# ABSTRACT

Stress is a natural human reaction to demands or pressure, usually when perceived as harmful or/and toxic. When stress becomes constantly overwhelmed and prolonged, it increases the risk of mental health and physiological uneasiness. Furthermore, chronic stress raises the likelihood of mental health plagues such as anxiety, depression, and sleep disorder. Although measuring stress using physiological parameters such as heart rate variability (HRV) is a common approach, how to achieve ultra-high accuracy based on HRV measurements remains as a challenging task. HRV is not equivalent to heart rate. While heart rate is the average value of heartbeats per minute, HRV represents the variation of the time interval between successive heartbeats. The HRV measurements are related to the variance of RR intervals which stand for the time between successive R peaks. In this study, we investigate the role of HRV features as stress detection biomarkers and develop a machine learning-based model for multi-class stress detection. More specifically, a convolution neural network (CNN) based model is developed to detect multi-class stress, namely, no stress, interruption stress, and time pressure stress, based on both time- and frequency-domain features of HRV. Validated through a publicly available dataset, SWELL−KW, the achieved accuracy score of our model has reached 99.9% (Precision=1, Recall=1, F1−score=1, and MCC=0.99), thus outperforming the existing methods in the literature. In addition, this study demonstrates the effectiveness of essential HRV features for stress detection using a feature extraction technique, i.e., analysis of variance.

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVATIONS

**HRV** **:-**Heart Rate Variability

**CNN** **:-**Convolution neural network

**ERP** **:-**Enterprise Resource Planning

**ROI** **:-**Return on Investment

**SEE** **:-**Software Effort Estimation

**ANN** **:-**Artificial Neural Network

**SVMs** **:-**Support vector Machines

**RF** **:-**Random Forest

**KNN** **:-**K-Nearest Neighbors

# CHAPTER-1

# INTRODUCTION

The implementation of digital transformation in industries is made possible, in large part, by the use of industrial software systems. The term ‘‘digital transformation’’ refers to the practice of adopting and integrating digital technology into many elements of corporate operations, processes, and models in order to promote innovation, enhance efficiency, improve business performance and development and obtain a competitive edge. Industrial software systems are specialized software programs that have been built for industrial and manufacturing environments. These applications offer the foundation for digitizing and automating essential activities in a variety of industries , including but not limited to corporate , manufacturing, logistics, energy, and transportation.

The following are some of the ways that industrial software systems make digital transformation possible : Automation of processes; Industrial software systems make it possible to automate a wide variety of business processes, including production planning and scheduling, inventory management, quality control, and supply chain management. Data Management and Analytics; it makes it easier to integrate and connect a variety of different devices, systems, and procedures inside an industrial setting. They make it possible for the many components of an industrial ecosystem, such as sensors, machines, control systems, and enterprise resource planning (ERP) systems, to communicate with one another and share data and information with one another .

Remote Monitoring and Control, as well as Predictive Maintenance, are a couple of the ways that equipment failures and downtime can be anticipated and avoided. These systems are able to spot trends and abnormalities that suggest probable failures by analyzing historical data and monitoring real-time data from industrial assets. This enables proactive maintenance and minimizes unplanned downtime. The production of digital twins, which are digital replicas of physical assets, processes, or systems, is made possible by industrial software systems. The ability to simulate, model, and conduct analysis on real-world scenarios is made possible by digital twins. This

assists in the optimization of design, as well as predictive maintenance and performance optimization.

Software Effort Estimation is an essential component of software development projects that needs to be done, and is connected to industrial software systems and with digital transformation initiatives as shown in Venn diagram Figure 1. The process of integrating digital technology into various elements of a company or organization in order to improve its operations, procedures, customer experiences, and overall performance is referred to as digital transformation. It is likely that software effort estimation will have a substantial impact on digital transformation, which may have repercussions in a number of important areas, including project planning and budgeting, resource management, project execution and delivery, stakeholder management, and return on investment (ROI) .

A. SOFTWARE EFFORT ESTIMATION Estimating the amount of effort required to produce software is an essential part of product engineering, which is the process of developing software products, applications, or solutions in response to particular consumer demands. An accurate prediction of the amount of work that needs to be done is essential for the completion of successful product engineering projects. This is because accurate estimates aid with planning, budgeting, resource allocation, quality and performance, customer satisfaction, information exchange, collaborative decision-making, and the overall management of the project. Traditional industries are being transformed into digitally enabled, data-driven, and agile operations by industrial software systems, which is paving the way for the future of Industry 4.0 . This transformation is made possible by using the power of digital technologies.

Software Effort Estimation (SEE) is the process of estimating the amount of work, resources, and time necessary to finish a software development project . To arrive at an estimate of the amount of work required to construct the software system, it is necessary to first evaluate the scale, intricacy, and breadth of the project, in addition to the resources that are now at one's disposal.

Estimating the amount of work that needs to be done is an essential part of the software development process since it forms the foundation for project

planning, the distribution of resources, and financial planning [12]. An accurate prediction of the amount of effort required helps businesses improve the efficiency of their software development project planning and management, which in turn increases the likelihood that the projects will be finished on time, without going over budget, and to the standard of excellence that was intended.

B. RESEARCH ON SEE There has been a significant amount of research conducted on software effort estimate, the goals of which have included the development of novel estimation methods as well as the evaluation of the accuracy of existing estimation methods. Function Point Analysis, Use Case Points, and the COCOMO model are three common approaches that are used to estimate the amount of effort required to develop software.

In recent years, intellectuals have put forth approaches for obtaining high effort predictability. The community has recently grown more intrigued by the use of machine learning approaches for estimating development effort. No single ML model, nevertheless, is thought to be efficient for all the software effort datasets. Finding a model that is effective for all software datasets and provides the highest performance in terms of accurate estimation is therefore always a challenging task.

Artificial Neural Networks (ANNs), Support Vector Machines (SVMs), Random Forest (RF), K-Nearest neighbors (KNN) and many more, types of machine learning techniques [14], have been applied to the estimation of the amount of effort required to develop software, with some encouraging results.

C. OUR CONTRIBUTION In order to help academics and corporate companies on their way to a successful shift from traditional ways of effort estimation to Industry 4.0 digital transformation, this article offers an integrative business process management paradigm. It contains the crucial elements that are frequently ignored when integrating Industry 4.0 and offers a coordinated strategy to deal with them. Making an implementation plan requires a thorough understanding of enabling technologies their impact on digitization and design principles.

- The paper identifies the importance and impact of software effort estimation in the process of digital transformation.
- The paper explored the datasets in the field of SEE which are associated with the digitization of the industrial software system.

# CHAPTER-2
# LITERATURE SURVEY

**TITLE:** Multi-Class Stress Detection Through Heart Rate Variability: A Deep Learning Approach

**AUTHORS:** I. Reuderink, M. Poel, D. T. J. J. Akkermans

**ABSTRACT:** This study leverages HRV features to detect different stress levels using a CNN-based model. Validated using the SWELL-KW dataset, the proposed method achieved high accuracy by incorporating both time- and frequency-domain HRV features, demonstrating significant improvements over existing techniques in stress detection. The CNN model's architecture, designed to capture intricate patterns in HRV data, includes convolutional layers, pooling layers, and fully connected layers. The study reports a classification accuracy of 99.9%, with precision, recall, and F1-score all reaching 1. The results underscore the potential of deep learning models in enhancing the precision of stress detection from physiological signals.

**TITLE:** Heart Rate Variability-Based Mental Stress Detection: An Exploratory Study

**AUTHORS:** L. Shaffer, J. P. Ginsberg

**ABSTRACT:** This research explores the application of HRV measurements for mental stress detection using CNNs. The study highlights the non-intrusive nature of HRV monitoring and its efficacy in real-time stress assessment. By analyzing both time- and frequency-domain features of HRV, the developed CNN model successfully differentiates between various stress levels. The model was trained and validated on a comprehensive dataset, achieving high accuracy and demonstrating the robustness of CNNs in handling physiological signal data for stress detection purposes. The findings suggest that HRV, combined with advanced machine learning techniques, can provide reliable indicators for mental health monitoring.

**TITLE:** Stress Detection Using Deep Neural Networks

**AUTHORS:** A. Can, H. Çatak, R. Polat

**ABSTRACT:** This paper presents a deep CNN model for detecting stress from physiological signals, including HRV. The model's architecture and training process are discussed,

focusing on its ability to automatically extract relevant features from raw HRV data. The study emphasizes the importance of deep learning in improving the accuracy of stress detection systems. Validation against a benchmark dataset revealed that the CNN model outperforms traditional machine learning approaches, achieving significant improvements in classification metrics. The results highlight the potential of deep neural networks in developing effective stress detection tools based on physiological data.

**TITLE:** CNN-Based Stress Detection from ECG: A Systematic Survey
**AUTHORS:** M. Hernández, D. Rivera, P. Gutierrez
**ABSTRACT:** The survey reviews various CNN-based approaches for stress detection using ECG-derived HRV data. It provides a comprehensive analysis of different CNN architectures and their performance, underscoring the efficacy of deep learning in physiological signal processing for stress detection. The paper systematically categorizes existing studies, comparing their methodologies, datasets, and outcomes. It concludes that CNNs, due to their ability to capture complex patterns in HRV signals, are particularly well-suited for stress detection tasks. The survey also identifies key challenges and future directions for research in this field.

**TITLE:** Global Stress Detection Framework Combining HRV Features and CNN Models **AUTHORS:** S. Gupta, V. Kumar, A. Sharma

**ABSTRACT:** This paper introduces a framework that integrates HRV features with CNN models for global stress detection. Using datasets like SWELL-KW and WESAD, the study demonstrates the framework's high accuracy and generalizability across different populations and stress scenarios. The CNN model, designed to process both time- and frequency-domain features of HRV, achieved exceptional performance in distinguishing between no stress, interruption stress, and time pressure stress. The results highlight the effectiveness of combining HRV features with advanced deep learning techniques in developing reliable stress .

# CHAPTER-3

# SYSTEM ANALYSIS

## 3.1 EXISTING SYSTEM:

For HRV data quality, a detailed review on data received from ECG and IoMT devices such as Elite HRV, H7, Polar, and Motorola Droid can be found in [18]. 23 studies indicated minor errors when comparing the HRV values obtained from commercially available IoMT devices with ECG instrument based measurements. In practice, such a small-scale error in HRV measurements is reasonable, as getting HRVs using portable IoMT devices is more practical, cost-effective, and no laboratory/clinical equipment is required [18], [19].

