1. INTRODUCTION

Stress is a complex physiological and psychological response to physical or mental imbalances, or noxious stimuli. It plays a crucial role in maintaining homeostasis by activating the autonomic nervous system (ANS). Chronic stress, however, leads to overactivity in the sympathetic nervous system, resulting in various adverse effects including increased risk of cardiovascular diseases, anxiety disorders, and other stress-related conditions. Heart Rate Variability (HRV), derived from electrocardiogram (ECG) signals, is areliable indicator of stress. HRV measures the variation in time intervals between consecutive heartbeats and typically increases with relaxation and decreases with stress, indicating an inverse relationship with heart rate. This makes HRV an essential biomarker for assessing the impact of stress on the ANS.

The advent of wearable Internet of Medical Things (IoMT) devices has revolutionized HRV monitoring. Devices like Elite HRV, H7, Polar, and Motorola Droid offer practical and cost-effective means for real- time HRV monitoring outside clinical settings, despite minor errors compared to traditional ECG measurements. These advancements have paved the way for more accessible stress monitoring and management.

Despite the development of numerous machine learning (ML) and deep learning (DL) algorithms forstress classification, achieving ultra-high accuracy in multi-class stress detection remains challenging. Existing studies, particularly those using datasets like SWELL-KW, have primarily focused on binary stressclassification and have not reached the desired accuracy levels for multi-class stress classification.

This study addresses this gap by introducing a novel one-dimensional convolutional neural network (1D CNN) model specifically designed for multi-class stress classification. By leveraging HRV features, the model achieves outstanding performance with 99.9% accuracy. Feature optimization techniques, such as theanalysis of variance (ANOVA) F-test, have been employed to identify the most relevant HRV features, reducing computational load during model training without compromising accuracy. This optimization not only enhances the model's efficiency but also ensures its practical applicability for real-time stress detection.

1.1 MOTIVATION

Stress, a pervasive issue in modern society, significantly impacts both physical and mental health. The ability to accurately detect and classify stress is crucial for effective stress management and prevention of related health conditions. Traditionally, stress measurement has relied on subjectivemethods, which may not provide accurate or timely insights. Heart Rate Variability (HRV) presents a reliable, objective biomarker for stress assessment, offering a physiological basis for understandingthe body's response to stressors.

The advent of wearable Internet of Medical Things (IoMT) devices has revolutionized the field of HRV monitoring, making it more accessible and practical. These devices enable real-time, non-invasive stress monitoring, providing valuable data outside clinical settings. Despite these advancements, existing machine learning (ML) and deep learning (DL) models for stress detection, particularly those utilizing the SWELL-KW dataset, have not achieved the desired accuracy for multi-class stress classification.

This study is motivated by the need to bridge this gap in research. By developing a novel one-dimensional convolutional neural network (1D CNN) model, we aim to achieve ultra-high accuracy in multi-class stress detection. This model not only leverages the power of deep learning but also incorporates advanced feature optimization techniques to enhance performance and reduce computational load. The success of this model can significantly improve real-time stress detection and management, providing a robust tool for enhancing mental health and well-being.

1.2 PROBLEM DEFINATION

The accurate detection and classification of stress levels are critical for effective stress management and preventing associated health issues. Traditional methods for stress measurement often rely on subjective assessments, which may not provide timely or accurate insights into an individual's stress state. Heart Rate Variability (HRV) offers a promising objective biomarker for stress assessment, reflecting the body's physiological response to stressors through variations in timeintervals between heartbeats. Despite advances in wearable Internet of Medical Things (IoMT) devices that enable real-time HRV monitoring, existing machine learning (ML) and deep learning (DL) models have not achieved the desired accuracy for multi-class stress classification, particularlyusing the SWELL-KW dataset. Most studies have focused on binary classification (stress vs. non- stress) and have not addressed the complexity of multi-class stress detection.

The challenge lies in developing a model that can accurately classify multiple stress levels with high precision and recall while also being computationally efficient. This requires optimizing HRV feature selection to reduce computational load without sacrificing accuracy. There is a need fora novel approach that leverages advanced deep learning techniques to achieve ultra-high accuracy inmulti-class stress detection using HRV data. This study aims to fill this research gap by developing aone-dimensional convolutional neural network (1D CNN) model for multi-class stress classification.The model will use optimized HRV features to enhance performance, making it a valuable tool for real-time stress detection and management in practical, everyday scenarios.

1.3 OBJECTIVE OF PROJECT

The primary objective of this project is to develop a highly accurate and efficient method formulti-class stress detection using Heart Rate Variability (HRV) data and a one-dimensional convolutional neural network (1D CNN) model. By leveraging HRV features as reliable biomarkers,the project aims to achieve the following specific goals:

1. **Accuracy Improvement:** Develop a 1D CNN model that significantly outperforms existing machine learning (ML) and deep learning (DL) approaches in terms of accuracy for multi-class stress classification. The target is to achieve an ultra-high accuracy rate of 99.9%, ensuring precisestress level detection.

2. **Feature Optimization:** Identify and select the most relevant HRV features through advanced feature optimization techniques, such as the analysis of variance (ANOVA) F-test, to reduce the computational load during model training without compromising accuracy. This will enhance themodel's efficiency and applicability in real-time scenarios.

3. **Practical Application:** Utilize wearable Internet of Medical Things (IoMT) devices to collect HRV data and demonstrate the practicality of the developed model for real-time stress monitoringand management. This includes validating the model's performance with data from various IoMTdevices and ensuring it can be seamlessly integrated into everyday use.

4. **Comparative Analysis:** Conduct a comprehensive evaluation of the 1D CNN model against state-of-the-art ML and DL models using publicly available datasets, such as the SWELL-KW dataset,to establish its superiority in multi-class stress detection.

5. **Impact on Mental Health:** Contribute to the field of stress assessment and mental health by providing a robust, accurate, and efficient tool for stress detection. The successful implementationof this project aims to support better stress management practices and improve overall well-being.

1.4 LIMITATIONS

- **Data Quality and Consistency**: The accuracy of HRV-based stress detection heavily relies on the quality and consistency of the HRV data collected from IoMT devices. Minor errors or inconsistencies in data can affect the model's performance.
- **Generalization**: The 1D CNN model is trained and validated on specific datasets like SWELL-KW. Its ability to generalize across different populations, environments, or activities not represented in the training data may be limited, potentially reducing its effectiveness in diverse real-world scenarios.
- **Feature Dependency**: While feature optimization reduces computational load, it may also lead to the exclusion of potentially relevant features not captured in the selected subset. This could limit themodel's ability to fully understand the complexity of stress responses.
- **Computational Resources**: Despite efforts to reduce computational load, training deep learning models such as 1D CNNs can still be resource-intensive, requiring significant computational power and time, particularly for large datasets or when fine-tuning hyperparameters.
- **Wearable Device Limitations**: Although IoMT devices offer practicality, their accuracy in measuring HRV compared to clinical ECG instruments can vary. Factors such as device placement,movement artifacts, and user compliance can impact data quality.
- **Real-time Application**: Implementing the model for real-time stress detection in wearable devices may present challenges, including the need for continuous data processing and analysis, power consumption, and maintaining user privacy and data security.
- **Interpretability**: Deep learning models, including 1D CNNs, often operate as "black boxes," makingit challenging to interpret how specific inputs influence the outputs. This can limit the ability to provide actionable insights or understand the physiological mechanisms underlying stress detection.
- **Ethical and Privacy Concerns**: Collecting and analyzing HRV data, particularly in real-time applications, raises ethical and privacy concerns. Ensuring data security, user consent, and adherenceto privacy regulations is crucial to addressing these issues.

1.5 ORGANIZATION OF DOCUMENTATION

Organize the documentation for the multi-class stress detection through heart rate variability project ina clear and structured manner. Consider the following sections. This report is organized into majorlyinto 8 different sections and each section provides brief descriptions about the project. The 8 sectionsmentioned are:

Chapter 1-Introduction:

This section provides the overview of the project, what's the major problem that is being addressed,objectives, methodologies and information about information about remaining part of the report.

Chapter 2-Literature Survey:

This section provides previous work of the project and their limitations. Multi-Class stress detection through heart rate variability Introduction.

Chapter 3- Analysis:

System Analysis is the document that describes about the existing system and proposed system inthe project

Chapter 4- Design:

System Design is the document that describes about the project modules, class Diagram, sequence Diagram and use case Diagram detailed in the project.

Chapter 5-Implementation & Results:

Implementation is the document that describes about the detailed concept of the project. It also describes about algorithm with detailed steps.

Chapter 6-Testing & Validation:

Testing is document that describes about unit testing, validation testing, functional testing, integration testing, user acceptance testing.

Chapter 7-Conclusion and Future Enhancement:

It is a document that describes about the brief summary of the project and undetermined events that will occur in the time.

2. LITERATURE SURVEY

2.1 INTRODUCTION

The study of stress detection using physiological signals, particularly Heart Rate Variability (HRV), has garnered significant attention in recent years. The ability to objectively measure and classify stress levels using HRV is crucial for developing effective stress management and intervention strategies. With advancements in wearable Internet of Medical Things (IoMT) devices, real-time monitoring of HRV has become more practical and accessible, paving the way for innovative applications in mental health.

Heart Rate Variability (HRV) serves as a reliable biomarker for stress, reflecting the autonomic nervous system's response to various stressors. As HRV varies inversely with heart rate, it provides valuable insights into an individual's stress state. Numerous machine learning (ML) and deep learning (DL) models have been explored to harness HRV data for stress detection, aiming to improve accuracy and efficiency.

Existing studies have primarily focused on binary stress classification (stress vs. non-stress). However, multi-class stress classification, which provides a more nuanced understanding of differentstress levels, remains a challenging task. The SWELL-KW dataset has been a valuable resource in this field, yet achieving ultra-high accuracy in multi-class stress detection using this dataset has proven difficult.

This literature survey delves into the various approaches and methodologies employed in recent studies on stress detection using HRV. It examines the performance of different ML and DL models, highlights the advancements in feature optimization techniques, and identifies the gaps that necessitate further research. By understanding the current state of the art, this survey aims to inform the development of more accurate and efficient models for multi-class stress detection, ultimately contributing to improved stress management and mental health outcomes.

2.2 EXISTING SYSTEM

The Existing systems for stress detection through Heart Rate Variability (HRV) primarily utilize machine learning (ML) and deep learning (DL) algorithms. These systems have madeconsiderable advancements in both binary and multi-class stress classification by

leveraging physiological data from wearable Internet of Medical Things (IoMT) devices and traditional ECG measurements. Wearable devices like Polar, and Motorola Droid enable real-time HRV monitoring, although they may exhibit minor discrepancies compared to tradition.

Machine learning approaches such as naive Bayes, k-nearest neighbor (KNN), support vectormachine (SVM), multilayer perceptron (MLP), random forest, and gradient boosting have been extensively used for stress classification. For instance, SVM with radial basis function (RBF) achieved accuracy scores of 83.33% using time-domain and frequency-domain HRV features.Publicly available datasets like SWELL-KW and WESAD are crucial for training and validating these models, with multi-class stress classification accuracy ranging from 81.65% to 90%. Furthermore, feature optimization techniques such as the analysis of variance F-test help in selecting the most relevant HRV features, thereby reducing the computational load while maintaining high accuracy. Despite these advances, the quest for ultra-high accuracy in multi-class stress detection continues, necessitating the development of novel ML and DL models that can enhance real-time stressmonitoring applications.

