A Project report on

STRESS DETECTION IN STUDENTS USING MACHINE LEARNING

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING (DATA SCIENCE)

By

C. SAI PRIYA (214G1A3291)

B. PURUSHOTHAM (214G1A3281)

C. NEHA (214G1A3268)

K.R. THARUN (214G1A32B3)

Under the Guidance of

Dr. P. CHITRALINGAPPA,M.Tech.,Ph.D.,
Associate Professor & Head

Department of Computer Science & Engineering (Data Science)

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY
(AUTONOMOUS)
Rotarypuram Village, B K Samudram Mandal, Ananthapuramu – 515701

2024-2025

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(AUTONOMOUS)

(Affiliated to JNTUA, Accredited by NAAC with 'A' Grade, Approved by AICTE, New Delhi & Accredited by NBA (EEE, ECE & CSE)

Rotarypuram Village, BK Samudram Mandal, Ananthapuramu-515701

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(DATA SCIENCE)

Certificate

This is to certify that the project report entitled STRESS DETECTION IN STUDENTS USING MACHINE LEARNING is the bonafide work carried out by **C. Sai Priya, B.Purushotham, C.Neha, K.R.Tharun** bearing Roll Number **214G1A3291, 214G1A3281, 214G1A3268, 214G1A32B3** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering (Data Science)** during the academic year 2024-2025.

Project Guide

Dr. P. Chitralingappa,M.Tech.,Ph.D.,
Associate Professor & Head

Head of the Department

Dr. P. Chitralingappa,M.Tech.,Ph.D.,
Associate Professor

Date:

External Examiner

Place: Rotarypuram

DECLARATION CERTIFICATE

We students of Computer Science & Engineering (Data Science), **SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY(AUTONOMOUS)**, Rotarypuram, hereby declare that the dissertation entitled **“STRESS DETECTION IN STUDENTS USING MACHINE LEARNING”** embodies the report of our project work carried out by us during IV year under the guidance of Dr. P. Chitralingappa, M.Tech.,Ph.D., Associate Professor & Head of Department of Computer Science & Engineering (Data Science), Srinivasa Ramanujan Institute of Technology, and this work has been submitted for the partial fulfillment of the requirements for the award of degree of Bachelor of Technology.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

Date:

Place:

S.No.	Name of the Student	Roll Number	Signature
1	C. Sai Priya	214G1A3291	
2	B. Purushotham	214G1A3281	
3	C. Neha	214G1A3268	
4	K.R. Tharun	214G1A32B3	

Vision & Mission of the SRIT

Vision:

To become a premier Educational Institution in India offering the best teaching and learning environment for our students that will enable them to become complete individuals with professional competency, human touch, ethical values, service motto, and a strong sense of responsibility towards environment and society at large.

Mission:

- Continually enhance the quality of physical infrastructure and human resources to evolve in to a center of excellence in engineering education.
- Provide comprehensive learning experiences that are conducive for the students to acquire professional competences, ethical values, life-long learning abilities and understanding of the technology, environment and society.
- Strengthen industry institute interactions to enable the students work on realistic problems and acquire the ability to face the ever changing requirements of the industry.
- Continually enhance the quality of the relationship between students and faculty which is a key to the development of an exciting and rewarding learning environment in the college.

Vision & Mission of the Department of CSE (Data Science)

Vision:

To evolve as a leading department by offering best comprehensive teaching and learning practices for students to be self-competent technocrats with professional ethics and social responsibilities.

Mission:

DM 1: Continuous enhancement of the teaching-learning practices to gain profound knowledge in theoretical & practical aspects of computer science applications.

DM 2: Administer training on emerging technologies and motivate the students to inculcate self-learning abilities, ethical values and social consciousness to become competent professionals.

DM 3: Perpetual elevation of Industry-Institute interactions to facilitate the students to work on real-time problems to serve the needs of the society.

Program Educational Objectives (PEOs)

An SRIT graduate in Computer Science & Engineering (Data Science), after three to four years of graduation will:

PEO 1: Lead a successful professional career in IT / ITES industry / Government organizations with ethical values.

PEO 2: Become competent and responsible computer science professional with good communication skills and leadership qualities to respond and contribute significantly for the benefit of society at large.

PEO 3: Engage in life-long learning, acquiring new and relevant professional competencies / higher academic qualifications.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express my gratitude for all of them.

It is with immense pleasure that we would like to express my indebted gratitude to my Guide **Dr. P. Chitralingappa,M.Tech.,Ph.D., Associate Professor & Head of the Department, Computer Science & Engineering (Data Science)**, who has guided me a lot and encouraged me in every step of the project work. We thank him for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We are very much thankful to **Dr. P. Chitralingappa,M.Tech.,Ph.D., Associate Professor & Head of the Department, Computer Science & Engineering (Data Science),** for his kind support and for providing necessary facilities to carry out the work.

We wish to convey my special thanks to **Dr. G. Bala Krishna,M.Tech.,Ph.D., Principal** of **Srinivasa Ramanujan Institute of Technology** for giving the required information in doing my project work. Not to forget, we thank all other faculty and non-teaching staff, and my friends who had directly or indirectly helped and supported me in completing my project in time.

We also express our sincere thanks to the Management for providing excellent facilities**.**

Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

Project Associates
214G1A3291
214G1A3281
214G1A3268
214G1A32B3

ABSTRACT

This project focuses on developing an intelligent system to classify emotion-based stress levels and detect stress using both image data and physiological features. The system utilizes an image dataset to analyze facial expressions for emotion-based stress classification, leveraging a Convolutional Neural Network (CNN) to detect emotions such as happiness, sadness, anger, and surprise. Additionally, the system detects stress levels through a CSV file dataset containing physiological features, including body temperature, hours of sleep, heart rate, snoring range, and respiration rate. For live stress classification, the system employs OpenCV to capture real-time video from the user's webcam, process facial expressions, and calculate a stress percentage. The backend is powered by Flask, integrating the machine learning models with an intuitive frontend for seamless real-time interaction. This approach combines visual cues and physiological data to accurately assess and monitor stress levels, offering applications in health monitoring, personal well-being, and stress management.

Keywords: Emotion recognition, Convolutional Neural Network (CNN), DeepFace, Flask, facial features, machine learning, computer vision.

Contents

List of Figures			ix
List of Tables			x
List of Abbreviations			xi
Chapter 1:	**Introduction**		1-3
	1.1	Motivation	1
	1.2	Problem Statement	1
	1.3	Objectives of the Project	1
	1.4	Scope of the Project	2
	1.5	Project Introduction	2
Chapter 2:	**Literature Survey**		4-6
	2.1	Related Work	4
Chapter 3:	**Planning**		7-9
	3.1	Existing System	7
	3.2	Proposed system	7
	3.3	Project flow	8
Chapter 4:	**Methodology**		10-11
Chapter 5:	**Requirement Analysis**		12-17
	5.1	Function and non-functional requirements	12
	5.2	Hardware Requirements	13
	5.3	Software Requirements	13
	5.4	Installation Of Python	14
	5.5	Installation Of Visual Studio	16
Chapter 6:	**System Design**		18-27
	6.1	Introduction of Input design	18
	6.2	UML Diagrams	19
	6.3	Data Flow diagrams	26

Chapter 7:	**Implementation**	28-29
	7.1 Modules	28
	7.1.1 User Module	28
	7.1.2 System Module	28
Chapter 8:	**System Study and Testing**	30-33
	8.1 Feasibility study	30
	8.2 Types of test & Test Cases	30
	8.2.1 Unit testing	31
	8.2.2 Integration testing	31
	8.2.3 Functional testing	31
	8.2.4 White Box Testing	32
	8.2.5 Black Box Testing	32
	8.2.6 Test Cases	33
Chapter 9:	**Results**	34-36
Conclusion		37
Future Enhancement		38
References		39-40

List of Figures

Fig. No	Description	Page No
3.1	Stress Detection System Workflow	09
5.1	Python download website	14
5.2	Python executable installer	14
5.3	Python path addition	15
5.4	Additional Features	15
5.5	Setup Successful	16
5.6	Download Visual Studio	16
5.7	Installing Visual Studio Code	17
5.8	Installation Completed	17
6.1	Use Case Diagram	21
6.2	Class Diagram	21
6.3	Sequence Diagram	22
6.4	Collaboration Diagram	23
6.5	Deployment Diagram	23
6.6	Activity Diagram	24
6.7	Component Diagram	25
6.8	ER Diagram	26
6.9	Data Flow Diagram Level 0	27
6.10	Data Flow Diagram Level 1	27
9.1	Home Page	34
9.2	Text Classification Level 0	35
9.3	Text Classification Level 1	35
9.4	Image Classification	36
9.5	Live Classification	36

List of Tables

Table No	Table Name	Page No
8.1	Parameters Test	33

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
CSV	Comma Separated Values
CNN	Convolutional Neural Network
DBMS	Database Management System
DFD	Data Flow Diagram
ECU	Electronic Control Unit
ER	Entity Relationship
GAN	Generative Adversarial Network
GANs	Generative Adversarial Networks
IDE	Integrated Development Environment
IOT	Internet Of Things
KNN	K-Nearest Neighbors
MICR	Magnetic Inc Character Recognition
OMR	Optical Mark Recognition
OPEN CV	Open Source Computer Vision Library
PSS	Perceived Stress Scale
RNN	Recurrent Neural Network
SVGA	Super Video Graphics Array
UI	User Interface
UML	Unified Modeling Language

CHAPTER - 1

INTRODUCTION

1.1 Motivation

The motivation behind this project arises from the increasing demand for effective and real-time stress monitoring solutions. Stress, if left unmanaged, can lead to various health problems, including anxiety, depression, and cardiovascular diseases. Current methods of stress detection often lack the ability to combine emotional and physiological data efficiently. This project aims to bridge this gap by developing a comprehensive system that combines real-time facial expression analysis with vital signs, enabling more accurate and timely detection of stress. Additionally, this system will empower individuals to monitor their stress levels and take proactive steps toward managing their mental and physical health.

