Industry Oriented Mini Project Report

On

REAL TIME FACIAL EMOTION DETECTION WITH SMART FEEDBACK

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

By

Boga Snehitha -22261A1211

Kandula Pradeep Reddy- 22261A1230

Under the guidance of

Mr.B.Tulasi Dasu

Assistant Professor

Department of IT

DEPARTMENT OF INFORMATION TECHNOLOGY

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY

(AUTONOMOUS)

(Affiliated to JNTUH, Hyderabad; Eight UG Programs Accredited by NBA; Accredited

by NAAC with ‘A++’ Grade)

Gandipet, Hyderabad, Telangana, Chaitanya Bharati (P.O), Ranga Reddy District, Hyderabad– 500075, Telangana

2024-2025

CERTIFICATE

This is to certify that the **Industry Oriented Mini Project** entitled **Real Time Facial Emotion Detection with Smart Feedback** submitted by **Snehitha Boga(22261A1211), Kandula Pradeep Reddy (22261A1230)** in partial fulfillment of the requirements for the Award of the Degree of Bachelor of Technology in Information Technology as specialization is a record of the bona fide work carried out under the supervision of **Mr.B.TULASI DASU**, and this has not been submitted to any other University or Institute for the award of any degree or diploma

Internal Supervisor:	**IOMP Supervisor:**
Mr.B.Tulasi Dasu	**Dr. U. Chaitanya**
Assistant Professor	Assistant Professor
Dept. of IT	Dept. of IT

EXTERNAL EXAMINAR	**Dr. D. Vijaya Lakshmi**
	Professor and HOD
	Dept. of IT

DECLARATION

We hear by declare that the **Industry Oriented Mini Project** entitled **Real Time Facial Emotion Detection with Smart Feedback** is an original and bona fide work carried out by us as a part of fulfilment of Bachelor of Technology in Information Technology, Mahatma Gandhi Institute of Technology, Hyderabad, under the guidance of **Mr.B.Tulasi Dasu** ,Department of IT, MGIT.

No part of the project work is copied from books /journals/ internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the project work done entirely by us and not copied from any other source.

Snehitha Boga-22261A1211

Kandula Pradeep Reddy-22261A1230

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without introducing the people who made it possible and whose constant guidance and encouragement crowns all efforts with success. They have been a guiding light and source of inspiration towards the completion of the **Industry Oriented Mini Project**.

We would like to express our sincere gratitude and indebtedness to our project guide **Mr.B.Tulasi Dasu**, Dept. of IT, who has supported us throughout our project with immense patience and expertise.

We are also thankful to our honourable Principal of MGIT **Prof. G. Chandramohan Reddy** and **Dr. D. Vijaya Lakshmi**, Professor and HOD, Department of IT, for providing excellent infrastructure and a conducive atmosphere for completing this **Industry Oriented Mini Project** successfully.

We are also extremely thankful to our Project Coordinator(s) **Dr.U.Chaitanya**Assistant Professor, **Mrs.B.Meenakshi**, Professor, Department of IT, for their valuable suggestions and guidance throughout the course of this project.

We convey our heartfelt thanks to the lab staff for allowing us to use the required equipment whenever needed.

Finally, we would like to take this opportunity to thank our families for their support all through the work. We sincerely acknowledge and thank all those who gave directly or indirectly their support for completion of this work.

Snehitha Boga-22261A1211

Kandula Pradeep Reddy-22261A1230

ABSTRACT

This project focuses on real-time human emotion detection using facial expressions. It utilizes a pre-trained deep learning model to analyze facial features and determine the user's current emotional state. The system leverages OpenCV for face detection and DeepFace for accurate emotion recognition. As part of the user experience, the system offers voice feedback corresponding to the detected emotion, making interactions more engaging and intuitive. A live emotion trend graph is displayed to provide a visual representation of how the user's emotions evolve over time.

In addition to the core functionalities, the system enhances user engagement by displaying desktop notifications and pickup lines tailored to the recognized emotions. These features not only add a touch of personalization but also make the interaction more enjoyable. This emotion-aware system proves valuable across multiple domains, including mental health monitoring, AI-based personal assistants, customer service enhancement, and security applications. By enabling machines to understand and respond to human emotions, the project contributes to more natural and empathetic human-computer interactions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 MOTIVATION

In an era of increasing emphasis on human-computer interaction, the ability of machines to understand human emotions has become critically important. Traditional computing systems lack the emotional intelligence required to respond empathetically or adaptively to users' emotional states. This shortfall is especially significant in areas such as mental health monitoring, AI assistants, customer support, and user experience design.

This project is driven by the need to bridge this gap through a smart, real-time facial emotion detection system that not only identifies emotions from facial expressions but also provides responsive feedback. By combining computer vision, deep learning, and natural interaction features like voice feedback and desktop notifications, the system aims to enhance user engagement and well-being while paving the way for emotionally aware computing solutions.

1.2 PROBLEM STATEMENT

Despite the rapid growth in artificial intelligence, most systems still fail to recognize and appropriately respond to human emotions. There is a lack of real-time emotion-aware systems that can process facial expressions and react dynamically in a human-like manner. The absence of such emotional intelligence in machines results in static user experiences, limited adaptability in real-time interaction, and missed opportunities for applications in mental health, safety, and entertainment.

The problem lies in developing an accurate and efficient system that can:

- Detect multiple facial emotions in real-time using a webcam.
- Provide smart, immediate feedback through voice and visual cues.
- Maintain a trend graph of emotional changes.
- Notify the user or others when certain emotions are detected.
- Be adaptable and scalable for various real-world applications.

1.3 EXISTING SYSTEMS

Most existing systems for emotion detection are either offline, rely on static image processing, or provide only limited feedback. Common characteristics include:

- Use of pre-labeled datasets without real-time processing.
- Basic rule-based response systems that lack personalization or adaptability.
- Limited emotion classes (usually 5–7) without complex emotional states.
- Absence of interactive features like voice feedback or user engagement tools.

These limitations reduce the practicality and user experience of emotion detection systems in real-world environments.

1.3.1 LIMITATIONS

- **Lack of Real-Time Performance**: Existing systems are often not optimized for live webcam feeds.
- **Limited Feedback Mechanism**: Most do not include auditory or visual feedback mechanisms.
- **Low Engagement**: Absence of features like emotion-based pickup lines, mood tracking, or chat interface.
- **Emotion Scope**: Many models only detect a narrow range of basic emotions.
- **No Personalization**: Responses are generic and not adapted to individual users or changing contexts.

1.4 PROPOSED SYSTEM

The proposed system is a real-time facial emotion detection solution powered by a pre-trained deep learning model. It uses OpenCV for webcam access and face detection, while emotion recognition is performed through a trained neural network. Key components include:

- **Real-Time Emotion Recognition**: Detects emotions like Happy, Sad, Angry, Surprise, etc., directly from webcam input.
- **Smart Voice Feedback**: Uses pyttsx3 or FreeTTS to give real-time verbal suggestions based on emotions.
- **Emotion Trend Graph**: Visualizes the dominant emotions over time using Matplotlib.
- **Pickup Lines and Notifications**: Displays fun, emotion-specific desktop notifications with corresponding pickup lines.
- **Modular and Scalable Design**: Can be expanded to detect 20+ emotions, integrate chatbots, or support other input modes.

