scalable, cost-effective, and efficient for long-term mental health monitoring.



ProjectBasic

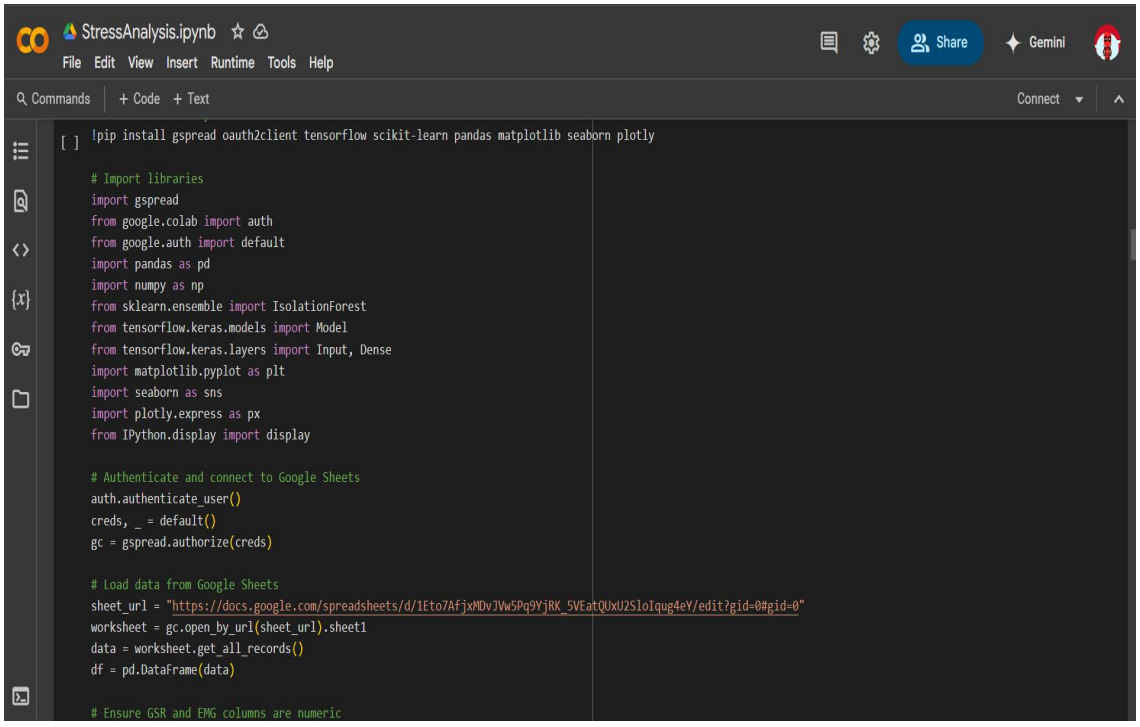
File Edit View Insert Format Data Tools Extens

100% $ % .0 .00 123

A1 fx Timestamp

	A	B	C
1	Timestamp	GSR	EMG
2	2/1/2025 16:25:12	174	507
3	2/1/2025 16:25:23	182	529
4	2/1/2025 16:25:34	182	474
5	2/1/2025 16:25:46	182	499
6	2/1/2025 16:25:57	182	524
7	2/1/2025 16:26:08	182	495
8	2/1/2025 16:26:18	181	522
9	2/1/2025 16:26:30	182	520
10	2/1/2025 16:26:41	182	495
11	2/1/2025 16:26:55	181	498
12	2/1/2025 16:30:32	175	495
13	2/1/2025 16:30:43	182	366
14	2/1/2025 16:30:54	182	506
15	2/1/2025 16:31:04	182	514
16	2/1/2025 16:31:15	182	481
17	2/1/2025 16:31:26	182	494
18	2/1/2025 16:31:37	182	487
19	2/1/2025 16:31:47	182	499
20	2/1/2025 16:31:58	182	502
21	2/1/2025 16:32:11	182	483

Fig 5.2: Fetching GSR and EMG Data to Google Sheets



StressAnalysis.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text Connect

```
[ ] !pip install gspread oauth2client tensorflow scikit-learn pandas matplotlib seaborn plotly

    # Import libraries
    import gspread
    from google.colab import auth
    from google.auth import default
    import pandas as pd
    import numpy as np
    from sklearn.ensemble import IsolationForest
    from tensorflow.keras.models import Model
    from tensorflow.keras.layers import Input, Dense
    import matplotlib.pyplot as plt
    import seaborn as sns
    import plotly.express as px
    from IPython.display import display

    # Authenticate and connect to Google Sheets
    auth.authenticate_user()
    creds, _ = default()
    gc = gspread.authorize(creds)

    # Load data from Google Sheets
    sheet_url = "https://docs.google.com/spreadsheets/d/1Eto7AfjxMDvJVw5Pq9YjRK_5VEatQUxU2SloIqug4eY/edit?gid=0#gid=0"
    worksheet = gc.open_by_url(sheet_url).sheet1
    data = worksheet.get_all_records()
    df = pd.DataFrame(data)

    # Ensure GSR and EMG columns are numeric
```

Fig 5.3: Google Colab Environment

5.2.3 ThingsBoard Cloud:

ThingsBoard Cloud is an open source platform for IoT devices, data collection, and visualization in real time. It supports several communication protocols, such as MQTT, HTTP and CPA, which makes it ideal for applications for the Internet of Things. Thanks to its customizable dashboard, real-time plotting, and notification system, ThingsBoard allows users to track sensor data in an efficient way. It offers

secure data management, user-friendly UI components and scalability, which makes it the preferred choice for monitoring systems running on the Internet of Things.In this project, the ThingsBoard Cloud as shown in figure 5.4, is used exclusively for real-time visualization of GSR and EMG data. The ESP8266 microcontroller transmits raw physiological signals to the platform, which allows users and health care professionals to dynamically monitor the level of stress. The dashboard provides live graphs, numerical data and trend analysis to help you interpret stress changes quickly. By integrating the ThingsBoard, the system provides a clear, interactive and remote display of GSR and EMG signals, enabling users to monitor their physiological response without the need for complicated setup.

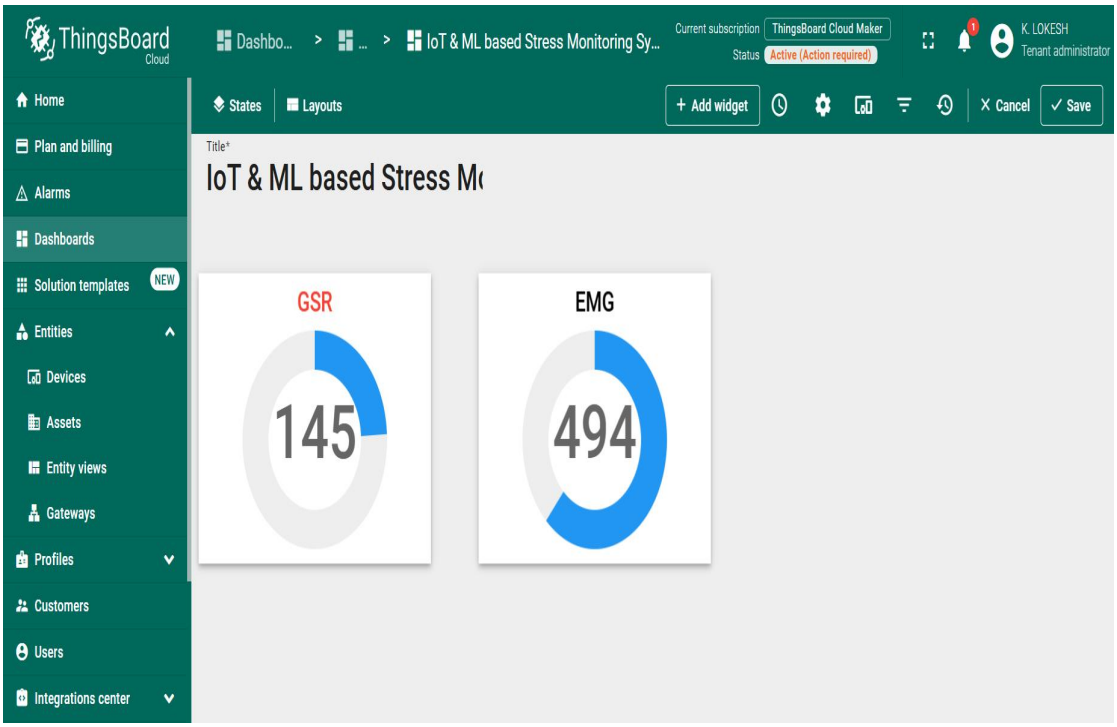


Fig 5.4: ThingsBoard Cloud Set up

5.3 Machine Learning Models:

The proposed system utilizes machine learning (ML) models to analyze Galvanic Skin Response (GSR) and Electromyography (EMG) signals, enabling both anomaly detection and disorder prediction. The anomaly detection phase employs Isolation Forest and Autoencoders to identify irregular physiological patterns, ensuring early detection of unusual stress responses. Once anomalies are detected, the disorder prediction phase classifies the likelihood of stress-related conditions such as Autism Spectrum Disorder (ASD) and ADHD using XGBoost, a powerful gradient-boosting algorithm. These models are implemented in Google Colab, leveraging cloud-based computing for efficient and high-accuracy analysis. By integrating real-

time anomaly detection and disorder classification, the system enhances early intervention capabilities, making it a scalable, accurate, and cost-effective solution for mental health monitoring.

5.3.1 Isolation Forest:

Isolation forest (IF) is an unsupervised machine learning algorithm specially designed for anomaly detection. It works on the principle that anomalies (outsiders) are easier to isolate than regular data points. Unlike traditional approaches based on clustering or density, the Isolation Forest does not calculate distance or density; it randomly partitions the data to quickly isolate anomalies.

The Isolation Forest algorithm detects anomalies by exploiting the principle that anomalies are easier to isolate than normal data points. The process starts with a random sub-sample selection where the data set is randomly split into several subsets. Within each subset, the algorithm performs recursive partitioning, selecting a random function and dividing it by a range within the range in which the data is distributed. This recursive process creates an iTree in which fewer splits indicate a higher probability of an anomaly. The path length of the data point is then calculated as the number of splits needed to isolate it, with anomalies having shorter path lengths because they are isolated more rapidly than normal data points.The anomaly score for a data point x is given by:

$$s(x,n) = 2^{\frac{-E(h(x))}{c(n)}} \qquad 5.1$$

where s(x,n) represents the anomaly score, E(h(x)) is the average path length across multiple isolation trees, and c(n) is the average path length of a balanced binary tree of size n. Based on this score, decision-making is performed: a higher anomaly score (closer to 1) indicates that the point is an anomaly, while a lower score (closer to 0) suggests that the point is normal. This approach allows Isolation Forest to efficiently detect outliers in high-dimensional datasets with minimal computational overhead.

5.3.2 Autoencoder:

An Autoencoder is a type of neural network designed to learn to represent input data efficiently by compressing it into smaller spaces and then reconstructing it. A system is designed with two main components: one that shrinks the size of incoming data while retaining its core features, and another that attempts to recreate the original data from this reduced version. The system learns by striving to make the

recreated data as close as possible to the initial input. In essence, for typical data, the system minimizes the differences between what it receives and what it produces. However, when the model encounters an unusual or invisible pattern, it struggles to reconstruct it accurately, leading to greater reconstruction errors. This principle is used to detect anomalies in your project, when the system detects stress-related abnormalities in the galvanic skin response (GSR) and electromyography (EMG) signals based on the reconstruction loss.Mathematically, given an input X, the encoder maps it to a latent space representation h using the function:

$$h=f(x)=\sigma\left(w_e x+b_e\right) \qquad 5.2$$

where We and be are the weights and biases of the encoder, and σ is the activation function. The decoder then reconstructs the input as X′ through:

$$x'=g(x)=\sigma\left(w_d h+b_d\right) \qquad 5.3$$

where Wd and bd are the weights and biases of the decoder. The difference between X and X′ is measured using Mean Squared Error (MSE), given by:

$$L=\frac{1}{n}\sum\left(x-x'\right)^2 \qquad 5.4$$

If the reconstruction loss L exceeds a set threshold (typically the 95th percentile), the data point is flagged as an anomaly.