1.2 Problem Statement

Despite the growing understanding of stress and its detrimental effects, existing systems for real-time stress detection often fail to incorporate both emotional and physiological data. While some methods focus on facial expressions or physiological parameters in isolation, they do not provide a comprehensive approach to stress detection. This project addresses the need for an integrated system that can assess stress levels based on both emotional cues from facial images and physiological features such as heart rate, body temperature, and sleep patterns. By combining these two sources of information, the system aims to provide an accurate and real-time assessment of an individual's stress level, enhancing the ability to monitor and manage stress effectively.

1.3 Objective of the project

The primary objective of this project is to build an intelligent system that can effectively classify stress levels by analyzing both emotional cues from facial expressions and physiological data. The system will detect emotions related to stress using image data and classify stress levels through physiological parameters stored in a CSV file. Additionally, the system aims to implement real-time stress detection using

OpenCV, capturing live video from a user's camera, and calculating the stress percentage based on visual and physiological cues.

1.4 Scope of the project

The scope of this project covers the development of a real-time stress detection system that integrates both facial expression analysis and physiological monitoring. The system will process facial images using Convolutional Neural Networks (CNNs) to detect emotions linked to stress, such as anger, sadness, and anxiety. Physiological data will be incorporated from a CSV dataset, including features like body temperature, sleep patterns, heart rate, snoring range, and respiration rate, to assess stress levels. Using OpenCV, the system will capture live video, providing real-time stress analysis and outputting a stress percentage. The solution will be implemented using Flask for easy integration between the backend machine learning models and the frontend interface.

1.5 Project Introduction

Stress is a natural and essential part of human experience, typically arising as a response to challenging situations. However, prolonged or excessive stress can have serious implications for both mental and physical health, contributing to conditions such as anxiety, depression, cardiovascular diseases, and impaired immune function. As a result, accurate and timely stress detection is vital for preventing these adverse outcomes and promoting well-being. Traditional methods for assessing stress, such as self-reports, surveys, or medical diagnostics, may not always provide real-time feedback or capture subtle changes in stress levels.

Advancements in technology, particularly in artificial intelligence and machine learning, have led to innovative approaches for stress detection, including emotion recognition from facial expressions and monitoring physiological signals. Facial expressions, which are often considered a direct reflection of emotional states, provide valuable insight into an individual's stress levels. At the same time, physiological features such as heart rate, respiration rate, body temperature, and sleep patterns offer important data that reflects the body’s response to stress. By combining these two sources of information—emotional cues from facial images and physiological signals it becomes possible to build a more accurate and dynamic system for stress detection.

This project aims to develop an intelligent system that classifies emotion-based stress and detects stress levels by integrating both visual and physiological data. The system utilizes an image dataset to detect emotions, such as happiness, sadness, and anger, which are often linked to different stress levels. In addition, a CSV file dataset containing key physiological features, such as body temperature, hours of sleep, heart rate, snoring range, and respiration rate, will be used to assess the user's stress level. For real-time monitoring, OpenCV will be employed to capture live video from a user's webcam, processing the video feed to analyze facial expressions and calculate the stress percentage. This combination of real-time facial analysis and physiological data offers a comprehensive, reliable, and user-friendly approach to stress monitoring, enabling individuals to better understand and manage their stress in everyday life.

CHAPTER - 2

LITERATURE SURVEY

2.1 Related Work

[1] Khan, S., & Shah, M. (2018). "Deep Learning for Real-Time Facial Emotion Recognition: A Review and Future Directions."

This paper reviews the use of deep learning techniques for real-time facial emotion recognition, emphasizing the importance of CNNs in extracting meaningful features from facial images. The authors explore the evolution of facial emotion recognition systems, from traditional methods like Haar Cascades to more advanced techniques like CNNs. They highlight how CNN-based models, specifically designed for facial expression classification, have achieved high accuracy rates, even under challenging conditions such as varying lighting and facial orientations. The paper also discusses several challenges faced by real-time emotion recognition systems, including real-world implementation limitations and model generalization across diverse demographics. Finally, the authors propose future directions, suggesting hybrid models combining CNNs with other techniques like Recurrent Neural Networks (RNNs) for better temporal consistency in emotion recognition. This review serves as a comprehensive guide for researchers and practitioners interested in advancing emotion recognition technologies.

[2] Mollah, M. M., & Fattah, S. (2019). "Real-time Age and Gender Detection using Deep Learning."

In this study, the authors present a real-time system for detecting age and gender from facial images using deep learning. They focus on using CNN architectures, which are particularly suited for image classification tasks due to their ability to learn complex patterns and features from raw pixel data. The system utilizes pre-trained models and fine-tunes them for the specific task of age and gender classification. The paper describes the deployment of the model in a web application that can process live webcam feeds, providing instant predictions for the user’s age group and gender. One key challenge addressed is ensuring the system works efficiently in real-time, with low latency. The authors also discuss performance evaluation and comparison with other traditional methods, demonstrating the advantages of deep learning for both accuracy and speed. This system is useful in applications such as human-computer interaction, security, and personalized content delivery.

[3] Singh, P., & Aggarwal, R. (2019). "Facial Emotion Recognition using Convolutional Neural Networks."

Singh and Aggarwal focus on the use of CNNs for facial emotion recognition, a critical aspect of human-computer interaction. The paper highlights how CNN-based models excel at identifying facial expressions such as happiness, sadness, anger, and surprise, by extracting spatial features directly from images. The authors implement a model trained on large facial emotion datasets, ensuring that it generalizes well across various subjects and environmental conditions. Real-time performance is a key objective of the system, enabling it to process live facial images via webcam input. The study also discusses the challenges of emotion recognition, particularly in dealing with subtle facial cues and variations in facial expressions across different cultures. The results demonstrate that CNNs outperform traditional machine learning methods in both accuracy and processing speed. This work contributes to the development of emotion-aware systems, which can be applied in areas like customer service, mental health monitoring, and adaptive interfaces.

[4] Hasan, M., & Zaki, M. (2021). "A Real-Time Face Emotion Recognition System Using Deep Learning."

Hasan and Zaki propose a real-time facial emotion recognition system utilizing deep learning techniques. The system employs a CNN-based architecture designed to classify facial emotions such as joy, sadness, anger, surprise, and neutral expressions. The authors describe how the system processes live video streams from webcams and uses CNNs to extract key facial features for emotion classification. One of the main goals of this system is to operate efficiently in real-time, which is crucial for applications in human-computer interaction and security. The paper also addresses the importance of a large, diverse dataset for training the model, ensuring the system’s robustness across different demographics. In their evaluation, the authors demonstrate that the proposed system can accurately recognize emotions with high precision, even in challenging conditions like poor lighting or occlusions. The work showcases how deep learning can enhance the accuracy and responsiveness of emotion recognition systems, making them viable for real-world deployment.

CHAPTER - 3

PLANNING

3.1 Existing System

Traditional methods of stress detection primarily rely on subjective reports, questionnaires, and manual assessments to gauge an individual’s stress level. These approaches often include surveys such as the Perceived Stress Scale (PSS), interviews, and self-reported questionnaires, which are completed by the individual and assessed by a clinician or therapist. Additionally, medical tests like cortisol level measurements or heart rate variability are sometimes used to assess physiological stress responses. However, these methods come with several key disadvantages:

- Subjectivity: Self-reports and interviews are inherently subjective, as they depend on the individual's self-awareness, honesty, and emotional state at the time of assessment. People may underreport or overreport their stress, leading to inaccurate data.
- Limited Scope: Many traditional methods focus solely on psychological or physiological indicators in isolation. For example, surveys may only assess perceived stress without considering physiological data, such as heart rate or sleep patterns. Similarly, medical tests may not assess emotional aspects of stress, which can play a significant role in a person’s overall well-being.
- Labor-Intensive and Expensive: Methods like medical tests (e.g., cortisol measurement) or frequent therapist visits can be expensive and time-consuming, making them inaccessible for routine monitoring, especially in large populations.

3.2 Proposed system

This overcomes the limitations of traditional methods by integrating both emotion-based stress classification from facial expressions and physiological data to provide a comprehensive, real-time stress detection solution. This system leverages advanced machine learning models, such as Convolutional Neural Networks (CNNs), to analyze facial expressions and detect emotional states, which are strongly linked to stress. At the same time, the system utilizes a CSV file dataset containing important physiological features, such as body temperature, hours of sleep, heart rate, snoring range, and respiration rate, to evaluate stress levels based on physical indicators.