1.4.1 ADVANTAGES

- **Emotion-Aware Interactions**: Enables systems to react empathetically to users.
- **Enhanced Mental Health Monitoring**: Provides visual feedback and voice suggestions that can help users self-reflect or feel supported.
- **Real-Time Processing**: Efficiently handles continuous webcam input with low latency.
- **Engaging User Experience**: Pickup lines and pop-ups create a more relatable and fun interaction model.
- **Multi-Domain Applications**: Can be extended for education, security, healthcare, and virtual assistants.

1.5 OBJECTIVES

- Accurately detect and classify real-time facial emotions using webcam input.
- Provide personalized voice feedback and emotion-based desktop notifications.
- Visualize emotional trends through a live-updating graph.

- Enhance user interaction with engaging content like emotion-specific pickup lines.
- Create a scalable system that can be extended with additional features like chatbots or stress detection modules.

1.6 HARDWARE AND SOFTWARE REQUIREMENTS

Software Requirements

Programming Language:

- Python
  The project is developed primarily in Python, a robust language widely used for artificial intelligence, computer vision, and backend services. Python's rich ecosystem supports real-time video processing, deep learning, and user interaction features.

Libraries and Frameworks:

- **OpenCV:** For real-time face detection using webcam input.
- **TensorFlow/Keras:** For loading and using the pre-trained facial emotion recognition model.
- **pyttsx3:** For generating voice feedback based on detected emotions.
- **Plyer:** For displaying desktop notifications.
- **Matplotlib:** For plotting real-time emotion trend graphs.
- **NumPy & Pandas:** For efficient data handling and preprocessing.
- **Tkinter (Optional):** For developing GUI interfaces, if included.

Development Environment:

- **IDE:** Visual Studio Code
  VS Code is used for development due to its flexibility, extension support, debugging capabilities, and Git integration.

Operating System Compatibility:

- Compatible with **Windows 10 or 11**, ensuring ease of installation and smooth operation with required dependencies.

Hardware Requirements

Operating System:

- Windows 10 / 11 (64-bit)

Processor:

- Intel Core i5
  A higher-performance processor is recommended to ensure smooth webcam processing, real-time prediction, and system responsiveness.

RAM:

- Minimum 8 GB
- Sufficient for running the webcam stream, model inference, and concurrent processes like audio output and graph plotting.

Webcam:

- Integrated or external webcam
  Required for real-time facial capture.

Storage:

- Minimum 256 GB (SSD preferred)
  For storing dependencies, models, logs, and additional datasets if extended.

2. LITERATURE SURVEY

Ozdemir et al. (2020) proposed a real-time facial emotion recognition system using Convolutional Neural Network (CNN) architecture. The system was designed to recognize seven basic emotions with a focus on real-time implementation. The study demonstrated that CNN-based architectures are highly effective for facial feature extraction and classification, achieving good accuracy under controlled conditions. However, the model's performance in dynamic environments and with occluded or low-resolution faces was not extensively explored.[1]

Joshi et al. (2021) introduced Real Time Emotion Analysis (RTEA), a system aimed at detecting facial expressions in real-time video streams. Their work integrated computer vision techniques with a deep learning framework to process live webcam feeds. The system offered promising results with reduced latency and was evaluated for its practical usability. Nonetheless, it faced challenges with varying lighting conditions and user diversity, pointing to the need for robust preprocessing and data augmentation strategies.[2]

Winyangkun et al. (2023) developed a method for real-time detection and classification of facial emotions. Their approach involved collecting facial data and using a classification model to identify emotional states. The system emphasized responsiveness and adaptability across different users, suggesting improvements in real-time detection algorithms. They highlighted that additional contextual factors like body language or speech could further improve emotion classification accuracy.[3]

Bhanupriya et al. (2025) presented *EmotionTracker*, which utilizes OpenCV for face detection and DeepFace for emotion recognition. This lightweight system was optimized for real-time applications and demonstrated ease of deployment with minimal hardware requirements. The study validated the use of pre-trained deep learning models to achieve high accuracy in emotion classification. However, the authors noted that emotions with subtle facial cues (like "fear" and "neutral") were harder to detect reliably.[4]

Table 2.1 Literature Survey of Real time facial emotion detection with smart feedback

	Author& year of publication	**Journal / Conference**	**Method / Approach**	**Merits**	**Limitations**	**Research Gap**
[1]	M. Bhanupriya, N. Kirubakaran and P. Jegadeeshwari (2025)	International Conference on Data Science, Agents & Artificial Intelligence (ICDSAAI), Chennai, India	EmotionTracker: Real-time Facial Emotion Detection with OpenCV and DeepFace	Combines traditional image processing with deep learning for enhanced detection	Less effective for complex emotions, limited emotion categories	Expansion to a broader range of emotions and real-world scenarios needed
[2]	T. Winyangkun, N. Vanitchanant, V. Chouvatut and B. Panyangam (2023)	15th International Conference on Knowledge and Smart Technology (KST), Phuket, Thailand	Real-Time Detection and Classification of Facial Emotions	Accurate classification, lightweight model for real-time applications	Performance degrades in low-light conditions, high sensitivity to noise	Lacks multi-modal emotion recognition integration
[3]	M. Bhanupriya, P. Jegadeeshwari (2023)	International Conference on Data Science, Agents & Artificial Intelligence (ICDSAAI), Chennai, India	Closed-loop acoustic stimulation system that adapts based on sleep stages to improve sleep quality	Personalized feedback; real-time monitoring	Continuous monitoring not suitable for all; noise sensitivity concerns	Develop silent/alternative feedback systems for sensitive users
[4]	M. A. Ozdemir, B. Elagoz, A. Alaybeyoglu, R. Sadighzadeh and A. Akan (2020)	2020 Medical Technologies Congress (TIPTEKNO), Izmir, Turkey	CNN Architecture for facial emotion recognition	High accuracy in emotion detection, efficient feature extraction	Requires large datasets for training, computationally expensive	Limited generalizability to diverse datasets, lack of real-time optimization

3. ANALYSIS AND DESIGN

Understanding human emotions in real-time is essential for improving human-computer interaction, especially in fields like mental health monitoring, AI assistance, and user experience design. This project presents a **Real-Time Facial Emotion Detection System with Smart Feedback**, which uses a pre-trained deep learning model to recognize human emotions from facial expressions via a webcam

The system utilizes **OpenCV** for live face detection and a deep learning model built with **TensorFlow/Keras** for emotion classification. Once an emotion is detected, the system provides **voice feedback** using **pyttsx3**, displays **real-time desktop notifications** with relevant pickup lines via **Plyer**, and visualizes emotion trends using **Matplotlib**. This multi-modal feedback loop makes the system not just responsive but also emotionally engaging and user-friendly. The application is highly useful in real-time environments for stress detection, virtual counseling, interactive AI systems, and more.