2.3 DISADVANTAGES OF EXISTING SYSTEM

The existing systems for stress detection using Heart Rate Variability (HRV) and machinelearning (ML) or deep learning (DL) algorithms have several disadvantages:

1. **Accuracy of Wearable Devices**: While wearable IoMT devices like Elite HRV, H7, Polar, and Motorola Droid provide practical real-time HRV monitoring, they can have minor discrepancies compared to traditional ECG instruments. These small errors, although acceptable in many cases,can affect the accuracy of stress detection.

2. **Limited Generalization**: Many existing models are trained on specific datasets like SWELL-KW or WESAD. This can limit their ability to generalize across different populations, environments, or activities not represented in the training data.
3. **Computational Requirements**: Training deep learning models, even with optimizations, often requires significant computational resources. This includes high processing power and memory, which can be a barrier to their widespread use, especially in resource-constrained settings.
4. **Binary Classification Focus**: A significant portion of existing studies focuses on binary stress classification (stress vs. non-stress), rather than the more complex multi-class classification.

5. **Feature Dependency**: Optimization techniques for feature selection, while reducing computational load, might exclude potentially relevant HRV features. This can limit the model's ability to fully capture the complexity of stress responses, potentially reducing its overall

2.4 PROPOSED SYSTEM

The proposed system enhances multi-class stress detection accuracy and efficiency using HRV data through a one-dimensional convolutional neural network (1D CNN) model. It captures spatial relationships in HRV data and optimizes feature selection using techniques like the ANOVA F-test to reduce computational load while maintaining high accuracy. Wearable IoMT devices like Elite HRV, H7, Polar, and Motorola Droid enable real-time HRV monitoring, and the collected data is pre-processed to ensure high quality. The system is evaluated using datasets like SWELL-KW, aiming for 99.9% accuracy. The ADAM optimizer improves computational efficiency. Extensive testing ensures the system's robustness and practicality in real-world scenarios, contributing to betterstress management and mental health outcomes.

2.5 CONCLUSION

The development of a one-dimensional convolutional neural network (1D CNN) model for multi-class stress detection using Heart Rate Variability (HRV) data significantly improves stress monitoring. This system leverages deep learning and advanced feature optimization techniques to achieve high accuracy. Real-time HRV data from wearable IoMT devices ensures practical, continuous stress assessment. The ADAM optimizer enhances computational efficiency, making the system suitable for devices with limited processing power. Extensive testing confirms the model's superior performance and robustness in diverse scenarios. Overall, the system offers an efficient toolfor real-time stress detection, contributing to better mental health management and outcomes.

3. ANALYSIS

3.1 INTRODUCTION

In the realm of machine learning and data science, analyzing the performance and efficacy of various classification algorithms is a critical step toward understanding their practical applications and limitations. This analysis involves examining decision tree classifiers, gradient boosting, K- nearest neighbors (KNN), logistic regression, Naïve Bayes, random forests, and support vector machines (SVM). Each of these algorithms possesses unique characteristics and advantages that makethem suitable for specific types of data and problem domains.

Decision tree classifiers, for instance, are lauded for their capability to capture decision-making knowledge from data, making them intuitive and easy to interpret. Gradient boosting enhances prediction accuracy by creating an ensemble of weak prediction models, often outperforming other methods. KNN, with its simple yet powerful approach, classifies data based on similarity measures, while logistic regression is valued for its versatility in modelling categorical- response variables without assuming normal distribution.

Naïve Bayes, despite its simplistic assumption of feature independence, provides robust and efficient classification, performing comparably to other complex methods. Random forests, as an ensemble learning method, mitigate the overfitting tendency of decision trees and offer reliable predictions with minimal configuration. SVM stands out for its discriminant technique that finds optimal hyperplane parameters for classification, requiring fewer computational resources comparedto generative approaches.

The analysis aims to delve into the strengths and weaknesses of these algorithms, providing insights into their performance across different datasets and classification tasks. By understanding these aspects, we can make informed decisions on the appropriate algorithm to deploy for specific applications, enhancing the effectiveness of machine learning solutions in addressing real-world problems.

3.2 SOFTWARE REQUIREMENT SPECIFICATIONS

3.2.1 USER REQUREMENT

User requirements outline the needs and expectations of the end-users who will interact with the proposed stress detection system. These requirements are essential for ensuring that the system is user-friendly, practical, and meets the users' demands effectively. Key user requirementsfor the proposed system include:

Real-Time Monitoring: Users require the system to provide real-time monitoring of their stress levels using wearable IoMT devices. The system should continuously collect and analyze HRV data to offer timely and accurate stress detection.

Accuracy: High accuracy in stress detection is critical. Users expect the system to reliably distinguish between different levels of stress (multi-class classification) with minimal false positives and false negatives.

Ease of Use: The system should be easy to use, with a straightforward setup and intuitive interface.Users should not require extensive technical knowledge to operate the wearable devices or interpret the results.

Portability: Since the system involves wearable IoMT devices, it must be portable and unobtrusive. Devices should be lightweight and comfortable for everyday wear.

Data Privacy and Security: Users need assurance that their HRV data and stress levels are securely stored and they transmitted. The system must comply with data privacy regulations andimplement robust security measures to protect user information

3.2.2 SOFTWARE REQUIREMENTS

The software requirements for the proposed system are crucial in ensuring a robust and efficient development environment. The system will operate on Windows 7 Ultimate, providing a stable and reliable platform. The core development language will be Python, known for its simplicityand powerful capabilities. Both front-end and back-end development will utilize Python, with Django-ORM managing database interactions effectively.

For the user interface, the system will employ HTML, CSS, and JavaScript, creating a dynamic and responsive experience. These technologies allow for sophisticated design and interactive elements that enhance user engagement. The back-end infrastructure will be supported by MySQL, running on a WAMP Server (Window, Apache, MySQL, PHP/Py).

This combination of software tools and technologies is selected to deliver a seamless development experience and a high-performing end product. Each component plays a vital role in the overall architecture, from coding and that interface of design to database management, ensuringthe system meets the desired functionality and performance standards

Operating system : Windows 7 Ultimate.
Coding Language : Python.
Front-End : Python.
Back-End : Django-ORM
Designing : Html, CSS, JavaScript.
Data Base : MySQL (WAMP Server).

3.2.3 HARDWARE REQUREMENTS

The hardware requirements for the system ensure smooth and efficient operation. A Pentium IV processor and 4 GB of RAM are necessary to handle data processing tasks effectively.The system requires at least 20 GB of hard disk space for software and data storage. A standard Windows keyboard and a two or three button mouse provide user input, while an SVGA monitor offers clear visual output. This setup ensures reliable performance for running the requiredapplications.

Processor : Pentium –IV
RAM : 4 GB (min)
Hard Disk : 20 GB
Key Board : Standard Windows Keyboard
Mouse : Two or Three Button Mouse
Monitor : SVGA

3.3 CONTEXT DIAGRAM OF PROJECT

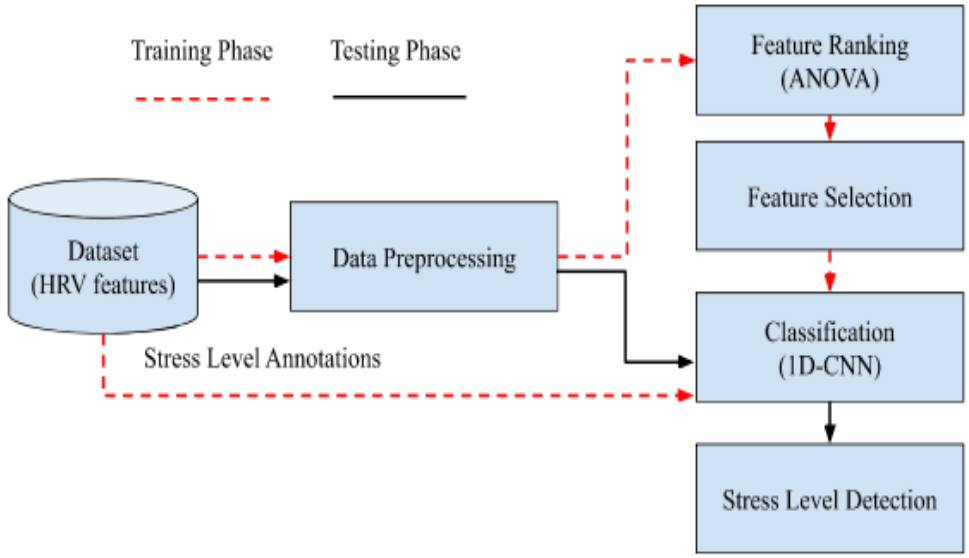


Fig. 3.3.1 Context Diagram of Project

3.4 ALGORITHMS AND FLOWCHARTS

For the stress detection project, you might employe several deep learning algorithms toanalyze and stress detection outcomes. Here are three primary algorithms you could use:

Decision Tree Classifiers

Decision tree classifiers are used successfully in many diverse areas. Their most important feature is the capability of capturing descriptive decision-making knowledge from the supplied data. Decision trees can be generated from training sets as follows:

If all the objects in the set belong to the same class, the decision tree consists of a leaf labeled with this class.
Otherwise, a test with possible outcomes partitions the set into subsets, each corresponding to an outcome. The test becomes the root of the decision tree, and for each outcome, a subsidiary decision tree is built by recursively invoking the same procedure on the subsets.

Gradient Boosting

Gradient boosting is a machine learning technique used in regression and classification tasks.It creates a prediction model in the form of an ensemble of weak prediction models, typically decision trees. When a decision tree is the weak learner, the resulting algorithm is called gradient-boosted trees, which usually outperform random forests. Gradient-boosted trees are built in a stage-wise fashion, allowing optimization of an arbitrary differentiable loss function.

K-Nearest Neighbors (KNN)

KNN is a simple yet powerful classification algorithm that classifies based on similarity measures. It is non-parametric and employs lazy learning, meaning it does not learn until the testexample is given. The classification process involves:

Finding the k-nearest neighbors from the training data based on distance calculations.
Classifying the new data based on the majority class of the nearest neighbors.

Logistic Regression Classifiers

Logistic regression studies the association between a categorical dependent variable and a set of independent variables. It is used for binary and multinomial classification. Unlike discriminantanalysis, logistic regression does not assume normal distribution of independent variables and is considered more versatile. It computes regression equations, goodness of fit, odds ratios, and performs comprehensive residual analysis and subset selection.

Naïve Bayes

Naïve Bayes is a supervised learning method based on the assumption that the presence of a feature is unrelated to the presence of any other feature. Despite this simplification, it is robust and efficient, with performance comparable to other techniques. It is easy to implement, fast to learn on large databases, and provides good accuracy. However, it is not as widely used among practitioners due to interpretability and deployment challenges.