The key advantages of the proposed system are:

- Comprehensive Analysis: By combining both facial expression analysis and physiological data (such as heart rate, sleep patterns, and respiration rate), the proposed system provides a holistic view of a person's stress level. This integration allows for a more accurate and well-rounded stress assessment, taking both emotional and physical factors into account.
- Objective and Data-Driven: The proposed system reduces the subjectivity found in traditional methods by relying on objective data derived from facial expressions and physiological parameters. The use of machine learning algorithms to classify emotions and calculate stress levels ensures more accurate and consistent results.
- User-Friendly and Accessible: The system is designed to be easy to use, employing a webcam to capture facial expressions, and can be accessed via a simple frontend interface, powered by Flask. Users do not need any specialized medical knowledge to interact with the system. Additionally, because it uses standard hardware (a webcam), the system is cost-effective and widely accessible compared to traditional medical tests or therapist visits.
- Continuous Monitoring: The proposed system can track stress over time and offer ongoing insights. Unlike traditional methods that require separate sessions, the system continuously monitors stress levels throughout the day and provides ongoing data to users for better stress management and self-awareness.

3.3 Project flow

The process of Stress Detection begins with loading a relevant dataset containing stress-related data, which may include physiological signals, facial expressions, or voice features. Once the dataset is loaded, data preprocessing is carried out to clean and normalize the data, handle missing values, and extract meaningful features essential for model training. The next step involves training machine learning models using algorithms such as Support Vector Machines (SVM), Random Forests, or

Deep Learning networks to identify patterns associated with stress. After training, the models are evaluated by analyzing their performance based on metrics like accuracy, precision, recall, and F1-score. Once a reliable model is selected, it is deployed for classification on a live webcam, enabling real-time detection of stress through continuous input data. The workflow concludes after achieving satisfactory performance and deployment, marking the end of the stress detection pipeline.

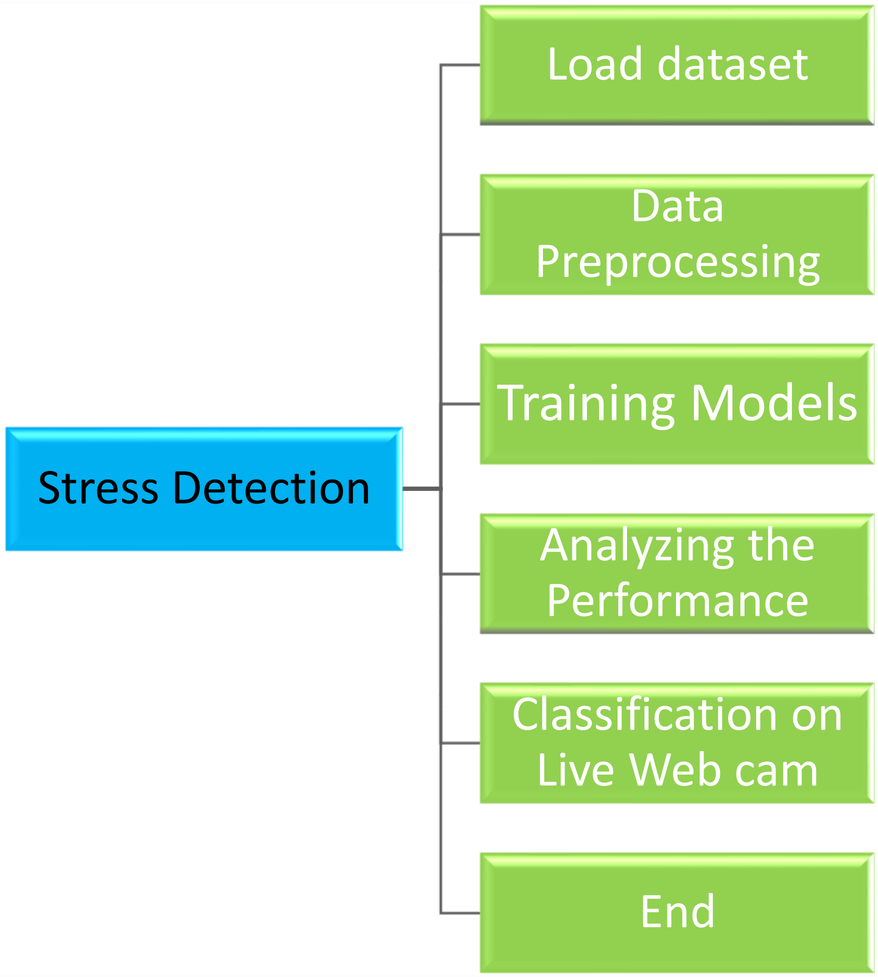


Fig. No. 3.1: Stress Detection System Workflow

CHAPTER - 4

METHODOLOGY

- The methodology of the proposed system integrates multiple data sources and machine learning techniques to detect and classify stress levels in real-time. The system leverages both physiological data and facial expression analysis to provide a comprehensive assessment of stress. For emotion-based stress classification, facial expressions are captured through live webcam feed and processed using computer vision techniques. Face detection is performed using OpenCV, and facial features are extracted for analysis.
- Facial expressions are evaluated using pre-trained deep learning models, which classify emotions such as anger, fear, and sadness, each of which correlates with higher stress levels. The detected emotions are mapped to stress categories, with emotions like "Angry Based Stress" and "Sad Based Stress" indicating high stress, while "Happy - No Stress" and "Neutral - No Stress" represent low stress states.
- The system also incorporates physiological data, such as body temperature, heart rate, and respiration rate, to predict stress levels. These features are processed using machine learning algorithms like K-Nearest Neighbors (KNN) to classify stress levels into multiple categories based on historical data. The system combines these two distinct data streams emotion-based analysis and physiological monitoring—to create a holistic stress assessment.
- For real-time classification, the system continuously captures video frames from the user’s webcam and processes them in sequence. Once a face is detected, the system analyzes the facial expressions to predict stress in real-time. The system also provides stress percentage output, showing the severity of the stress based on visual and physiological data.
- By using OpenCV and DeepFace for emotion recognition, the system can make accurate stress predictions during live interactions. It is designed to provide immediate feedback, allowing users to monitor their stress levels continuously throughout the day. After processing the webcam feed, the results are displayed on a simple user interface for easy understanding.

- The system is designed to be interactive and responsive, offering users a comprehensive stress evaluation. The real-time aspect of the system makes it an effective tool for stress management and well-being monitoring. After completing each task, the system can be reset or adapted for further use, ensuring a flexible and efficient stress classification process.
- The proposed system represents a powerful integration of computer vision, machine learning, and real-time data analysis, providing users with an innovative way to track and manage stress.

CHAPTER - 5

REQUIREMENT ANALYSIS

5.1 Function and non-functional requirements

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

Functional Requirements: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

1) Authentication of user whenever he/she logs into the system
2) System shutdown in Solar prediction.
3) A verification email is sent to user whenever he/she register for the first time on some software system.

Non-functional requirements: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability

- Scalability
- Performance
- Reusability
- Flexibility

Examples of non-functional requirements:

1) Emails should be sent with a latency of no greater than 12 hours from such an activity.
2) The processing of each request should be done within 10 seconds
3) The site should load in 3 seconds whenever of simultaneous users are > 10000

5.2 Hardware Requirements

Processor	- I3/Intel Processor
Hard Disk	- 160GB
Key Board	- Standard Windows Keyboard
Mouse	- Two or Three Button Mouse
Monitor	- SVGA
RAM	- 8GB

5.3 Software Requirements

Operating System	: Windows 7/8/10
Programming Language	: Python
Libraries	: Pandas, Numpy, scikit-learn.
IDE/Workbench	: Visual Studio Code.

5.4 Installation of Python

Python has become one of the most popular programming languages of the 21st century. It is being used for multiple purposes in various sectors of business. Developers use Python for building applications and developing websites. Data Engineers use python for performing data analysis, statistical analysis, and building machine learning models. However, you can check if it exists on the system by running one line of command on the command prompt: python--version.