3.1 MODULES

User Interaction Module

- **Function:** Manages real-time video feed and displays detected emotions to the user.
- **Input:** Live webcam stream (face).
- **Output:** Emotion label on screen and corresponding feedback (voice + notification).

2. Preprocessing Module

- **Function:** Handles image preprocessing such as grayscale conversion, face alignment, and resizing for consistent model input.
- **Input:** Raw image frames from webcam.

- **Output:** Preprocessed image suitable for emotion classification.

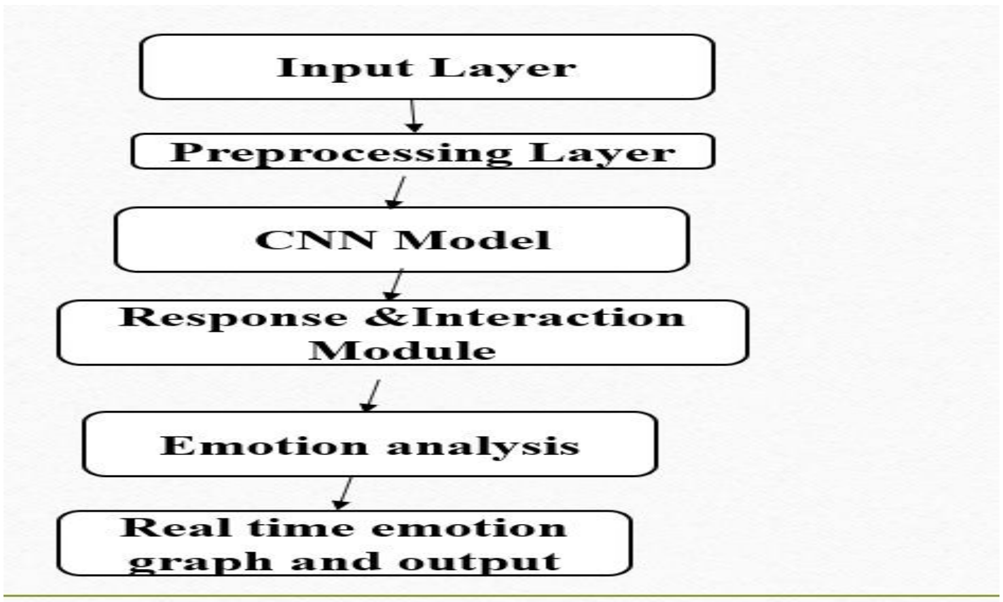
3. Prediction Module

- **Function:** Uses a pre-trained deep learning model to classify emotions from the face image.
- **Input:** Preprocessed facial image.
- **Output:** Detected emotion (e.g., Happy, Sad, Angry, etc.).

4. Smart Feedback Module

- **Function:** Provides responsive feedback including:
    - Voice message using pyttsx3
    - Desktop notification with personalized pickup line using plyer
    - Emotion trend visualization using matplotlib
- **Input:** Detected emotion.
- **Output:** Audio-visual response and trend updates.

3.2 ARCHITECTURE



Input Layer
Preprocessing Layer
CNN Model
Response &Interaction Module
Emotion analysis
Real time emotion graph and output

Fig. 3.2.1 Architecture of Real tine facial emotion detection

The architecture of the system, as shown in **Fig. 3.2.1**, is designed to detect facial emotions in real-time and provide smart feedback based on the detected emotion. The system begins with the **Input Capture module**, which uses a **webcam** to continuously stream live video of the user's face.

The next step is the **Face Detection & Preprocessing module**, which uses **OpenCV** to detect faces in the video frames and preprocesses them (e.g., grayscale conversion, resizing) to make them suitable for input into the deep learning model.

The preprocessed face is passed to the **Emotion Recognition Model**, which is a **pre-trained deep learning model** built using **TensorFlow/Keras**. It classifies the facial expression into one of several predefined emotion categories such as **Happy, Sad, Angry, Neutral**, etc.

Once an emotion is detected, it is sent to the **Smart Feedback Engine**, which triggers multiple responsive actions:

- **Voice feedback** using pyttsx3 (e.g., saying “Cheer up!” if Sad is detected),
- **Desktop notifications** with related messages or pickup lines using plyer,
- **Real-time emotion trend graph** using matplotlib to show how the user's mood changes over time.

Finally, the **User Interface module** presents all these outputs to the user in a clean, interactive way—displaying live camera feed, detected emotion labels, notifications, and the mood trend graph, all in real time.

3.3 UML DIAGRAMS

3.3.1 USE CASE DIAGRAMS

A **use case diagram** is a visual tool used to represent the interactions between external entities (called actors) and the system under development. It illustrates the various ways users or other systems interact with the application to accomplish specific tasks or goals. Use case diagrams play a critical role in system analysis and design by providing a clear overview of the system's functionality from the user's perspective.

In this project, actors can include the **User**, **System Administrator**, or any external service interfacing with the system. The use cases represent the core functionalities such as **Start Real-Time Emotion Detection**, **Provide Voice Feedback**, **Display Emotion Trends**, **Send Notifications**, and **Log Emotion Data**.

The connections between actors and use cases, shown as lines in the diagram (see Fig. 3.3.1.1), illustrate how each actor participates in or triggers a given system function. This visual representation helps stakeholders understand the system scope, requirements, and user interactions before proceeding to detailed design and implementation phases.

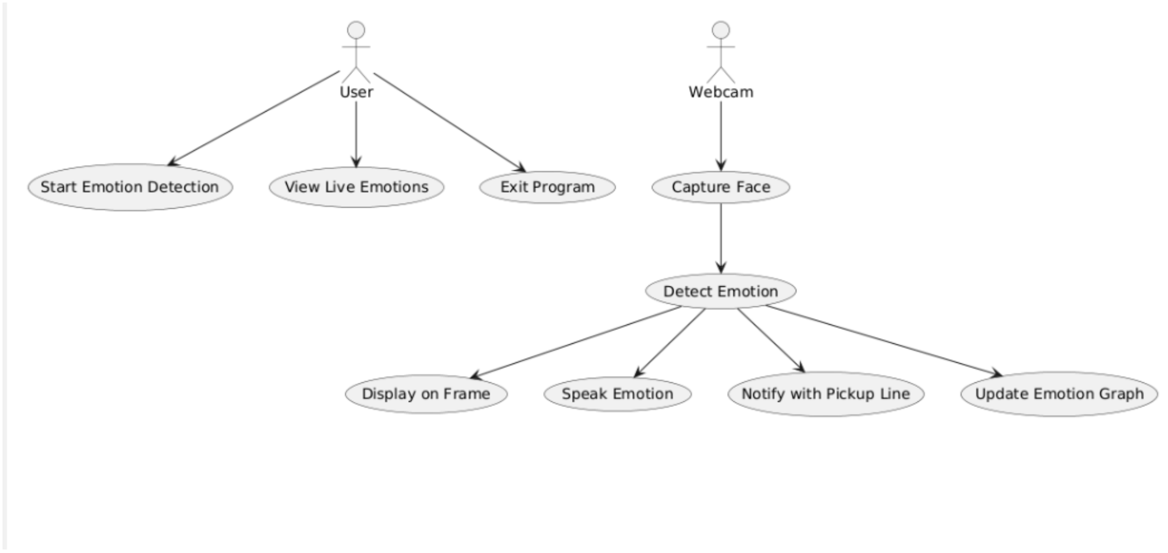


Fig. 3.3.1.1 Use Case Diagram

Actors:

1. **User**: The individual interacting with the system, providing input data (such as sleep time, mood, etc.) and viewing personalized wellness recommendations.
2. **System**: The backend system that processes user data, trains models, and generates predictions based on the user's inputs.