In our project, the Autoencoder is trained on normal GSR and EMG values. During real-time monitoring, it processes incoming signals and reconstructs them. If the MSE is high, indicating an unusual pattern, the instance is classified as an anomaly. By integrating Autoencoders with Isolation Forest, your system ensures accurate stress anomaly detection while minimizing false positives, making it highly robust for real-time stress monitoring applications.

In this project, Isolation Forest and Autoencoders are working together on a hybrid approach to anomaly detection, which allows for accurate detection of stress-related GSR and EMG signals. The Isolation tree detects anomalies by analyzing how easy it is to isolate a data point, while the Autoencoders learn normal patterns and flag instances with high reconstruction errors as anomalous. By combining these techniques, the system uses both statistical and deep learning methods to ensure high accuracy and robustness in the detection of stress-related abnormalities. This

integrated approach improves real-time monitoring of stress, allowing early detection of abnormal physiological responses and making the system more robust and adaptable to the real-life application of health care.

5.3.3 Extreme Gradient Boosting (XGBoost):

Extreme Gradient Boosting (XGBoost) is a strong supervised machine learning technique based on gradient boosting that is designed for speed, efficiency, and accuracy. It is commonly used for classification and regression problems, with strong performance on big datasets containing missing values and complicated patterns. Unlike classic decision tree models, XGBoost creates an ensemble of weak learners (decision trees) in a sequential fashion, with each tree correcting the flaws of its predecessor. This procedure continues until the model achieves optimal accuracy, making XGBoost an excellent choice for non-linear connections. In this study, XGBoost is used to forecast disorders by evaluating Galvanic Skin Response (GSR) and Electromyography (EMG) data and classifying stress-related illnesses such as Autism Spectrum Disorder (ASD) and Attention Deficit Hyperactivity Disorder (ADHD).

XGBoost employs the Gradient Boosting Decision Trees (GBDTs) technique, which involves training multiple trees progressively to minimize errors at each stage. The first tree produces an initial forecast, which is then refined by succeeding trees that focus on residual mistakes. The approach computes a loss function, uses gradients to determine the direction of progress, and grows new trees to reduce mistakes even further. Finally, the outputs of all trees are merged in a weighted summation to create a powerful classifier. XGBoost's mathematical formulation minimizes the goal function:

$$Obj = \sum_{i=1}^{n} L\left(yi - \widehat{yi}\right) + \sum_{k=1}^{k} \Omega\left(f_k\right) \qquad 5.5$$

where:L(yi,yi') is the loss function, which quantifies the difference between actual and predicted values.Ω(fk) is the regularization term, which controls model complexity and prevents overfitting.

The final prediction is computed as:

$$\hat{y} = \sum_{k=1}^{k} f_k\left(x\right) \qquad 5.6$$

where fk(x) represents the output of each decision tree in the

ensemble.XGBoost effectively models complex dependencies between GSR and EMG signals. Feature importance ranking identifies significant physiological signals. Regularization techniques prevent overfitting. High efficiency allows for real-time stress classification. XGBoost is integrated with gaussian matrix model(GMM) for disorder classification. Data acquisition, preprocessing, and clustering are performed. Feature engineering extracts key statistical properties. XGBoost classifier is trained and evaluates model performance. System combines GMM and XGBoost for accurate stress disorder prediction.

The Software Implementation chapter discussed the development tools, programming environments, and machine learning models utilized in the Smart Stress Monitoring System Using IoT and ML. The combination of C++ for microcontroller programming, Python for data processing, and cloud platforms for real-time monitoring results in an efficient and scalable solution. The system detects and classifies stress levels properly using machine learning techniques like Isolation Forest, Autoencoders, and XGBoost. This technology is useful for continuous stress monitoring and early intervention because it combines sensor data processing, cloud connectivity, and machine learning-based analysis.

CHAPTER 6

RESULTS

6.1 Anomaly Detection:

Anomalies detected in Galvanic Skin Response (GSR) and Electromyography (EMG) signals serve as indicators of stress patterns. The use of Isolation Forest and Autoencoder algorithms proved effective in identifying these anomalies. For instance, the Isolation Forest algorithm utilized 100 estimators and a 5% contamination rate to pinpoint anomalies in the data. Visualizations such as heatmaps provided a clear confirmation of these anomalies. On the other hand, the Autoencoder algorithm was able to learn normal patterns by employing a two-layer encoder and a linear decoder. The calculation of reconstruction errors using Mean Squared Error (MSE) allowed for the identification of anomalies based on the 95th percentile of MSE values.

By combining the results from both Isolation Forest and Autoencoder, the sensitivity of anomaly detection was significantly enhanced. This fusion approach was further illustrated through the visualization of anomaly distribution using a pie chart. Additionally, time-series graphs were utilized to track the fluctuations in GSR and EMG over time, showcasing the effectiveness of both algorithms in detecting stress-related anomalies. The identification of these anomalies is crucial for predicting disorders, especially in the context of stress monitoring for children with mental disabilities and individuals diagnosed with ADHD and Autism Spectrum Disorder (ASD). Overall, the utilization of Isolation Forest and Autoencoder algorithms has demonstrated their efficacy in detecting and analyzing stress-related anomalies, providing valuable insights for further research and applications in the field.

	GSR	EMG	Isolation_Anomaly	Autoencoder_Anomaly	Final_Anomaly
0	174.000000	507.000000	Normal	Normal	Normal
1	182.000000	529.000000	Normal	Normal	Normal
2	182.000000	474.000000	Normal	Normal	Normal
3	182.000000	499.000000	Normal	Normal	Normal
4	182.000000	524.000000	Normal	Normal	Normal
5	182.000000	495.000000	Normal	Normal	Normal
6	181.000000	522.000000	Normal	Normal	Normal
7	182.000000	520.000000	Normal	Normal	Normal
8	182.000000	495.000000	Normal	Normal	Normal
9	181.000000	498.000000	Normal	Normal	Normal
10	175.000000	495.000000	Normal	Normal	Normal

Fig 6.1: Subject is in Normal Condition

100	53.000000	504.000000	Normal	Normal	Normal
101	51.000000	517.000000	Normal	Normal	Normal
102	61.000000	974.000000	Anomaly	Anomaly	Anomaly
103	57.000000	456.000000	Normal	Normal	Normal
104	57.000000	796.000000	Normal	Normal	Normal
105	58.000000	796.000000	Anomaly	Normal	Anomaly
106	57.000000	764.000000	Normal	Normal	Normal
107	69.000000	974.000000	Anomaly	Anomaly	Anomaly
108	62.000000	974.000000	Anomaly	Anomaly	Anomaly
109	62.000000	497.000000	Normal	Normal	Normal
110	60.000000	494.000000	Normal	Normal	Normal

Fig 6.2: Subject is in Stressed Condition

Figure 6.1 shows the subject in a normal condition, with stable GSR and EMG readings indicating no signs of stress.Figure 6.2 represents the subject in a stressed condition, where elevated GSR and EMG values reflect physiological stress responses.

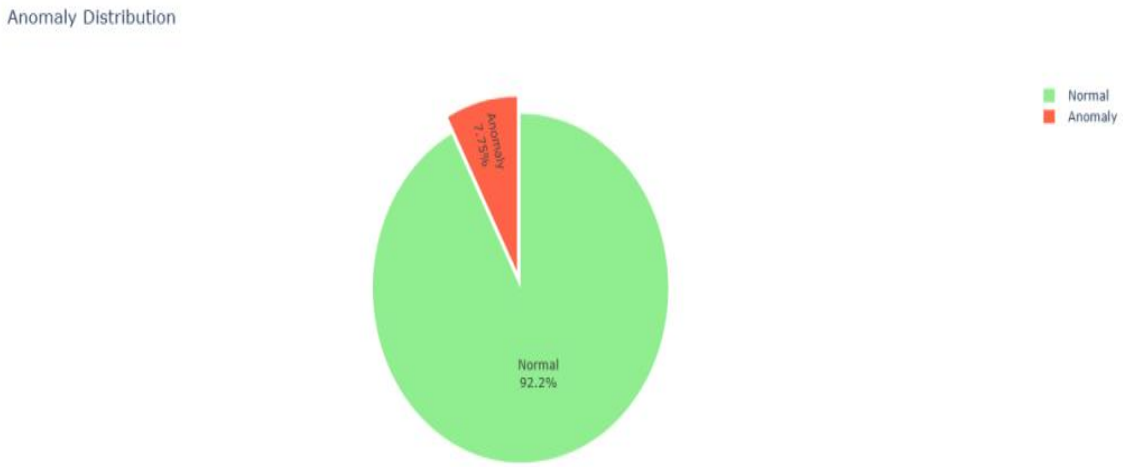


Fig 6.3: Anomaly Distribution Pie Chart

The anomaly detection findings reveal significant insights on the stress monitoring system utilizing GSR and EMG sensor data. The correlation heatmap depicts the correlations between raw and normalized sensor readings, with GSR and EMG exhibiting a modest negative correlation, indicating unique stress response patterns. The anomaly distribution pie chart shows that 7.75% of the collected data points are classed as anomalies, indicating probable stress-related occurrences. The time-series display reveals more abnormalities, including rapid spikes and decreases in GSR and EMG readings. The discovery of such abnormalities is critical for early intervention because they may signify periods of high stress or emotional instability in people with ADHD and ASD. This investigation validated the efficiency of the Isolation Forest and Autoencoder models.

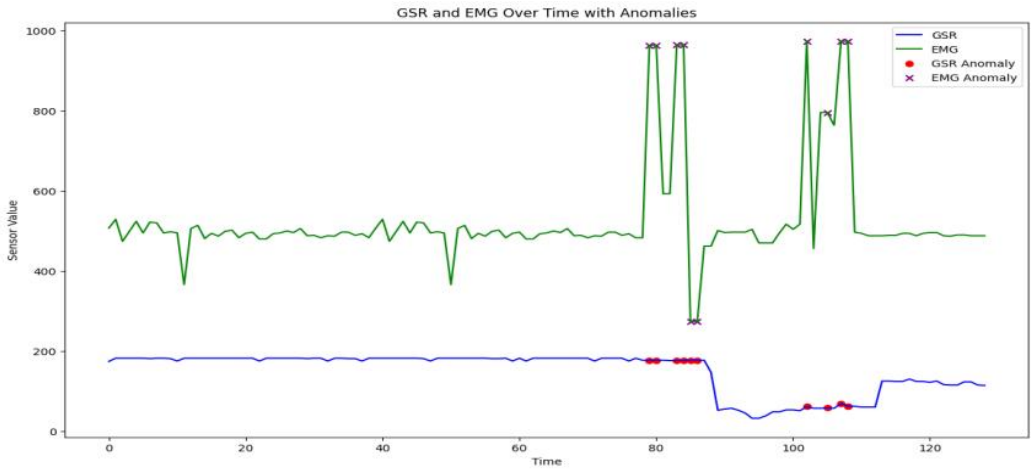


Fig 6.4: GSR and EMG Over Time with Anomalies

6.2 Disorder Prediction:

The disorder prediction model utilizes GSR and EMG sensor data to classify individuals into different health conditions, including ASD, ADHD, Cognitive Impairment, and Normal. The model is built using XGBoost, a powerful gradient-boosting algorithm, to analyze physiological signals and make accurate predictions[5].The data is first preprocessed and normalized using MinMaxScaler to ensure all values are within a uniform range[7]. A labeling function is applied to categorize data points based on GSR and EMG thresholds, mapping them to the corresponding disorders.The categorized information was divided into two distinct groups: a larger segment, comprising 80% of the total, allocated for model training, and a smaller portion, representing the remaining 20%, designated for performance assessment[3].The model achieves an overall accuracy of 94.74%, demonstrating its effectiveness in distinguishing between different stress-related conditions. The classification report further highlights the model's precision, recall, and F1-score, indicating strong performance in identifying each disorder.A confusion matrix heatmap is used to visualize the model's predictions, showing how well it differentiates between conditions. Additionally, the system provides a color-coded table displaying actual vs. predicted conditions, allowing for a clear assessment of the model's decision-making process.