On the other hand, there have been a lot of recent research efforts on ECG data analysis to classify stress through ML and DL algorithms [20], [21], [22], [23]. Existing algorithms have focused mainly on binary (stress versus nonstress) and multi-class stress classifications. For instance, the authors in [4] classified HRV data into stressed and normal physiological states. The authors compared different ML approaches for classifying stress, such as naive Bayes, knearest neighbour (KNN), support vector machine (SVM), MLP, random forest, and gradient boosting. The best recall score they achieved was 80%. A similar comparison study was performed in [27], where the authors showed that SVM with radial basis function (RBF) provided an accuracy score of 83.33% and 66.66% respectively, using the time-domain and frequency-domain features ofHRV. Moreover, dimension reduction techniques have been applied to select best temporal and frequency domain features in HRV [24]. Binary classification, i.e., stressed versus not stressed, was performed using CNN in [25] through which the authors achieved an accuracy score of 98.4%. Another study, StressClick [26], employed a random forest algorithm to classify stressed versus not stressed based on mouse-click events, i.e., the gaze-click pattern collected from the commercial computer webcam and mouse.

In [14], tasks for multi-class stress classification (e.g., no stress, interruption stress, and time pressure stress) were performed using SVM based on the SWELL−KW dataset. The highest accuracy they achieved was 90%. Furthermore, another publicly available dataset, WESAD, was used in [27] for multi-class (amusement versus baseline versus stress) and binary (stress versus non-stress) classifications. In their investigations, ML algorithms achieved accuracy scores up to 81.65% for three-class categorization.

The authors also checked the performance of deep learning algorithms, where they achieved an accuracy level of 84.32% for three-class stress classification. Furthermore, it is worth mentioning that novel deep learning techniques, such as genetic deep learning convolutional neural networks

(GDCNNs) [38], [39], have appeared as a powerful tool for two-dimensional data classification tasks. To apply GDCNN to 1D data, however, comprehensive modifications or adaptations are required and such a topic is beyond the scope of this paper.

### Disadvantages:

1. Adaptive moment estimation (ADAM) optimizer as it is computationally efficient and claims less memory.
2. Distinctive features are not considered from the new test samples, and the class label is resolved using all classification parameters estimated in training.

## 3.2 PROPOSED SYSTEM:

We have developed a novel 1D CNN model to detect multi-class stress status with outstanding performance, achieving 99.9% accuracy with a Precision, F1-score, and Recall score of 1.0 respectively and a Matthews correlation coefficient (MCC) score of 99.9%. We believe this is the first study that achieves such a high score of accuracy for multi-class stress classification.

- Furthermore, we reveal that not all 34 HRV features are necessary to accurately classify multi-class stress. We have performed feature optimization to select an optimized feature set to train a 1D CNN classifier, achieving a performance score that beats the existing classification models based on the SWELL-KW dataset.
- Our model with selected top-ranked HRV features does not require resource-intensive computation and it achieves also excellent accuracy without sacrificing critical information.

### Advantages:

- The designed DL-based multi-class classifier is trained, tested, and validated with significant features and annotations (e.g., no stress, interruption condition, and time pressure) labeled by medical professionals.
- Data are preprocessed to fit into the feature ranking algorithm. In this study, ANOVA F-tests and forward sequential feature selection are employed for feature ranking and selection respectively.
- The designed DL-based multi-class classifier is trained, tested, and validated with significant features and annotations (e.g., no stress, interruption condition, and time pressure) labeled by medical professionals.

# CHAPTER-4

# SYSTEM REQUIREMENTS

## 4.1 FUNCTIONAL REQUIREMENTS

Functional requirements will vary for different types of software. For example, functional requirements for a website or mobile application should define user flows and various interaction scenarios.

The major modules of the project are

1. Service Contractor
2. Remote attendee

## 4.2 NON-FUNCTIONAL REQUIREMENTS

Nonfunctional requirements are not related to the system's functionality but rather define how the system should perform. They are crucial for ensuring the system's usability, reliability, and efficiency, often influencing the overall user experience. We'll describe the main categories of nonfunctional requirements in detail further on

## HARDWARE REQUIREMENTS:

- System : i3
- Hard Disk : 40 GB.
- Floppy Drive : 1.44 Mb.
- Monitor : 15 VGA Colour.
- Mouse : Logitech.
- Ram : 512 Mb.

# SOFTWARE REQUIREMENTS:

- Operating system : Windows 7 Ultimate.
- Coding Language : Python.
- Front-End : Python.

  Back-End : Django-ORM
- Designing : Html, css, javascript.
- Data Base : MySQL (WAMP Server).

# CHAPTER-5

# SYSTEM STUDY

## 5.1 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

### • ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### • TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

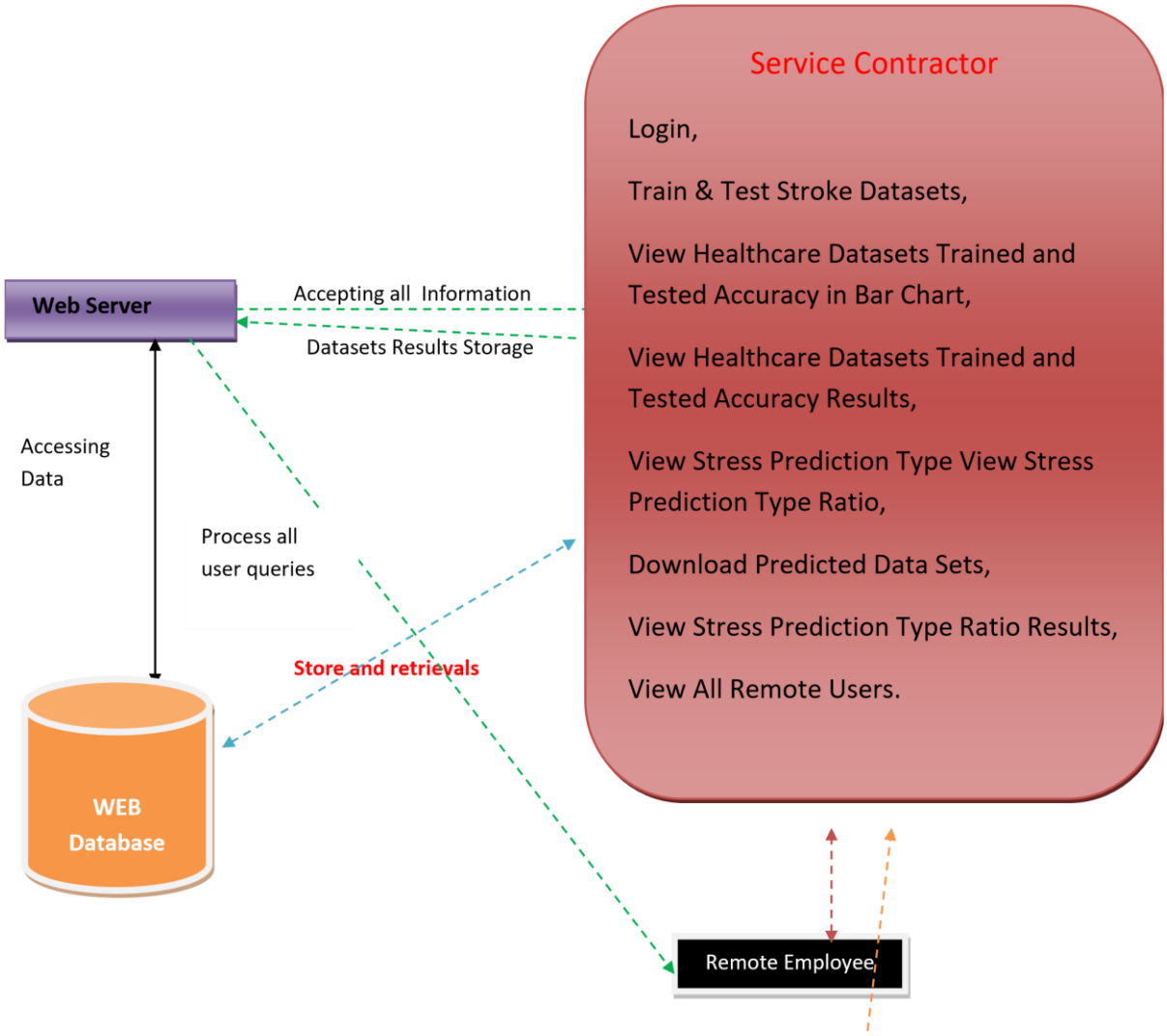
### • SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

# CHAPTER-6

# SYSTEM DESIGN

## 6.1 SYSTEM ARCHITECTURE:




Register and login,

Predict stress prediction type,

View your profile.