Use Cases:

1. **Register**: The user can create an account to access the system's features, including personalized recommendations for sleep and wellness.
2. **Login**: After registration, the user logs in to the system using their credentials to access personalized data and recommendations.
3. **Selecting Coin**: The user selects a cryptocurrency (Bitcoin or Ethereum) to predict its price, starting the prediction process.
4. **Prediction of Coin Price**: The system predicts the price of the selected cryptocurrency based on the trained models.
5. **Predicted Price in Incremental Timeframes**: The system provides predictions for different timeframes, such as hourly, daily, or weekly.
6. **Visualizing Trained Models and Predicted Price**: The system displays the trained machine learning models, showing graphs and charts of the predicted cryptocurrency price trends.

3.3.2 CLASS DIAGRAM

A class diagram is a visual representation that models the static structure of a system, showcasing the system's classes, their attributes, methods (operations), and the relationships between them as seen in Fig. 3.3.2.1. It is a key tool in object-oriented design and is commonly used in software engineering to define the blueprint of a system.

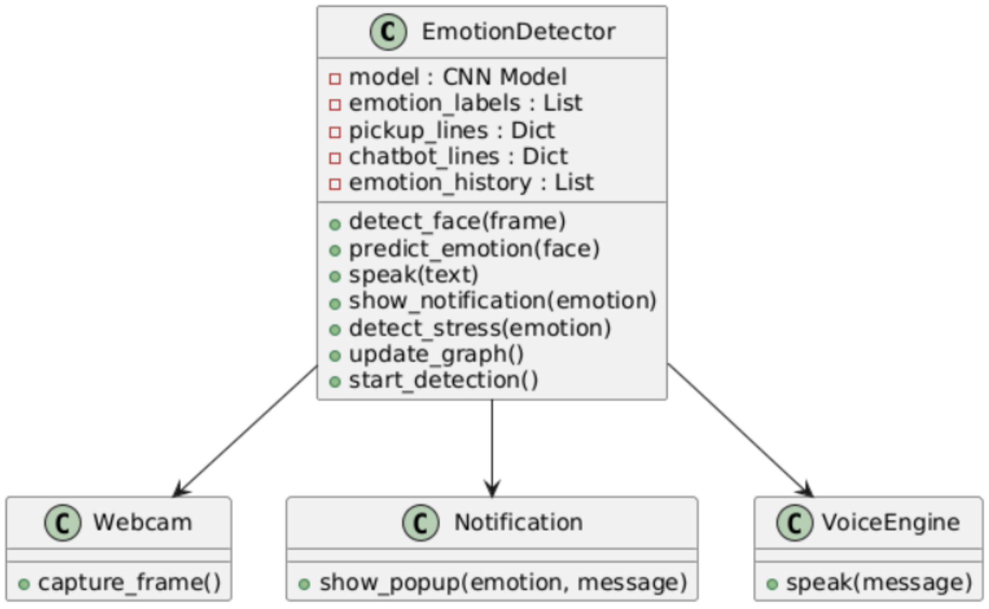


Fig. 3.3.2.1 Class Diagram

Relationships:

1. **User → Emotion Detection System**

- The user interacts with the system by starting or stopping the real-time emotion detection process and viewing feedback.
- Users can receive voice feedback, desktop notifications, and emotion trend visualizations.

2.Emotion Detection System → Webcam

- The system accesses the webcam to capture live video frames for facial emotion analysis.

3. **Emotion Detection System → Face Detector**

- The face detector module processes video frames to identify and isolate faces for emotion classification.

4.Emotion Detection System → Emotion Recognition Model

- The emotion recognition model receives preprocessed face data and predicts the user’s current emotion**.**

5.Emotion Recognition Model → Feedback Engine

- The model sends the detected emotion to the feedback engine.

6.Feedback Engine → Voice Feedback Module

- The feedback engine triggers voice responses (e.g., using pyttsx3) corresponding to detected emotions.

7.Feedback Engine → Notification Module

- The feedback engine sends desktop notifications related to the user’s current emotional state**.**

8.Feedback Engine → Emotion Trend Visualizer

- The feedback engine updates the real-time emotion trend graph showing mood changes over time.

3.3.3 ACTIVITY DIAGRAM

An Activity Diagram is a type of behavioral diagram used in Unified Modeling Language (UML) to represent the flow of control or data through the system as seen in Fig. 3.3.3.1. It focuses on the flow of activities and actions, capturing the sequence of steps in a particular process or workflow. Activity diagrams are commonly used to model business processes, workflows, or any sequential activities in a system.

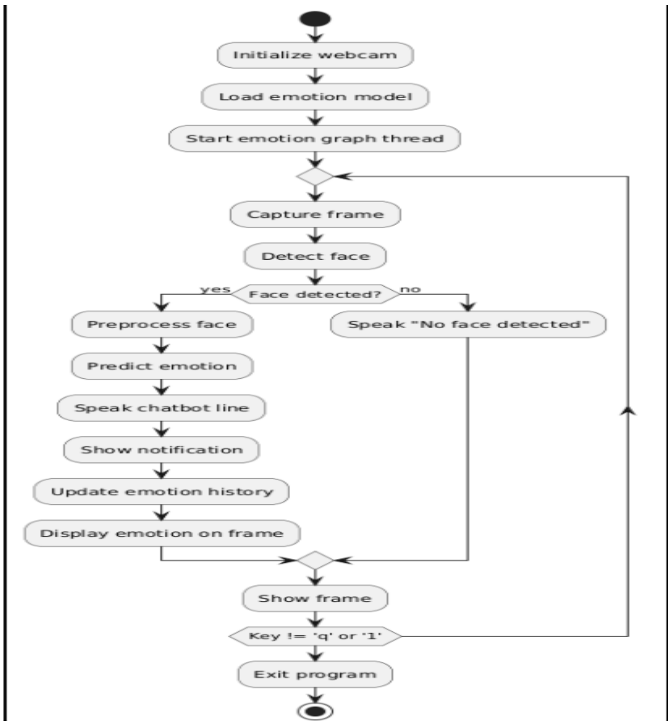


Fig. 3.3.3.1 Activity Diagram

Flow Explanation:

1. **Start System**

   The user launches the facial emotion detection application.

2. **InitializeWebcam**

   The system accesses and initializes the webcam to capture live video frames.

3. **Capture**

   The webcam continuously captures video frames in real-time.

4. **DetectFaceinFrame:**

   The face detection module processes each frame to detect the presence and location of faces.

5. **Is Face Detected?**

   - **Yes:** Proceed to next step.
   - **No:** Provide voice feedback like "No face detected" and continue capturing frames.