The results suggest that GSR and EMG signals can serve as reliable indicators for detecting stress-related disorders[4]. The model's ability to classify conditions with high accuracy makes it a promising tool for early diagnosis and intervention, especially for individuals with ASD and ADHD[1]. Future improvements could focus on incorporating additional physiological markers and

refining classification thresholds for even greater accuracy.

Table 6.1: Thresholds for Health Condition Classification

Health Condition	GSR Threshold	EMG Threshold
ASD	GSR > 0.6	EMG > 0.6
Cognitive Impairment	GSR > 0.5	EMG < 0.5
ADHD	GSR <0.3	EMG > 0.5
Normal	All Other Cases	All Other Cases

Table 6.2: Performance Evaluation

Class	Precision	Recall	F1-Score
ADHD	1.00	1.00	1.00
ASD	1.00	1.00	1.00
Cognitive Impairment	0.96	0.96	0.96
Normal	0.90	0.90	0.90
Macro Avg.	0.97	0.97	0.97
Weighted Avg.	0.95	0.95	0.95
Overall Accuracy	94.74%		

Misclassification happens when a model predicts a class label inaccurately, which means that the anticipated health status differs from the actual condition. This might be due to overlapping characteristics, sensor noise, or difficulties in threshold-based categorization. In the context of this project, false positives (detecting a condition mistakenly when the individual is actually normal) and false negatives (failing to detect a disorder when it exists) might have an influence on the system's dependability.However, misclassification is limited in this project thanks to the XGBoost model's excellent accuracy of 94.74% and great performance. The accuracy and recall scores for each class are consistently high, demonstrating that the model can distinguish between ASD, ADHD, Cognitive Impairment, and Normal conditions. The confusion matrix confirms this by demonstrating that the number of inaccurate guesses is quite low.

Table 6.1 defines the threshold values for classifying health conditions based on GSR and EMG sensor readings, categorizing subjects into ASD, ADHD, Cognitive Impairment, or Normal. Table 6.2 presents the performance evaluation of the XGBoost model, showing high precision, recall, and F1-scores for all classes, with an overall accuracy of 94.74%. Together, these tables demonstrate the system’s effectiveness in predicting stress-related disorders with strong classification accuracy.

This highlights the system's potential for early diagnosis and intervention in mental health monitoring applications. Figure 6.5 illustrates the confusion matrix for health condition prediction which shows how much accurate the model we have used.

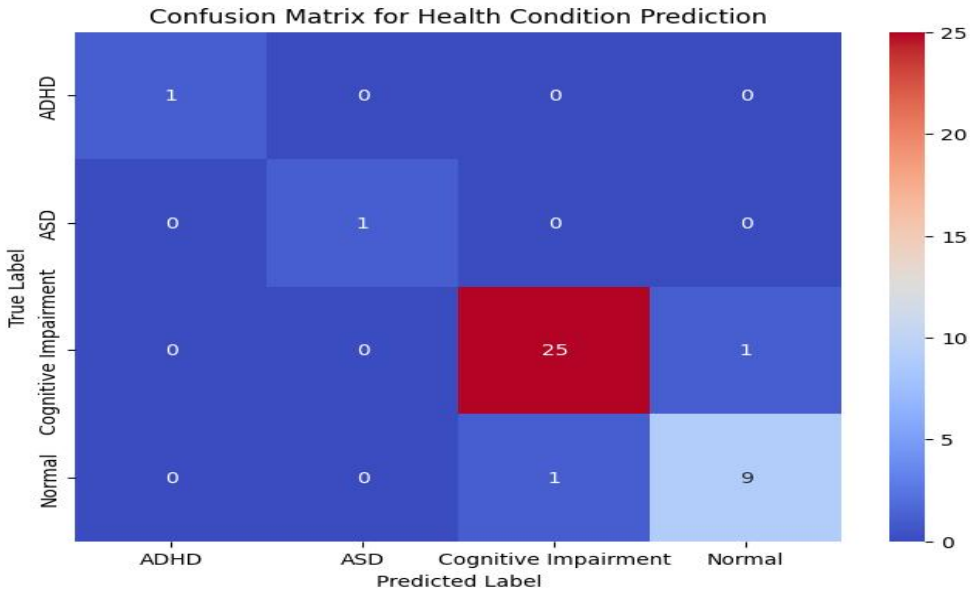


Fig 6.5: Confusion Matrix for Health Condition Prediction

In the end the results show that the proposed IoT-ML-based stress monitoring system is successful at detecting anomalies as well as classifying disorders. The anomaly detection models (Isolation Forest and Autoencoder) correctly identified stress-related anomalies in GSR and EMG signals, while the XGBoost-based disorder prediction model achieved a 94.74% accuracy in classifying ASD, ADHD, Cognitive Impairment, and Normal conditions. The performance assessment measures, including as accuracy, recall, and F1-score, show high model dependability and low misclassification. The system's capacity to identify physiological stress reactions in real time demonstrates its potential for early diagnosis and intervention, making it an important tool for mental health evaluation.

CONCLUSION AND FUTURE SCOPE

Conclusion:

Individuals of all ages are now frequently feeling stress as a result of a variety of circumstances, including mental health issues, cognitive impairments, and high-pressure situations. Stress typically goes unnoticed, particularly in youngsters with mental problems and the elderly, making early identification difficult. This project was created with the goal of delivering an effective, real-time, and automated solution for stress monitoring via IoT and ML technologies.

Smart Stress Monitoring System Using IoT and ML benefits caregivers, guardians, and healthcare professionals by lowering the load of manual stress evaluation and allowing for continuous monitoring. The system is affordable, portable, and simple to operate, making it suitable for a variety of real-world applications. Future enhancements might include adding more physiological sensors, creating a mobile application, strengthening security features, and boosting ML model accuracy. With future development, this project might become a comprehensive tool for proactive stress management and mental health monitoring.

Future Scope:

Our approach has shown promising results in real-time stress detection and early disorder prediction, there are various areas for improvement and growth. One significant enhancement is the use of additional physiological sensors, such as heart rate variability (HRV) and electroencephalography (EEG), which can enhance multi-modal stress detection by collecting a wider variety of physiological responses. By adding these additional biosignals, the system may give a more complete and accurate evaluation of stress levels, enhancing both short-term monitoring and long-term trend analysis. Furthermore, optimizing machine learning algorithms is critical for improving classification accuracy and decreasing the number of false positives and false negatives. Using improved feature selection techniques, fine-tuning hyperparameters, and ensemble learning algorithms can dramatically enhance prediction reliability. Furthermore, using real-time adaptive learning methods might allow the system to grow over time depending on user-specific data, making stress detection more personalized and precise.Advanced deep learning models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), can help improve stress pattern identification.

This program not only solves the limits of existing stress monitoring

methods, but it also establishes a solid platform for future advances in mental health care. By providing real-time, data-driven insights and early intervention capabilities, the system has the potential to revolutionize how stress-related disorders are identified and managed. As technology advances, combining AI-driven analytics with IoT-based monitoring can empower individuals, caregivers, and healthcare professionals, resulting in enhanced mental health and proactive stress management measures.

Advantages:

1. **Specific Focus on Stress and Neurological Disorders:** By monitoring GSR and EMG, the system provides targeted insights into mental health and stress-related conditions, offering a more specialized solution compared to general health monitoring systems.
2. **Comprehensive Real-Time Monitoring:** The integration of IoT dashboards and cloud-based analysis ensures a seamless flow of real-time data, enabling immediate visualization and long-term tracking.
3. **Advanced Anomaly Detection:** Hybrid models combining Isolation Forests and Autoencoders ensure robust detection of irregularities in GSR and EMG signals, even in dynamic environments.
4. **Cost-Effective & Scalable:** The system uses low-cost IoT components (ESP8266, GSR, EMG sensors), making it an affordable and scalable solution.

Disadvantages:

1. **Sensor Accuracy & Noise:** EMG and GSR signals are prone to noise and motion artifacts, which may affect the system's accuracy.
2. **Dependence on Internet Connectivity:** While edge processing is implemented, the system still relies on Wi-Fi connectivity for cloud-based analysis and remote access.
3. **Limited to EMG & GSR Metrics:** The system does not incorporate additional physiological markers like ECG, temperature, or SpO2, which could improve stress assessment accuracy.
4. **Data Privacy Concerns:** Although the system minimizes cloud dependency, stress data being transmitted over Wi-Fi still poses potential privacy risks, especially if not properly encrypted.

Applications:

1. **Mental Health Monitoring:** Helps detect stress and emotional responses in individuals with ADHD, ASD, and other cognitive impairments.
2. **Workplace Stress Assessment:** Can be deployed in corporate offices and high-

stress environments to track and manage employee well-being, reducing burnout and improving overall productivity and job satisfaction.

3. **Rehabilitation & Therapy:** Assists doctors and therapists in monitoring stress patterns during psychological therapy and rehabilitation programs.
4. **Educational Institutions:** Useful in schools and universities to track students' stress levels, improving mental health support systems and allowing educators to implement strategies for stress reduction and academic performance enhancement.

BIBLIOGRAPHY

[1] Yali Zheng, Chen Wu, Peizheng Cai, Zhiqiang Zhong, Hongda Huang, Yuqi Jiang, "Tiny-PPG: A Lightweight Deep Neural Network for Real-Time Detection of Motion Artifacts in Photoplethysmogram Signals on Edge Devices," Internet of Things, Vol. 25, 2024.

[2] Zhu, L., Spachos, P., Ng, P. C., Yu, Y., Wang, Y., Plataniotis, K., & Hatzinakos, D., "Stress Detection Through Wrist-Based Electrodermal Activity Monitoring and Machine Learning," IEEE Journal of Biomedical and Health Informatics, Vol. 27, No. 5, pp. 2155-2165, 2023.

[3] Patil, A. N., Preeti, B. M., & Laxmi, "Real-Time Stress Level Monitoring Using IoT," International Conference on Integrated Intelligence and Communication Systems (ICESC), 2023.

[4] Ariayudha, R. A., Nuha, H. H., & Irsan, M., "Reading Stress Levels and Setting Emotional Patterns with Sensors Based on Galvanic Skin Response (GSR) Method," International Conference on Data Science and Its Application (ICoDSA), 2023.

[5] Ranjha, Azlaan, Laiba Jabbar, and Osaid Ahmed, "Cloud-Connected Wireless Holter Monitor Machine with Neural Networks Based ECG Analysis for Remote Health Monitoring," 2023.

[6] Giannakakis, G., Grigoriadis, D., Giannakaki, K., Simantiraki, O., Roniotis, A., Tsiknakis, M., "Review on Psychological Stress Detection Using Biosignals," IEEE Transactions on Affective Computing, Vol. 13, No. 1, pp. 440-460, Jan.-Mar. 2022.

[7] Pourmohammadi, S., Maleki, A., "Continuous Mental Stress Level Assessment Using Electrocardiogram and Electromyogram Signals," Biomedical Signal Processing and Control, Vol. 68, July 2021, 102694.

[8] Memar, M., Mokaribolhassan, A., "Stress Level Classification Using Statistical Analysis of Skin Conductance Signal While Driving," SN Applied Sciences, Vol. 3, p. 64, 2021.

[9] Mozafari, M., Firouzi, F., & Farahani, B., "Towards IoT-Enabled Multimodal Mental Stress Monitoring," Sharif University of Technology, Duke University, Shahid Beheshti University, 2020.

[10] Elgendi, M., Menon, C., "Machine Learning Ranks ECG as an Optimal Wearable Biosignal for Assessing Driving Stress," IEEE Access, Vol. 8, pp. 34362-34374, 2020.