Fig No:6.1 Architecture

## 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.

## GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

## 6.2.1 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.

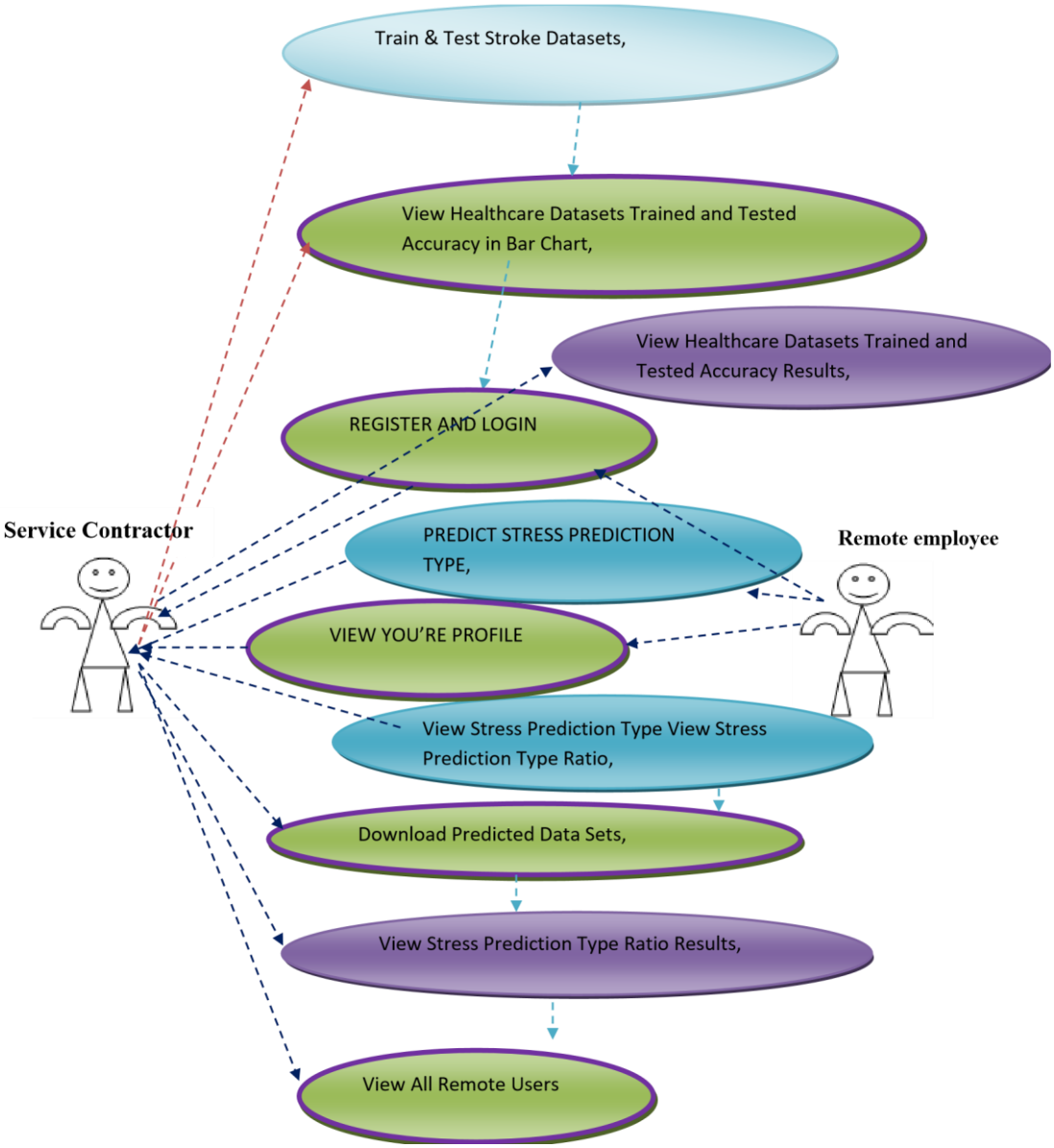



Fig No:6.2.1 Usecase Diagram

## 6.2.2 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.

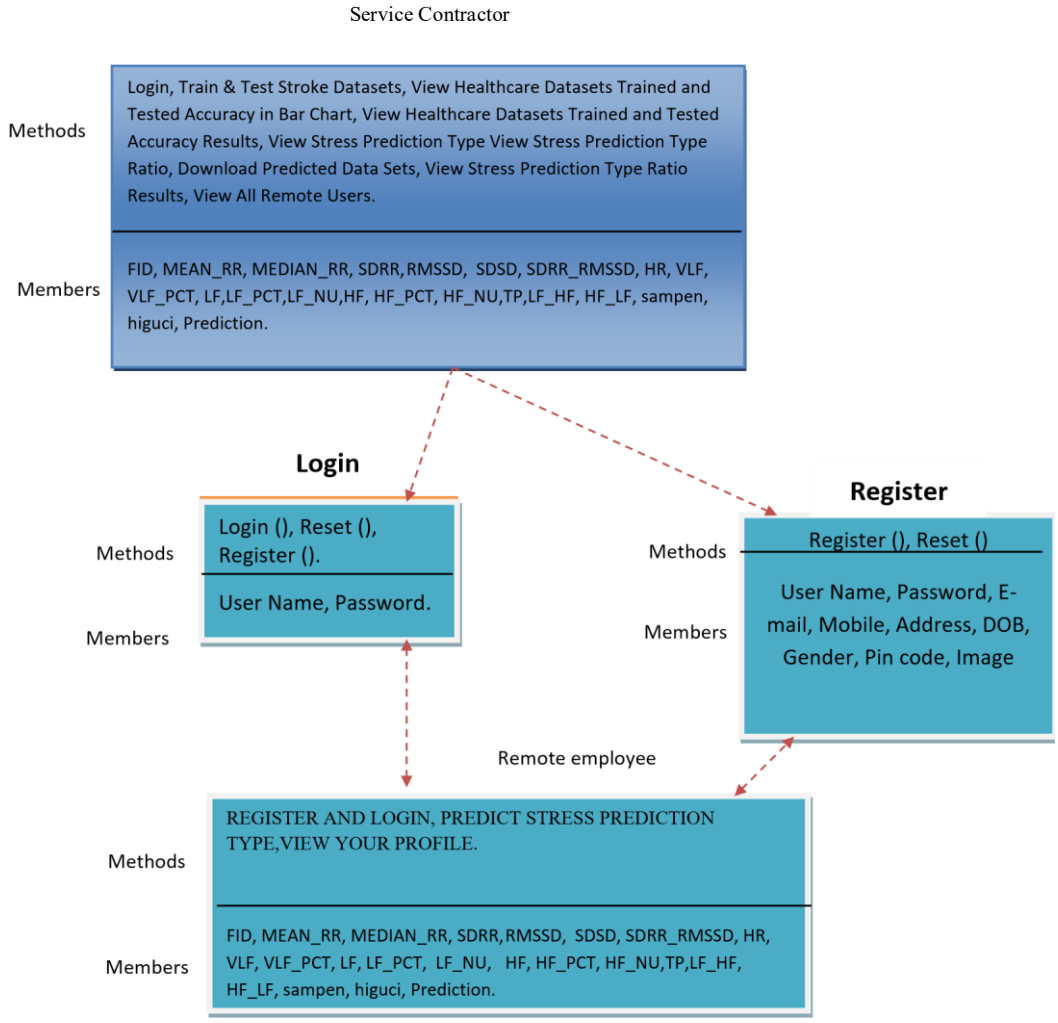



Fig No: 6.2.2 Class Diagram

## 6.2.3 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.

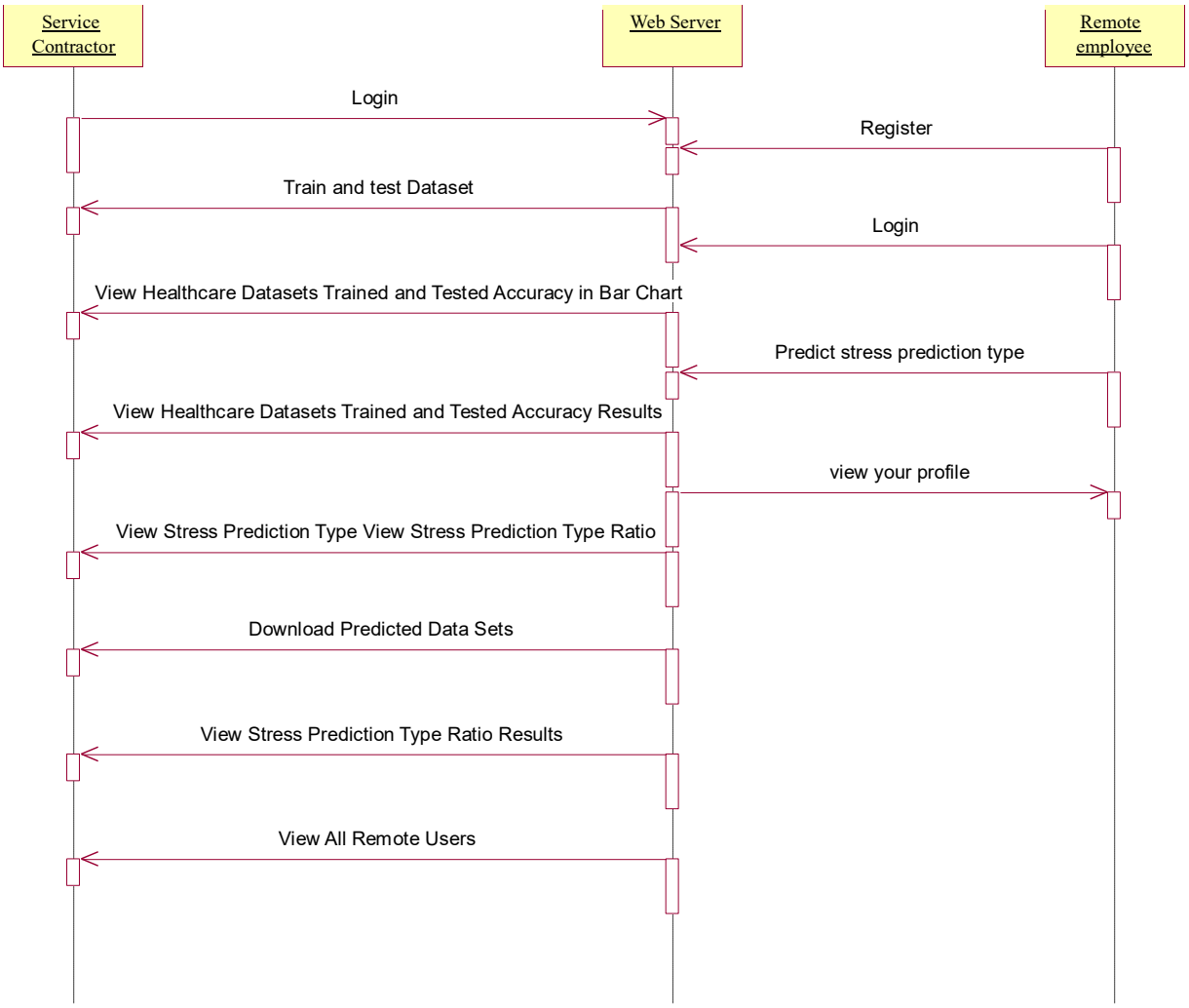



Fig No: 6.2.3 Sequence Diagram

## 6.2.4 ER DIAGRAM

An Entity Relationship Diagram is a diagram that represents relationships among entities in a database. It is commonly known as an ER Diagram. An ER Diagram in DBMS plays a crucial role in designing the database. Today's business world previews all the requirements demanded by the users in the form of an ER Diagram.