6. **PreprocessFaceImage**

   Extract the detected face region and preprocess it for emotion recognition (e.g., resizing, normalization).

7. **PredictEmotion**

   The emotion recognition model analyzes the processed face image and predicts the current emotion.

8. **GenerateFeedback**

   Based on the detected emotion, the system:

   - Provides voice feedback (e.g., encouraging words).
   - Displays desktop notifications relevant to the emotion.
   - Updates the real-time emotion trend graph.

9. **LogEmotionData**

   The detected emotion and timestamp are logged for trend analysis.

10. **User Continues or Stops**

    - If the user chooses to continue, the system loops back to step 3 for continuous detection.
    - If the user stops, proceed to shutdown.

11. **ShutdownSystem**

    Release webcam and system resources, then end the application

3.3.4 SEQUENCE DIAGRAM

A sequence diagram illustrates the flow of interactions between actors and system components over time as seen in Fig. 3.3.4.1, emphasizing the order in which messages are exchanged to achieve specific functionalities. Actors represent external entities that interact with the system, while lifelines depict the system components involved in the process. Messages are shown as arrows, indicating the flow of information or actions between these elements. By providing a step-by-step view of workflows, sequence diagrams help in understanding and designing the dynamic behaviour of a system.

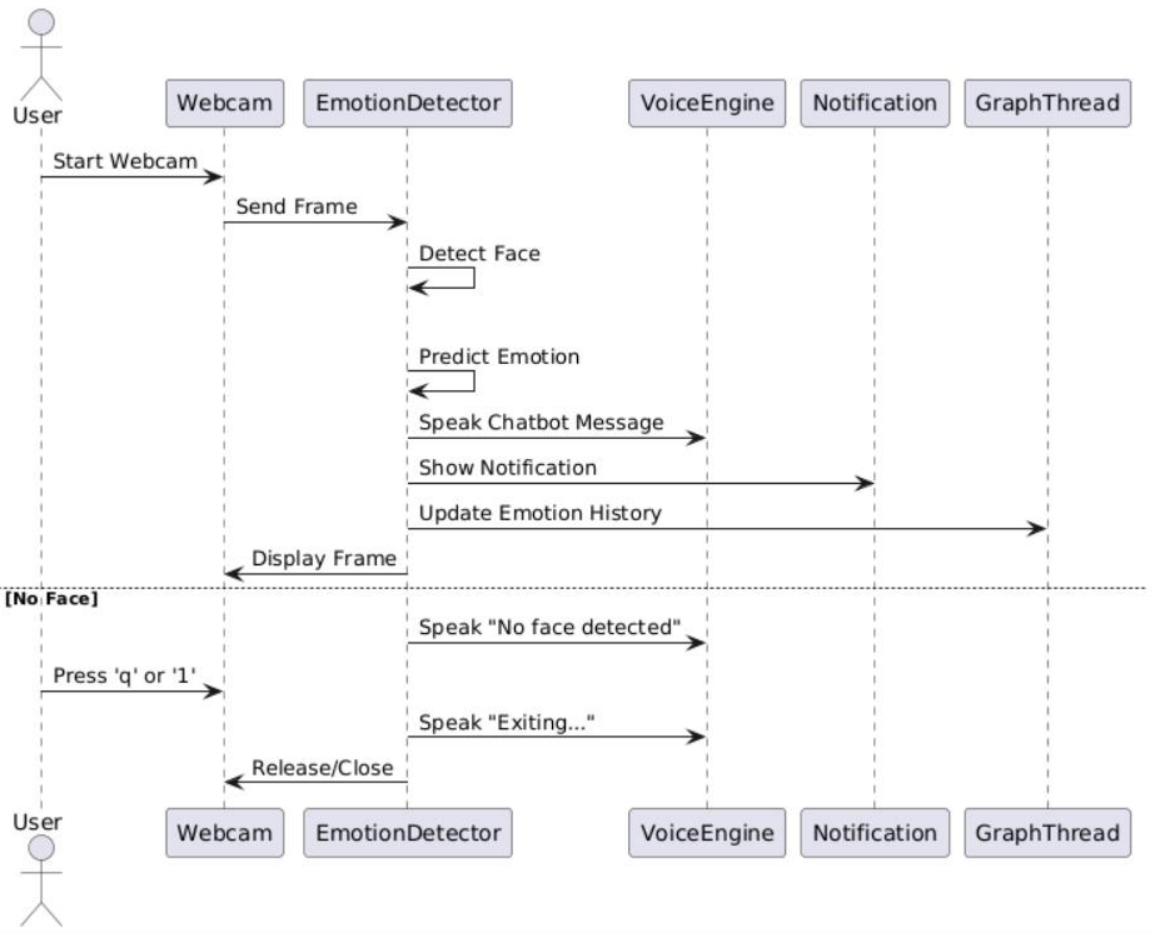


Fig. 3.3.4.1 Sequence Diagram

Key Interactions and Relationships

1. User starts the application.
2. Webcam starts capturing video frames.
3. For each frame:
   - Face Detection Module receives frame and detects face.
   - If a face is detected, the detected face image is passed to the Emotion Recognition Model.
4. The Emotion Recognition Model predicts the emotion.
5. The predicted emotion is sent to the Feedback System to:
   - Provide voice feedback.
   - Show desktop notifications.
   - Update the real-time emotion trend graph.

6. The Logger stores the detected emotion with timestamp.
7. Steps 2-6 repeat continuously until the User stops the application.

On stop, system releases resources and ends

3.3.5 COMPONENT DIAGRAM

A Component Diagram is a type of structural diagram used in software engineering to represent the components of a system and how they interact or depend on each other. It shows how the components (which could be software modules, subsystems, or other significant parts) are organized and connected within a system. In this diagram, each component encapsulates a set of related functionalities and interfaces as shown in Fig. 3.3.5.1.

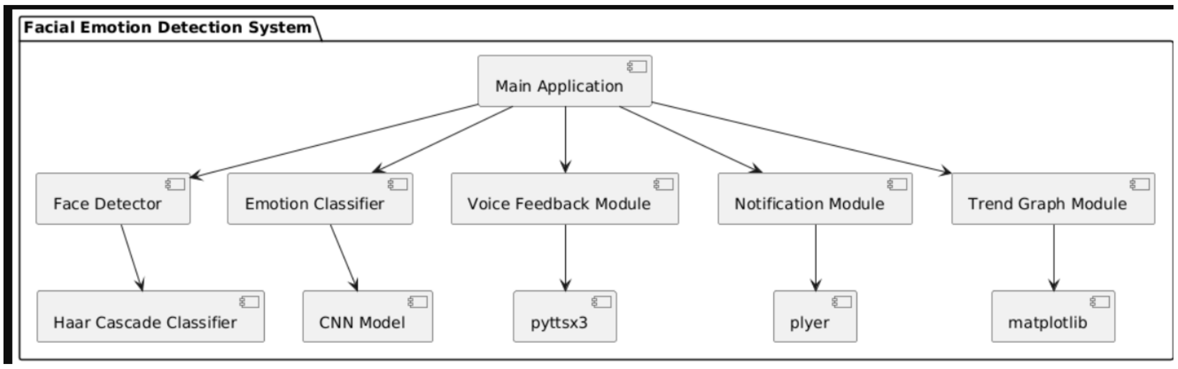


Fig. 3.3.5.1 Component Diagram

Main Components:

- User Interface (UI)
    - Captures webcam input
    - Displays detected emotions, graphs, notifications
    - Provides user interaction for start/stop, settings
- Face Detection Module
    - Detects faces in real-time frames using OpenCV

- Emotion Recognition Module
    - Loads pre-trained deep learning model
    - Classifies emotions based on face data
- Feedback System
    - Generates voice feedback using pyttsx3 or similar
    - Shows desktop notifications and pickup lines
    - Plots emotion trend graphs
- Data Manager
    - Handles storage of detected emotion data
    - Manages logs or session data for trend analysis

3.3.6 DEPLOYMENT DIAGRAM

The Deployment Diagram illustrates the physical deployment of software components across hardware nodes in the **Rhythm Restore** system. It captures the interactions between the **user device**, **server**, and **database**, highlighting how system components communicate and are distributed in a real-world deployment scenario.

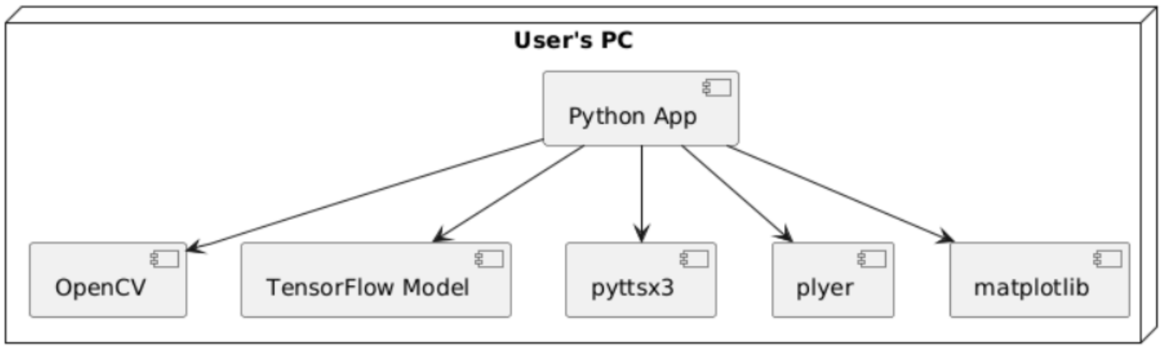


Fig. 3.3.6.1 Deployment Diagram

The deployment diagram shows how the system components are distributed across three main nodes:

- User Device (Laptop/Desktop)

    - Runs the Application (with GUI to capture webcam input, show feedback)
    - Connects to Webcam
- Webcam
    - Captures real-time video feed
    - Sends video frames to the application
- Processing Unit (Local Machine)
    - Runs Face Detection Module (OpenCV)
    - Runs Emotion Recognition Model (Deep Learning model)
    - Runs Feedback System (Voice output, notifications, graph plotting)

3.4 METHODOLOGY

Data Acquisition

The system continuously captures live video feed using a webcam. Each video frame is processed in real time to detect human faces. To ensure accurate emotion detection, face images are extracted from the video frames using OpenCV's face detection algorithms. This live streaming data forms the basis for real-time emotion recognition

Preprocessing

Detected faces are preprocessed by resizing to the input dimensions expected by the deep learning model. Image normalization and grayscale conversion are applied as necessary to standardize the input for consistent performance.

Emotion Recognition Model

A pre-trained deep learning model, such as a convolutional neural network (CNN), is employed to classify facial emotions from the preprocessed face images. The model has been trained on a diverse dataset covering multiple emotions such as happiness, sadness, anger, surprise, and neutrality.

How it works:

- The model processes each detected face image and outputs probabilities for each emotion class.

- The emotion with the highest probability is selected as the recognized emotion for that frame.
- Continuous emotion predictions are logged for trend analysis.

Why it's Used: LSTM's ability to store and manage information over long periods allows it to capture gradual, long-term trends in cryptocurrency prices, making it particularly effective for long-term predictions in volatile markets.

Smart Feedback Module

The system uses the detected emotions to provide intelligent feedback to the user:

- Voice Feedback: Using text-to-speech libraries (e.g., pyttsx3), the system generates spoken messages tailored to the detected emotion to engage the user.
- Visual Notifications: Desktop notifications and on-screen pickup lines or encouraging messages are displayed corresponding to the emotion.
- Emotion Trends: A real-time graph plots the frequency of detected emotions over the session, offering insight into emotional changes.

4. CODE AND IMPLEMENTATION

4.1 CODE

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import load_model
import cv2
import pyttsx3
from plyer import notification
import matplotlib.pyplot as plt
from collections import Counter
import threading
import time
import os
import tkinter as tk
model = load_model("fer2013_model.h5")
emotion_labels = ['Angry', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral']
engine = pyttsx3.init()
exit_button_width = 150
exit_button_height = 50

pickup_lines = {
    'Angry': "Your anger makes thunder look soft... but I bet I can handle your storm!",
    'Disgust': "Even with that look, you're still charming! You're like a fine wine... with
a twist of lemon.",
    'Fear': "No fear, I’m here for you. Just don’t be afraid to smile... it won’t bite, I
promise!",
    'Happy': "Your smile could light up the city! Seriously, I think I need sunglasses just
to look at you.",
    'Sad': "If tears were diamonds, you'd be rich by now. But hey, don't worry, you’ve
got a treasure chest of happiness ahead!",
    'Surprise': "That surprised look suits you! You should wear it more often – it’s like
your new superpower.",
    'Neutral': "Mysterious and composed – love that vibe. You're like a cool breeze on a hot day... seriously, I could use that right now."
}

chatbot_lines = {
    'Angry': "Hey, I can sense some anger. Let's take a deep breath together!",
    'Disgust': "Hmm, looks like something's bothering you. Let's talk about it.",
    'Fear': "You're safe now, don't worry.",
    'Happy': "I'm happy to see you happy! Keep it going!",
    'Sad': "I understand you're feeling down. Let's cheer you up.",
    'Surprise': "That caught you off guard, huh?",
    'Neutral': "You seem calm. That's great!"
}
def speak(text):
    engine.say(text)
    engine.runAndWait()
def detect_stress(emotion):
    suggestion = chatbot_lines.get(emotion, "You're doing great, keep going!")
    speak(suggestion)
def show_notification(emotion):
    line = pickup_lines.get(emotion, "Keep smiling!")
    try:
        notification.notify(
            title=f"Emotion: {emotion}",
            message=line,
            timeout=3
        )
    except:
        print(f"Notification failed for {emotion}")
emotion_history = []
running = True
def create_exit_gui():
    def exit_app():
        global running
        running = False
        speak("Exiting the application now!")
        root.destroy()