[11] R. K. Nath, H. Thapliyal, A. Caban-Holt and S. P. Mohanty, "Machine Learning Based Solutions for Real-Time Stress Monitoring," in IEEE Consumer Electronics Magazine, vol. 9, no. 5, pp. 34-41, 1 Sept. 2020

[12] Kyriakou, K., Resch, B., Sagl, G., Petutschnig, A., Werner, C., Niederseer, D., Liedlgruber, M., Wilhelm, F. H., Osborne, T., Pykett, J., “Detecting Moments of Stress from Measurements of Wearable Physiological Sensors,” Sensors, Vol. 19, p. 3805, 2019.

[13] Pandey, P. S., “Machine Learning and IoT for Prediction and Detection of Stress,” 17th International Conference on Computational Science and Its Applications (ICCSA), Trieste, Italy, 2017, pp. 1-5.

[14] Boon-Leng, L., Dae-Seok, L., & Boon-Giin, L., “Mobile-Based Wearable-Type of Driver Fatigue Detection by GSR and EMG,” TENCON 2015 - 2015 IEEE Region 10 Conference, Macao, China, 2015, pp. 1-4.

PROJECT OUTCOMES MAPPED WITH PROGRAMME SPECIFIC OUTCOMES (PSOs) AND PROGRAMME OUTCOMES (POs)

Classification of Project	Application	Product	Research	Review
		✓	✓	

Project Outcomes:

1. **Real-Time Stress Monitoring:** By leveraging IoT and machine learning algorithms like Isolation Forest and XGBoost, the device ensures real-time tracking and anomaly detection of stress levels, thus helping caregivers and healthcare professionals make informed decisions instantly.
2. **ML-Driven Anomaly Detection for Early Intervention:** Leveraging machine learning models such as Isolation Forest and XGBoost, the system detects irregular stress patterns and early signs of neurodevelopmental conditions. This supports proactive mental health management, helping caregivers and professionals respond before escalation.
3. **Enhanced Support for Special Needs Individuals:** The project provides a continuous and non-intrusive monitoring solution for children with mental disabilities, supporting educators, therapists, and parents in understanding emotional triggers and behavioral changes to offer better personalized care.
4. **Enhanced Caregiver Support and Engagement:** By transmitting data to platforms like ThingsBoard and Google Sheets, the system offers caregivers and healthcare professionals accessible dashboards for stress tracking. This improves communication, promotes timely decision-making, and enhances care quality for individuals with special needs.

PROGRAMME SPECIFIC OUTCOMES (PSOs):

The Internet of Things program Graduates will be equipped with the ability of

PSO1: Implementing innovative, cost effective processes for producing energy efficient and eco-friendly IoT applications.

PSO2: Applying the knowledge of Core, Disciplinary and Interdisciplinary elective courses of Electronics, Communications and Computer Engineering to achieve a logical solution to real world problems.

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

1. **Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/ development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multi-disciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multi-

disciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Mapping Table

Project Outcomes	Programme Outcomes (POs)												PSOs	
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
Outcome 1	3	3	3	2	3	2	2	1	2	2	2	3	3	3
Outcome 2	3	3	3	3	3	2	1	2	2	2	2	3	2	3
Outcome 3	2	2	2	2	2	3	2	3	3	2	1	2	2	2
Outcome 4	2	2	2	2	2	3	2	2	3	3	2	2	2	2

Note: Map each project outcomes with POs and PSOs with either 1 or 2 or 3 based on level of mapping as follows:

1-Slightly (Low) mapped 2-Moderately (Medium) mapped 3-Substantially (High) mapped

APPENDIX A

// GSR Arduino Code :

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClientSecure.h>
// WiFi credentials
const char* ssid = "Device";
const char* password = "1234567890";
// Google Apps Script URL
const char* googleSheetsUrl =
"https://script.google.com/macros/s/AKfycbxLGJie08tyrdYmQfZxe17p1H1_pkQ5dNmaZ
wSjX7n6ejv857Jp-hDoropFC4R8grQuXw/exec";
// ThingsBoard server and token
const char* TOKEN = "Q62rn79ctYuIKa6pJtyQ"; // Replace with your ThingsBoard
token
const char* thingsBoardUrl = "http://thingsboard.cloud/api/v1/"; // ThingsBoard server
URL
WiFiClientSecure clientGoogle; // Client for Google Sheets (HTTPS)
WiFiClient clientThingsBoard;  // Client for ThingsBoard
#define GSR_PIN A0
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to WiFi");
  // Set up the Google Sheets client to trust all certificates
  clientGoogle.setInsecure();
}
void sendGSRDataToGoogleSheets(int gsrValue) {
  if (WiFi.status() == WL_CONNECTED) {
```

```
    HTTPClient http;
    String payload = "{\"sensorType\":\"GSR\",\"value\":" + String(gsrValue) + "}";
    Serial.println("Sending GSR data to Google Sheets: " + payload);
    http.begin(clientGoogle, googleSheetsUrl); // Initialize HTTPS client
    http.addHeader("Content-Type", "application/json");
    int httpResponseCode = http.POST(payload);
    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println("Google Sheets Response: " + response);
    } else {
      Serial.print("Error on sending POST to Google Sheets: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  } else {
    Serial.println("WiFi not connected for Google Sheets!");
  }
}
void sendGSRDataToThingsBoard(int gsrValue) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String fullServerPath = String(thingsBoardUrl) + TOKEN + "/telemetry";
    String payload = "{\"GSR\":" + String(gsrValue) + "}";
    Serial.println("Sending GSR data to ThingsBoard: " + payload);
    http.begin(clientThingsBoard, fullServerPath); // Initialize ThingsBoard client
    http.addHeader("Content-Type", "application/json");
    int httpResponseCode = http.POST(payload);
    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println("ThingsBoard Response: " + response);
    } else {
      Serial.print("Error on sending POST to ThingsBoard: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  } else {
```

```
        Serial.println("WiFi not connected for ThingsBoard!");
    }
}
void loop() {
    int gsrValue = analogRead(GSR_PIN);
    Serial.print("GSR: ");
    Serial.println(gsrValue);
    // Send data to Google Sheets
    sendGSRDataToGoogleSheets(gsrValue);
    // Send data to ThingsBoard
    sendGSRDataToThingsBoard(gsrValue);
    delay(500); // Delay before the next reading
}
```

//EMG Arduino Code:

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClientSecure.h>
// WiFi credentials
const char* ssid = "Device";
const char* password = "1234567890";
// Google Apps Script URL
const char* googleSheetsUrl =
"https://script.google.com/macros/s/AKfycbxLGJie08tyrdYmQfZxe17p1H1_pkQ5dNmaZ
wSjX7n6ejv857Jp-hDoropFC4R8grQuXw/exec";
// ThingsBoard server and token
const char* TOKEN = "Q62rn79ctYuIKa6pJtyQ"; // Replace with your ThingsBoard
token
const char* thingsBoardUrl = "http://thingsboard.cloud/api/v1/"; // ThingsBoard server
URL
WiFiClientSecure clientGoogle;  // Secure client for Google Sheets
WiFiClient clientThingsBoard;   // Client for ThingsBoard
#define EMG_PIN A0
void setup() {
    Serial.begin(115200);
    WiFi.begin(ssid, password);
```

```
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to WiFi");
  // Set up the Google Sheets client to trust all certificates
  clientGoogle.setInsecure();
}
void sendEMGDataToGoogleSheets(int emgValue) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String payload = "{\"sensorType\":\"EMG\",\"value\":" + String(emgValue) + "}";
    Serial.println("Sending EMG data to Google Sheets: " + payload);
    http.begin(clientGoogle, googleSheetsUrl);  // Initialize with HTTPS client
    http.addHeader("Content-Type", "application/json");
    int httpResponseCode = http.POST(payload);
    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println("Google Sheets Response: " + response);
    } else {
      Serial.print("Error on sending POST to Google Sheets: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  } else {
    Serial.println("WiFi not connected for Google Sheets!");
  }
}
void sendEMGDataToThingsBoard(int emgValue) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String fullServerPath = String(thingsBoardUrl) + TOKEN + "/telemetry";
    String payload = "{\"EMG\":" + String(emgValue) + "}";
    Serial.println("Sending EMG data to ThingsBoard: " + payload);
    http.begin(clientThingsBoard, fullServerPath); // Initialize ThingsBoard client
    http.addHeader("Content-Type", "application/json");
```

```
        int httpResponseCode = http.POST(payload);
        if (httpResponseCode > 0) {
            String response = http.getString();
            Serial.println("ThingsBoard Response: " + response);
        } else {
            Serial.print("Error on sending POST to ThingsBoard: ");
            Serial.println(httpResponseCode);
        }
        http.end();
    } else {
        Serial.println("WiFi not connected for ThingsBoard!");
    }
}
void loop() {
    int emgValue = analogRead(EMG_PIN);
    Serial.print("EMG: ");
    Serial.println(emgValue);
    // Send data to Google Sheets
    sendEMGDataToGoogleSheets(emgValue);
    // Send data to ThingsBoard
    sendEMGDataToThingsBoard(emgValue);
    delay(500);  // Delay before the next reading
}
```

#Data Analysis:

#Anomaly Detection:

```
# Install necessary libraries
!pip install gspread oauth2client tensorflow scikit-learn pandas matplotlib seaborn plotly
# Import libraries
import gspread
from google.colab import auth
from google.auth import default
import pandas as pd
import numpy as np
from sklearn.ensemble import IsolationForest
```

```
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from IPython.display import display
# Authenticate and connect to Google Sheets
auth.authenticate_user()
creds, _ = default()
gc = gspread.authorize(creds)
# Load data from Google Sheets
sheet_url =
"https://docs.google.com/spreadsheets/d/1Eto7AfjxMDvJVw5Pq9YjRK_5VEatQUxU2Slo
Iqug4eY/edit?gid=0#gid=0"
worksheet = gc.open_by_url(sheet_url).sheet1
data = worksheet.get_all_records()
df = pd.DataFrame(data)
# Ensure GSR and EMG columns are numeric
df['GSR'] = pd.to_numeric(df['GSR'], errors='coerce')
df['EMG'] = pd.to_numeric(df['EMG'], errors='coerce')
# Handle any NaNs by forward-filling, then backward-filling remaining NaNs
df = df.fillna(method='ffill').fillna(method='bfill')
# Normalize GSR and EMG data
df['GSR_norm'] = (df['GSR'] - df['GSR'].mean()) / df['GSR'].std()
df['EMG_norm'] = (df['EMG'] - df['EMG'].mean()) / df['EMG'].std()
# Simplified feature engineering without rolling window
X = df[['GSR_norm', 'EMG_norm']]
# Isolation Forest for Anomaly Detection
iso_forest = IsolationForest(n_estimators=100, max_samples='auto', contamination=0.05,
random_state=42)
df['Isolation_Anomaly'] = iso_forest.fit_predict(X)
df['Isolation_Anomaly'] = df['Isolation_Anomaly'].map({1: 'Normal', -1: 'Anomaly'})
# Autoencoder Model for Anomaly Detection
input_dim = X.shape[1]
encoding_dim = 2
input_layer = Input(shape=(input_dim,))
```

```
encoder = Dense(encoding_dim, activation="relu")(input_layer)
decoder = Dense(input_dim, activation="linear")(encoder)
autoencoder = Model(inputs=input_layer, outputs=decoder)
autoencoder.compile(optimizer='adam', loss='mse')
# Train Autoencoder
history = autoencoder.fit(X, X, epochs=50, batch_size=16, validation_split=0.2, verbose=0)
# Autoencoder-based anomaly detection
reconstructions = autoencoder.predict(X)
mse = np.mean(np.power(X - reconstructions, 2), axis=1)
threshold = np.percentile(mse, 95)  # Using 95th percentile as threshold
df['Autoencoder_Anomaly'] = mse > threshold
df['Autoencoder_Anomaly'] = df['Autoencoder_Anomaly'].map({True: 'Anomaly', False:
'Normal'})
# Combine results from Isolation Forest and Autoencoder
df['Final_Anomaly'] = np.where((df['Isolation_Anomaly'] == 'Anomaly') |
(df['Autoencoder_Anomaly'] == 'Anomaly'), 'Anomaly', 'Normal')
# Define function to color anomalies in the table
def color_anomalies(val):
    if val == 'Anomaly':
        color = 'background-color: #FF6347; color: white; font-weight: bold;'
    elif val == 'Normal':
        color = 'background-color: #90EE90; color: black; font-weight: bold;'
    else:
        color = ''
    return color
# Apply border and color styling to the DataFrame
styled_df = df[['GSR', 'EMG', 'Isolation_Anomaly', 'Autoencoder_Anomaly',
'Final_Anomaly']].style \
    .applymap(color_anomalies, subset=['Isolation_Anomaly', 'Autoencoder_Anomaly',
'Final_Anomaly']) \
    .set_table_styles([{'selector': 'th, td', 'props': [('border', '1px solid black')]}])
# Display styled DataFrame in Colab
display(styled_df)
# Visualization 1: Anomaly Distribution (Pie Chart)
fig = px.pie(df, names='Final_Anomaly', title="Anomaly Distribution",
color_discrete_sequence=["#90EE90", "#FF6347"])
```

```
fig.update_traces(textinfo='percent+label', pull=[0.1, 0])
fig.show()
# Visualization 2: GSR and EMG Over Time with Anomalies
plt.figure(figsize=(14, 8))
plt.plot(df.index, df['GSR'], label="GSR", color="blue", linewidth=1.5)
plt.plot(df.index, df['EMG'], label="EMG", color="green", linewidth=1.5)
plt.scatter(df[df['Final_Anomaly'] == 'Anomaly'].index, df[df['Final_Anomaly'] ==
'Anomaly']['GSR'], color='red', label='GSR Anomaly', marker='o')
plt.scatter(df[df['Final_Anomaly'] == 'Anomaly'].index, df[df['Final_Anomaly'] ==
'Anomaly']['EMG'], color='purple', label='EMG Anomaly', marker='x')
plt.title("GSR and EMG Over Time with Anomalies")
plt.xlabel("Time")
plt.ylabel("Sensor Value")
plt.legend()
plt.show()
# Visualization 3: Heatmap of GSR and EMG values
plt.figure(figsize=(12, 6))
sns.heatmap(df[['GSR', 'EMG']].transpose(), cmap='coolwarm', annot=True, cbar=True)
plt.title("Heatmap of GSR and EMG Sensor Values")
plt.show()
# Visualization 4: Correlation Heatmap
correlation_matrix = df[['GSR', 'EMG', 'GSR_norm', 'EMG_norm']].corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='Blues', fmt=".2f")
plt.title("Correlation Heatmap of GSR and EMG Sensors")
plt.show()
# Visualization 5: Pairplot with Anomaly Highlight
sns.pairplot(df, hue="Final_Anomaly", vars=["GSR", "EMG"], palette={'Normal': 'green',
'Anomaly': 'red'})
plt.suptitle("Pairplot of GSR and EMG with Anomaly Highlight", y=1.02)
plt.show()
# Visualization 6: Distribution of Sensor Values (KDE Plot)
plt.figure(figsize=(14, 6))
sns.kdeplot(df['GSR'], label='GSR', fill=True, color="blue")
sns.kdeplot(df['EMG'], label='EMG', fill=True, color="green")
plt.title("Distribution of GSR and EMG Sensor Values")
```

```
plt.xlabel("Sensor Value")
plt.legend()
plt.show()
```

#Early Indication of Disorders:

```
import gspread
import pandas as pd
from google.colab import auth
from google.auth import default
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
from tabulate import tabulate
import xgboost as xgb
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import display, HTML
auth.authenticate_user()
creds, _ = default()
gc = gspread.authorize(creds)
sheet_url =
"https://docs.google.com/spreadsheets/d/1Eto7AfjxMDvJVw5Pq9YjRK_5VEatQUxU2Slo
Iqug4eY/edit?gid=0#gid=0"  # Replace with your sheet URL
worksheet = gc.open_by_url(sheet_url).sheet1
data = worksheet.get_all_records()
df = pd.DataFrame(data)
df['GSR'] = pd.to_numeric(df['GSR'], errors='coerce').ffill().bfill()
df['EMG'] = pd.to_numeric(df['EMG'], errors='coerce').ffill().bfill()
# Normalize GSR and EMG
scaler = MinMaxScaler()
df[['GSR', 'EMG']] = scaler.fit_transform(df[['GSR', 'EMG']])
def label_health_condition(row):
    if row['GSR'] > 0.6 and row['EMG'] > 0.6:
        return "ASD"
    elif row['GSR'] > 0.5 and row['EMG'] < 0.5:
        return "Cognitive Impairment"
```

```
    elif row['GSR'] < 0.3 and row['EMG'] > 0.5:
        return "ADHD"
    # elif row['GSR'] > 0.4 and row['EMG'] > 0.4:
    #     return "PTSD/Trauma"
    else:
        return "Normal"
df['HealthCondition'] = df.apply(label_health_condition, axis=1)
# Oversampling (same as before)
for condition in df['HealthCondition'].value_counts().index:
    while df['HealthCondition'].value_counts()[condition] < 10:
        df = pd.concat([df, df[df['HealthCondition'] == condition]])
label_encoder = LabelEncoder()
df['HealthConditionEncoded'] = label_encoder.fit_transform(df['HealthCondition'])
X = df[['GSR', 'EMG']]
y = df['HealthConditionEncoded']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
xgb_clf = xgb.XGBClassifier(use_label_encoder=False, eval_metric='mlogloss')
xgb_clf.fit(X_train, y_train)
y_pred = xgb_clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
class_report = classification_report(y_test, y_pred, target_names=label_encoder.classes_,
output_dict=True)
# Display performance summary in a table
performance_data = [["Metric", "Value"], ["Accuracy", f"{accuracy * 100:.2f}%"]]
for condition, metrics in class_report.items():
    if isinstance(metrics, dict):
        performance_data.append([f"{condition} Precision", f"{metrics['precision']:.2f}"])
        performance_data.append([f"{condition} Recall", f"{metrics['recall']:.2f}"])
        performance_data.append([f"{condition} F1 Score", f"{metrics['f1-score']:.2f}"])
print("\nHealth Condition Prediction Performance Summary:\n")
print(tabulate(performance_data, headers="firstrow", tablefmt="fancy_grid"))
# Add model predictions to DataFrame for visualization
df['PredictedHealthCondition'] =
label_encoder.inverse_transform(xgb_clf.predict(df[['GSR', 'EMG']]))
# Ensure unique indices to avoid errors in styling
df = df.reset_index(drop=True)
```

```
# Function to color-code health conditions in the table
def color_health_condition(val):
    if val == "ASD":
        return 'background-color: #FFCDD2; color: black'  # Light Red
    elif val == "Cognitive Impairment":
        return 'background-color: #BBDEFB; color: black'  # Light Blue
    elif val == "ADHD":
        return 'background-color: #FFF9C4; color: black'  # Light Yellow
    # elif val == "PTSD/Trauma":
    #     return 'background-color: #C8E6C9; color: black'  # Light Green
    else:
        return 'background-color: #E0E0E0; color: black'  # Light Gray for Normal
# Apply styling to display original and predicted conditions
styled_df = df[['Timestamp', 'GSR', 'EMG', 'HealthCondition',
'PredictedHealthCondition']].style.applymap(
    color_health_condition, subset=['HealthCondition',
'PredictedHealthCondition']).set_table_styles([
    {'selector': 'thead th', 'props': [('background-color', '#4CAF50'), ('color', 'white'), ('font-
weight', 'bold')]},
    {'selector': 'tbody td', 'props': [('border', '1px solid black')]}
])
# Display the styled table
display(styled_df)
# Visualization: Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="coolwarm",
xticklabels=label_encoder.classes_, yticklabels=label_encoder.classes_)
plt.title("Confusion Matrix for Health Condition Prediction")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()
```

APPENDIX B

RKCOLLEGEOFENGINEERING

(AUTONOMOUS)

Kethanakonda (V),Ibrahimpatnam(M),Vijayawada,AMARAVATI–521456

A National Conference on

Advanced Trends in Engineering Science & Technology (ATEST-2025)

Certificate of Presentation

This is to certify that, Prof. / Dr. / Mr. K. Lokesh Babu of Gudlavalleru Engineering College has presented a paper entitled Smart Stress Monitoring System Using IoT and Machine Learning at a National Conference on Advanced Trends in Engineering Science & Technology(ATEST-2025) organized by IQAC, R K College of Engineering on 11th and 12thApril 2025.The academician's scholarly participation is highly appreciated.

Coordinator **Convener** **Principal**

Smart Stress Monitoring System Using IoT and Machine Learning

Dr.Y. Syamala, K. Lokesh Babu, B. Suma, Y. Jyothi, T. Yeswanth Babu
Department of Internet of Things
Seshadri Rao Gudlavalleru Engineering College
Gudlavalleru, Andhra Pradesh, India
kattalokeshbabuk@gmail.com

Abstract **– Stress is increasingly recognized as a critical factor affecting mental and physical health, especially in children with mental disorders and elderly individuals, where stress symptoms are often difficult to identify. This project presents a compact and integrated stress monitoring system leveraging IoT and machine learning (ML) to track physiological stress indicators in real time. Designed to measure key parameters such as Electromyography (EMG) and Galvanic Skin Response (GSR), the device provides continuous monitoring to assess stress levels accurately. EMG signals help detect muscle tension, a primary indicator of stress, while elevated GSR readings reflect emotional arousal and heightened sympathetic nervous system activity. By combining on-site data collection with ML-driven anomaly detection techniques, the device identifies unusual stress patterns and predicts possible stress-related disorders. The collected data is visualized on cloud platforms, enabling caregivers to monitor stress levels remotely. This system empowers users with timely feedback and helps in identifying stress triggers, making it particularly useful in environments where individuals may struggle to communicate their emotional state. Targeted toward children with mental disabilities and elderly individuals, this project supports improved mental health management and early intervention strategies.**

Keywords **– IoT-Based Health Monitoring, Stress Monitoring, Machine Learning, Galvanic Skin Response, Electromyography.**

I. Introduction

Stress-related disorders are a growing global concern, affecting both mental and physical health [4]. Conventional health monitoring systems rely on basic physiological metrics such as heart rate and blood pressure, which do not effectively capture stress-related responses. The integration of Galvanic Skin Response (GSR) and Electromyography (EMG) offers a more comprehensive approach by analyzing autonomic and muscular responses linked to stress [3].The emergence of Internet of Things (IoT) technology has enabled real-time physiological monitoring through wearable devices. Combined with machine learning (ML), IoT systems can efficiently process large volumes of physiological data, allowing for accurate classification, anomaly detection, and early stress prediction [2]. Despite these advancements, challenges such as data privacy, cloud dependency, and adaptability to real-world conditions remain.

This paper proposes an IoT-ML-based stress monitoring system utilizing ESP8266 microcontrollers to collect GSR and EMG signals in real-time. The data is visualized on an IoT dashboard, logged into Google Sheets, and analyzed using deep learning models in Google Colab, including Isolation Forest, Autoencoders, and XGBoost [6]. This dual-layer approach ensures real-time monitoring and predictive insights, making stress detection more accessible and actionable. The proposed system aims to address existing gaps by offering a low-cost, scalable, and privacy-conscious solution for stress assessment.