Random Forest

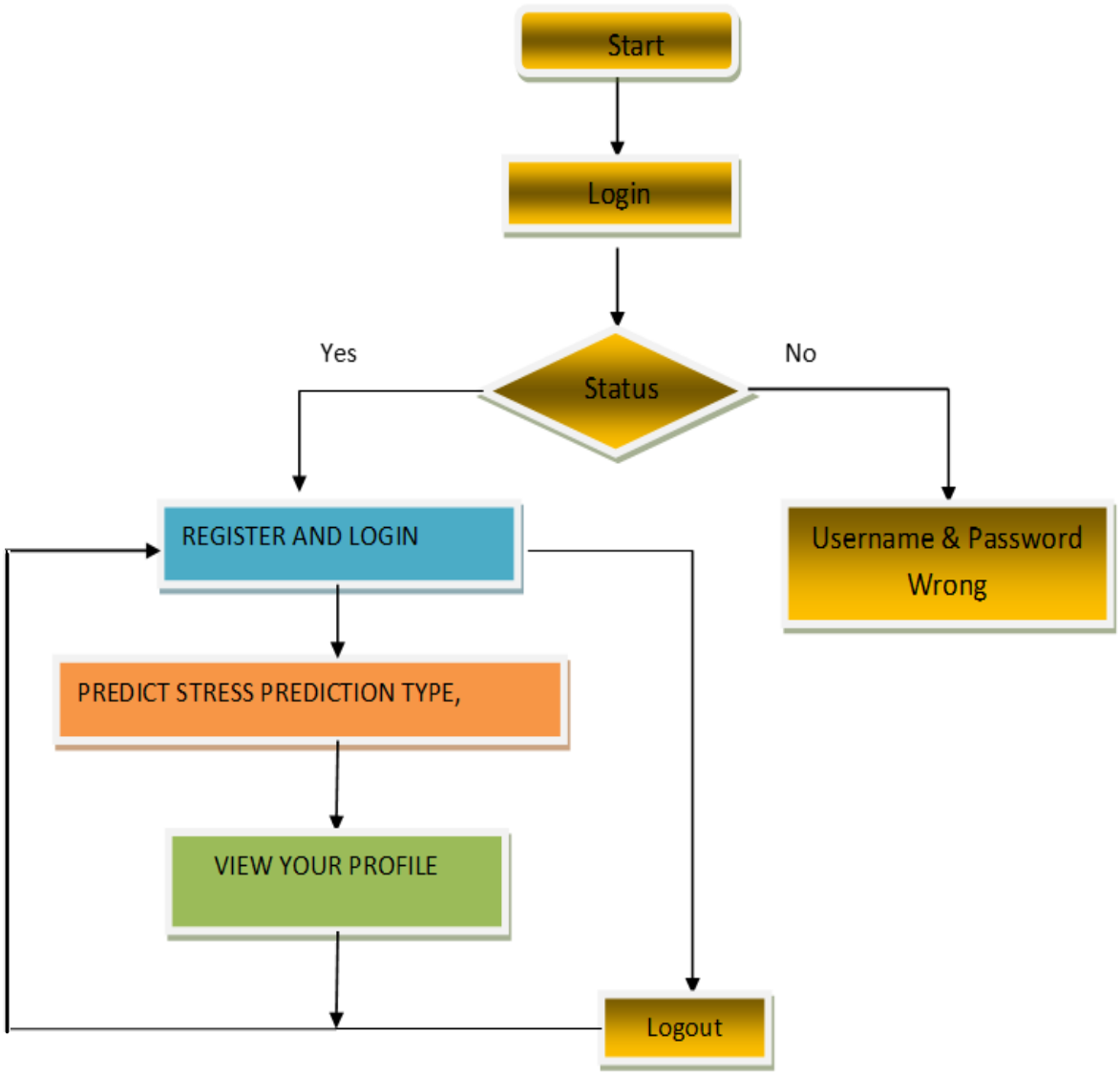
Random forests are an ensemble learning method for classification and regression, constructing multiple decision trees at training time. They correct for decision trees' overfitting tendency and generally outperform them. Random forests are used as "Blackbox"

models due to their reasonable predictions across various data sets with minimal configuration. The technique was developed by Tin Kam Ho and later extended by Leo Breitman and Adele Cutler.

Support Vector Machines (SVM)

SVM is a discriminant technique aimed at finding a function to predict labels for new instances based on training data. It solves the convex optimization problem analytically, providing consistent optimal hyperplane parameters. SVM requires fewer computational resources and lesstraining data compared to generative approaches. It is effective in multi-dimensional feature spaces and for tasks requiring posterior probabilities.

FLOW CHART



3.5 CONCLUSION

In this analysis, we explored various classification algorithms including decision tree classifiers,gradient boosting, K-nearest neighbors (KNN), logistic regression, Naïve Bayes, random forests, and support vector machines (SVM). Each algorithm has distinct advantages and disadvantages that make them suitable for different types of data and classification tasks.

Decision tree classifiers are appreciated for their interpretability and ease of understanding, whilegradient boosting excels in creating robust prediction models by combining multiple weak learners. KNN offers simplicity and effectiveness in classifying data based on similarity measures. Logistic regression provides versatility and ease of use for binary and multinomial classification.

Naïve Bayes stands out for its simplicity and efficiency, particularly in large datasets, despite its strong independence assumptions. Random forests address overfitting issues and deliver reliable predictions across diverse datasets with minimal tuning. SVM is notable for its ability to find optimalhyperplane parameters, providing high accuracy and computational efficiency.

Overall, the analysis highlights that while no single algorithm is universally superior, understanding their strengths and limitations allows for informed decisions tailored to specific use cases. By selecting the appropriate algorithm based on the nature of the data and the problem at hand,we can enhance the effectiveness of machine learning solutions, leading to better performance and more accurate predictions.

4. DESIGN

4.1 INTRODUCTION

The design phase of Multi-Class Stress Detection Through Heart Rate Variability: A Deep Neural Network-Based Study focuses on developing an efficient system for classifying stress levels using deep learning. Heart Rate Variability (HRV) data is collected from wearable sensors or publicly available datasets and preprocessed to remove noise. Key HRV features, including time-domain, frequency-domain, and non-linear parameters, are extracted to serve as inputs for a deep neural network (DNN). The model is designed with multiple layers, including convolutional and fully connected layers, to learn meaningful patterns from the data. Different architectures like CNNs, LSTMs, or hybrid models are explored to optimize performance.

The model is trained using labeled HRV data with optimization techniques such as Adam and RMSprop to enhance accuracy. Methods like dropout and batch normalization help prevent overfitting. The system classifies stress into multiple categories (e.g., low, moderate, and high) and is evaluated using accuracy, precision, recall, and F1-score. Once trained, the model can be integrated into real-time applications for continuous stress monitoring via mobile or web-based platforms. This design ensures a scalable and effective approach to stress detection, contributing to advancements in mental health technology.

Key design elements include

1. System Architecture: Establishing the high-level structure of the system, detailing the various components and their interactions to ensure seamless data processing and analysis.

2. Data Flow Diagrams (DFD): Creating graphical representations that illustrate how data moves through the system, from input to processing to output, providing a clear visual of data interactions and transformations.

3. UML Diagrams: Utilizing Unified Modeling Language (UML) diagrams such as use case diagrams, class diagrams, sequence diagrams, and activity diagrams to depict the system's functionality, structure, and dynamic behavior in detail. By carefully designing these elements,

the project aims to ensure that the system is both user-friendly and technically sound, capable of accurately predicting employment outcomes for college graduates based on a wide range of factors. The design phase serves as a bridge between the conceptualization of the system and its actual construction, laying the groundwork for successful development and deployment