    root = tk.Tk()
    root.title("Exit Panel")
    root.geometry("200x100")
    tk.Button(root, text="Exit App", command=exit_app, bg="red", fg="white",
font=("Arial", 14)).pack(expand=True)
    root.mainloop()
def plot_emotion_graph():
    plt.ion()
    fig, ax = plt.subplots()
    while running:
        if emotion_history:
            ax.clear()
            counts = Counter(emotion_history[-30:])
            ax.bar(counts.keys(), counts.values(), color='skyblue')
            ax.set_title('Emotion Trend')
            ax.set_ylabel('Count')
            plt.pause(2)
    plt.close('all')
threading.Thread(target=create_exit_gui, daemon=True).start()
threading.Thread(target=plot_emotion_graph, daemon=True).start()
cap = cv2.VideoCapture(0)
cv2.namedWindow("Emotion Detection")
prev_emotion = None
last_spoken_time = time.time()
last_emotion_time = time.time()
while running:
    ret, frame = cap.read()
    if not ret:
        break
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.3, minNeighbors=5)
    for (x, y, w, h) in faces:
        face = gray[y:y + h, x:x + w]
        face = cv2.resize(face, (48, 48))
        face = face.astype('float32') / 255.0
        face = np.expand_dims(face, axis=-1)
        face = np.expand_dims(face, axis=0)
        emotion_probs = model.predict(face, verbose=0)
        max_index = np.argmax(emotion_probs[0])
        predicted_emotion = emotion_labels[max_index]
    if len(faces) == 0:
        if time.time() - last_spoken_time > 3:
            speak("Umm! No face detected!")
            last_spoken_time = time.time()
        if predicted_emotion != prev_emotion or (time.time() - last_emotion_time > 5):
            emotion_history.append(predicted_emotion)
            detect_stress(predicted_emotion)
            show_notification(predicted_emotion)
            prev_emotion = predicted_emotion
            last_emotion_time = time.time()
        cv2.putText(frame, f"{predicted_emotion}", (x, y - 10),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 255), 2)
     cv2.putText(frame, f"{pickup_lines.get(predicted_emotion)}", (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 255, 255), 1)
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 100, 100), 2)

    cv2.imshow("Emotion Detection & Chatbot", frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        running = False
cap.release()
cv2.destroyAllWindows()
```

```
time.sleep(1)

    if positive_emotions >= negative_emotions:
        mood_message = "The person seems to be in a good and calm mood! 😊"
    else:
        mood_message = "The person appears stressed or tensed. Consider taking a break
or relaxing. 😟"
def send_final_mood_notification():
    final_counts = Counter(emotion_history)
    positive_emotions = final_counts['Happy'] + final_counts['Neutral']
    negative_emotions = sum(final_counts[emo] for emo in emotion_labels if emo not
in ['Happy', 'Neutral'])
    try:
        notification.notify(
            title="▯ Final Mood Summary",
            message=mood_message,
            timeout=6
        )
    except:
        print("Final mood notification failed.")

    print("📊 Final Mood Notification Sent.")
    speak(mood_message)
send_final_mood_notification()
```

4.2 IMPLEMENTATION

The project is organized into a directory named RealTimeEmotionDetection/ with the following structure:

bash

CopyEdit

```
RealTimeEmotionDetection/
│
├── main.py                  # Main Python script to run the emotion detection system
├── fer2013_model.h5         # Pre-trained facial emotion recognition model
├── requirements.txt         # List of required Python packages
├── README.md                # Project description and setup instructions
├── resources/               # Folder for images, notifications, and other resources
└── utils/                   # Utility scripts (optional, e.g., for data preprocessing)
```

Software and Package Installation

To run the system, install the required Python packages with:

bash

CopyEdit

```
pip install numpy tensorflow opencv-python pyttsx3 plyer matplotlib
```

- **numpy**: for numerical operations on image data
- **tensorflow**: for loading and running the deep learning model
- **opencv-python**: for real-time video capture and face detection
- **pyttsx3**: for text-to-speech voice feedback
- **plyer**: for desktop notifications
- **matplotlib**: for plotting emotion trend graphs

Code Overview

Model Loading

The system loads a pre-trained Keras model (fer2013_model.h5) for facial emotion classification into one of seven categories:

python

CopyEdit

```
emotion_labels = ['Angry', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral']
model = load_model("fer2013_model.h5")
```

Real-Time Video Capture and Face Detection

- Webcam feed is captured using OpenCV.
- Faces are detected in each frame using OpenCV's Haar Cascade classifier.
- Detected face regions are extracted, resized, normalized, and passed to the model.

Emotion Prediction and Feedback

- The model predicts the emotion label for each detected face.
- If the predicted emotion differs from the previous one or after a time gap, the system:
    - Adds the emotion to history for trend analysis.
    - Provides voice feedback through pyttsx3 using predefined chatbot messages.
    - Displays desktop notifications with emotion-specific pickup lines.
    - Shows emotion label and pickup line text overlay on the video feed.

GUI and Exit Handling

- A separate Tkinter window runs as an exit panel with an "Exit App" button.
- Clicking the button stops the real-time video loop and closes the application with voice confirmation.

Emotion Trend Visualization

- A live-updating matplotlib bar chart shows the count of emotions detected over the last 30 frames.
- This graph provides a visual summary of the user's emotional trends in real time.

Final Mood Notification

- When the application exits, a summary notification is sent based on the counts of positive vs. negative emotions detected during the session.
- Voice feedback announces the final mood status to the user.

Sample Code Snippet (Main Loop)

python

CopyEdit

```
while running:
    ret, frame = cap.read()
    if not ret:
        break

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.3, minNeighbors=5)

    if len(faces) == 0 and time.time() - last_spoken_time > 3:
        speak("Umm! No face detected!")
        last_spoken_time = time.time()
    for (x, y, w, h) in faces:
        face = gray[y:y+h, x:x+w]
        face = cv2.resize(face, (48,48))
        face = face.astype('float32') / 255.0
        face = np.expand_dims(face, axis=(0, -1))
        emotion_probs = model.predict(face, verbose=0)
        max_index = np.argmax(emotion_probs[0])
        predicted_emotion = emotion_labels[max_index]
        if predicted_emotion != prev_emotion or (time.time() - last_emotion_time > 5):
            emotion_history.append(predicted_emotion)
            detect_stress(predicted_emotion)
            show_notification(predicted_emotion)
            prev_emotion = predicted_emotion
            last_emotion_time = time.time()

        # Draw rectangle and text overlays on the frame
        cv2.rectangle(frame, (x,y), (x+w,y+h), (255,100,100), 2)
        cv2.putText(frame, predicted_emotion, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX,
1, (0,255,255), 2)
        cv2.putText(frame, pickup_lines.get(predicted_emotion), (10,30),
cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255,255,255), 1)
```