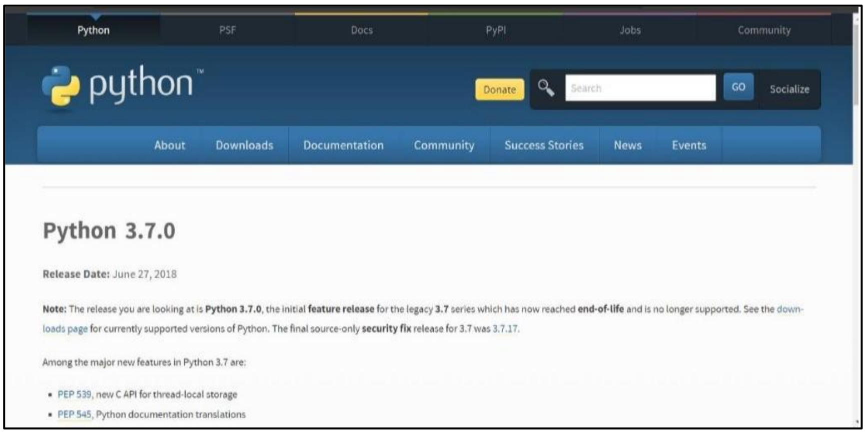
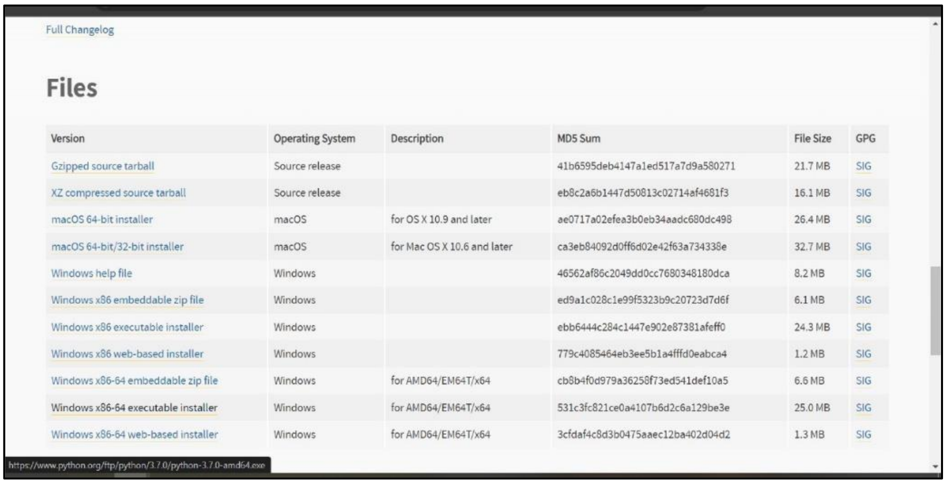


Fig. No. 5.1: Python download website

To download python, open the chrome and browse by typing python download. Download “python 3.7.0” version from the internet because it is compatable with our project.


Full Changelog

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		41b6595deb4147a1ed517a7d9a580271	21.7 MB	SIG
XZ compressed source tarball	Source release		eb8c2a6b1447d50813c02714af4681f3	16.1 MB	SIG
macOS 64-bit installer	macOS	for OS X 10.9 and later	ae0717a02efea3b0eb34aadc680dc498	26.4 MB	SIG
macOS 64-bit/32-bit installer	macOS	for Mac OS X 10.6 and later	ca3eb84092d0ff6d02e42f63a734338e	32.7 MB	SIG
Windows help file	Windows		46562af86c2049dd0cc7680348180dca	8.2 MB	SIG
Windows x86 embeddable zip file	Windows		ed9a1c028c1e99f5323b9c20723d7d6f	6.1 MB	SIG
Windows x86 executable installer	Windows		ebb6444c284c1447e902e87381afeff0	24.3 MB	SIG
Windows x86 web-based installer	Windows		779c4085464eb3ee5b1a4fffd0eabca4	1.2 MB	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	cb8b4f0d979a36258f73ed541def10a5	6.6 MB	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	531c3fc821ce0a4107b6d2c6a129be3e	25.0 MB	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	3cfdaf4c8d3b0475aaec12ba402d04d2	1.3 MB	SIG

https://www.python.org/ftp/python/3.7.0/python-3.7.0-amd64.exe

Fig. No. 5.2: Python executable installer

Select the "windows x36-64 executable installer".

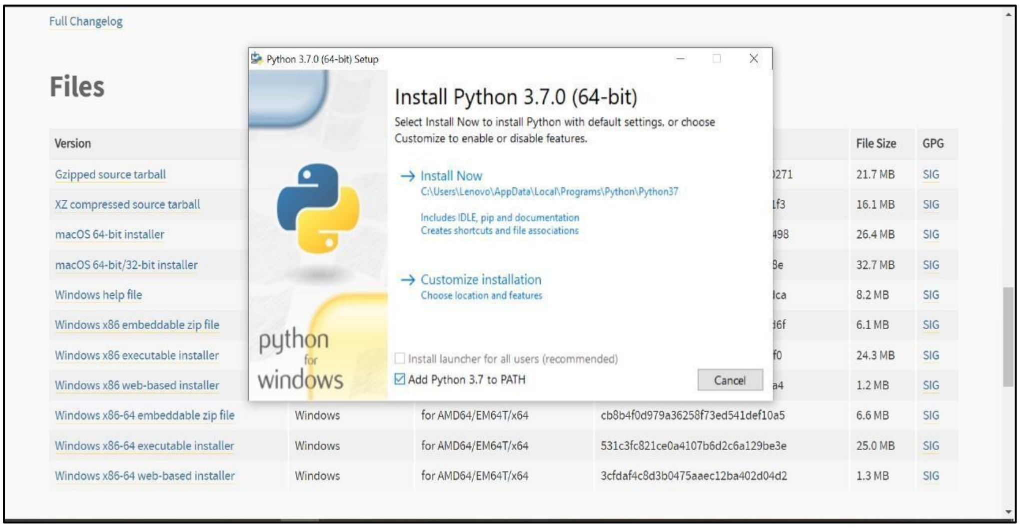


Fig. No. 5.3: Python path addition

Upon selecting the suitable version, a dialogue box is popped showing the options "install now" and "customize Installation". Select the "Customize Installation" and check box the "Add python 3.7 to PATH".

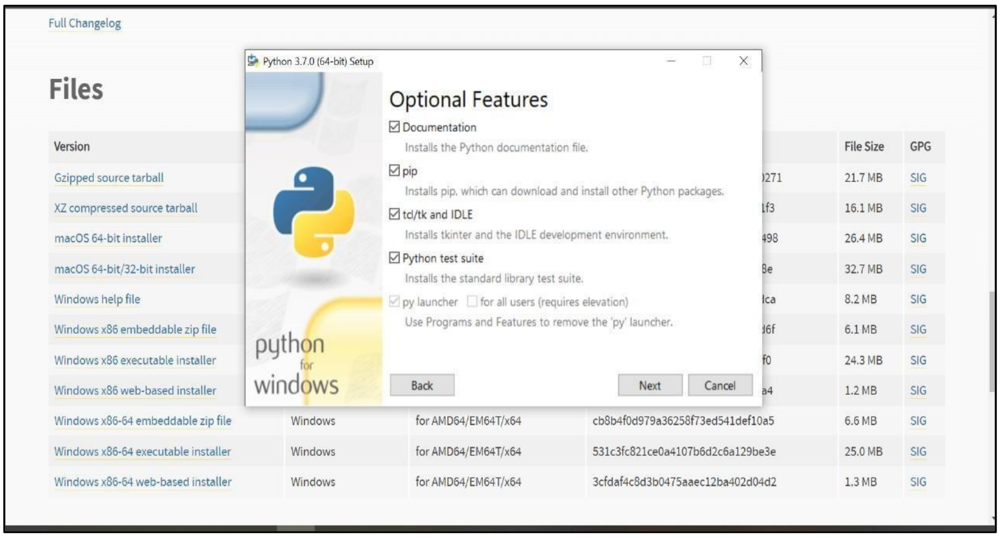


Fig. No. 5.4: Additional Features

Adding all the additional features that are required for the execution of the code. Click on "Next" button after selecting the features. Another dialogue box names Advanced Options will be displayed. Check marks the box "Install for all Users". We can also change the location where the python software to be installed.

It is preferred to choose the default location option for installation. Click on "Install" button for successful installation of Python. Python software is installing all the executable files and libraries.

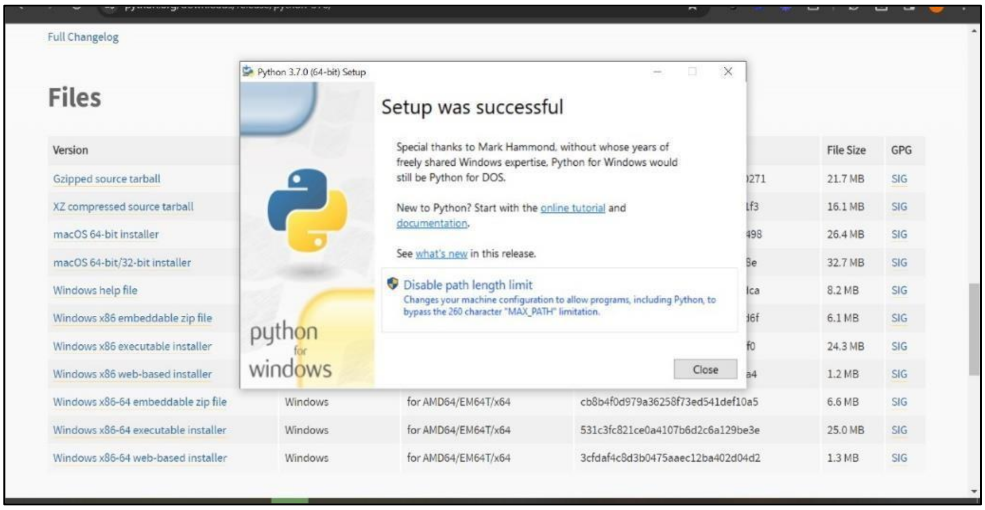


Fig. No. 5.5: Setup Successful

5.5 Installation of Visual Studio

Visual Studio is an integrated development environment (IDE) developed by Microsoft. It provides comprehensive tools and features for software development across various platforms, including web, mobile, desktop, cloud, and more. Installing Visual Studio is essential for developers looking to leverage its powerful capabilities for writing, debugging, and deploying applications efficiently.