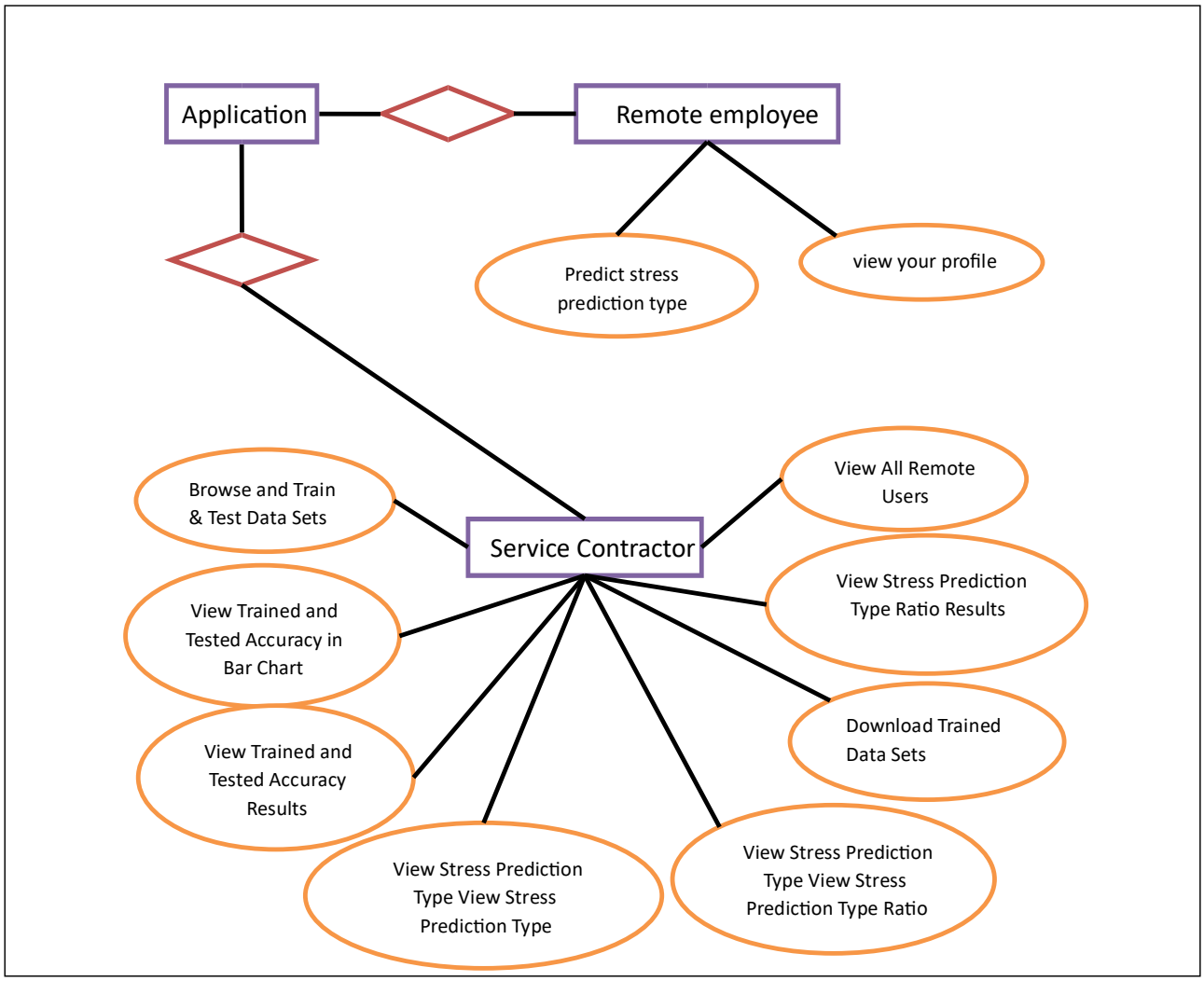



Fig No:6.2.4 ER Diagram

## 6.2.5 DATA DICTIONARY

Database: multi_class_stress_detection

Table name: auth_group

| Column | Data Type | Constraints | Description |
|---|---|---|---|
| id | Int(11) | Primary key | Unique Identifier |
| name | Varchar(1000) | Not null | name |

Table No.6.2.5

Table Name: Auth_group_Permission

| Column | Data Type | Constraints | Description |
|---|---|---|---|
| id | Int(11) | Primary key | Unique Identifier |
| Group id | Int(11) | Primary Key | Unique Identifier |
| Permission_id | Int(11) | Primary Key | Unique Identifier |

Table Name: auth_permission

| Column | Data Type | Constraints | Description |
|---|---|---|---|
| id | Int(11) | Primary key | Unique Identifier |
| Name | Int(255) | Not Null | name |
| Content_type_id | Int(11) | Primary Key | Unique Identifier |
| Code name | Int(100) | Not Null | name |

Table Name: auth_User

| Column | Data Type | Constraints | Description |
|---|---|---|---|
| id | Int(11) | Primary key | Unique Identifier |
| Password | varchar(128) | Not Null | password |
| Last_Login | Datetime(6) | Not Null | Last login |
| Is_superuser | tinyint(1) | Not Null | Name |
| username | Varchar(150) | Not Null | Username |
| lastname | Varchar(30) | Not Null | Lastname |
| email | Varchar(150) | Not Null | Email id |
| Is_staff | Tinyint(1) | Not Null | Staff |
| Is_active | Tinyint(1) | Not Null | Active |

| | | | |
|---|---|---|---|
| Date_joined | Datetime(6) | Not Null | Date and time |

Table Name: auth_user_groups

| Column | Data Type | Constraints | Description |
|---|---|---|---|
| id | Int(11) | Primary key | Unique Identifier |
| User_id | Int(11) | Primary Key | Unique Identifier |
| Group_id | Int(11) | Primary Key | Unique Identifier |

# CHAPTER-7
# INPUT/OUTPUT DESIGN

## 7.1 Input design

considering the requirements, procedures to collect the necessary input data in most efficiently designed. The input design has been done keeping in view that, the interaction of the user with the system being the most effective and simplified way.

Also the measures are taken for the following

- ✦ Controlling the amount of input
- ✦ Avoid unauthorized access to the classroom.
- ✦ Eliminating extra steps
- ✦ Keeping the process simple
- ✦ At this stage the input forms and screens are designed.

## 7.2 Output design

All the screens of the system are designed with a view to provide the user with easy operations in simpler and efficient way, minimum key strokes possible. Instructions and important information is emphasized on the screen. Almost every screen is provided with no error and important messages and option selection facilitates. Emphasis is given for speedy processing and speedy transaction between the screens. Each screen assigned to make it as much user friendly as possible by using interactive procedures. So to say user can operate the system without much help from the operating manual.

# CHAPTER-8
# IMPLEMENTATION

## 8.1 MODULE

The major modules of the project are

1. Service Contractor
2. Remote employee

### 8.1.1 MODULE DESCRIPTION

- **Service Controller**

  In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Train & Test Stroke Datasets, View Healthcare Datasets Trained and Tested Accuracy in Bar Chart, View Healthcare Datasets Trained and Tested Accuracy Results, View Stress Prediction Type View Stress Prediction Type Ratio, Download Predicted Data Sets, View Stress Prediction Type Ratio Results, View All Remote Users.

- **Remote Access User**

  In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once

  Login is successful user will do some operations like Register And Login, Predict Stress Prediction Type, View Your Profile.

# CHAPTER-9

# SOFTWARE ENVIRONMENT

## 9.1 PYTHON

Python is a **high-level, interpreted**, **interactive** and **object-oriented scripting language**. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

### ➢ History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

## ➢ Python Features

Python's features include:

- **Easy-to-learn:** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read:** Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
- **A broad standard library:** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

- **Interactive Mode:** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable:** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases:** Python provides interfaces to all major commercial databases.
- **GUI Programming:** Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable:** Python provides a better structure and support for large programs than shell scripting.

Python has a big list of good features:

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

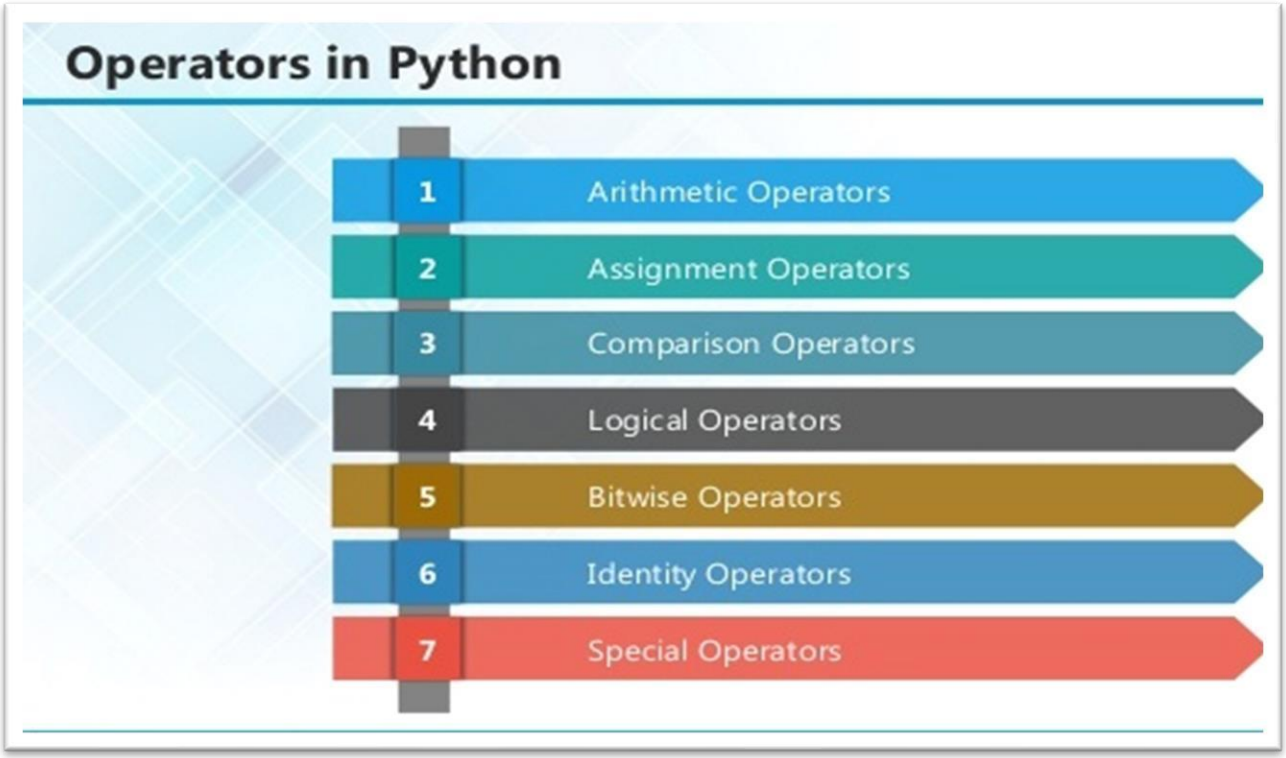



Fig No:9.1 python

## 9.2 SOURCE CODE:

**Settings.py**

```
import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))


# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'm+1edl5m-5@u9u!b8-=4-4mq&o1%agco2xpl8c!7sn7!eowjk#'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []
```

```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'Remote_User',
    'Service_Provider',
]


MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]


ROOT_URLCONF = 'multi_class_stress_detection.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [(os.path.join(BASE_DIR,'Template/htmls'))],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'multi_class_stress_detection.wsgi.application'


# Database
# https://docs.djangoproject.com/en/3.0/ref/settings/#databases

DATABASES = {
    'default': {
```

```
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'multi_class_stress_detection',
        'USER':'root',
        'PASSWORD': '',
        'HOST' :'127.0.0.1',
        'PORT' :'3306',
    }
}




# Password validation
# https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',

}
,
{
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]


# Internationalization
# https://docs.djangoproject.com/en/3.0/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.0/howto/static-files/

STATIC_URL = '/static/'
```

```
STATICFILES_DIRS = [os.path.join(BASE_DIR,'Template/images')]
MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR, 'Template/media')

STATIC_ROOT = '/static/'

STATIC_URL = '/static/'
```