II. Background

The surveillance of physiological signals is essential in healthcare, providing significant insights into both physical and mental well-being [9]. With the increasing prevalence of stress-related diseases such as Autism Spectrum Disorder(ASD) and Attention Deficit Hyperactivity Disorder(ADHD), there is a greater demand for real-time monitoring systems. Traditional methods typically monitor heart rate and blood pressure, but they do not capture the complex stress reactions needed for accurate evaluation. Galvanic Skin Response (GSR) and Electromyography (EMG) have developed as important physiological markers, providing more insight into autonomic and muscle responses to stress [3].The Internet of Things (IoT) has revolutionized health monitoring by facilitating real-time data gathering and transfer via wearable devices. When combined with machine learning (ML), IoT systems can evaluate massive amounts of physiological data to recognize trends, categorize stress levels, and identify abnormalities with high accuracy [2]. However, issues like data privacy, cloud dependence, and real-world adaptation persist. This research tackles these shortcomings by providing an IoT-ML-based stress monitoring system that uses GSR and EMG to improve mental health evaluation [6].

III. Motivation and Ojectives

1. Motivation

Current IoT-enabled health monitoring systems frequently fail to provide the thorough insights required for evaluating stress and mental health, predominantly concentrating on conventional measures such as heart rate while neglecting Galvanic Skin Response (GSR) and Electromyography (EMG). These systems usually rely on cloud processing, which

increases latency and raises privacy problems, and they struggle to react to real-world data variation [6]. As a result, devices capable of using GSR and EMG signals are required [3].

To solve these restrictions, our study proposes a novel IoT-based deep learning system optimized for GSR and EMG data [2]. Using low-cost IoT sensors and Google Colab, our solution offers real-time monitoring and advanced data processing, such as anomaly detection and health status categorization [5]. This strategy provides a more accurate, dependable, and proactive solution for stress and mental health monitoring in modern healthcare by using hybrid deep learning techniques and prioritizing privacy through reduced cloud dependence [12].

2. Objectives

The main aim of this project is to create an IoT-enabled health monitoring system that uses machine learning to analyze Galvanic Skin Response (GSR) and Electromyography (EMG) signals for the real-time evaluation of stress and mental health issues. The specific objectives include:

2.1 Real-Time Monitoring of GSR and EMG: Design and implement a system to capture GSR and EMG signals using low-cost IoT devices like the ESP8266, ensuring real-time data acquisition and visualization [9].

2.2 Anomaly Detection: Utilize advanced machine learning models, such as Isolation Forests and Autoencoders, to detect anomalies in GSR and EMG signals, ensuring accurate identification of irregular physiological patterns [3].

2.3 Health Condition Classification: Develop a classification framework using machine learning models like XGBoost to identify conditions such as stress, (ASD) and ADHD, based on GSR and EMG data [2].

2.4 Integration with IoT Dashboard: Create an intuitive IoT dashboard for real-time visualization of GSR and EMG data, allowing immediate access to insights for healthcare providers and users.

2.5 Data Logging and Historical Analysis: Implement a robust system to log data into Google Sheets for long-term storage and historical trend analysis, enabling better understanding and management of health conditions.

2.6 Machine Learning Analysis in Google Colab: Leverage Google Colab for advanced machine learning analysis, enabling high-accuracy anomaly detection and health condition classification [5].

2.7 Predictive Health Analytics: Incorporate predictive modeling to forecast potential stress-related events or mental health deterioration, facilitating early intervention and preventive care.

2.8 Cost-Effective and Scalable Architecture: Develop a cost-effective system using readily available sensors and open-source tools, ensuring scalability and accessibility for diverse user groups.

IV. Proposed Methods

1. System Architecture

The suggested system is an IoT-enabled framework for real-time monitoring of stress, capturing Galvanic Skin Response (GSR) and Electromyography (EMG) data. It is comprised of low-cost IoT sensors, an ESP8266 microcontroller, and a cloud-based processing pipeline for anomaly detection and stress categorization [3]. The device is designed to continuously monitor physiological data, allowing for early diagnosis of stress-related disorders [2].

The acquired physiological data is shown in real-time via ThingsBoard Cloud, allowing users and healthcare professionals to dynamically monitor stress levels [6]. The data is then analyzed using machine learning models such as Isolation Forest, Autoencoders, and XGBoost to detect abnormal stress patterns and diagnose mental health disorders such as Autism Spectrum Disorder (ASD) and ADHD [5].To maintain efficiency and security, the system uses optimized data handling techniques, such as edge processing on ESP8266, which reduces latency and cloud dependence [12]. Furthermore, secure communication protocols are employed to safeguard sensitive health data, resulting in a dependable, scalable, and cost-effective solution for real-world healthcare applications [8].

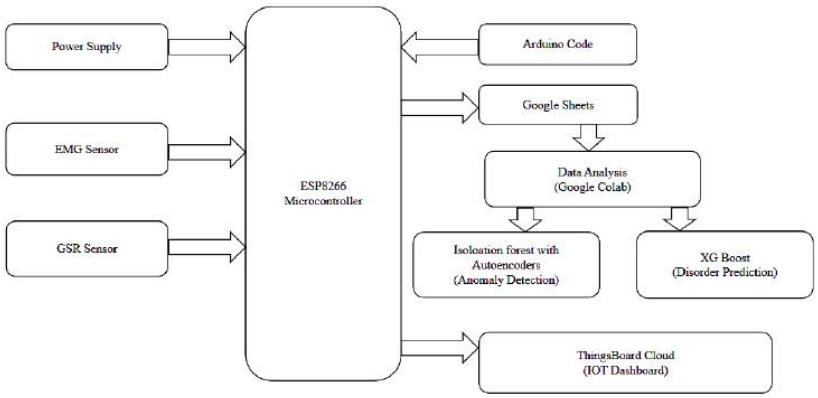


Fig.1. System Architecture

1.1 ESP8266 Microcontroller: The ESP8266 is an economical, Wi-Fi-capable microcontroller extensively utilized in IoT applications owing to its high efficiency, tiny dimensions, and integrated networking features. It has a 32-bit Tensilica L106 CPU, runs at 80 MHz, and includes TCP/IP stack integration, making it perfect for real-time data transfer. With many GPIO ports, it provides for smooth sensor connection while using little power, making it ideal for continuous monitoring applications.

In this project, the ESP8266 is the primary processing unit, gathering real-time GSR and EMG sensor data and sending it to the cloud for additional analysis. It conducts the first data preprocessing, ensuring that only relevant physiological data is delivered to the ThingsBoard Cloud for display and Google Sheets for recording. Furthermore, the ESP8266 supports edge computing, minimizing the system's dependency on cloud-based processing through on-device filtering of sensor information. This solution reduces latency, optimizes battery usage, and improves data security by processing sensitive health data locally before transmitting it to Google Colab for machine learning-based stress analysis.

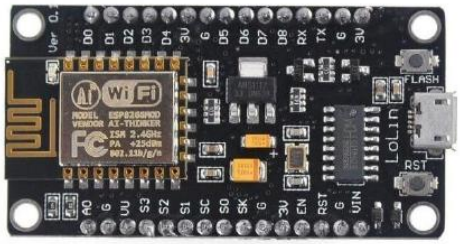


Fig.2. ESP8266 Microcontroller

1.2 Galvanic Skin Response Sensor: The Galvanic Skin Response (GSR) sensor is a physiological sensor that measures electrical conductance changes in the skin as a result of sweat gland activity controlled by the autonomic nervous system. When a person is stressed, agitated, or anxious, perspiration production rises, resulting in increased skin conductivity. The GSR sensor detects these changes by delivering a tiny electrical current through the skin and measuring its resistance, which is inversely proportional to stress levels. It is frequently utilized in stress management, deception detection, and psychological research because of its capacity to give objective insights into emotional and cognitive processes.

In this research, the GSR sensor is critical for stress detection since it continually measures skin conductance fluctuations. The gathered data is transferred to the ESP8266 microcontroller, which analyzes it before sending it to the ThingsBoard Cloud for real-time viewing. Simultaneously, the data is entered into Google Sheets for historical trend analysis. Google Colab's machine learning models use Isolation Forest and Autoencoders to evaluate GSR data for aberrant stress patterns, while XGBoost classifies stress levels and predicts disorders such as ASD and ADHD. By combining GSR data with EMG measurements, the method improves stress assessment accuracy, allowing for early identification and intervention for mental health monitoring.

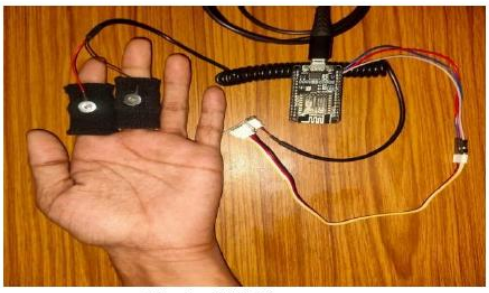


Fig.3. GSR Sensor

1.3 Electromyography Sensor: The Electromyography (EMG) sensor monitors electrical activity caused by muscular contractions. When a muscle contracts, it produces little electrical impulses that the EMG sensor monitors using surface electrodes. EMG is utilized extensively in biomedical research, neuromuscular diagnosis, rehabilitation, and stress monitoring. Because muscle activity is impacted by stress, tension, and neurological diseases, EMG sensors aid in the analysis of muscle response patterns, providing vital information about an individual's physical and mental health.

In this research, the EMG sensor is critical for stress detection since muscular tension rises under stress, resulting in larger EMG amplitude readings. The ESP8266 microcontroller captures, analyzes, and transmits EMG data to ThingsBoard Cloud for real-time viewing, as well as logging it in Google Sheets for historical study. Google Colab's machine learning models examine EMG data, employing Isolation Forest and Autoencoders to find anomalies and XGBoost to diagnose ASD and ADHD. The method enhances stress assessment accuracy by merging EMG and GSR data, allowing for real-time monitoring and early intervention for those facing mental health difficulties.

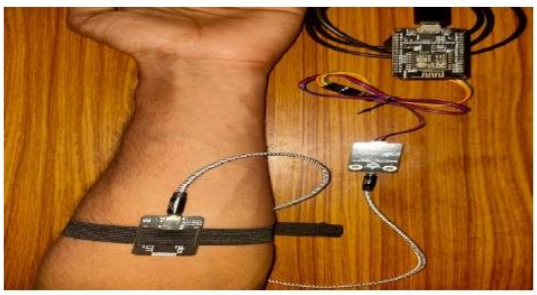


Fig.3. EMG Sensor

1.4 Arduino IDE: The Arduino IDE (Integrated Development Environment) is an open source software used to write, compile, and upload code for microcontrollers such as the ESP8266. It provides a simple and user-friendly interface that supports both C and C++ programming, and is therefore ideal for the development of embedded systems. The IDE incorporates an embedded code editor, a serial monitor, and a large pre-built library of functions that make it easier to communicate with hardware components such as sensors, displays, and actuators. In addition, it supports various microcontroller boards, enabling developers to develop and debug applications for the Internet of Things efficiently. In this project, the ESP8266 microcontroller is programmed with the Arduino IDE, which allows it to collect, process and transmit GSR and EMG sensor data. Code sent in the Arduino IDE configures sensor connections, Wi-Fi communications, and data transfer protocols, ensuring a seamless integration with the ThingsBoard Cloud for real-time visualization and the Google Sheet for data logging. ESP8266 is also programmed to preprocess sensor data to reduce noise and improve accuracy before sending it to Google Colab for machine learning analysis. Using the Arduino IDE, the system provides efficient control of sensors, real-time data processing, and reliable connection to the Internet of Things for stress monitoring and mental health evaluation.

1.5 ThingsBoard Cloud: ThingsBoard Cloud is an open source platform for IoT devices, data collection, and visualization in real time. It supports several communication protocols, such as MQTT, HTTP and CPA, which makes it ideal for applications for the Internet of Things. Thanks to its customizable dashboard, real-time plotting, and notification system, ThingsBoard allows users to track sensor data in an efficient way. It offers secure data management, user-friendly UI components and scalability, which makes it the preferred choice for monitoring systems running on the Internet of Things.In this project, the ThingsBoard Cloud is used exclusively for real-time visualization of GSR and EMG data. The ESP8266 microcontroller transmits raw physiological signals to the platform, which allows users and health care professionals to dynamically monitor the level of stress. The dashboard provides live graphs, numerical data and trend analysis to help you interpret stress changes quickly. By integrating the ThingsBoard, the system provides a clear, interactive and remote display of GSR and EMG signals, enabling users to monitor their physiological response without the need for complicated setup.