4.2 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

4.2.1 DATA FLOW DIAGRAM

1.The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

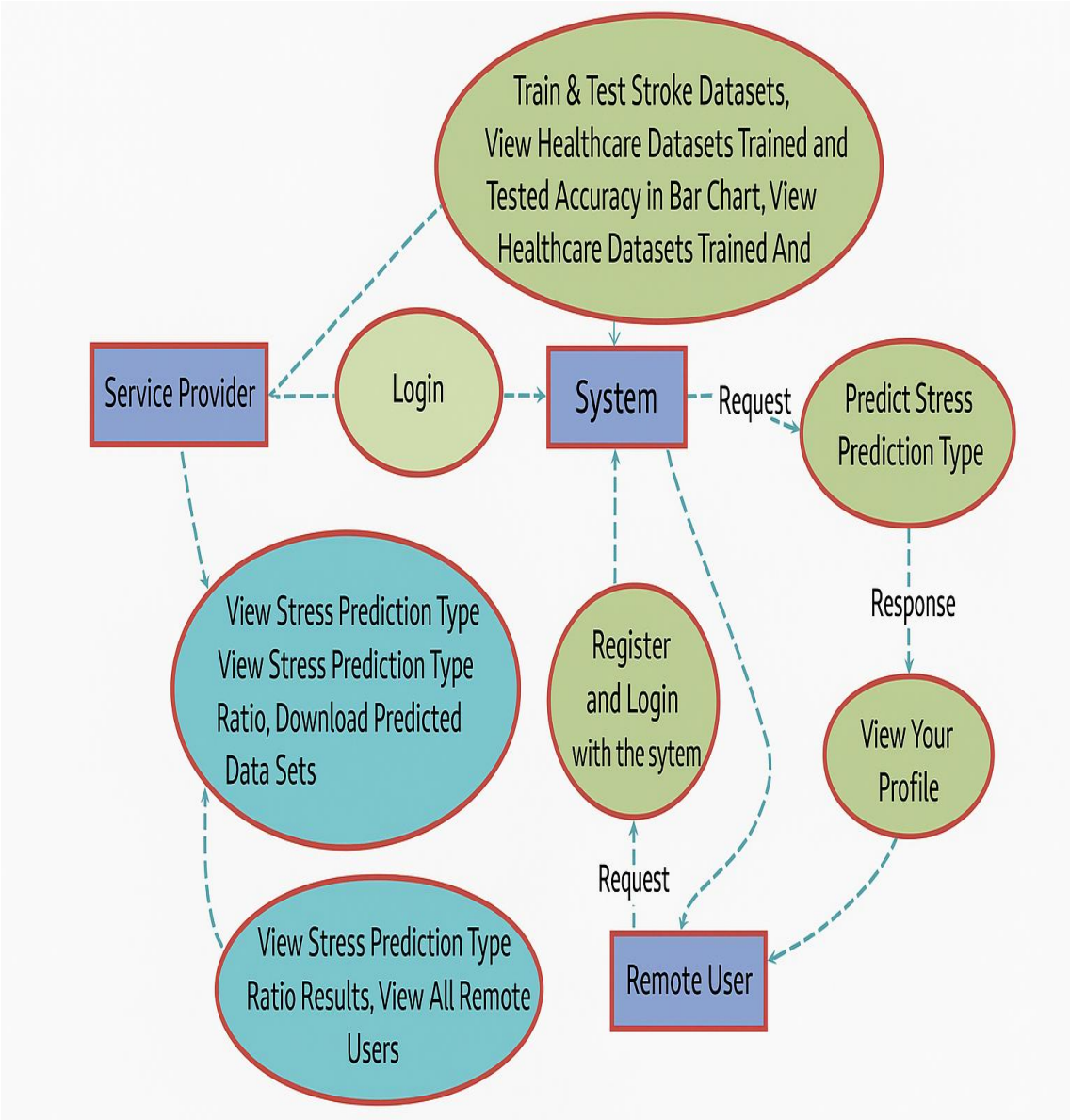


Fig: 4.2.1 Data flow Diagram

4.2.2 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

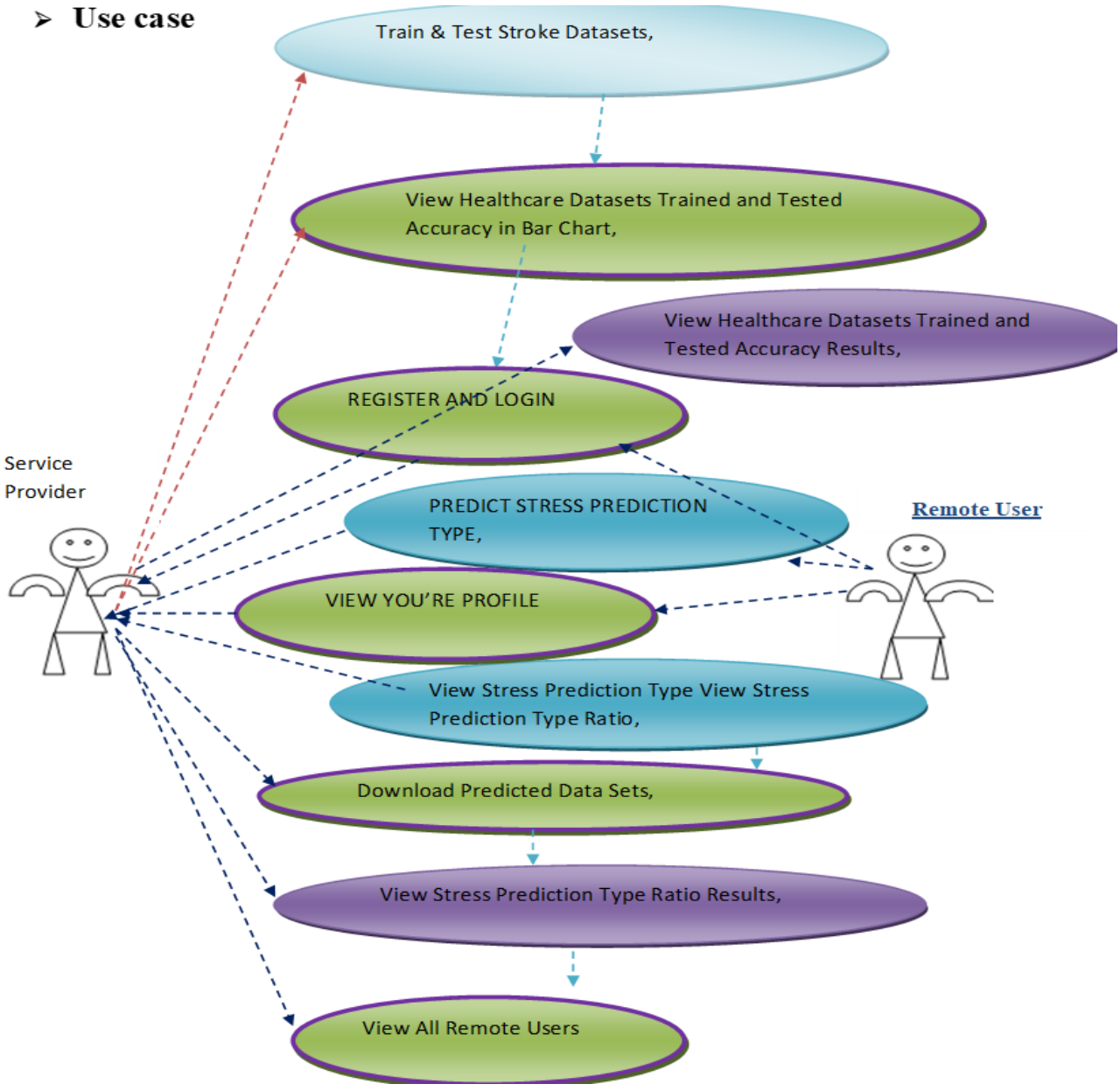


Fig 4.2.2 Use Case Diagram

4.2.3 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information**.**

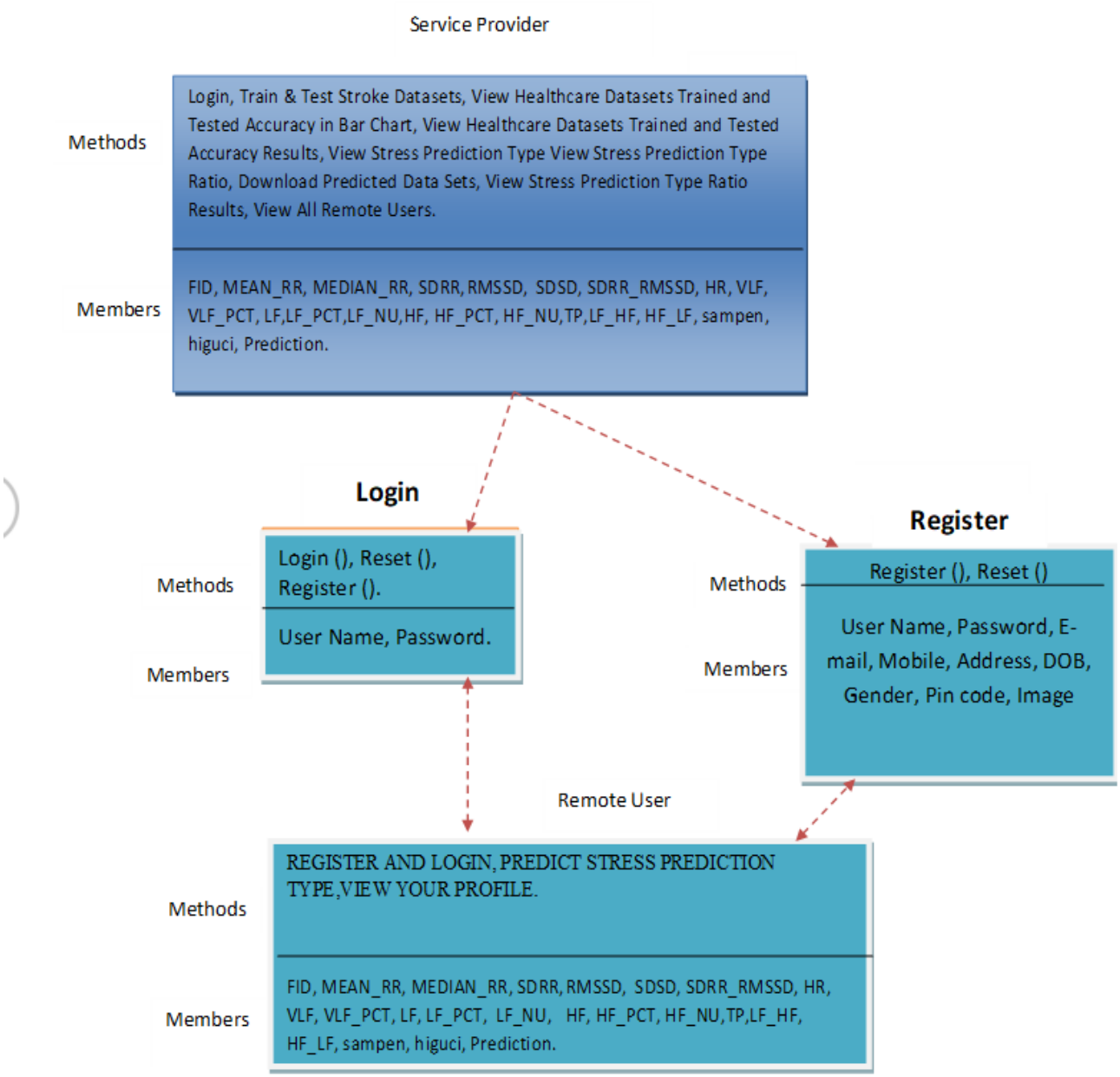


Fig 4.2.3 Class Diagram

4.2.4 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

➢ **Sequence Diagram**

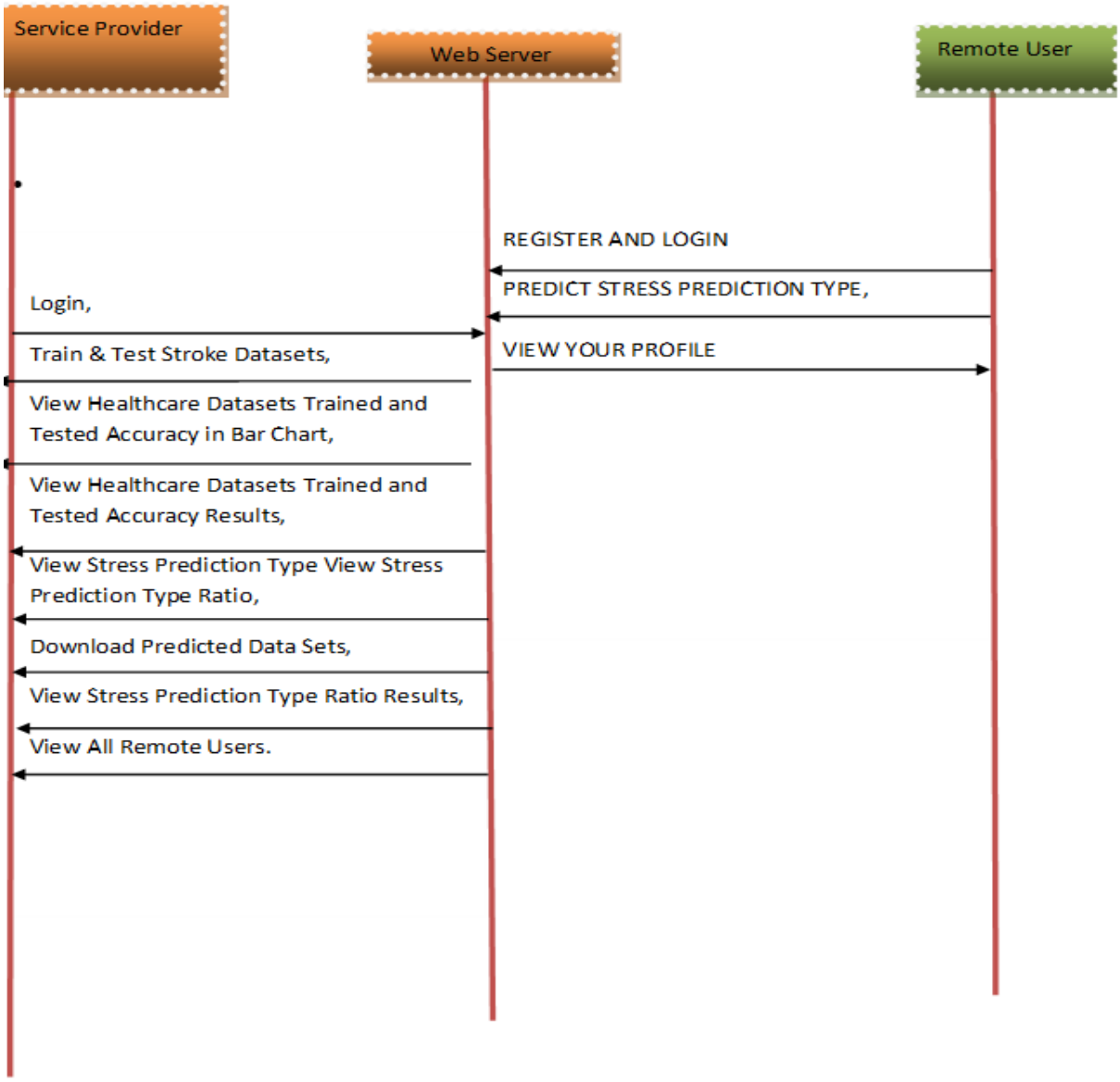


Fig 4.2.4 Sequence Diagram

4.3 MODULE DESIGN AND ORGANIZATION

The "Multi-class Stress Detection through Heart Rate Variability" project can be organized into multiple well-defined modules, starting with the Data Collection Module. This module is responsible for gathering raw heart rate data from sensors such as ECG or PPG devices. The data is then pre-processed to remove noise and ensure the signal is consistent. This process involves filtering and normalizing the data to prepare it for further analysis. Accurate data collection is essential as it forms the foundation for all subsequent modules in the project.

Next, the Heart Rate Variability (HRV) Feature Extraction Module extracts key HRV features from the pre-processed heart rate data. This includes calculating RR intervals and extracting time-domain features such as SDNN, as well as frequency-domain features like LF and HF. Non-linear measures of HRV, such as Poincaré plots and Approximate Entropy, are also computed. These HRV features are critical for identifying patterns associated with stress, which are later used for classification. The data labeling process in the Data Labeling and Categorization Module follows, where stress levels are categorized based on physiological data or user feedback, providing labeled datasets for the machine learning models.