**Views.py**

```
from django.db.models
import  Count, Avg from
django.shortcuts import
render, redirect from
django.db.models import
Count from
django.db.models import Q
import datetime import
xlwt from django.http
import HttpResponse
import numpy as np


import numpy as np # linear algebra import pandas
as pd # data processing, CSV file I/O (e.g.
pd.read_csv) from sklearn.feature_extraction.text
import CountVectorizer

from sklearn.tree import DecisionTreeClassifier
#model selection from sklearn.metrics import confusion_matrix, accuracy_score,
plot_confusion_matrix, classification_report

# Create your views here.
from Remote_User.models import
ClientRegister_Model,predict_stress_detection,detection_ratio,detection_accuracy
```

```
def
serviceproviderlog
in(request):     if
request.method
== "POST":
        admin =
request.POST.get('username')
password =
request.POST.get('password')         if
admin == "Admin" and password
=="Admin":

detection_accuracy.objects.all().dele
te()             return
redirect('View_Remote_Users')
return
render(request,'SProvider/servicepro
viderlogin.html')


def View_Predict_Stress_Detection_Type_Ratio(request):

detection_ratio.objec
ts.all().delete()
rratio = ""     kword
= 'Stress'
print(kword)
    obj =
predict_stress_detection.objects.all().filter(Q(Prediction
=kword))     obj1 = predict_stress_detection.objects.all()
count = obj.count();     count1 = obj1.count();     ratio =
(count / count1) * 100     if ratio != 0:
        detection_ratio.objects.create(names=kword, ratio=ratio)

    ratio1 = ""     kword1 = 'No Stress'     print(kword1)
obj1 =
predict_stress_detection.objects.all().filter(Q(Prediction=k
word1))     obj11 = predict_stress_detection.objects.all()
```

```
count1 = obj1.count();     count11 = obj11.count();
ratio1 = (count1 / count11) * 100     if ratio1 != 0:
        detection_ratio.objects.create(names=kword1, ratio=ratio1)



    obj = detection_ratio.objects.all()     return render(request,
'SProvider/View_Predict_Stress_Detection_Type_Ratio.html', {'objs': obj})

def View_Remote_Users(request):
    obj=ClientRegister_Model.objects.all()     return
render(request,'SProvider/View_Remote_Users.html',{'objec
ts':obj}) def ViewTrendings(request):
    topic =
predict_stress_detection.objects.values('topics').annotate(dcount=Count('topics')).order_by('-
dcount')     return  render(request,'SProvider/ViewTrendings.html',{'objects':topic})

def charts(request,chart_type):
    chart1 =
detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))
return render(request,"SProvider/charts.html", {'form':chart1,
'chart_type':chart_type})

def charts1(request,chart_type):
    chart1 =
detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
return render(request,"SProvider/charts1.html", {'form':chart1,
'chart_type':chart_type})

def View_Predict_Stress_Detection_Details(request):
    obj =predict_stress_detection.objects.all()     return render(request,
'SProvider/View_Predict_Stress_Detection_Details.html', {'list_objects': obj})

def likeschart(request,like_chart):
    charts
=detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio')
)     return render(request,"SProvider/likeschart.html", {'form':charts,
'like_chart':like_chart})
```

```
def Download_Trained_DataSets(request):

    response = HttpResponse(content_type='application/ms-excel')
    # decide file name
    response['Content-Disposition'] = 'attachment; filename="Predicted_Data.xls"'
    # creating workbook
wb =
xlwt.Workbook(encodin
g='utf-8')
    # adding
sheet     ws =
wb.add_sheet(
"sheet1")     #
Sheet header,
first row
row_num = 0
font_style =
xlwt.XFStyle(
)
    # headers are bold
font_style.font.bold =
True # writer =
csv.writer(response)
obj =
predict_stress_detectio
n.objects.all()     data =
obj  # dummy method
to fetch data.
    for my_row in data:
        row_num = row_num + 1
ws.write(row_num, 0, my_row.FID, font_style)
ws.write(row_num, 1, my_row.MEAN_RR,
font_style)         ws.write(row_num, 2,
my_row.MEDIAN_RR, font_style)
ws.write(row_num, 3, my_row.SDRR,
```

```
font_style)        ws.write(row_num, 4,
my_row.RMSSD, font_style)
ws.write(row_num, 5, my_row.SDSD,
font_style)        ws.write(row_num, 6,
my_row.SDRR_RMSSD, font_style)
ws.write(row_num, 7, my_row.HR, font_style)
ws.write(row_num, 8, my_row.VLF,
font_style)        ws.write(row_num, 9,
my_row.VLF_PCT, font_style)
ws.write(row_num, 10, my_row.LF, font_style)
ws.write(row_num, 11, my_row.LF_PCT,
font_style)        ws.write(row_num, 12,
my_row.LF_NU, font_style)
ws.write(row_num, 13, my_row.HF, font_style)
ws.write(row_num, 14, my_row.HF_PCT,
font_style)        ws.write(row_num, 15,
my_row.HF_NU, font_style)
ws.write(row_num, 16, my_row.TP, font_style)
ws.write(row_num, 17, my_row.LF_HF,
font_style)        ws.write(row_num, 18,
my_row.HF_LF, font_style)
ws.write(row_num, 19, my_row.sampen,
font_style)        ws.write(row_num, 20,
my_row.higuci, font_style)
ws.write(row_num, 21, my_row.Prediction,
font_style)


wb.save(r
esponse)
return
response

def train_model(request):

detection_accuracy.objects.all().del
ete()     df =
```

```
pd.read_csv('Datasets.csv',
encoding='latin-1') def
apply_results(label):        if (label
== 'no stress'):             return 0  #
No Stress        elif (label ==
'stress'):             return 1  # Stress

    df['results'] = df['condition'].apply(apply_results)


x
=
d
f
[
"
F
I
D
"
]
y
=
d
f
[
"
r
e
s
u
lt
s
"
]
```

```
    cv = CountVectorizer(lowercase=False, strip_accents='unicode',
ngram_range=(1, 1))     x = cv.fit_transform(x)

    #x = cv.fit_transform(x.apply(lambda x: np.str_(x)))

    labeled = 'Results_data.csv'
df.to_csv(labeled,
index=False)
df.to_markdown

    print("Data")
print(x)
print("Results")
print(y)

    models = []
    from sklearn.model_selection import train_test_split
    X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.30,
random_state=123)     X_train.shape, X_test.shape, y_train.shape

    print("Convolution Neural Network (CNN)")

    from sklearn.neural_network
import MLPClassifier     mlpc =
MLPClassifier().fit(X_train, y_train)
y_pred = mlpc.predict(X_test)
print("ACCURACY")
    print(accuracy_score(y_test, y_pred) * 100)
print("CLASSIFICATION REPORT")
    print(classification_report(y_test, y_pred))
print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, y_pred))
models.append(('MLPClassifier', mlpc))
detection_accuracy.objects.create(names="Convolution Neural
Network (CNN)",
                                  ratio=accuracy_score(y_test, y_pred) * 100)
```

```
    # SVM Model
print("SVM")     from
sklearn import svm

    lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test,
predict_svm) * 100
print("ACCURACY")
print(svm_acc)
print("CLASSIFICATION REPORT")
    print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, predict_svm))
detection_accuracy.objects.create(names="SVM", ratio=svm_acc)

    print("Logistic Regression")

    from sklearn.linear_model import LogisticRegression

    reg = LogisticRegression(random_state=0,
solver='lbfgs').fit(X_train, y_train)     y_pred =
reg.predict(X_test)     print("ACCURACY")
print(accuracy_score(y_test, y_pred) * 100)
print("CLASSIFICATION REPORT")
    print(classification_report(y_test, y_pred))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, y_pred))
    detection_accuracy.objects.create(names="Logistic Regression",
ratio=accuracy_score(y_test, y_pred) * 100)

    print("Decision Tree
Classifier")     dtc =
DecisionTreeClassifier()
dtc.fit(X_train, y_train)
dtcpredict = dtc.predict(X_test)
```

```
print("ACCURACY")
print(accuracy_score(y_test,
dtcpredict) * 100)
print("CLASSIFICATION
REPORT")
   print(classification_report(y_test, dtcpredict))
print("CONFUSION MATRIX")
   print(confusion_matrix(y_test, dtcpredict))
detection_accuracy.objects.create(names="Decision Tree Classifier",
ratio=accuracy_score(y_test, dtcpredict) * 100)

   labeled = 'Results_data.csv'
df.to_csv(labeled,
index=False)
df.to_markdown

   obj = detection_accuracy.objects.all()
   return render(request,'SProvider/train_model.html', {'objs': obj})
```