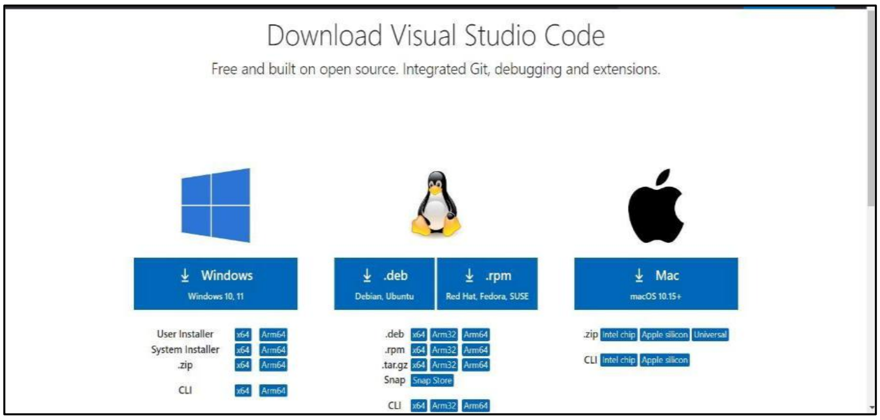


Fig. No. 5.6: Download Visual Studio

As Visual Studio is available for many Operating Systems, we have to select the type of the operating system that is supported by the Computer/Laptop. For setting up the Visual Studio Code, select the location to where it has to be installed. We can change the location by clicking on “Browse” button. Click on the “Next” button after selecting the location. Selecting all the additional features that are required for the successful installation of the visual studio code and click on “Next”. A Dialogue is shown displaying the Destination Location, Menu Folder and Additional Tools for the confirmation. Proceed by clicking on the “Install” button to install Visual Studio Code.

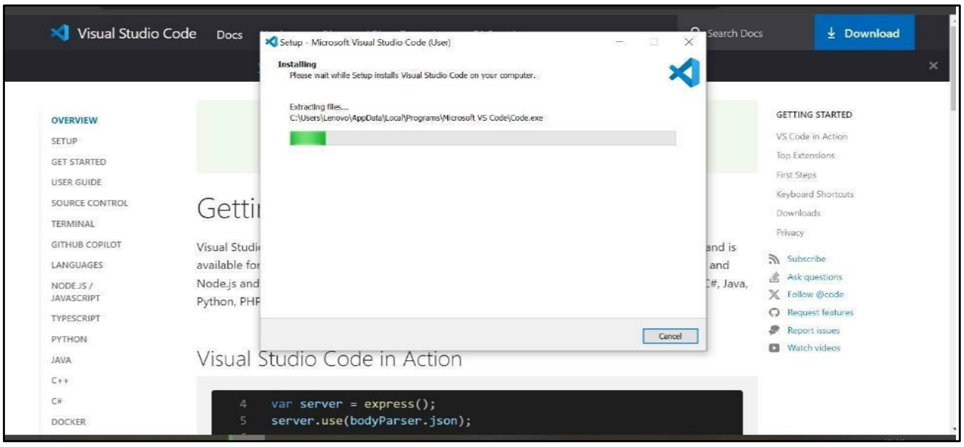


Fig. No. 5.7: Installing Visual Studio Code

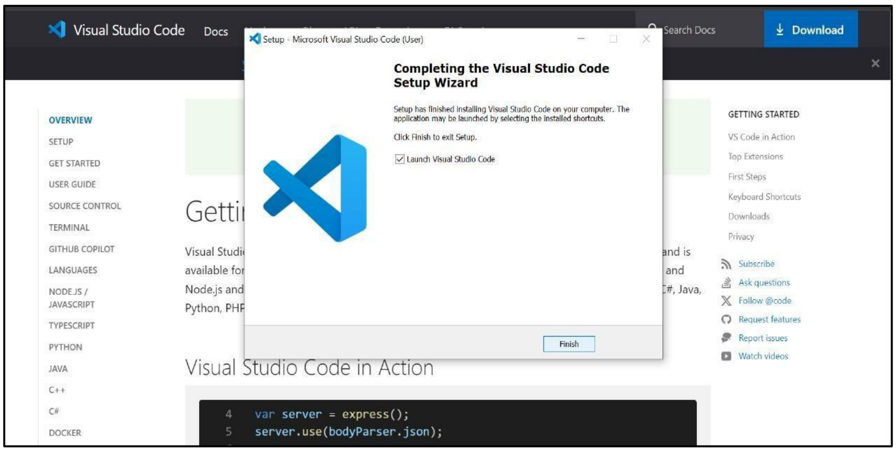


Fig. No. 5.8: Installation Completed

Visual Studio Code is installed.

CHAPTER - 6

SYSTEM DESIGN

6.1 Introduction of Input design

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well-designed input forms and screens have following properties −

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- It ensures proper completion with accuracy.
- It should be easy to fill and straightforward.
- It should focus on user’s attention, consistency, and simplicity.
- All these objectives are obtained using the knowledge of basic design principles regarding

What are the inputs needed for the system?

How end users respond to different elements of forms and screens.

Objectives for Input Design:

The objectives of input design are −

- To design data entry and input procedures
- To reduce input volume
- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.
- To use validation checks and develop effective input controls.

Output Design:

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design:

The objectives of input design are:

- To develop output design that serves the intended purpose and eliminates the production of unwanted output.
- To develop the output design that meets the end user's requirements.
- To deliver the appropriate quantity of output.
- To form the output in appropriate format and direct it to the right person.
- To make the output available on time for making good decisions.

6.2 UML diagrams

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling 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 Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artefacts of software system, as well as for business modelling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modelling 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. UML includes a rich set of diagram types, typically categorized into structure diagrams (like Class Diagrams, Object Diagrams, and Component Diagrams) and behavior diagrams (such as Use Case Diagrams, Sequence Diagrams, and Activity Diagrams).

GOALS:

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

1. Provide users a ready-to-use, expressive visual modelling 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 modelling 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.

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.

Use case diagrams are especially useful during the early stages of system design to capture functional requirements and to provide a high-level view of the system's intended behavior. They often serve as a communication tool between stakeholders such as clients, developers, and testers, ensuring that all parties share a common understanding of how the system is expected to function.