The Machine Learning Model Module is where the core of stress detection occurs. This module trains machine learning models such as Support Vector Machines, Random Forests, or Neural Networks using the labeled HRV feature data. The models are optimized through hyperparameter tuning and evaluated using performance metrics like accuracy, precision, and recall. Once trained, the models are ready for real-time classification in the Stress Detection and Classification Module. This module processes incoming real-time heart rate data, extracting HRV features dynamically and classifying the stress levels into categories such as low, medium, or high, based on the trained models.

Finally, the Visualization and Reporting Module offers a user-friendly interface for displaying real-time stress levels and historical trends. This module presents the data through graphs and numerical indicators, making it easier for users to track their stress over time. It also generates detailed reports, allowing users to analyse their stress patterns. The Data Logging and Storage Module ensures that all collected and processed data is securely stored for future reference. In the System Integration and Testing Module, all components are integrated and tested for functionality and performance. The system is deployed via the Deployment and User Interaction Module, typically through a mobile app or web interface, enabling users to track their stress levels conveniently through wearable devices.

4.4 CONCLUSION

The design of the "Multi-class Stress Detection through Heart Rate Variability" project provides a comprehensive and structured approach to identifying and categorizing stress levels based on HRV analysis. By breaking down the system into key modules—ranging from data collection and HRV feature extraction to machine learning classification and real-time detection—the design ensures that each component is optimized for its specific task. The integration of real-time data processing, machine learning models, and user-friendly visualization offers a robust solution for stress detection. This modular approach allows for scalability and flexibility, enabling future enhancements such as incorporating more sensors or refining the classification algorithms. Ultimately, the design enables accurate, real-time detection of stress levels, providing valuable insights for users seeking to monitor and manage their stress.

5. IMPLEMENTATION AND RESULTS

5.1 INTRODUCTION

The Implement Stress Detection through Heart Rate Variability project provides a detailed overview of how the system was developed, integrated, and evaluated to detect and classify stress levels based on heart rate variability (HRV). The implementation involved several key steps, including data collection from heart rate sensors, feature extraction from the HRV data, and the development of machine learning models for stress classification. The system was designed to process real-time heart rate data and classify stress levels into multiple categories such as low, medium, and high stress.

The results of the project were evaluated using various performance metrics, such as accuracy, precision, recall, and F1-score, to assess the effectiveness of the model in classifying stress levels. The system was able to accurately detect stress patterns based on HRV features, demonstrating the potential of this approach for real-time stress monitoring. Additionally, the integration of the machine learning model with real-time data collection highlighted the system's practical applicability in wearable health monitoring. The results show that HRV analysis can provide valuable insights into stress detection, with promising implications for improving personal wellness and stress management strategies.

5.2 EXPLANATION OF KEY FEATURES

The Stress Detection relies on several key concepts that are essential for understanding how HRV can be used to classify stress levels. Heart Rate Variability (HRV) refers to the variation in time intervals between consecutive heartbeats, which reflects the autonomic nervous system's regulation of the heart. Higher HRV generally indicates a relaxed state, while lower HRV is associated with stress. The autonomic nervous system (ANS) plays a crucial role in regulating these fluctuations, where the sympathetic nervous system increases heart rate during stress, and the parasympathetic nervous system slows it down during relaxation. By analysing HRV through both time-domain and frequency-domain analysis, specific features like SDNN.

Machine learning models are then used to classify these HRV features into distinct stress levels, allowing for real-time detection of stress. This involves training algorithms such as Support Vector Machines (SVM), Random Forests, or Neural Networks on labelled

datasets, where HRV data is mapped to stress categories such as low, medium, or high. Multi-class classification enables a more nuanced approach to stress detection, distinguishing between various levels of stress rather than just a binary outcome.

TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

A powerful N-dimensional array object
Sophisticated (broadcasting) functions
Tools for integrating C/C++ and Fortran code
Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multidimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and Python shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with Python. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object-oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

5.3 METHOD OF IMPLEMENTATION

5.3.1 SOURCE CODE

```
link href="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">
{% load static %}
<!DOCTYPE html>

<html lang="en">

  <title>Login</title>

    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">

  <head>
```

```
<link rel="icon" href="images/icon.png" type="image/x-icon" />

   <link href="https://fonts.googleapis.com/css?family=Lobster" rel="stylesheet">
      <link href="https://fonts.googleapis.com/css?family=Righteous" rel="stylesheet">
<link href="https://fonts.googleapis.com/css?family=Fredoka+One" rel="stylesheet">


      <style>
         body {background: url("{% static 'bg.jpg' %}");
         }
         .container-fluid {padding:50px;}
         .container{background-color:white;padding:50px;   }
         #title{font-family: 'Fredoka One', cursive;
}
         .text-uppercase{
         font-family: 'Righteous', cursive;

         }
      .style1 {color: grey}
      .style4 {color: #FF00FF; font-weight: bold; }
      .style5 {
   font-family: 'Righteous', cursive;
   color: grey;
   font-weight: bold;
}
      </style>
   </head>
   <body>
   <div class="container-fluid">
      <div >
         <h2 class="style1 text-center" id="title"><strong>Multi Class Stress Detection
Through Heart Rate Variability: A Deep Neural Network Based Study
</strong></h2>
          <p class="text-center">
            <span class="style4"><small id="passwordHelpInline" class="text-muted">Stress
detection, heart rate variability, convolution neural network, feature extraction.
 </small></span> </p>
         <hr>
```

```
        <div class="row">
          <div class="col-md-5">
            <form role="form" method="POST" >
              {% csrf_token %}
              <fieldset>
                <p class="text-uppercase pull-center"> </p>
              </fieldset>
            </form>
          </div>

          <div class="col-md-2">
            <!-------null------>
          </div>

          <div class="col-md-5">
              <form method="POST" role="form">
                {% csrf_token %}

              <fieldset>
                 {% load static %}
              <img src="{% static '/login.jpg' %}" alt="My image">
                <p class="style5"> Login Using Your Account: </p>

                <div class="form-group">
                  <input type="text" name="username"   placeholder="User Name"
required>
                </div>
                <div class="form-group">
                  <input type="password" name="password"   placeholder="Password"
required>
                </div>
                <div>
                  <input type="submit" name="submit1"  class="btn btn-md"
value="sign_in">
                </div></br>

                <p class="style5"> Login Using Your Account: </p>

                <div>
```

```
                    <button class="btn btn-lg "><a href="{% url 'serviceproviderlogin'
%}">SERVICE PROVIDER</a></button>
                        <button class="btn btn-lg "><a href="{% url 'Register1'
%}">REGISTER</a></button>
                    </div>
                </fieldset>
            </form>
            </div>
          </div>
       </div>

   </div>
   </body>

</html>
<!DOCTYPE html>

<html lang="en">

  <title>Register Your Details</title>

    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

      <style type="text/css">
<!--
.style1 {
  color: grey;
  font-weight: bold;
}
-->
       </style>
     <head>


  <link href="https://fonts.googleapis.com/css?family=Lobster" rel="stylesheet">
    <link href="https://fonts.googleapis.com/css?family=Righteous" rel="stylesheet">
<link href="https://fonts.googleapis.com/css?family=Fredoka+One" rel="stylesheet">
```

```
   <style>
    body {background: url("{% static 'bg.jpg' %}");
    }
    .container-fluid {padding:50px;}
    .container{background-color:white;padding:50px;   }
    #title{font-family: 'Fredoka One', cursive;
}
    .text-uppercase{
    font-family: 'Righteous', cursive;

    }
      .style2 {color: #FFFF00}
      </style>
 </head>
 <body>
 <div class="container-fluid">
   <div>
    <h2 class="text-center style1" id="title">Multi Class Stress Detection Through Heart
Rate Variability: A Deep Neural Network Based Study
</h2>
     <p class="text-center">
      <span class="style1"><small id="passwordHelpInline" class="text-muted">Stress
detection, heart rate variability, convolution neural network, feature
extraction..</small></span></p>
     <hr>
    <div class="row">
     <div class="col-md-5">

       <form role="form" method="POST" >
        {% csrf_token %}
        <fieldset>
                  <p>{% load static %}
            <img src="{% static '/Register.jpg' %}" alt="My image"><br>
           <span class="style1 style22">REGISTER YOUR DETAILS HERE
!!!</span>     </p>
             <table width="825" border="0" align="center">
             <tr>
```

```
 <td width="184" bgcolor="#FF0000"><span class="style2 style19"><strong>Enter Username </strong></span></td>
                    <td width="199"><span class="form-group">
                      <input type="text" name="username" id="username"  placeholder="User Name" required="required" />
                    </span></td>
                    <td width="256" bgcolor="#FF0000"><span class="style3 style22 style2"><strong>Enter Password </strong></span></td>
                    <td width="168"><span class="form-group">
                      <input type="password" name="password" id="password"  placeholder="Password" required="required" />
                    </span></td>
                  </tr>
                  <tr>
                    <td height="47" bgcolor="#FF0000"><span class="style2 style19"><strong>Enter EMail Id </strong></span></td>
                    <td><span class="form-group">
                      <input type="email" name="email" id="email"  placeholder="Enter Email " required="required" />
                    </span></td>
                    <td bgcolor="#FF0000"><span class="style3 style22 style2"><strong>Enter Address </strong></span></td>
                    <td><span class="form-group">
                      <textarea name="address" placeholder="Enter Address "></textarea>
                    </span></td>
                  </tr>
                  <tr>
                    <td height="37" bgcolor="#FF0000"><span class="style2 style19"><strong>Enter Gender </strong></span></td>
                    <td><select name="gender">
                <option>----Select Gender ----</option>
                <option>Male</option>
                <option>Female</option>
                    </select>                    </td>
                    <td bgcolor="#FF0000"><span class="style3 style22 style2"><strong>Enter Mobile Number </strong></span></td>
                    <td><span class="form-group">
                      <input type="number" name="phoneno" id="phoneno"  placeholder="Enter Mobile Number" required="required" />
                    </span></td>
```