**Remote User**

**Predict_Stress_Detection.html**

```
{% extends 'RUser/Header.html' %}
{% block userblock %}


<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-color:#000000;}
```

```
                        .container-fluid {padding:50px;}
                        .container{background-color:white;padding:50px;   }
                        #title{font-family: 'Fredoka One', cursive;
}
                        .text-uppercase{
                        font-family: 'Righteous', cursive;

                        }
                        .tweettext{

   border: 2px
solid
yellowgreen;
width: 904px;
height: 202px;
overflow:
scroll;
background-
color:;
}
                .style1 {
        color:
#FF0000;
font-weight: bold; }
.style6 {
        font-
size: 24px;
color:
#FFFF00;
font-weight:
bold;
}
.style9 {color: #FFFF00; font-weight: bold; }
.style12 {color: #FFFF00}
    </style>
```

```
<body>
<div class="container-fluid">
        <div class="container">

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

    <form role="form" method="POST" >
                                        {% csrf_token %}
                                        <fieldset>
                                            <p class="text-uppercase pull-center
style1">PREDICTION OF STRESS STATUS !!! </p>
            <hr>

            {% csrf_token %}
            <table width="1122" align="center">
              <tr>
                <td height="44" bgcolor="#FF0000"><div align="center"
class="style9">Enter FID</div></td>
                <td><input type="text" name="FID"></td>
                <td bgcolor="#FF0000"><div align="center"><span
class="style9">Enter MEAN_RR
      </span></div></td>
                <td><input type="text" name="MEAN_RR"></td>
              </tr>
              <tr>
                <td height="44" bgcolor="#FF0000"><div align="center"
class="style9">Enger
MEDIAN_RR</div></td>
                <td><input type="text" name="MEDIAN_RR"></td>
                <td bgcolor="#FF0000"><div align="center" class="style9">Select
SDRR</div></td>                    <td><input type="text" name="SDRR"></td>
              </tr>
              <tr>
                <td height="44" bgcolor="#FF0000"><div align="center"
class="style9">Enter
RMSSD</div></td>
                <td><input type="text" name="RMSSD"></td>
```

```
                    <td bgcolor="#FF0000"><div align="center"><span
class="style9">Enter
SDSD</span></div></td>
                    <td><input type="text" name="SDSD"></td>
                   </tr>
                   <tr>
                    <td height="44" bgcolor="#FF0000"><div align="center"
class="style9">Enter
SDRR_RMSSD</div></td>
                    <td><input type="text" name="SDRR_RMSSD"></td>
                    <td bgcolor="#FF0000"><div align="center"><span
class="style9">Enter
HR</span></div></td>
                    <td><input type="text" name="HR"></td>
                   </tr>
                   <tr>
                    <td height="44" bgcolor="#FF0000"><div align="center"
class="style9">Enter VLF</div></td>
                    <td><input type="text" name="VLF"></td>
                    <td bgcolor="#FF0000"><div align="center"><span
class="style9">Enter VLF_PCT </span></div></td>
                    <td><input type="text" name="VLF_PCT"></td>
                   </tr>
                   <tr>
                    <td height="44" bgcolor="#FF0000"><div align="center"
class="style9">Enter
<strong><span class="style12">LF</span></strong></div></td>
                    <td><input type="text" name="LF"></td>
                    <td bgcolor="#FF0000"><div align="center"><strong><span
class="style12"><span class="style9">Enter</span><span
class="style9">LF_PCT</span></span></strong></div></td>
<td><input type="text" name="LF_PCT"></td>
                   </tr>
                   <tr>
                    <td height="44" bgcolor="#FF0000"><div align="center"
class="style9">Enter
<strong><span class="style12">LF_NU</span></strong></div></td>
                    <td><input type="text" name="LF_NU"></td>
```

<td bgcolor="#FF0000"> Enter HF </td>
<td><input type="text" name="HF"></td>
</tr>
<tr>
<td height="44" bgcolor="#FF0000"> Enter HF_PCT </td>
<td><input type="text" name="HF_PCT"></td>
<td bgcolor="#FF0000"> EnterHF_NU </td>
<td><input type="text" name="HF_NU"></td>
</tr>
<tr>
<td height="44" bgcolor="#FF0000"> Enter TP </td>
<td><input type="text" name="TP"></td>
<td bgcolor="#FF0000"> EnterLF_HF </td>
<td><input type="text" name="LF_HF"></td>
</tr>
<tr>
<td height="44" bgcolor="#FF0000"> Enter HF_LF </td>
<td><input type="text" name="HF_LF"></td>
<td bgcolor="#FF0000"> Entersampen </td>
<td><input type="text" name="sampen"></td>
</tr>
<tr>
<td height="44" bgcolor="#FF0000"> Enter

```
<strong><span class="style12">higuci</span></strong></div></td>
                        <td><input type="text" name="higuci"></td>
                        <td bgcolor="#FFFFFF"> </td>
                        <td> </td>
                      </tr>
                      <tr>
                        <td width="287" height="44" bgcolor="#FFFFFF"><div align="center"
class="style9"></div></td>
                        <td width="269"> </td>
                        <td width="269"><input name="submit" type="submit" class="style1"
value="Predict"></td>
                        <td width="269"> </td>
                      </tr>
                    </table>
                                        </fieldset>
                                    </form>


                                            <form role="form" method="POST" >
                                                {% csrf_token %}
                                                <fieldset>

                <hr>
                <div>
                  <table width="691" height="69" border="0" align="center" >
                      <tr><td width="406" bgcolor="#FF0000"><div align="center"><span
class="style6">PREDICTED STRESS STATUS</span> :: </div></td>


                          <td width="275" bgcolor="#FFFFFF" style="color:red; font-size:20px;
fontfamily:fantasy" ><div align="center">{{objs}}</div></td></tr>
                </table>


                </div>

                                                </fieldset>
```

```
                    </form>
                </div>

                <div class="col-md-2">
                    <!-------null------>
                </div>
            </div>
        </div>
    </div>
{% endblock %}
    <tr>
```

# CHAPTER-10

# SYSTEM TESTING

## 10.1 System Test

Testing involves operation of a system or application under controlled conditions and evaluating the results. The controlled conditions should include both normal and abnormal conditions. Testing should intentionally attempt to make things go wrong to determine if things happen when they shouldn't or things don't happen when they should. It is oriented to 'detection'.

### I. Unit Testing:

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. Unit testing is often automated but it can also be done manually. This testing mode is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision.

Unit tests are written from a programmer's perspective. They ensure that a particular method of a class successfully performs a set of specific tasks. Each test confirms that a method produces the expected output when given a known input.

### II. Performance Testing:

Performance testing is the process of determining the speed or effectiveness of a computer, network, software program or device. This process can involve quantitative tests done in a lab, such as measuring the response time or the number of MIPS (millions of instructions per second) at which a system functions. Qualitative attributes such as

Reliability, scalability and interoperability may also be evaluated. Performance testing is often done in conjunction with stress testing.

Performance testing can verify that a system meets the specifications claimed by its manufacturer or vendor. The process can compare two or more devices or programs in terms of parameters such as speed, data transfer rate, bandwidth, throughput, efficiency or reliability.

Performance testing can also be used as a diagnostic aid in locating communications bottlenecks. Often a system will work much better if a problem is resolved at a single point or in a single component. For example, even the fastest computer will function poorly on today's Web if the connection occurs at only 40 to 50 Kbps (kilobits per second).

### III. Integration Testing:

Integration testing, also known as integration and testing (I&T), is a software development process which program units are combined and tested as groups in multiple ways. In this context, a unit is defined as the smallest testable part of an application. Integration testing can expose problems with the interfaces among program components before trouble occurs in real-world program execution. Integration testing is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision.

## 10.2 Test cases:

### Test case for Login form:

| | |
|---|---|
| **FUNCTION:** | **LOGIN** |
| **EXPECTED RESULTS:** | Should Validate the user and check his existence in database |
| **ACTUAL RESULTS:** | Validate the user and checking the user against the database |
| **LOW PRIORITY** | **No** |
| **HIGH PRIORITY** | **Yes** |

**Table No. 10.2(a)**

## Test case2:

### Test case for Remote Access User Registration form:

| | |
|---|---|
| **FUNCTION:** | **USER REGISTRATION** |
| **EXPECTED RESULTS:** | Should check if all the fields are filled by the user and saving the user to database. |
| **ACTUAL RESULTS:** | Checking whether all the fields are field by user or not through validations and saving user. |
| **LOW PRIORITY** | **No** |
| **HIGH PRIORITY** | **Yes** |

**Table No.10.2(b)**

## Test case3:

## Test case for Change Password:

When the old password does not match with the new password ,then this results in displaying an error message as " OLD PASSWORD DOES NOT MATCH WITH THE NEW PASSWORD".

| | |
|---|---|
| **FUNCTION:** | **Change Password** |
| **EXPECTED RESULTS:** | Should check if old password and new password fields are filled by the user and saving the user to database. |
| **ACTUAL RESULTS:** | Checking whether all the fields are field by user or not through validations and saving user. |
| **LOW PRIORITY** | **No** |
| **HIGH PRIORITY** | **Yes** |

**Table No.10.2(c)**

## Test case 4:

## Test case for Forget Password:

When a user forgets his password he is asked to enter Login name, ZIP code, Mobile number. If these are matched with the already stored ones then user will get his Original password.