1.6 Google Sheets and Google Colab: Google Sheets is a cloud-based spreadsheet tool that enables real-time data logging and structured storage of incoming sensor readings. It allows seamless integration with IoT applications, making it a suitable platform for storing, organizing, and analyzing physiological data over time. Google Colab, on the other hand, is a cloud-based Python development environment that provides a powerful machine learning (ML) platform with built-in support for deep learning frameworks such as TensorFlow and Scikit-learn. Colab enables high-performance computations without requiring local resources, making it ideal for processing large datasets and running ML models efficiently.

In this project, Google Sheets serves as the primary data storage unit, where GSR and EMG signals are continuously logged for historical analysis. The raw sensor data is then fetched into Google Colab, where advanced ML models—including Isolation Forest, Autoencoders, and XGBoost—are applied for anomaly detection and stress classification. This combination ensures real-time data storage, seamless cloud-based ML processing, and accurate stress pattern analysis, making the system scalable, cost-effective, and efficient for long-term mental health monitoring.

2. Machine Learning Models

The proposed system utilizes machine learning (ML) models to analyze Galvanic Skin Response (GSR) and Electromyography (EMG) signals, enabling both anomaly detection and disorder prediction. The anomaly detection phase employs Isolation Forest and Autoencoders to identify irregular physiological patterns, ensuring early detection of unusual stress responses. Once anomalies are detected, the disorder prediction phase classifies the likelihood of stress-related conditions such as Autism Spectrum Disorder (ASD) and ADHD using XGBoost, a powerful gradient-boosting algorithm. These models are implemented in Google Colab, leveraging cloud-based computing for efficient and high-accuracy analysis. By integrating real-time anomaly detection and disorder classification, the system enhances early intervention capabilities, making it a scalable, accurate, and cost-effective solution for mental health monitoring.

2.1 Isolation Forest: Isolation forest (IF) is an unsupervised machine learning algorithm specially designed for anomaly detection. It works on the principle that anomalies (outsiders) are easier to isolate than regular data points. Unlike traditional approaches based on clustering or density, the Isolation Forest does not calculate distance or density; it randomly partitions the data to quickly isolate anomalies.

The Isolation Forest algorithm detects anomalies by exploiting the principle that anomalies are easier to isolate than normal data points. The process starts with a random sub-sample selection where the data set is randomly split into several subsets. Within each subset, the algorithm performs recursive partitioning, selecting a random function and dividing it by a range within the range in which the data is distributed. This recursive process creates an iTree in which fewer splits indicate a higher probability of an anomaly. The path length of the data point is then calculated as the number of splits needed to isolate it, with anomalies having shorter path lengths because they are isolated more rapidly than normal data points.The anomaly score for a data point x is given by:

$$s(x,n) = 2^{\frac{-E(h(x))}{c(n)}} \quad (1)$$

where s(x,n) represents the anomaly score, E(h(x)) is the average path length across multiple isolation trees, and c(n) is the average path length of a balanced binary tree of size n. Based on this score, decision-making is performed: a higher anomaly score (closer to 1) indicates that the point is an anomaly, while a lower score (closer to 0) suggests that the point is normal. This approach allows Isolation Forest to efficiently detect outliers in high-dimensional datasets with minimal computational overhead.

2.2 Autoencoder: An Autoencoder is a type of neural network designed to learn to represent input data efficiently by compressing it into smaller spaces and then reconstructing it. It consists of two main components: an encoder which reduces the dimensionality of the input and extracts the most important elements, and a decoder which reconstructs the original input from this compressed representation. The network is trained to minimize the difference between input and reconstructed output, which means that the reconstruction error is kept to a low level for normal data. However, when the model encounters an unusual or invisible pattern, it struggles to reconstruct it accurately, leading to greater reconstruction errors. This principle is used to detect anomalies in your project, when the system detects stress-related abnormalities in the galvanic skin response (GSR) and electromyography (EMG) signals based on the reconstruction loss.
Mathematically, given an input X, the encoder maps it to a latent space representation h using the function:

$$h = f(x) = \sigma\left(w_e x + b_e\right) \quad (2)$$

where We and be are the weights and biases of the encoder, and σ is the activation function. The decoder then reconstructs the input as X′ through:

$$x' = g(x) = \sigma\left(w_d h + b_d\right) \quad (3)$$

where Wd and bd are the weights and biases of the decoder. The difference between X and X′ is measured using Mean Squared Error (MSE), given by:

$$L = \frac{1}{n}\sum\left(x - x'\right)^2 \quad (4)$$

If the reconstruction loss L exceeds a set threshold (typically the 95th percentile), the data point is flagged as an anomaly. In our project, the Autoencoder is trained on normal GSR and EMG values. During real-time monitoring, it processes incoming signals and reconstructs them. If the MSE is high, indicating an unusual pattern, the instance is classified as an anomaly. By integrating Autoencoders with Isolation Forest, your system ensures accurate stress anomaly detection while minimizing false positives, making it highly robust for real-time stress monitoring applications.

In this project, Isolation Forest and Autoencoders are working together on a hybrid approach to anomaly detection, which allows for accurate detection of stress-related GSR and EMG signals. The Isolation tree detects anomalies by analyzing how easy it is to isolate a data point, while the Autoencoders learn normal patterns and flag instances with high reconstruction errors as anomalous. By combining these techniques, the system uses both statistical and deep learning methods to ensure high accuracy and robustness in the detection of stress-related abnormalities. This integrated approach improves real-time monitoring of stress, allowing early detection of abnormal physiological responses and making the system more robust and adaptable to the real-life application of health care.

2.3 Extreme Gradient Boosting (XGBoost): Extreme Gradient Boosting (XGBoost) is a strong supervised machine learning technique based on gradient boosting that is designed for speed, efficiency, and accuracy. It is commonly used for classification and regression problems, with strong performance on big datasets containing missing values and complicated patterns. Unlike classic decision tree models, XGBoost creates an ensemble of weak learners (decision trees) in a sequential fashion, with each tree correcting the flaws of its predecessor. This procedure continues until the model achieves optimal accuracy, making XGBoost an excellent choice for non-linear connections. In this study, XGBoost is used to forecast disorders by evaluating Galvanic Skin Response (GSR) and Electromyography (EMG) data and classifying stress-related illnesses such as Autism Spectrum Disorder (ASD) and Attention Deficit Hyperactivity Disorder (ADHD).

XGBoost employs the Gradient Boosting Decision Trees (GBDTs) technique, which involves training multiple trees progressively to minimize errors at each stage. The first tree produces an initial forecast, which is then refined by succeeding trees that focus on residual mistakes. The approach computes a loss function, uses gradients to determine the direction of progress, and grows new trees to reduce mistakes even further. Finally, the outputs of all trees are merged in a weighted summation to create a powerful classifier. XGBoost's mathematical formulation minimizes the goal function:

$$Obj = \sum_{i=1}^{n} L\left(yi - yi\right) + \sum_{k=1}^{k} \Omega\left(f_k\right) \qquad (5)$$

where:

L(yi,yi') is the loss function, which quantifies the difference between actual and predicted values.

Ω(fk) is the regularization term, which controls model complexity and prevents overfitting.

The final prediction is computed as:

$$y = \sum_{k=1}^{k} f_k\left(x\right) \qquad (6)$$

where fk(x) represents the output of each decision tree in the ensemble.

XGBoost effectively models complex dependencies between GSR and EMG signals. Feature importance ranking identifies significant physiological signals. Regularization techniques prevent overfitting. High efficiency allows for real-time stress classification. XGBoost is integrated with gaussian matrix model(GMM) for disorder classification. Data acquisition, preprocessing, and clustering are performed. Feature engineering extracts key statistical properties. XGBoost classifier is trained and evaluates model performance. System combines GMM and XGBoost for accurate stress disorder prediction.

V. RESULTS

1. Anomaly Detection

Anomalies detected in Galvanic Skin Response (GSR) and Electromyography (EMG) signals serve as indicators of stress patterns. The use of Isolation Forest and Autoencoder algorithms proved effective in identifying these anomalies. For instance, the Isolation Forest algorithm utilized 100 estimators and a 5% contamination rate to pinpoint anomalies in the data. Visualizations such as heatmaps provided a clear confirmation of these anomalies. On the other hand, the Autoencoder algorithm was able to learn normal patterns by employing a two-layer encoder and a linear decoder. The calculation of reconstruction errors using Mean Squared Error (MSE) allowed for the identification of anomalies based on the 95th percentile of MSE values.

By combining the results from both Isolation Forest and Autoencoder, the sensitivity of anomaly detection was significantly enhanced. This fusion approach was further illustrated through the visualization of anomaly distribution using a pie chart. Additionally, time-series graphs were utilized to track the fluctuations in GSR and EMG over time, showcasing the effectiveness of both algorithms in detecting stress-related anomalies. The identification of these anomalies is crucial for predicting disorders, especially in the context of stress monitoring for children with mental disabilities and individuals diagnosed with ADHD and Autism Spectrum Disorder (ASD). Overall, the utilization of Isolation Forest and Autoencoder algorithms has demonstrated their efficacy in detecting and analyzing stress-related anomalies, providing valuable insights for further research and applications in the field.

	GSR	EMG	Isolation_Anomaly	Autoencoder_Anomaly	Final_Anomaly
0	174.000000	507.000000	Normal	Normal	Normal
1	182.000000	529.000000	Normal	Normal	Normal
2	182.000000	474.000000	Normal	Normal	Normal
3	182.000000	499.000000	Normal	Normal	Normal
4	182.000000	524.000000	Normal	Normal	Normal
5	182.000000	495.000000	Normal	Normal	Normal
6	181.000000	522.000000	Normal	Normal	Normal
7	182.000000	520.000000	Normal	Normal	Normal
8	182.000000	495.000000	Normal	Normal	Normal
9	181.000000	498.000000	Normal	Normal	Normal
10	175.000000	495.000000	Normal	Normal	Normal

Fig.4. Subject is in Normal Condition

100	53.000000	504.000000	Normal	Normal	Normal
101	61.000000	617.000000	Normal	Normal	Normal
102	61.000000	974.000000	Anomaly	Anomaly	Anomaly
103	57.000000	456.000000	Normal	Normal	Normal
104	57.000000	796.000000	Normal	Normal	Normal
105	58.000000	796.000000	Anomaly	Normal	Anomaly
106	57.000000	764.000000	Normal	Normal	Normal
107	69.000000	974.000000	Anomaly	Anomaly	Anomaly
108	62.000000	974.000000	Anomaly	Anomaly	Anomaly
109	62.000000	497.000000	Normal	Normal	Normal
110	60.000000	494.000000	Normal	Normal	Normal

Fig.5. Subject is in Stressed Condition

The anomaly detection findings reveal significant insights on the stress monitoring system utilizing GSR and EMG sensor data. The correlation heatmap depicts the correlations between raw and normalized sensor readings, with GSR and EMG exhibiting a modest negative correlation, indicating unique stress response patterns. The anomaly distribution pie chart shows that 7.75% of the collected data points are classed as anomalies, indicating probable stress-related occurrences. The time-series display reveals more abnormalities, including rapid spikes and decreases in GSR and EMG readings. The discovery of such abnormalities is critical for early intervention because they may signify periods of high stress or emotional instability in people with ADHD and ASD. This investigation validated the efficiency of the Isolation Forest and Autoencoder models.