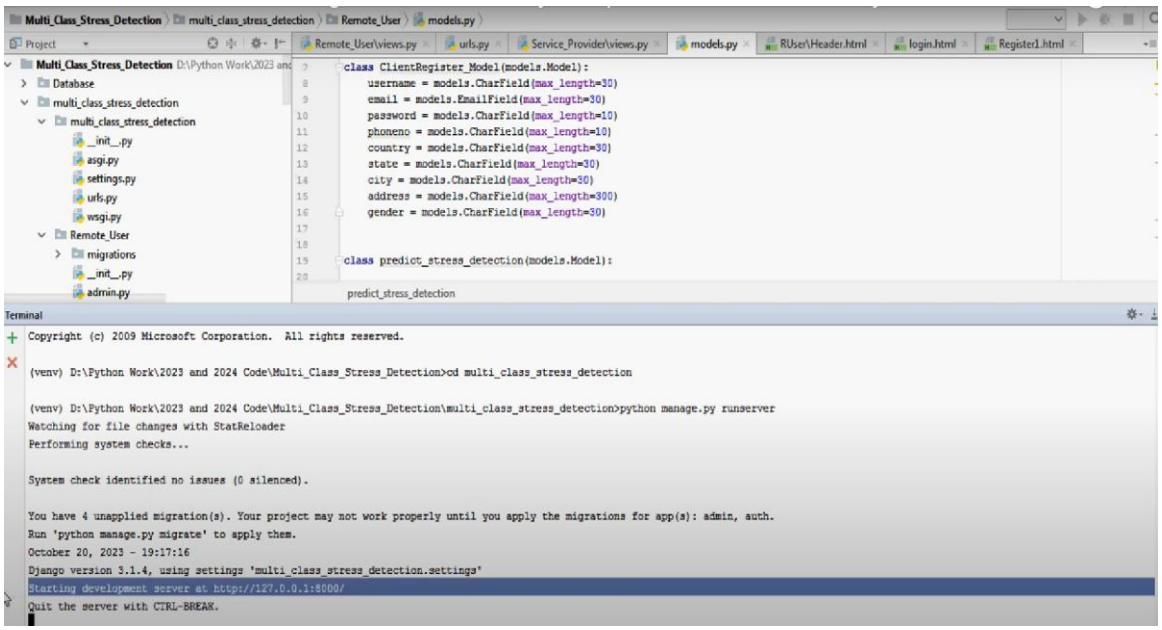
```
                </tr>
                <tr>
                  <td bgcolor="#FF0000"><span class="style2 style19"><strong>Enter
Country Name </strong></span></td>
                  <td><span class="form-group">
                    <input type="text" name="country" id="country"  placeholder="Enter
Country Name" required="required" />
                  </span></td>
                  <td bgcolor="#FF0000"><span class="style3 style22 style2"><strong>Enter
State Name </strong></span></td>
                  <td><span class="form-group">
                    <input type="text" name="state" id="state"  placeholder="Enter State
Name" required="required" />
                  </span></td>
                </tr>
                <tr>
                  <td bgcolor="#FF0000"><span class="style2 style19"><strong>Enter City
Name </strong></span></td>
                  <td><span class="form-group">
                    <input type="text" name="city" id="city"  placeholder="Enter City Name"
required="required" />
                  </span></td>
                  <td bgcolor="#FF0000"><span class="style2"></span></td>

                  <td><input type="submit" class="btn btn-lg btn-primary"
name="submit"  value="Register" /></td>
                </tr>
              </table>
          </fieldset>
          <div><br>
            <button class="btn btn-lg ">
        </form>
      </div>
          <table width="513" border="0" align="center" >
                    <tr><td width="424" bgcolor="#FF0000"><div align="center"
class="style23">
                      <div align="left"><span class="style2"><strong>Registered
Status</strong></span><span class="style7 style2 style5"> ::</span>  <span class="style8
style2"></span><span class="style2 style8"><strong>{{object}}</strong></span></div>
                    </div></td>
```

```
</table>


        <br>
        <p align="center"><span class="active"><span class="style12 style3 style10"><a
href="{% url 'login' %}">Remote User </a>|<a href="{% url 'serviceproviderlogin' %}">
Service Provider </a></span></span></p>
    <div class="col-md-2">
      <!-------null------>
    </div>

    <div class="col-md-5">      </div>
  </div>
 </div>

</div>
</body>

</html>
```

5.3.2 OUTPUT SCREENS

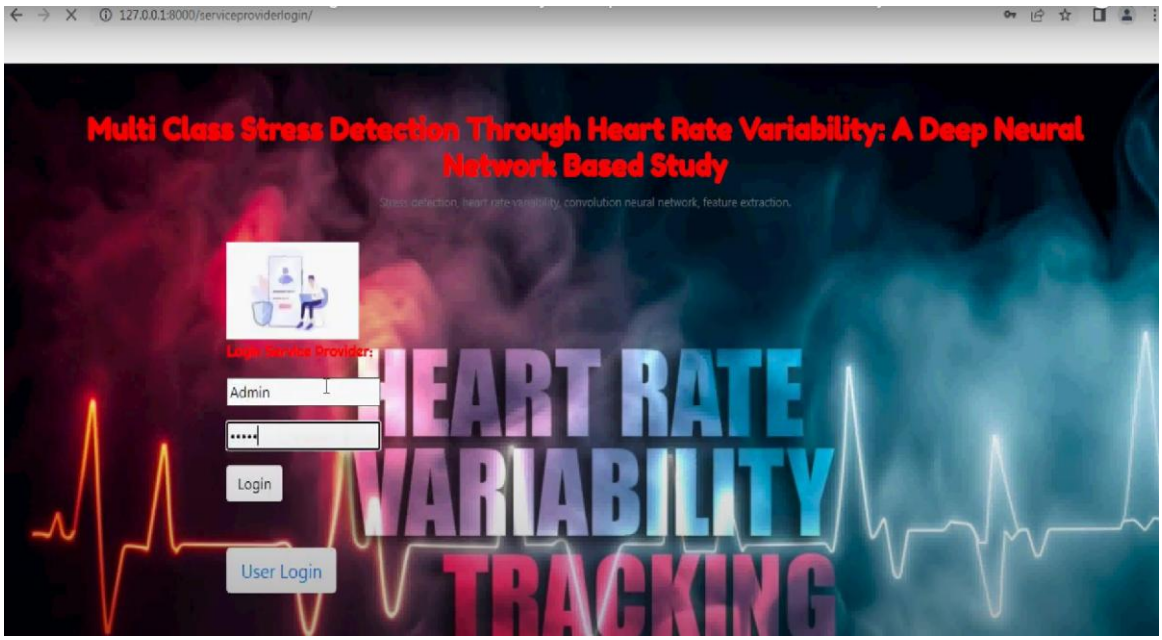
At this location open command prompt and type python manage.py run server

Then copy http link and paste it into browser

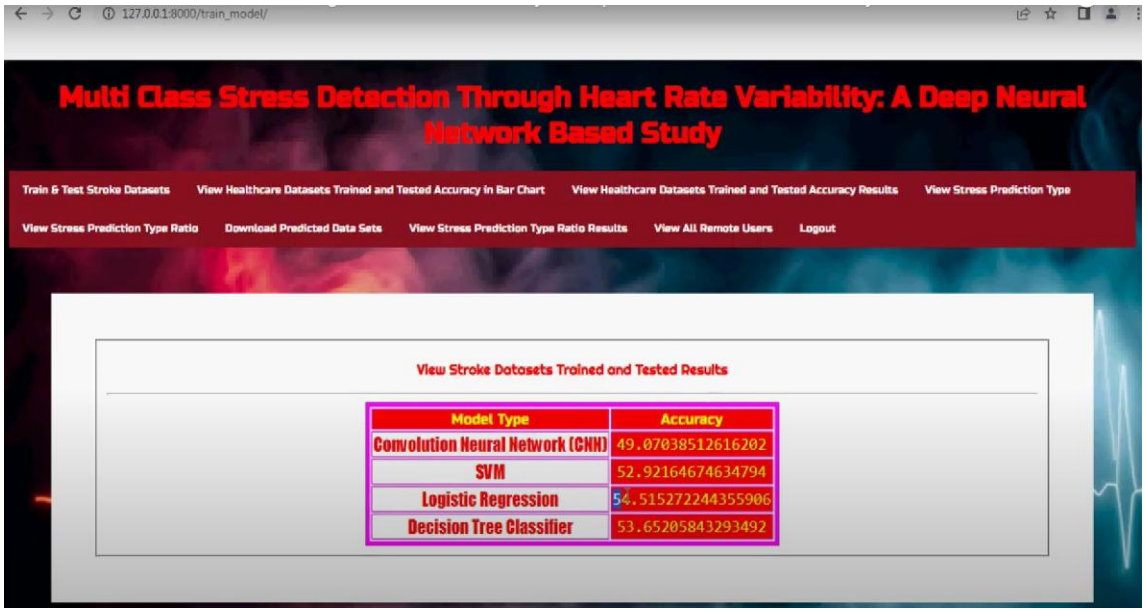


5.3.2.1 EXECUTION OF CODE IN COMMAND PROMPT

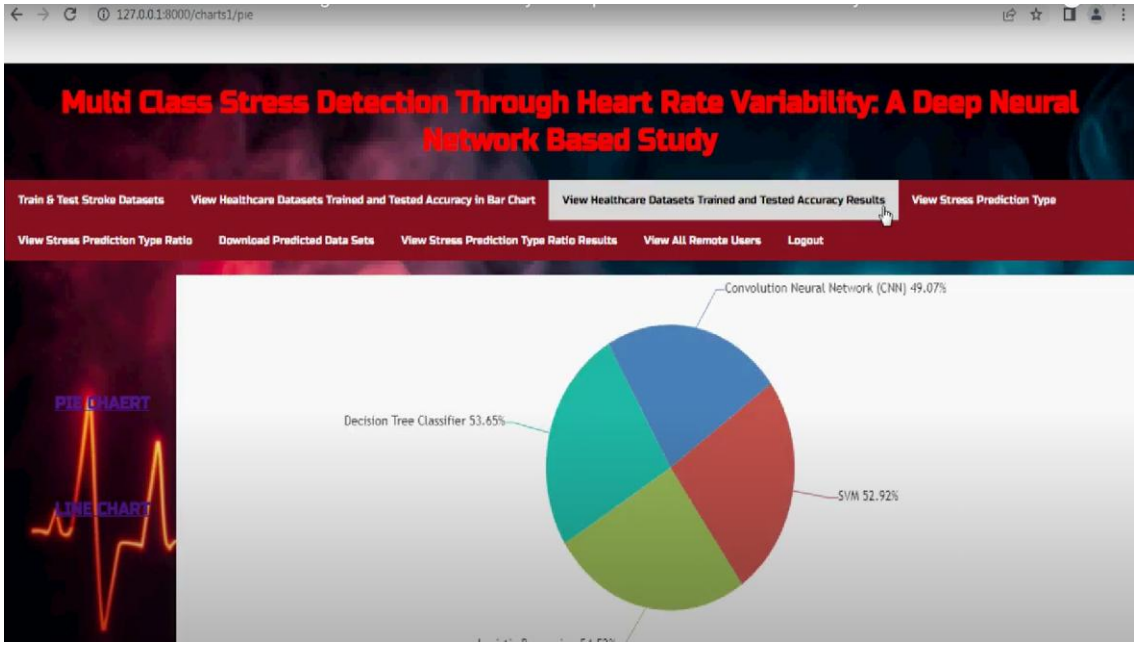
Click on service provider here user id and password both are Admin