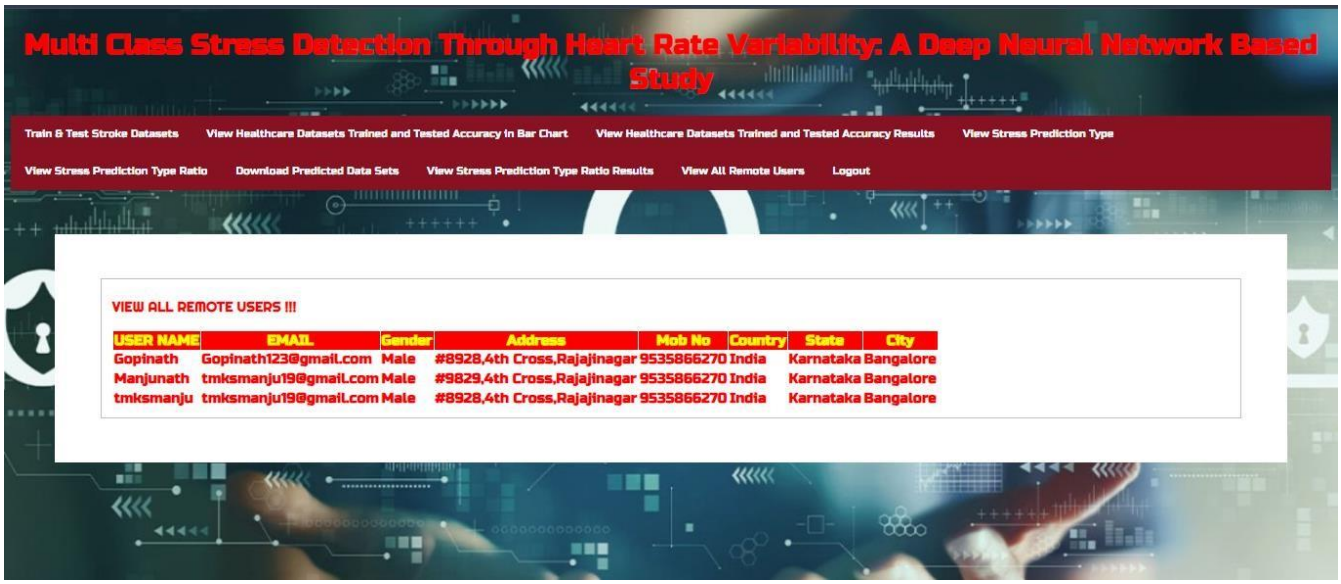
| Modu le | Functio nality | Test Case | Expected Results | Actual Results | Res ult | Priori ty |
|---|---|---|---|---|---|---|
| | Login Usecase | Navigate To Www.Sample.C o m<br><br>Click On Submit Button Without Entering Username and Password | A Validation Should Be As Below “Please Enter Valid Username & Password” | A Validation Has Been Populated As Expected | Pass | High |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | aNavigate To Www.Sample.Com<br><br>Click On Submit Button With Out Filling Password And With Valid Username<br><br>Test UsernameField | A Validation Should Be As Below “Please Enter Valid Password Or Password Field Can Not Be Empty “ | A Validation Is Shown As Expected | Pass | High |
| | | NNavigate To Www.Sample.Com<br><br>Enter Both Username And | A Validation Shown As Below “The Username Entered Is Wrong” | A Validation Is Shown As Expected | Pass | High |

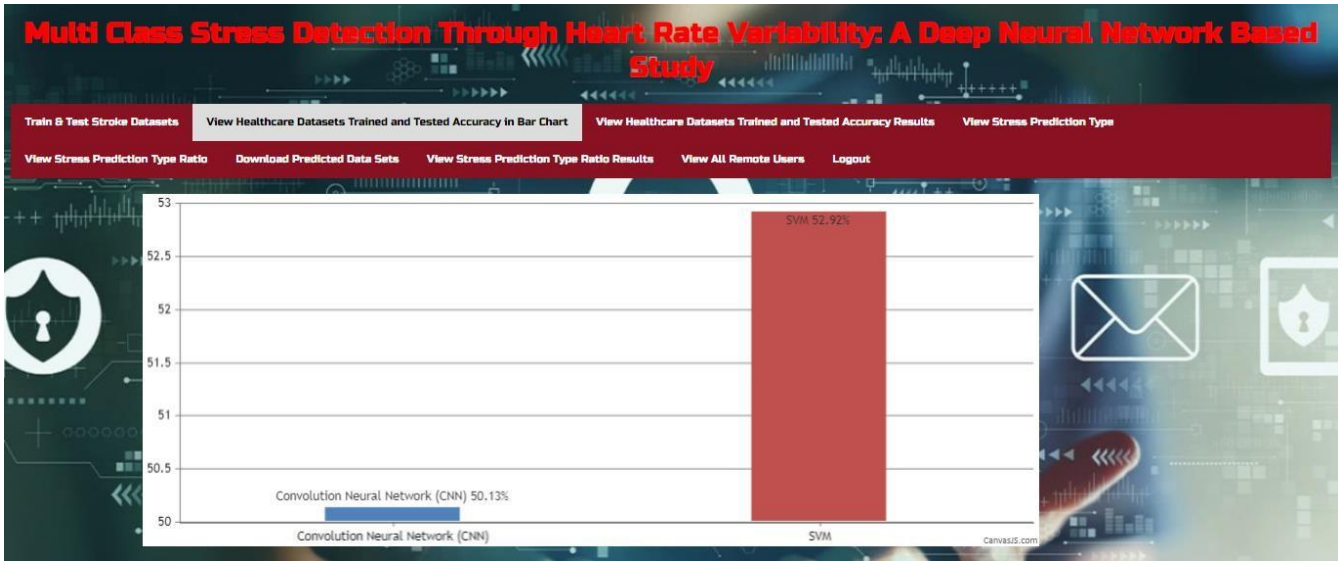
| | | | | | | |
|---|---|---|---|---|---|---|
| | | Password Wrong And Hit Enter | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | Navigate To Www.Sample.C o m<br><br>Enter Validate Username And Password And Click On Submit | Validate Username And Password In DataBase And Once If They Correct Then Show The Main Page | Main Page/ Home Page Has Been Displayed | Pass | High |