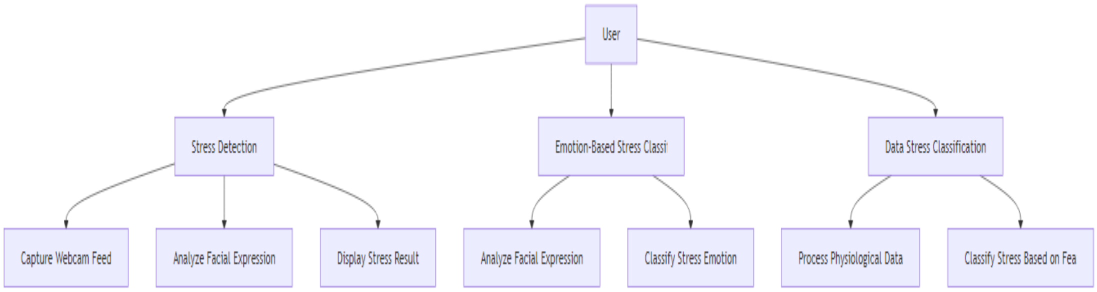


Fig. No. 6.1: Use Case Diagram

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

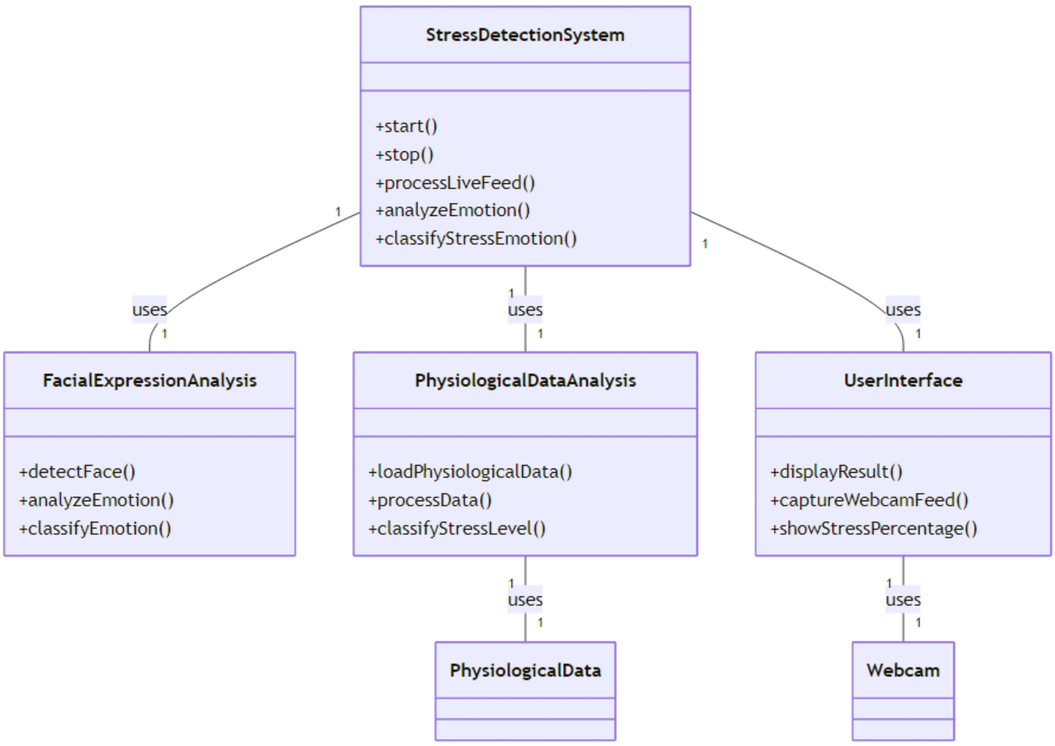


Fig. No. 6.2: Class Diagram

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

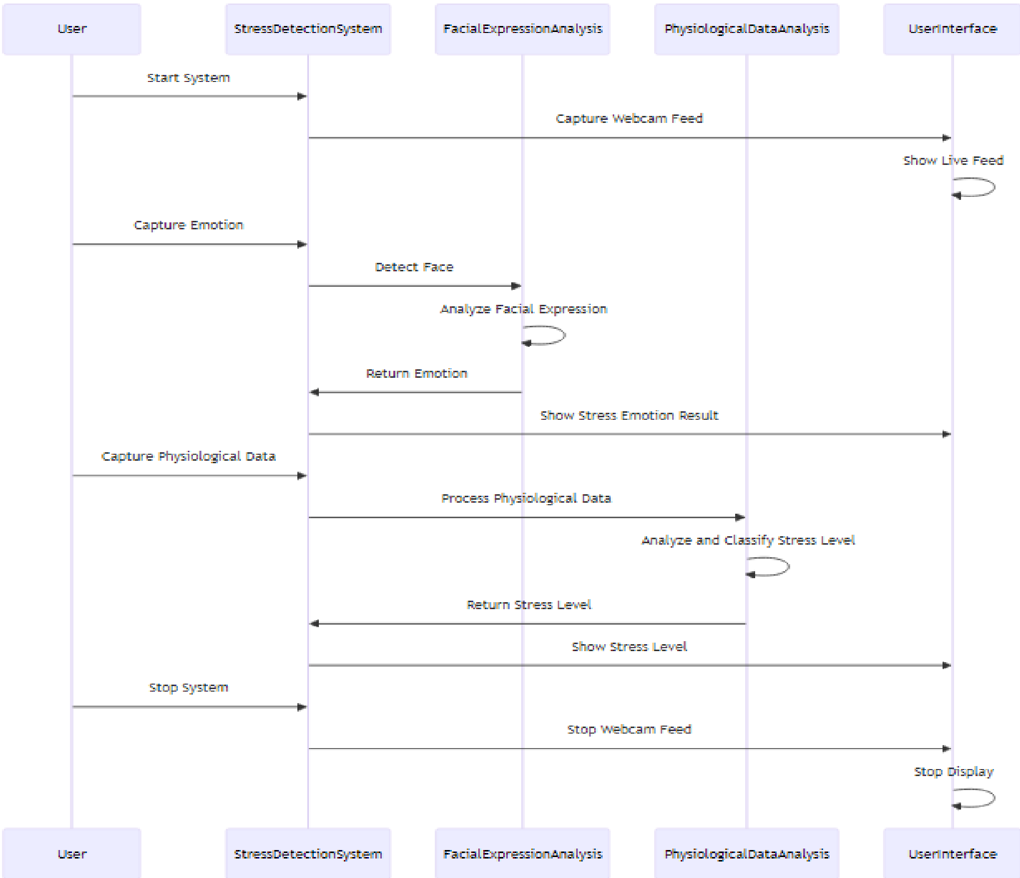


Fig. No. 6.3: Sequence Diagram

COLLABORATION DIAGRAM:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.

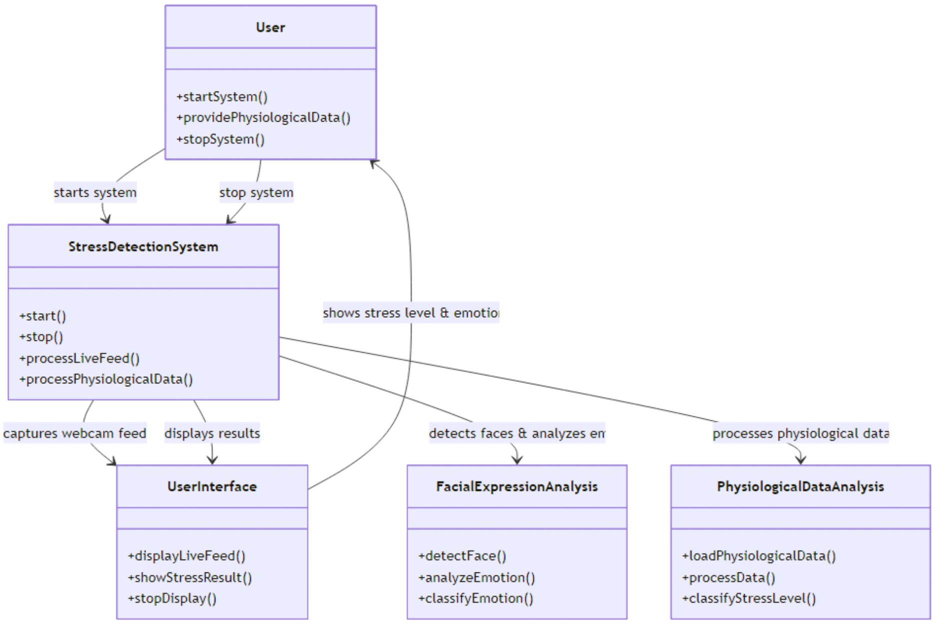


Fig. No. 6.4: Collaboration Diagram

DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.

Fig. No. 6.5: Deployment Diagram

ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

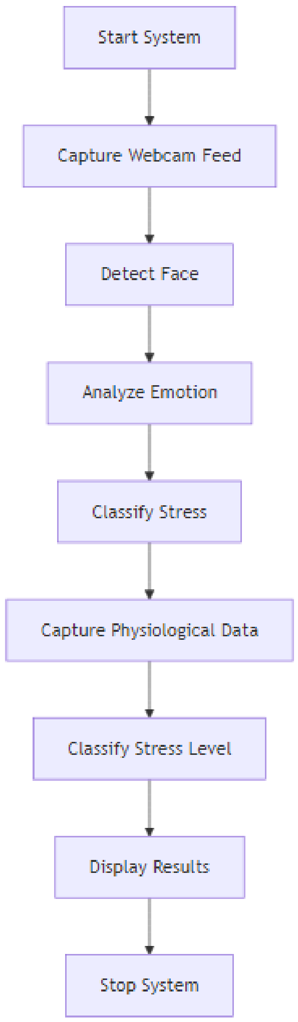


Fig. No. 6.6: Activity Diagram

COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.

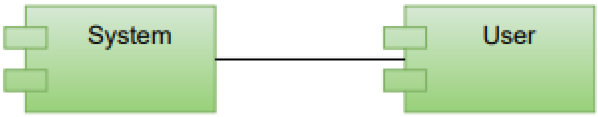


Fig. No. 6.7: Component Diagram

ER DIAGRAM:

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let’s have a look at a simple ER diagram to understand this concept.

ER models are essential during the database design phase, ensuring that data requirements are clearly understood and logically represented, which in turn helps to avoid redundancy, maintain integrity, and facilitate easier database maintenance and expansion.

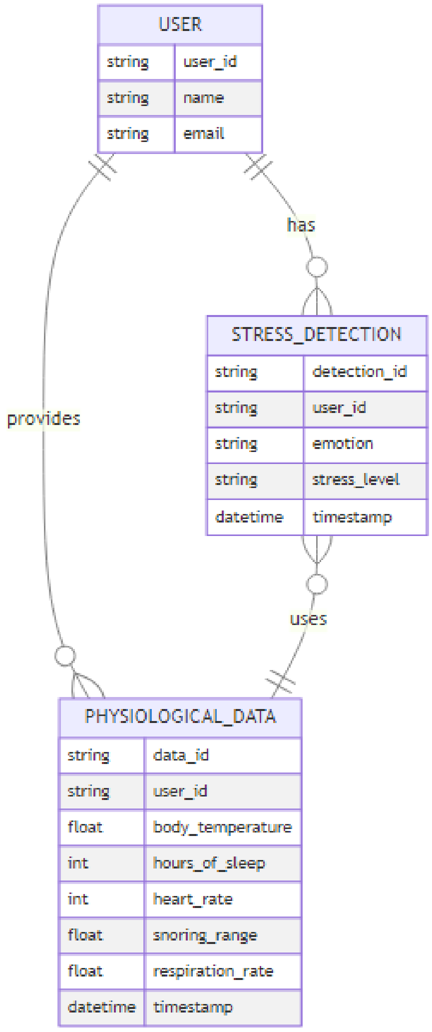


Fig. No. 6.8: ER Diagram

6.3 Data Flow diagrams

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

Contrast Level:

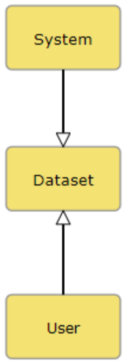


Fig. No. 6.9: Data Flow Diagram Level 0

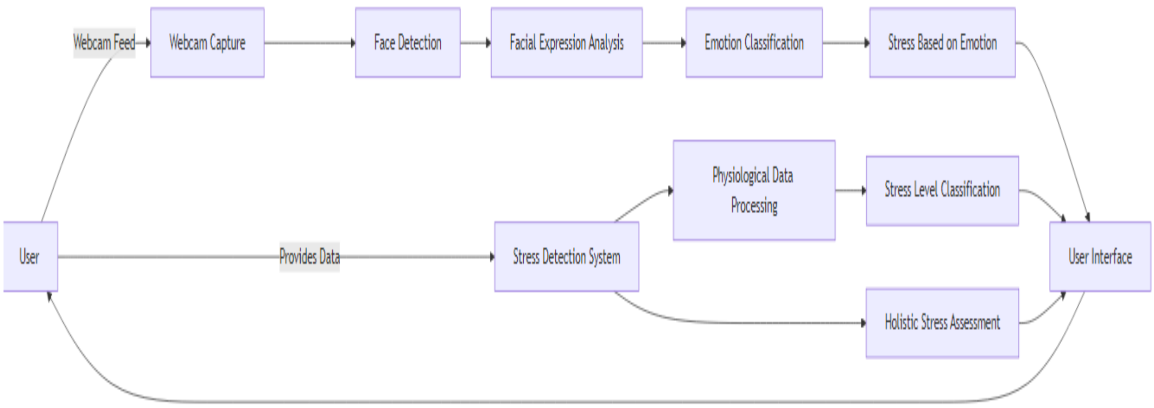


Fig. No. 6.10: Data Flow Diagram Level1

CHAPTER - 7

IMPLEMENTATION

7.1 Modules

7.1.1 User Module:

Index Page

The user starts at the index page, which provides an overview of the application. It includes links to the "About" page, explaining the system's features and functionality.

About Page

When the user clicks on the "About" link, they are directed to a page that describes the project, its objectives, and how the system works, providing users with an understanding of the technology behind the application.

Prediction Feature

On the index page, the user can click a "Predict" button to activate the webcam. Once clicked, the system opens the webcam in real time, detects the user's face, and predicts their age, gender, and emotion. The predictions are displayed on the same page for immediate feedback.

7.1.2 System Module

Face Detection and Analysis:

This module handles the core functionality of detecting faces from live webcam feeds. It utilizes Convolutional Neural Networks (CNN) to classify gender and emotions, and DeepFace for age prediction. The system processes the incoming video stream, detects faces, and performs attribute classification in real time.

Model Integration:

This module integrates the emotion detection (CNN), and (DeepFace) models. It manages the flow of data between the models and ensures accurate predictions based on the live feed from the webcam.

Flask Web Application:

The system is hosted on a Flask-based web application. This module ensures that the backend handles requests like opening the webcam, processing the images, and serving the predictions to the frontend. It provides routes for different pages, like the homepage and prediction page. It also defines various routes for different sections of the system, including the homepage, image and text classification pages, and the live prediction feature. This modular structure allows for efficient request handling, easy navigation, and scalability, making the application lightweight and user-friendly.

CHAPTER - 8

SYSTEM STUDY AND TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.1 Feasibility study

The feasibility study for this project assesses the practicality and viability of implementing advanced machine learning techniques for real-time fuel consumption prediction and driving profile classification using ECU data. Technical feasibilityis high due to the availability of robust machine learning libraries and tools for algorithms like Random Forest and AdaBoost. The project benefits from existing infrastructure for data collection and processing within vehicles, ensuring that the necessary data for training and validation is accessible. Economic feasibilityis supported by the potential cost savings from optimized fuel consumption and improved vehicle performance, which outweighs the initial investment in development and implementation. Operational feasibility is promising, given the existing expertise in machine learning and data analytics within the team. Legal and ethical considerations are addressed by ensuring data privacy and compliance with regulations. Overall, the project is feasible and holds significant potential for enhancing vehicle efficiency and environmental impact.

8.2 Types of test & Test Cases

8.2.1 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.

8.2.2 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.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

8.2.3 Functional testing

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.

8.2.4 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.

8.2.5 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. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

8.2.6 Test cases

S.NO	Test cases	I/O	Expected O/T	Actual O/T	P/F
1	Read the dataset.	Dataset path.	Dataset need to read successfully.	Dataset fetched successfully.	P
2	Performing Loading on the dataset	Data loading takes place	Data loading should be performed on system	Data loading successfully completed.	P
3	Performing data preprocessing	Image dataset is provided to process the data	Processed data will be the output emotion gender	Processed data will successfully completed	P
4	Model Building	Model Building for the clean data	Need to create model using required algorithms	Model Created Successfully.	P
5	Prediction	Start a live webcam	Based on the input frames Age, emotion and gender	Predicted successfully	P

Table 8.1: Parameters Test

CHAPTER - 9

RESULTS

Home Page:

The Home page provides an overview of the project, highlighting the importance of stress detection through facial expressions. It introduces the problem statement and sets the context for the system’s purpose and applications.

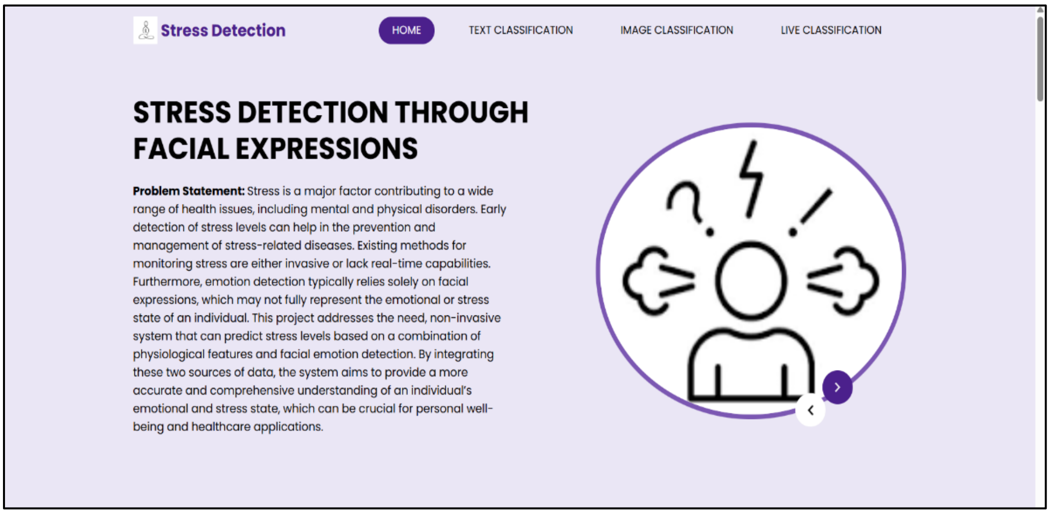


Fig. No. 9.1: Home Page

Text Classification:

In this Text classification system, users enter input values such as body temperature, hours of sleep, heart rate, snoring range, and respiration rate. After submitting the data, the system provides an output indicating the user's stress level.

The system applies text classification to analyze a CSV file dataset containing important physiological features, such as body temperature, hours of sleep, heart rate, snoring range, and respiration rate.It classifies and predicts stress levels based on these dataset feature