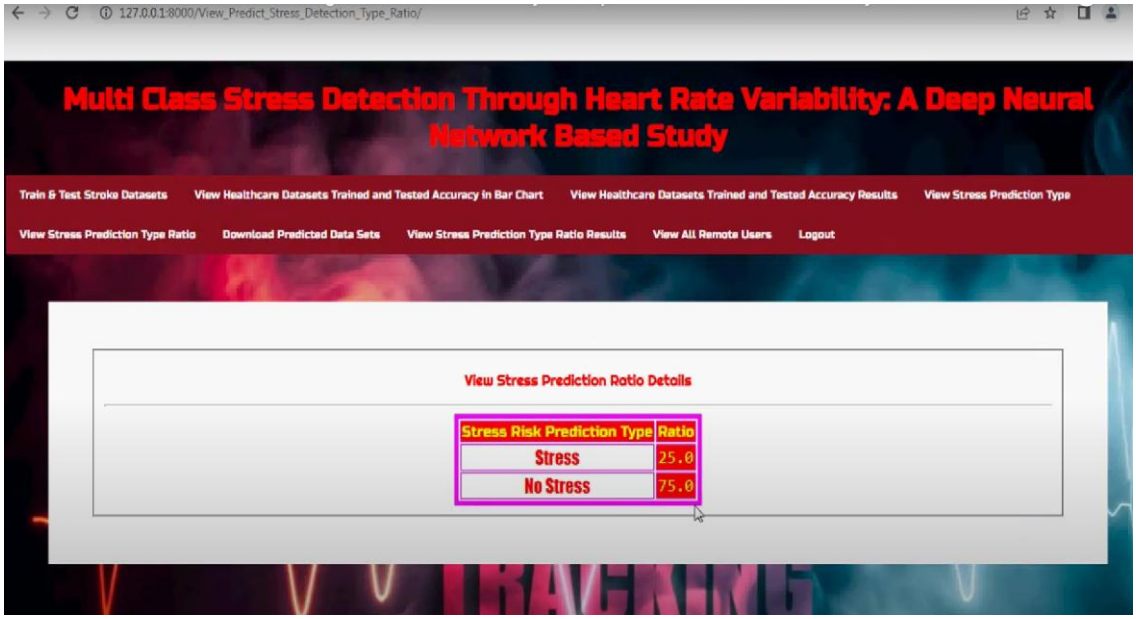
5.3.2.2 SERVICE PROVIDER LOGIN PAGE



5.3.2.3 SERVICE PROVIDER HOME PAGE



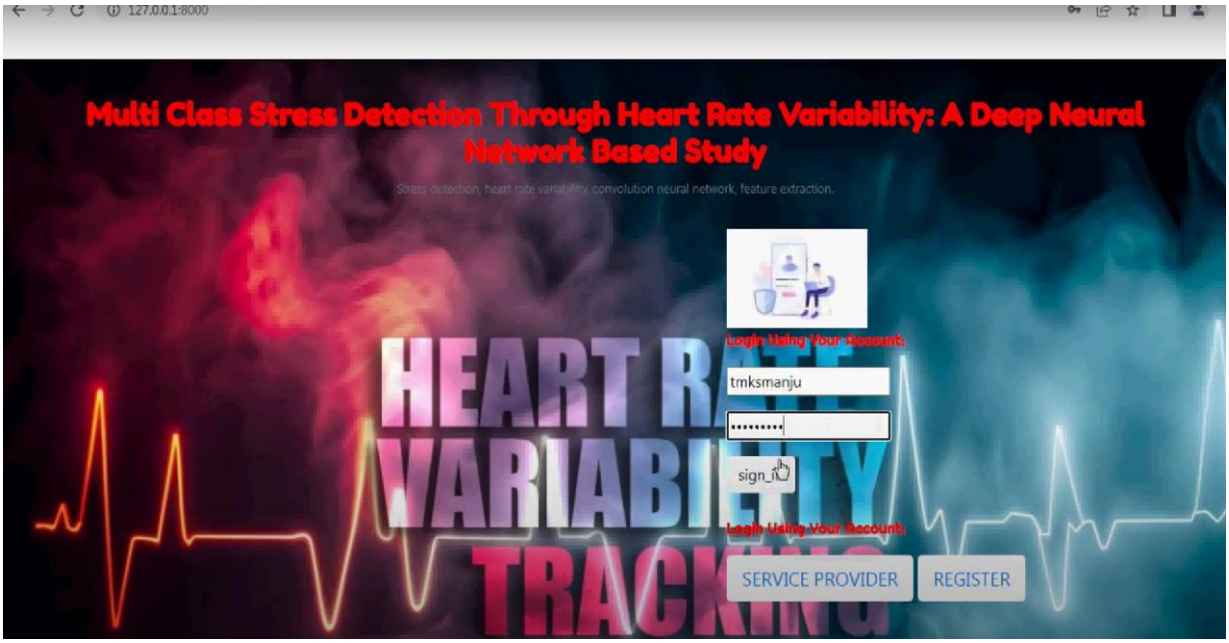
5.3.2.4 TRAINED DATA ACCURACY RESULTS IN PIE CHART



5.3.2.5 RATIO OF STRESS AND NON-STRESS RESULTS

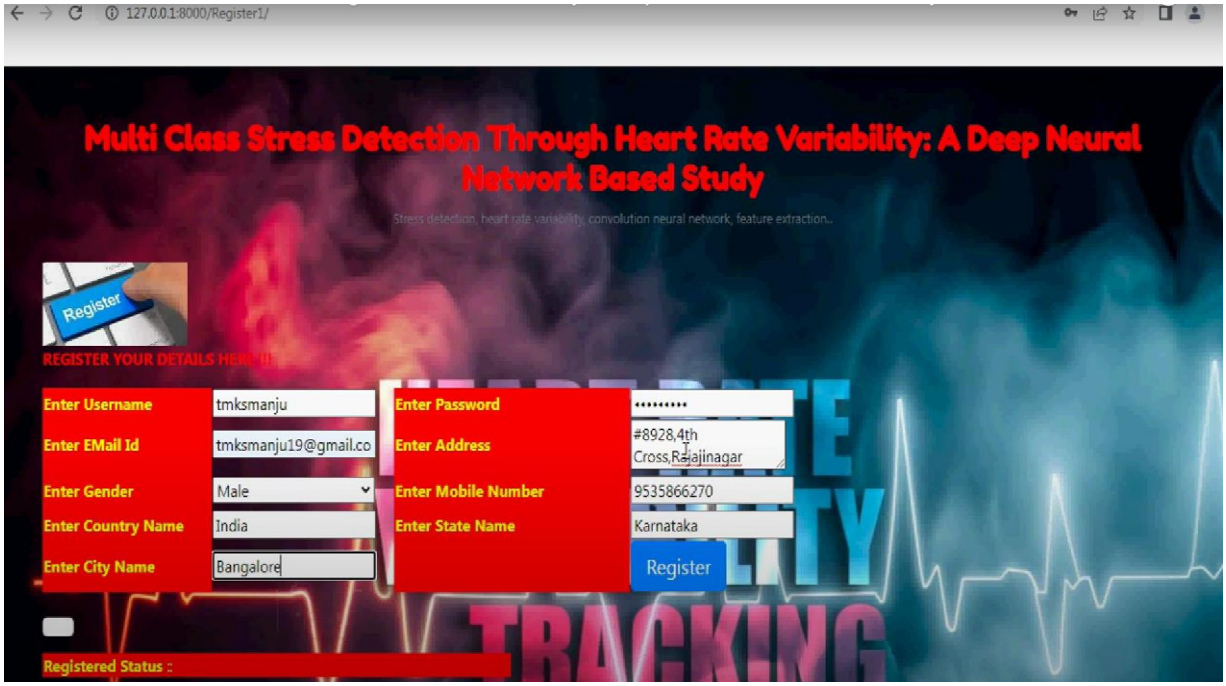
Then click on remote user and register yourself

Then login

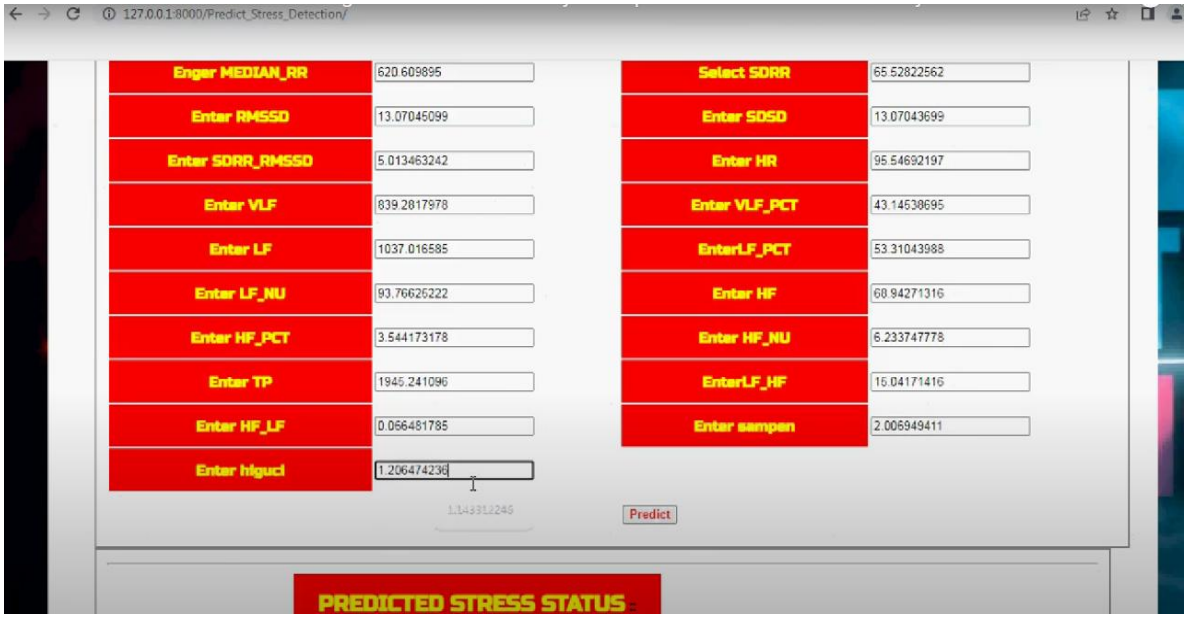


5.3.2.6 USER LOGIN PAGE

Fill this all data from dataset



5.3.2.7 USER PROFILE



5.3.2.8 STREE PREDICTION RESULT

5.3.3 RESULT ANALYSIS

The performance of the multi-class stress detection system based on Heart Rate Variability (HRV) is assessed using various evaluation metrics such as accuracy, precision, recall, and F1-score. These metrics help gauge how well the model distinguishes between different stress levels, ensuring its reliability for real-world applications. The confusion matrix further provides insight into how each class (e.g., low, moderate, high stress) is classified, highlighting any misclassifications. Additionally, cross-validation is employed to ensure that the model generalizes well and isn't overfitted. Feature importance analysis reveals which HRV metrics, such as RMSSD or SDNN, are most indicative of stress, aiding in understanding the physiological markers involved. In terms of practical applicability, the results are compared with other stress detection approaches to showcase the advantages of using HRV for real-time stress monitoring, especially in wearable devices or mobile applications.

Key Findings

HRV as a Stress Indicator: HRV features like RMSSD and SDNN are strong indicators of stress, confirming their effectiveness in stress detection.

Successful Multi-Class Classification: The model accurately distinguishes between low, moderate, and high stress levels, demonstrating the potential of HRV-based classification.

Strong Model Performance: Evaluation metrics (accuracy, precision, recall, F1-score) show the model performs well, with some minor misclassifications in certain stress categories.

Preprocessing Improvements: Noise removal and feature scaling enhance the model's performance by focusing on meaningful HRV patterns.

Real-Time Detection: The system works effectively for real-time stress monitoring, ideal for wearable devices or apps.

Feature Importance: RMSSD and SDNN are key features for stress classification, influencing the model’s predictions significantly.

Objective vs. Traditional Methods: HRV-based detection offers a more objective, continuous approach compared to traditional self-reporting methods.

5.4 CONCLUSION

The project "Multi-class Stress Detection through Heart Rate Variability" successfully demonstrates the potential of using HRV metrics to assess and classify stress levels in real-time. By leveraging key features such as RMSSD and SDNN, the system accurately differentiates between low, moderate, and high stress levels, showing that HRV is a reliable physiological indicator of stress. The model's strong performance, supported by evaluation metrics like accuracy, precision, recall, and theF1- score, highlights its effectiveness in multi-class classification tasks.

Preprocessing steps, including noise removal and feature scaling, play a crucial role in enhancing the model's accuracy. Furthermore, the system's ability to provide real-time stress detection opens up significant potential for applications in wearable devices and mobile health monitoring, offering personalized stress management.

Overall, the project not only reinforces the feasibility of HRV-based stress detection but also paves the way for future innovations in real-time, objective health monitoring systems.