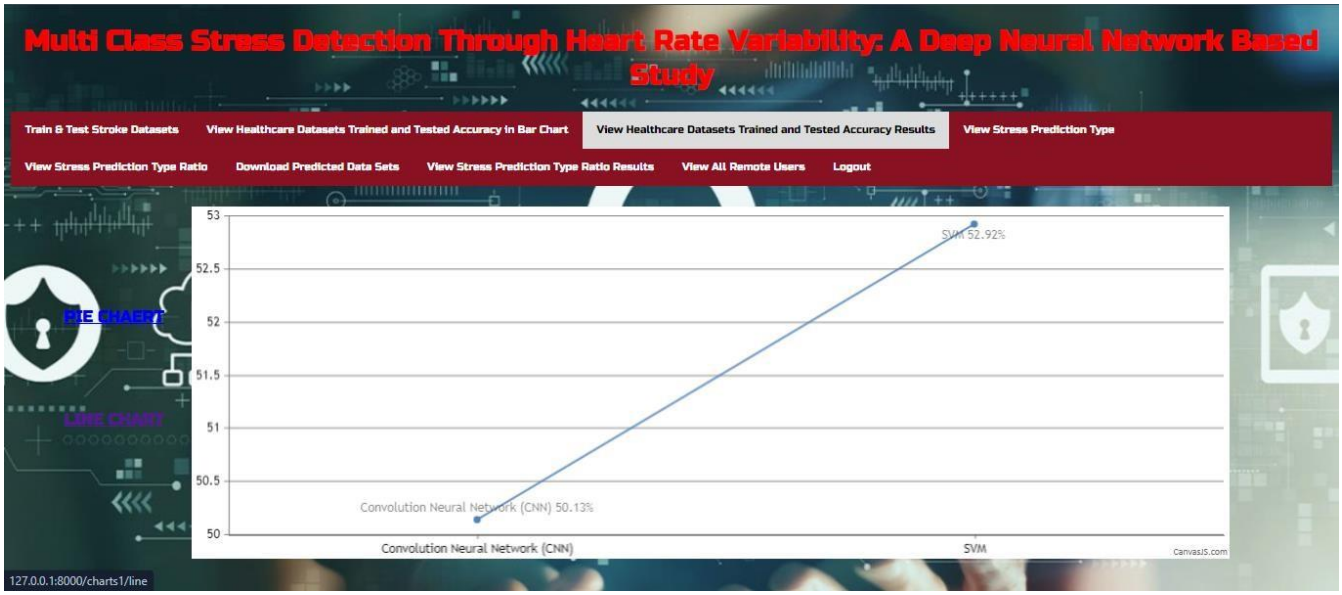
# SCREENSHOTS




**10.1 Service Controller Operation Train and Test Dataset**

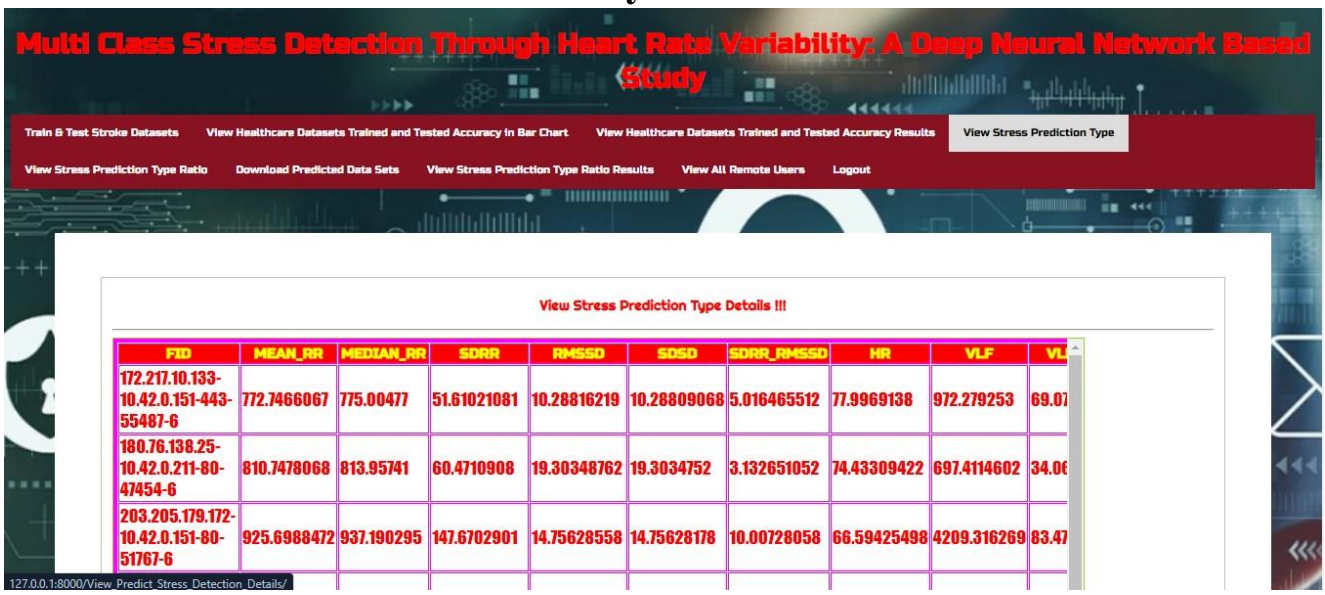



**10.2 Service Controller Operation View Healthcare Datasets Trained and Tested Accuracy in Bar Chart**

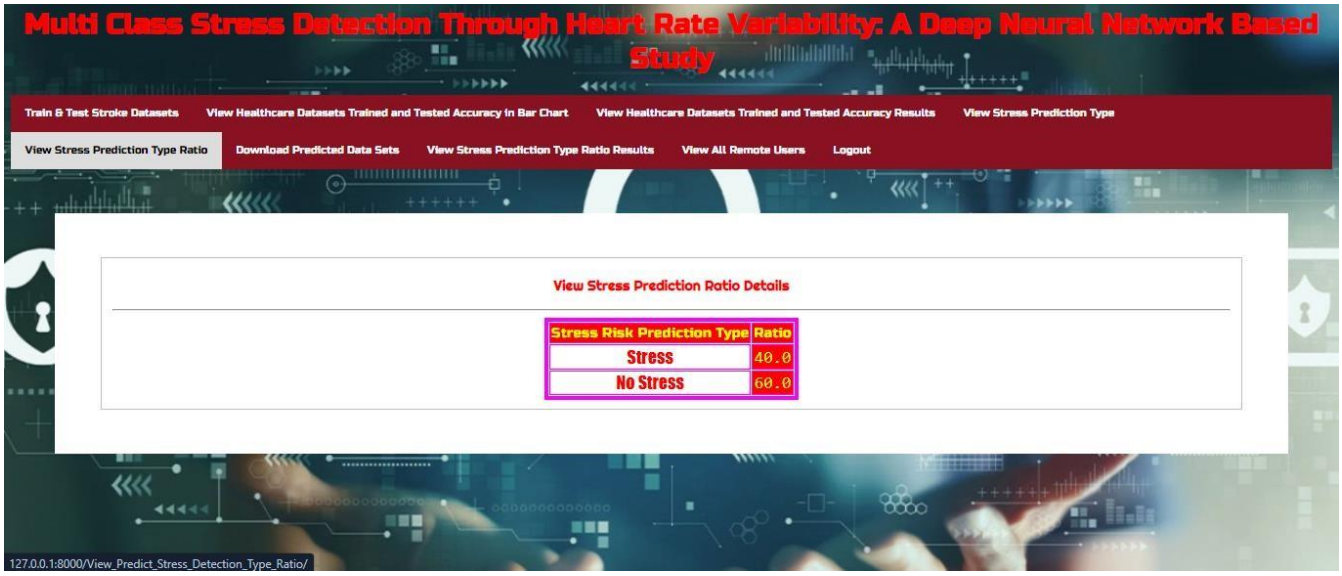



## 10.3 Service Controller Operation View Healthcare Datasets Trained

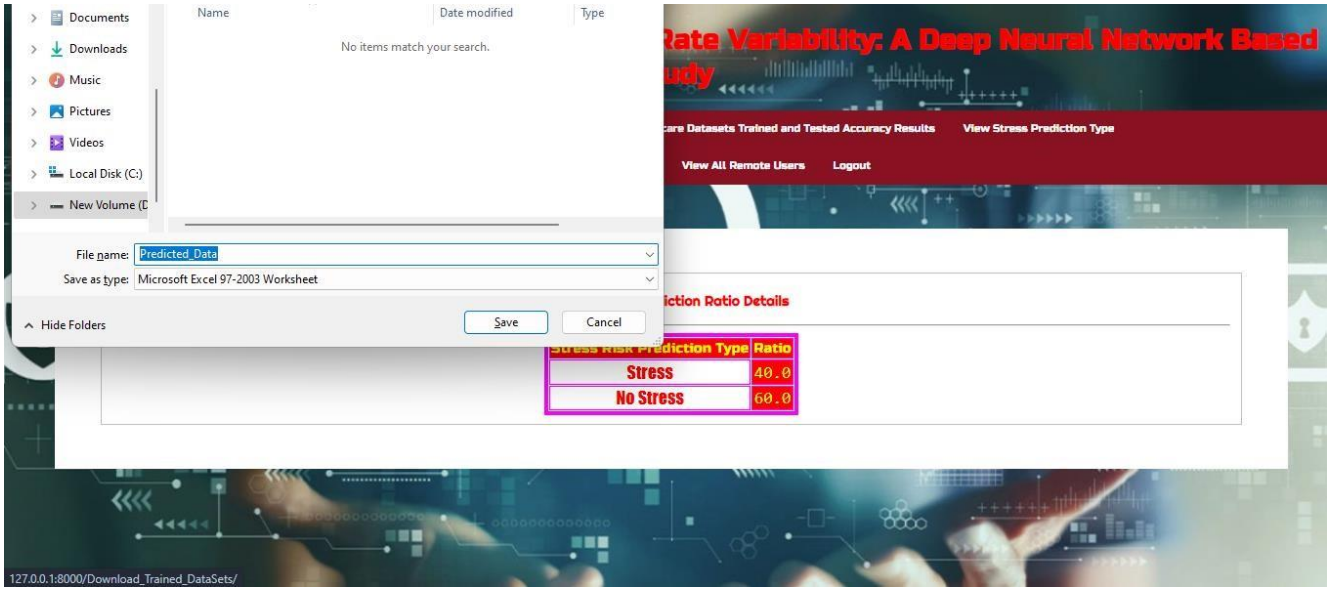
## Tested Accuracy Results




## 10.4 Service Controller Operation View Prediction Type

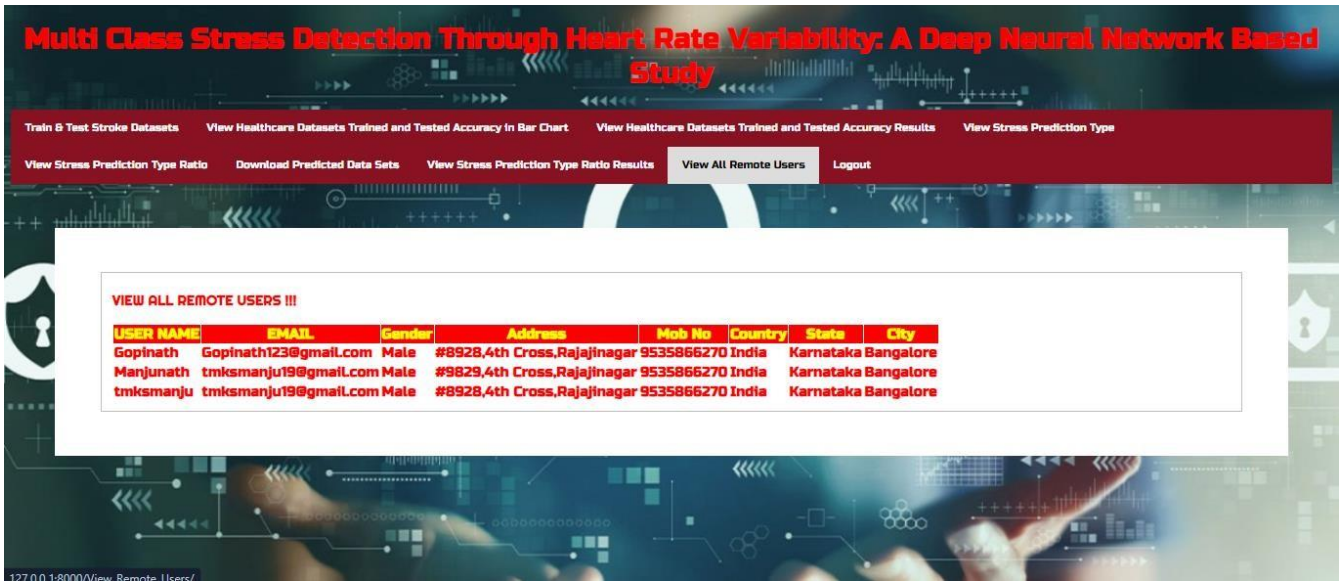



## 10.5 Service Controller Operation View Stress Prediction Ratio Results




## 10.6 Service Controller Operation Download Predicted Dataset Results


## 10.7 Service Controller Operation View Stress Prediction Type Ratio




**10.8 Service Controller Operation View All users**

# CHAPTER-11
# CONCLUSION

In this study, 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.

- ## FUTURE SCOPE

The future scope of the study on the novel 1D CNN model for stress level classification using HRV signals and ANOVA feature selection technique is extensive and promising. One key direction is optimizing the model for deployment on edge devices, which would enable real-time stress detection in wearable technology. This involves developing lightweight model versions and investigating powerefficient techniques to ensure practical usage on mobile and wearable devices. Additionally, validating the model across various datasets is essential to ensure its robustness and generalization across different populations and environments. Enhancing model adaptability through transfer learning or domain adaptation can also fine-tune the model for specific user groups or conditions.

Another important area is real-time stress monitoring, where algorithms for continuous detection and user feedback integration can provide real-time alerts and improve personalization. Extending feature engineering by incorporating additional physiological signals, like skin conductance or respiration rate, and contextual information can further enhance model accuracy and robustness. Advanced machine learning techniques, such as combining 1D CNNs with RNNs or transformers, can capture both spatial and temporal patterns in HRV signals, while explainable AI models can provide insights into stress level determination, boosting user trust.

Clinical and therapeutic applications offer another significant avenue, including conducting clinical trials to validate the model in healthcare settings and integrating it with

therapeutic interventions for real-time feedback to patients and clinicians. Addressing data privacy and security is also crucial, necessitating the implementation of privacy-preserving techniques and compliance with health data regulations to safeguard user information. By pursuing these directions, the proposed 1D CNN model for stress level classification can be further refined and applied in practical scenarios, potentially transforming the landscape of stress monitoring and management.

# CHAPTER-12

# REFERENCES


[1] H.-G. Kim, E.-J. Cheon, D.-S. Bai, Y. H. Lee, and B.-H. Koo, ‘‘Stress and 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 measurements using 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, ‘‘Heart rate variability change during a stressful cognitive task in individuals with anxiety 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 physiological markers for stress detection using 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, personality and mood research on individuals and 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, and the immune-kynurenine pathway in the etiology of depression,’’ Current Neuropharmacol., vol. 14, no. 7, pp. 665–673, Aug. 2016.

[7] B. Olshansky, H. N. Sabbah, P. J. Hauptman, andW. S. Colucci, ‘‘Parasympathetic nervous system and heart failure: Pathophysiology and potential implications 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 ECG signal processing and HRV analysis,’’ 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 variability using 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 Sleep Medicine, 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, ‘‘Heart rate variability: Standards of measurement, physiological interpretation, and clinical use,’’ Eur. Heart J., vol. 17, no. 3, pp. 354–381, Mar. 1996.