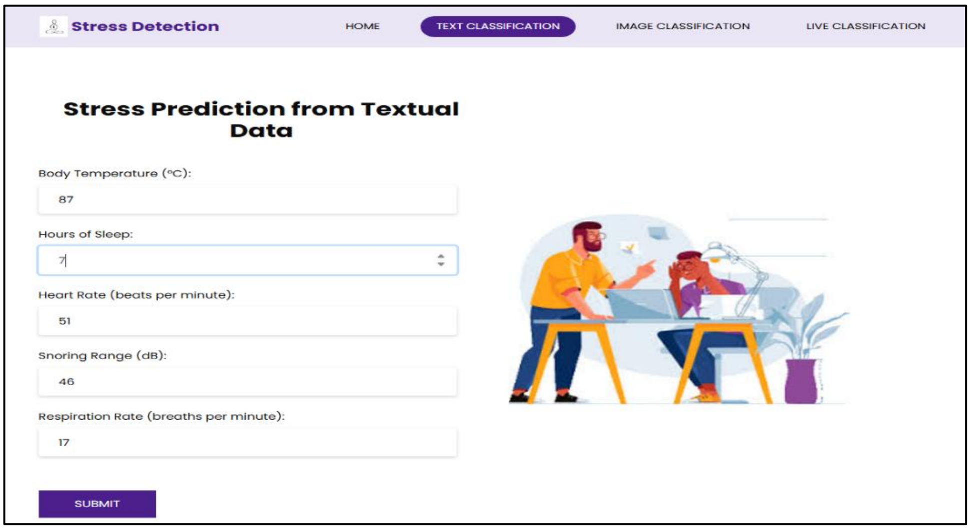


Fig. No. 9.2: Text Classification Level 0

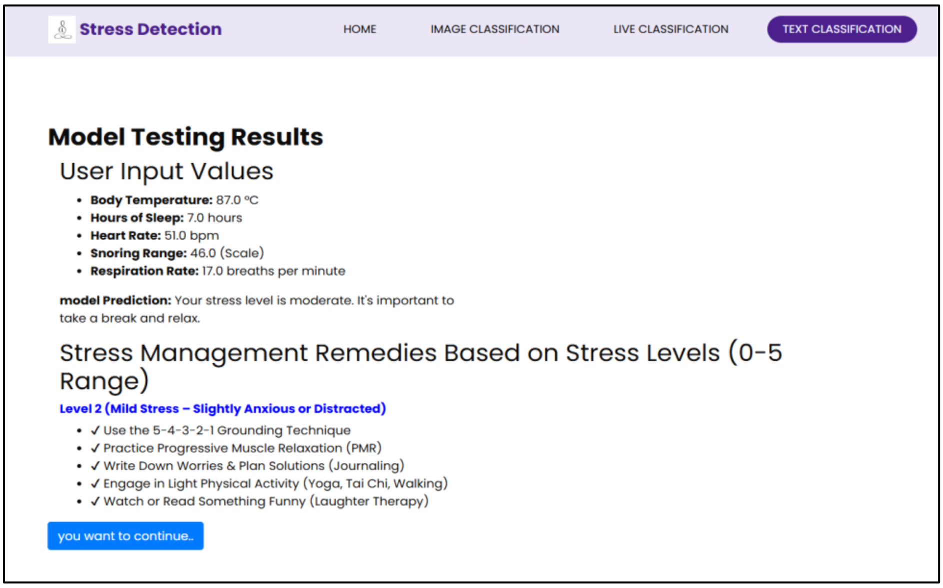


Fig. No. 9.3: Text Classification Level 1

Image Classification:

In this classification method, users upload an image, and the system analyzes it to display an output such as Angry-based Stress, Sad-based Stress, or Fear-based Stress. This prediction is based on facial expressions detected using a deep learning model.

The system uses image classification to detect and classify human emotions such as happy, sad, and disgusted based on facial expressions.

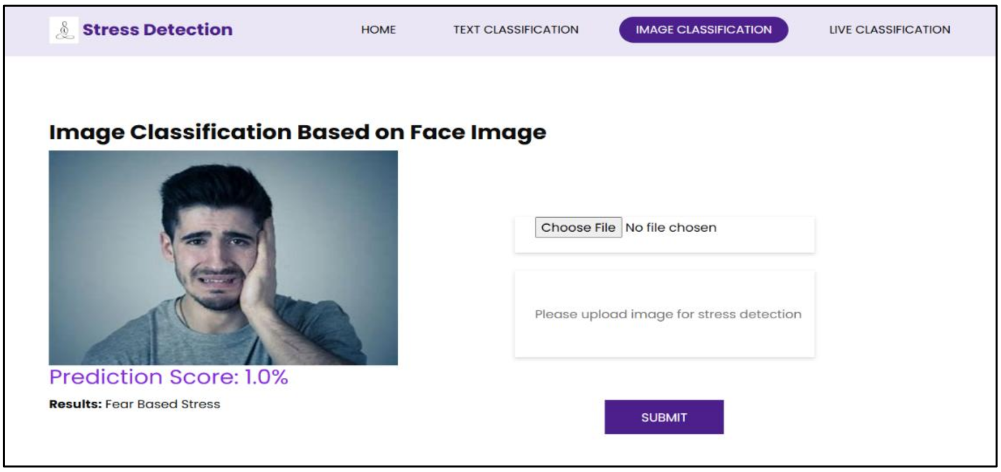


Fig. No. 9.4: Image Classification

Live Classification:

In this classification, we use OpenCV to access the live camera and detect stress levels in real time. If stress is detected, it displays a percentage—typically 90% or higher—and this value changes dynamically based on facial expressions. This live feedback helps users become more aware of their emotional state instantly.

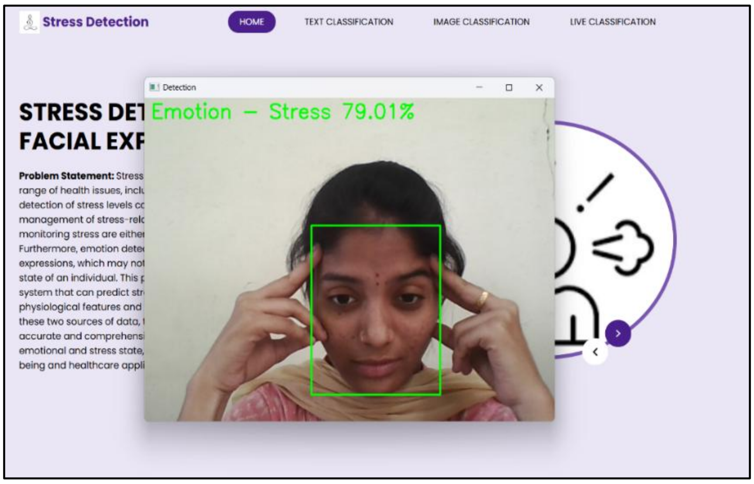


Fig. No. 9.5: Live Classification

CONCLUSION

we developed an intelligent system capable of recognizing emotions and predicting stress levels using both facial and physiological data. The system utilizes advanced machine learning models, with Convolutional Neural Networks (CNN) for image classification and K-Nearest Neighbors (KNN) for stress level prediction. The performance of these models has been evaluated, with both achieving an impressive accuracy of 97%, highlighting the system's effectiveness and robustness. For the stress detection dataset, the KNN model demonstrated high performance, accurately predicting stress levels based on various physiological features, including snoring range, respiration rate, body temperature, limb movement, and heart rate. The ability to process these features in real-time allows the system to offer valuable insights into the user's emotional and physical state.

In the emotion detection task, the CNN model performed exceptionally well, classifying emotions such as happiness, sadness, anger, and surprise from live facial images with an accuracy of 97%. This high level of accuracy emphasizes the potential of CNNs in recognizing complex emotional expressions with high precision.The combination of real-time emotion recognition and stress level prediction makes this system a powerful tool for personalized healthcare, human-computer interaction, and tailored user experiences.

FUTURE ENHANCEMENT

This project has significant potential for future enhancements, especially for academic exploration and real-world applications. As a student, you can expand the system's functionality by integrating additional facial attributes, such as ethnicity, facial landmarks, and health-related insights like stress detection or fatigue monitoring. Incorporating advanced deep learning models like Vision Transformers or more efficient CNN architectures can improve accuracy and processing speed, even in low-light or occluded environments.

To make the system more versatile, it can be extended to recognize group emotions in crowds or analyze human behavior for use in public safety and surveillance. Implementing multi-language support and voice commands in the interface could improve accessibility and usability. Furthermore, integrating with IoT devices like smart cameras could enable seamless real-time monitoring and applications in smart homes or workplaces.For academic research, this project can be a platform to explore cutting-edge algorithms such as Generative Adversarial Networks (GANs) for enhancing image quality or few-shot learning for better predictions with limited datasets. It also opens pathways for exploring ethical AI practices by developing bias-reduction techniques for gender, age, or emotion recognition models.

In summary, this project offers an excellent foundation for learning, innovation, and impactful real-world applications, with ample scope for future advancements.

REFERENCES

[1] **Khan, S., & Shah, M. (2018).** Deep Learning for Real-Time Facial Emotion Recognition: A Review and Future Directions. Journal of Imaging Science and Technology, 62(5), 50604.

[2] **Mollah, M. M., & Fattah, S. (2019).** Real-time Age and Gender Detection using Deep Learning. IEEE Access, 7, 155144-155152.

[3] **Singh, P., & Aggarwal, R. (2019).** Facial Emotion Recognition using Convolutional Neural Networks. International Journal of Computer Applications, 178(6), 27-32.

[4] **Liu, Y., & Wei, Y. (2020).** Facial Age Estimation via Deep Convolutional Neural Networks for Real-Time Applications. Springer Nature.

[5] **Sharma, H., & M, S. (2020).** Real-Time Gender Recognition System for Human-Computer Interaction using CNN. Procedia Computer Science, 167, 1911-1918.

[6] **Wang, S., & Xu, W. (2020).** Multi-Task Learning for Real-Time Emotion and Gender Recognition. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 4364-4372.

[7] **Hasan, M., & Zaki, M. (2021).** A Real-Time Face Emotion Recognition System Using Deep Learning. International Journal of Image and Graphics, 21(2), 2150007.

[8] **Kumar, V., & Gupta, V. (2021).** Emotion and Gender Recognition Using Deep Learning Techniques in Real-Time Video Streams. Proceedings of the 2021 IEEE 2nd International Conference on Artificial Intelligence, Big Data, Computing, and Data Communication Systems (icACDC), 98-104.

[9] **Shah, R., & Patel, A. (2021).** Age Estimation in Real-Time Using DeepFace and Convolutional Neural Networks. Journal of Artificial Intelligence and Soft Computing Research, 11(3), 171-177.

[10] **Raj, G., & Meena, A. (2022).** Real-Time Face Emotion Recognition System using Convolutional Neural Networks. Journal of Ambient Intelligence and Humanized Computing, 13(7), 3913-3925.

[11] **Ashutosh Singh, Khushdeep Singh (2024)** Machine Learning Algorithms for detecting Mental Stress in College Student**s,** *Journal of* 9th International Conference for Convergence in Technology,(I2CT).