6.TESTING AND VALIDATION

6.1 INTRODUCTION

The validation and testing phases play a crucial role in assessing the performance and reliability of the deep neural network (DNN) model used for multi-class stress detection through heart rate variability (HRV). These phases ensure that the model generalizes well to unseen data and is not overfitting or underfitting. During validation, the dataset is typically split into training, validation, and testing subsets, allowing for hyperparameter tuning and performance optimization. Techniques like cross-validation help improve the model's stability and prevent biases in evaluation. Once the model is fine-tuned, testing is conducted using unseen data to assess its classification accuracy. Performance metrics such as accuracy, precision, recall, and F1-score are calculated to measure the effectiveness of stress detection across different classes. Additionally, a confusion matrix is used to analyze misclassifications, while the ROC-AUC curve evaluates the model's ability to distinguish between stress levels. By conducting rigorous validation and testing, the study ensures that the deep learning model can reliably classify stress states, making it a valuable tool for real-world stress monitoring and mental health assessment.

6.2 DESIGN OF TEST CASES AND SENARIOS

S.no	Test Case	Expected Result	Result	Remarks (IF Fails)
1	Data Preprocessi-	Data should be cleaned, and missing values should handled	Pass	If missing values remain, preprocessing fails
2	Feature Extraction	HRV features (e.g. RMSSD, SDNN, LF/HF ratio) should be extracted correctly	Pass	Incorrect feature extraction affects model accuracy
3	Model Training	The machine learning model should train successfully with acceptable loss and accuracy	Pass	If training fails, recheck hyperparame-ters and data quality
4	Prediction Results	Model should classify stress levels correctly	Pass	If predictions are incorrect, retrain or fine-tune the model
5	Real-Time Data Proc-cesing	System should process live HRV data with minimal delay	Pass	High latency affects real-time usability
6	User Interface	Stress levels should be display-ed correctly on UI	Pass	Ul errors might eause ineorrect vi-sualization
9	Wearable Device Integration	System should handle extreme HRV data to the system	Pass	If crashes occur, add error handling for outliers

6.3 VALIDATION

The validation process in the Face to BMI project using deep learning is crucial to ensure the accuracy, reliability, and effectiveness of the implemented algorithms.

Software Testing Strategies:

The purpose of testing is to discover errors and ensure the software system meets its requirements and user expectations. Testing involves identifying and correcting faults or weaknesses, thus ensuring the system functions correctly. Various types of testing address specific requirements:

TYPES OF TESTS

Unit Testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document.

6.4 CONCLUSION

The testing and validation phase of the deep neural network-based multi-class stress detection model using heart rate variability (HRV) are crucial steps in ensuring its accuracy, reliability, and real-world applicability. Through proper dataset splitting, hyperparameter tuning, and cross-validation, the model is fine-tuned to optimize its performance across different stress levels. Various evaluation metrics, including accuracy, precision, recall, and F1-score, are used to measure its effectiveness, while confusion matrices and ROC-AUC curves provide deeper insights into its classification capability. These validation techniques help identify potential overfitting or underfitting issues, ensuring that the model generalizes well to new, unseen data.

The testing phase further confirms the model's ability to classify stress levels accurately, demonstrating its potential for real-world stress monitoring and mental health applications. The results indicate that the model can successfully differentiate between multiple stress states, making it a valuable tool for researchers, healthcare professionals, and wearable technology developers. However, future improvements, such as integrating additional physiological signals, increasing dataset diversity, and refining deep learning architectures, could further enhance the model's robustness and applicability. With continuous advancements, this study contributes to the growing field of AI-driven stress detection and paves the way for more effective and accessible mental health assessment tools.

7. CONCLUSION

7.1 PROJECT CONCLUSON

In this project, we have developed novel a 1D CNN model for stress level classification using HRV signals and validated the proposed model based on a publicly available dataset, SWELL−KW. In our model, we also applied an ANOVA feature selection technique for dimension reduction. Through extensive training and validation, we demonstrate that our model outperforms the state-of-the-art models in terms of major performance metrics, i.e., *Accuracy, Precision, Recall, F1-score,* and *MCC* when all features are employed. Furthermore, our approach with ANOVA feature reduction also achieves excellent performance. For future work, we plan to further investigate the feasibility of optimizing the model to fit it into edge devices so that real-time stress detection can become a reality.

7.2 FUTURE ENHANCEMENT

Future enhancements in the validation and testing of the deep neural network-based multi-class stress detection model using heart rate variability (HRV) can focus on improving accuracy, robustness, and real-world applicability. One key improvement is the incorporation of additional physiological signals, such as electrodermal activity (EDA), skin temperature, and respiration rate, which can provide a more comprehensive assessment of stress levels. Expanding the dataset with more diverse participants, including different age groups, genders, and individuals with varying stress responses, will enhance the model's generalization and reliability across different populations. Additionally, advanced deep learning techniques, such as transfer learning and ensemble models, could be explored to improve classification performance and adaptability to different environments.

Moreover, future work could focus on real-time validation and deployment of the model in wearable devices and mobile health applications to facilitate continuous stress monitoring. Implementing federated learning approaches can allow for decentralized training on edge devices while preserving user privacy and security. Further research into explainability techniques, such as SHAP (Shapley Additive Explanations) and Grad-CAM (Gradient-weighted Class Activation Mapping), could enhance model interpretability, making it more transparent for healthcare professionals and users. By integrating these advancements, the study can contribute to developing more reliable, efficient, and accessible AI-driven stress detection systems for personalized health monitoring and mental well-being management.

8. REFERENCE

[1] H.-G. Kim, E.-J. Cheon, D.-S. Bai, Y. H. Lee, and B.-H. Koo, ‘‘Stressand heart rate variability: A meta-analysis and review of the literature,’’*Psychiatry Invest.*, vol. 15, no. 3, pp. 235–245, Mar. 2018.

[2] D. Muhajir, F. Mahananto, and N. A. Sani, ‘‘Stress level measurementsusing heart rate variability analysis on Android based application,’’ *Proc.Comput. Sci.*, vol. 197, pp. 189–197, Jan. 2022.

[3] J. Held, A. Vîslă, C. Wolfer, N. Messerli-Bürgy, and C. Flückiger, ‘‘Heartrate variability change during a stressful cognitive task in individuals withanxiety and control participants,’’ *BMC Psychol.*,vol.9,no. 1, p. 44,Mar. 2021.

[4] K. M. Dalmeida and G. L. Masala, ‘‘HRV features as viable physiologicalmarkers for stress detectionusing wearable devices,’’ *Sensors*, vol. 21,no. 8, p. 2873, Apr. 2021.

[5] J. A. Miranda-Correa, M. K. Abadi, N. Sebe, and I. Patras, ‘‘AMIGOS:A dataset for affect, personalityand mood research on individualsand groups,’’ *IEEE Trans. Affect. Comput.*, vol. 12, no. 2,pp. 479– 493,Apr./Jun. 2021.

[6] E. Won and Y.-K. Kim, ‘‘Stress, the autonomic nervous system, andthe immune-kynureninepathwayin the etiology of depression,’’ *CurrentNeuropharmacol.*, vol. 14, no. 7, pp. 665–673, Aug. 2016.

[7] B. Olshansky, H. N. Sabbah, P. J. Hauptman, andW. S. Colucci, ‘‘Parasympatheticnervous system and heart failure: Pathophysiology and potentialimplications for therapy,’’ *Circulation*, vol. 118, no. 8,pp.863–871,Aug. 2008.

[8] S. Goel, P. Tomar, and G. Kaur, ‘‘ECG feature extraction for stress recognition in automobile drivers,’’

Electron. J. Biol., vol. 12, no. 2, pp. 156–165,Mar. 2016.

[9] V. N. Hegde, R. Deekshit, and P. S. Satyanarayana, ‘‘A review on ECGsignal processing and HRVanalysis,’’ *J. Med. Imag. Health Informat.*,vol. 3, no. 2, pp. 270–279, Jun. 2013.

[10] M. Vollmer, ‘‘A robust, simple and reliable measure of heart rate variabilityusing relative RR intervals,’’ in *Proc. Comput. Cardiol. Conf. (CinC)*,Sep. 2015, pp. 609–612.

[11] M. H. Kryger, T. Roth, andW. C. Dement, *Principles and Practice of SleepMedicine*, 5th ed. Amsterdam, The Netherlands: Elsevier, 2011.

[12] M. Malik, J. T. Bigger, A. J. Camm, R. E. Kleiger, A. Malliani, A. J. Moss,and P. J. Schwartz, ‘‘Heartrate variability: Standards of measurement,physiological interpretation, and clinical use,’’ *Eur.Heart J.*, vol. 17, no. 3,pp. 354–381, Mar. 1996.

[13] S. Koldijk, M. Sappelli, S. Verberne, M. A. Neerincx, and W. Kraaij,‘‘The SWELL knowledge work dataset for stress and user modelingresearch,’’ in *Proc. 16th Int. Conf. Multimodal Interact.*, Nov. 2014,pp. 291–298.

[14] S. Koldijk, M. A. Neerincx, and W. Kraaij, ‘‘Detecting work stress inoffices by combiningunobtrusive sensors,’’ *IEEE Trans. Affect. Comput.*,vol. 9, no. 2, pp. 227–239, Apr. 2018.

[15] M. Albaladejo-González, J. A. Ruipérez-Valiente, and F. G. Mármol,‘‘Evaluating differentconfigurations of machine learning models and theirtransfer learning capabilities for stress detection using heart rate,’’ *J. AmbientIntell. Human. Comput.*, pp. 1–11, Aug. 2022, doi:10.1007/s12652-022-04365-z.

[16] R. Walambe, P. Nayak, A. Bhardwaj, and K. Kotecha, ‘‘Employingmultimodal machine learning forstress detection,’’ *J. Healthcare Eng.*,vol. 2021, Oct. 2021, Art. no. 9356452.

[17] A. Ibaida, A. Abuadbba, and N. Chilamkurti, ‘‘Privacy-preserving compressionmodel for efficient IoMT ECG sharing,’’ *Comput. Commun.*,vol. 166, pp. 1–8, Jan. 2021.

[18] W. C. Dobbs, M. V. Fedewa, H. V. MacDonald, C. J. Holmes, Z. S. Cicone,D. J. Plews, and M. R. Esco, ‘‘The accuracy of acquiring heart ratevariability from portable devices: A systematic review and meta-analysis,’’*Sports Med.*, vol. 49, no. 3, pp. 417–435, Mar. 2019.

[19] C.-M. Chen, S. Anastasova, K. Zhang, B. G. Rosa, B. P. L. Lo,H. E. Assender, and G.-Z. Yang, ‘‘Towards wearable and flexible sensorsand circuits integration for stress monitoring,’’ *IEEE J. Biomed.HealthInformat.*, vol. 24, no. 8, pp. 2208–2215, Aug. 2020.

[20] R. A. Rahman, K. Omar, S. A. M. Noah, M. S. N. M. Danuri,and M. A. Al-Garadi, ‘‘Application ofmachine learning methods inmental health detection: A systematic review,’’ *IEEE Access*, vol. 8,pp. 183952–183964, 2020.

[21] S. H. Jambukia, V. K. Dabhi, and H. B. Prajapati, ‘‘Application of machinelearning methods in mental health detection:Asystematic review,’’ in *Proc.Int. Conf. Adv. Comput. Eng. Appl.*, 2015, pp. 714–721.

[22] S. Celin and K. Vasanth, ‘‘ECG signal classification using various machinelearning techniques,’’ *J.Med. Syst.*, vol. 42, no. 12, p. 241, Oct. 2018.

[23] A. Padha and A. Sahoo, ‘‘A parametrized quantum LSTM model forcontinuous stress monitoring,’’in *Proc. 9th Int. Conf. Comput. Sustain.Global Develop. (INDIACom)*, Mar. 2022, pp. 261–266.

[24] S. Sriramprakash, V. D. Prasanna, and O. V. R. Murthy, ‘‘Stress detectionin working people,’’ *Proc.Comput. Sci.*, vol. 115, pp. 359–366, Dec. 2017