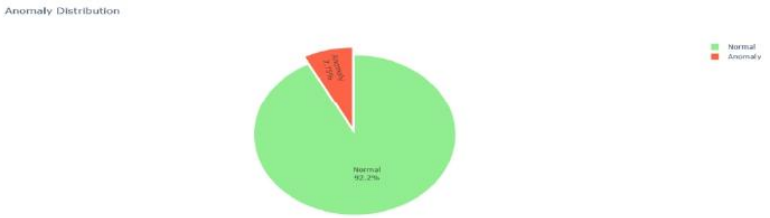


Fig.6. Anomaly Distribution Pie Chart

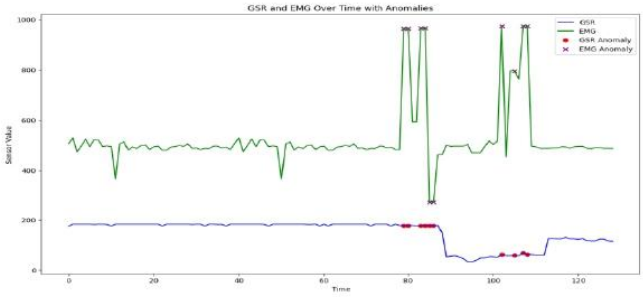


Fig.7.GSR and EMG Over Time with Anomalies

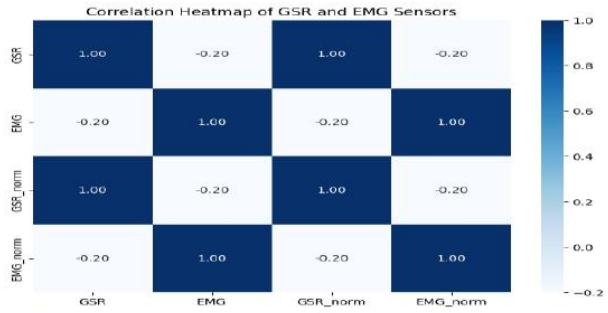


Fig.8.Correlation Heatmap of GSR and EMG Sensors

2. Disorder Prediction

The disorder prediction model utilizes GSR and EMG sensor data to classify individuals into different health conditions, including ASD, ADHD, Cognitive Impairment, and Normal. The model is built using XGBoost, a powerful gradient-boosting algorithm, to analyze physiological signals and make accurate predictions[5].The data is first preprocessed and normalized using MinMaxScaler to ensure all values are within a uniform range[7]. A labeling function is applied to categorize data points based on GSR and EMG thresholds, mapping them to the corresponding disorders. The labeled data is then split into training and testing sets, with 80% of the data used for training and 20% for evaluation[3].The model achieves an overall accuracy of 94.74%, demonstrating its effectiveness in distinguishing between different stress-related conditions. The classification report further highlights the model's precision, recall, and F1-score, indicating strong performance in identifying each disorder.A confusion matrix heatmap is used to visualize the model's predictions, showing how well it differentiates between conditions. Additionally, the system provides a color-coded table displaying actual vs. predicted conditions, allowing for a clear assessment of the model's decision-making process.

The results suggest that GSR and EMG signals can serve as reliable indicators for detecting stress-related disorders[4]. The model's ability to classify conditions with high accuracy makes it a promising tool for early diagnosis and intervention, especially for individuals with ASD and ADHD[1]. Future improvements could focus on incorporating additional physiological markers and refining classification thresholds for even greater accuracy.

Table I: Thresholds for Health Condition Classification

Health Condition	GSR Threshold	EMG Threshold
ASD	GSR > 0.6	EMG > 0.6
Cognitive Impairment	GSR > 0.5	EMG < 0.5
ADHD	GSR <0.3	EMG > 0.5
Normal	All Other Cases	All Other Cases

Table III: Performance Evaluation

Class	Precision	Recall	F1-Score
ADHD	1.00	1.00	1.00
ASD	1.00	1.00	1.00
Cognitive Impairment	0.96	0.96	0.96
Normal	0.90	0.90	0.90
Macro Avg.	0.97	0.97	0.97
Weighted Avg.	0.95	0.95	0.95
Overall Accuracy	94.74%		

Misclassification happens when a model predicts a class label inaccurately, which means that the anticipated health status differs from the actual condition. This might be due to overlapping characteristics, sensor noise, or difficulties in threshold-based categorization. In the context of this project, false positives (detecting a condition mistakenly when the individual is actually normal) and false negatives (failing to detect a disorder when it exists) might have an influence on the system's dependability.However, misclassification is limited in this project thanks to the XGBoost model's excellent accuracy of 94.74% and great performance. The accuracy and recall scores for each class are consistently high, demonstrating that the model can distinguish between ASD, ADHD, Cognitive Impairment, and Normal conditions. The confusion matrix confirms this by demonstrating that the number of inaccurate guesses is quite low.

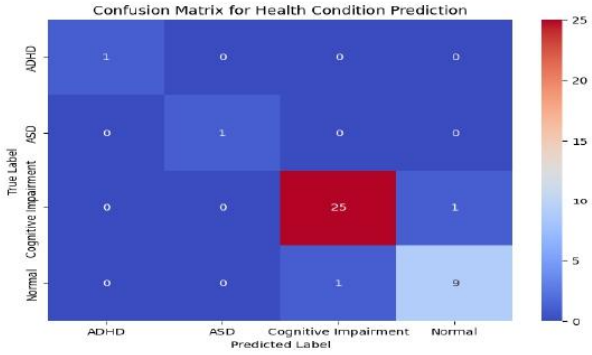


Fig.9.Confusion Matrix for Health Condition Prediction

The results show that the proposed IoT-ML-based stress monitoring system is successful at detecting anomalies as well as classifying disorders. The anomaly detection models (Isolation Forest and Autoencoder) correctly identified stress-related anomalies in GSR and EMG signals, while the XGBoost-based disorder prediction model achieved a 94.74% accuracy in classifying ASD, ADHD, Cognitive Impairment, and Normal conditions. The performance assessment measures, including as accuracy, recall, and F1-score, show high model dependability and low misclassification. The system's capacity to identify physiological stress reactions in real time demonstrates its potential for early diagnosis and intervention, making it an important tool for mental health evaluation.

VI. Conclusion and Future Scope

In this paper, we present an IoT-based stress monitoring system that uses machine learning to identify stress in real time and predict early disorders in children with mental impairments, adults with ADHD and ASD, and the elderly. The system is organized into three layers: data collecting, processing, and intelligence. It has Galvanic Skin Response (GSR) and Electromyography (EMG) sensors that are linked to an ESP8266 microprocessor to detect physiological signals associated with stress. The collected data is transported to cloud platforms, where it is visualized and monitored using Google Sheets and ThingsBoard, allowing for smooth real-time analysis.

To achieve accurate and dependable stress categorization, we used a variety of powerful machine learning approaches. Isolation Forest, paired with Autoencoders, was used to detect anomalies, allowing the system to discover unusual stress patterns. Furthermore, XGBoost was used for predictive modeling, which improved classification accuracy by exploiting its capacity to handle complicated correlations in physiological data. The model was rigorously evaluated using numerous performance measures, including accuracy, recall, and F1-score, demonstrating its ability to identify varying stress levels with high precision.

The findings show that our suggested system is a cost-effective, scalable, and data-driven method to mental health monitoring. The technology is especially useful for people who need continuous monitoring and early intervention since it provides real-time physiological analysis and stress evaluation. Its capacity to assess stress levels remotely and send timely warnings makes it a vital tool for caregivers, healthcare professionals, and academics seeking to enhance the well-being of persons suffering mental health difficulties.

While our approach has shown promising results in real-time stress detection and early disorder prediction, there are various areas for improvement and growth. One significant enhancement is the use of additional physiological sensors, such as heart rate variability (HRV) and electroencephalography (EEG), which can enhance multi-modal stress detection by collecting a wider variety of physiological responses. By adding these additional biosignals, the system may give a more complete and accurate evaluation of stress levels, enhancing both short-term monitoring and long-term trend analysis. Furthermore, optimizing machine learning algorithms is critical for improving classification accuracy and decreasing the number of false positives and false negatives. Using improved feature selection techniques, fine-tuning hyperparameters, and ensemble learning algorithms can dramatically enhance prediction reliability. Furthermore, using real-time adaptive learning methods might allow the system to grow over time depending on user-specific data, making stress detection more personalized and precise.Advanced deep learning models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), can help improve stress pattern identification. These designs excel in extracting complicated patterns from time-series physiological data, allowing the system to detect small fluctuations in stress reactions with greater precision. The system can go beyond traditional categorization models and detect stress patterns over longer periods of time by utilizing deep learning.

This program not only solves the limits of existing stress monitoring methods, but it also establishes a solid platform for future advances in mental health care. By providing real-time, data-driven insights and early intervention capabilities, the system has the potential to revolutionize how stress-related disorders are identified and managed. As technology advances, combining AI-driven analytics with IoT-based monitoring can empower individuals, caregivers, and healthcare professionals, resulting in enhanced mental health and proactive stress management measures.

REFERENCES

[1] Yali Zheng, Chen Wu, Peizheng Cai, Zhiqiang Zhong, Hongda Huang, Yuqi Jiang, "Tiny-PPG: A Lightweight Deep Neural Network for Real-Time Detection of Motion Artifacts in Photoplethysmogram Signals on Edge Devices," *Internet of Things*, Vol. 25, 2024.

[2] Zhu, L., Spachos, P., Ng, P. C., Yu, Y., Wang, Y., Plataniotis, K., & Hatzinakos, D., "Stress Detection Through Wrist-Based Electrodermal Activity Monitoring and Machine Learning," *IEEE Journal of Biomedical and Health Informatics*, Vol. 27, No. 5, pp. 2155-2165, 2023.

[3] Patil, A. N., Preeti, B. M., & Laxmi, "Real-Time Stress Level Monitoring Using IoT," *International Conference on Integrated Intelligence and Communication Systems (ICESC)*, 2023.

[4] Ariayudha, R. A., Nuha, H. H., & Irsan, M., "Reading Stress Levels and Setting Emotional Patterns with Sensors Based on Galvanic Skin Response (GSR) Method," *International Conference on Data Science and Its Application (ICoDSA)*, 2023.

[5] Ranjha, Azlaan, Laiba Jabbar, and Osaid Ahmed, "Cloud-Connected Wireless Holter Monitor Machine with Neural Networks Based ECG Analysis for Remote Health Monitoring," 2023.

[6] Giannakakis, G., Grigoriadis, D., Giannakaki, K., Simantiraki, O., Roniotis, A., Tsiknakis, M., "Review on Psychological Stress Detection Using Biosignals," *IEEE Transactions on Affective Computing*, Vol. 13, No. 1, pp. 440-460, Jan.-Mar. 2022.

[7] Pourmohammadi, S., Maleki, A., "Continuous Mental Stress Level Assessment Using Electrocardiogram and Electromyogram Signals," *Biomedical Signal Processing and Control*, Vol. 68, July 2021, 102694.

[8] Memar, M., Mokaribolhassan, A., "Stress Level Classification Using Statistical Analysis of Skin Conductance Signal While Driving," *SN Applied Sciences*, Vol. 3, p. 64, 2021.

[9] Mozafari, M., Firouzi, F., & Farahani, B., "Towards IoT-Enabled Multimodal Mental Stress Monitoring," *Sharif University of Technology, Duke University, Shahid Beheshti University,* 2020.

[10] Elgendi, M., Menon, C., "Machine Learning Ranks ECG as an Optimal Wearable Biosignal for Assessing Driving Stress," *IEEE Access*, Vol. 8, pp. 34362-34374, 2020.

[11] Kyriakou, K., Resch, B., Sagl, G., Petutschnig, A., Werner, C., Niederseer, D., Liedlgruber, M., Wilhelm, F. H., Osborne, T., Pykett, J., "Detecting Moments of Stress from Measurements of Wearable Physiological Sensors," *Sensors*, Vol. 19, p. 3805, 2019.

[12] Pandey, P. S., "Machine Learning and IoT for Prediction and Detection of Stress," *17th International Conference on Computational Science and Its Applications (ICCSA)*, Trieste, Italy, 2017, pp. 1-5.

[13] Boon-Leng, L., Dae-Seok, L., & Boon-Giin, L., "Mobile-Based Wearable-Type of Driver Fatigue Detection by GSR and EMG," *TENCON 2015 - 2015 IEEE Region 10 Conference*, Macao, China, 2015, pp. 1-4.