```
cv2.imshow("Emotion Detection & Chatbot", frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
    running = False
```

5. TESTING

5.1 INTRODUCTION TO TESTING

Testing is a critical aspect of software development aimed at verifying, validating, and ensuring the quality and reliability of software components. It helps to minimize risks and optimize resource utilization throughout the development lifecycle. Implementing testing early allows for the identification and resolution of defects before they escalate, improving overall system robustness.

Testing involves evaluating the software under different conditions and environments to assess its functionality, performance, and other essential attributes. Various testing methodologies are employed depending on the nature and objectives of the software being developed.

In this project, testing ensures that every module of the real-time facial emotion detection system functions effectively, contributing to the overall reliability and performance of the system. The focus is on assessing the system's ability to accurately detect facial emotions in real-time, provide appropriate smart feedback through voice and notifications, and maintain stability during continuous operation.

5.2 Testing Objectives

The primary objectives of testing this system include:

- Verifying accurate real-time detection of facial emotions across various facial expressions.
- Ensuring timely and appropriate voice feedback corresponding to detected emotions.
- Validating the display of notifications and pickup lines aligned with the emotion detected.
- Assessing the stability of webcam feed processing over extended periods.
- Checking the responsiveness and functionality of the exit GUI button.
- Confirming that the system handles no-face detection scenarios gracefully.

5.3 TEST CASES:

Table 5.1 Test Cases of Real time facial emotion detection with smart feedback

Test Case ID	**Test case Name- Facial Expression**	**Test Description**	**Expected Output**	**Actual Output**	**Remarks**
TC_01	Neutral	When the user is in Constant mood	The expression has captured and also it will give a voice feedback	Expression based voice feedback with a pop up message and a live graph is displayed.	Success
TC_02	Happy	When the user is in Happy mood	The expression has captured and also it will give a voice feedback	Expression based voice feedback with a pop up message and a live graph is displayed.	Success
TC_03	Angry	When the user is in angry mood	The expression has captured and also it will give a voice feedback	Expression based voice feedback with a pop up message and a live graph is displayed.	Success
TC_04	Sad	When the user is in sad mood	The expression has captured and also it will give a voice feedback	Expression based voice feedback with a pop up message and a live graph is displayed.	Success
TC_05	Fear	When the user is in fear mood	The expression has captured and also it will give a voice feedback	Expression based voice feedback with a pop up message and a live graph is displayed.	Success
TC_06	Surprise	When the user is in surprise mood	The expression has captured and also it will give a voice feedback	Expression based voice feedback with a pop up message and a live graph is displayed.	Success

6. RESULTS

[6.1] FACIAL EXPRESSION:NEUTRAL

Facial Expression :

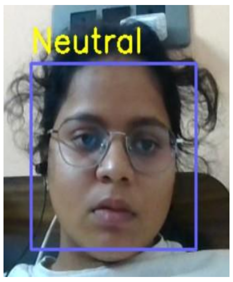


Fig6.1.1 Neutral exp

Fig 6.1.1 displays the detected emotion as neutral

Bargraph:

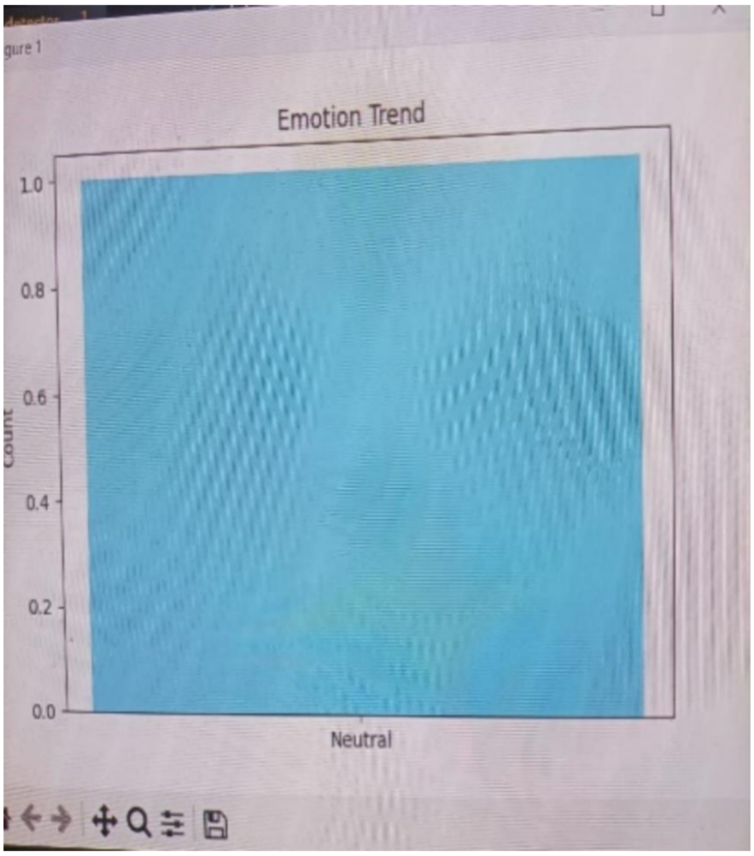
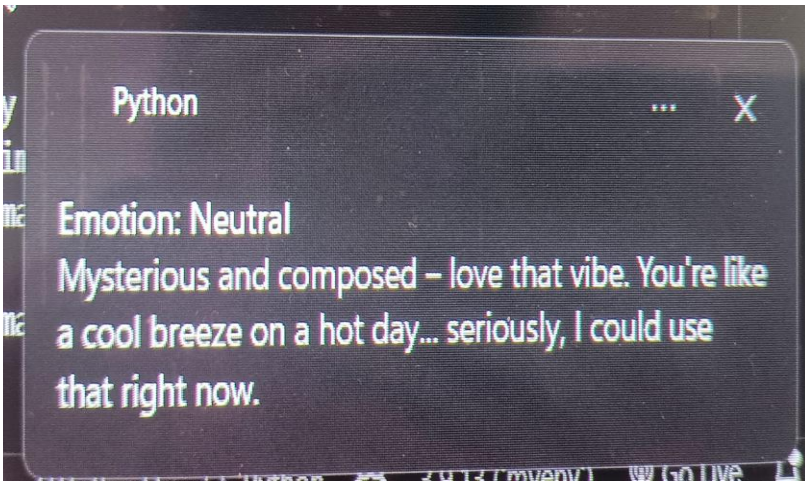


Fig 6.1.2. Bargraph

Fig 6.1.2 displays the bargraph based on the detected emotion.

Pop-up Message


Python

Emotion: Neutral
Mysterious and composed – love that vibe. You're like a cool breeze on a hot day... seriously, I could use that right now.

Fig 6.1.3 Pop-up message

fig 6.1.3 displays the pop up message and makes the person happy with the given voice based suggestion

[6.2] FACIAL EXPRESSION:HAPPY

Facial Expression

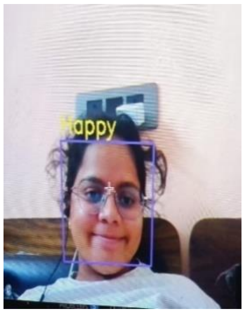


Fig 6.2.1. Happy exp

Fig 6.2.1 displays that the given expression is happy

Bargraph

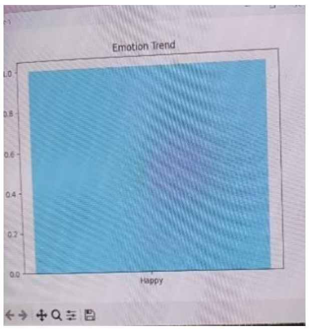
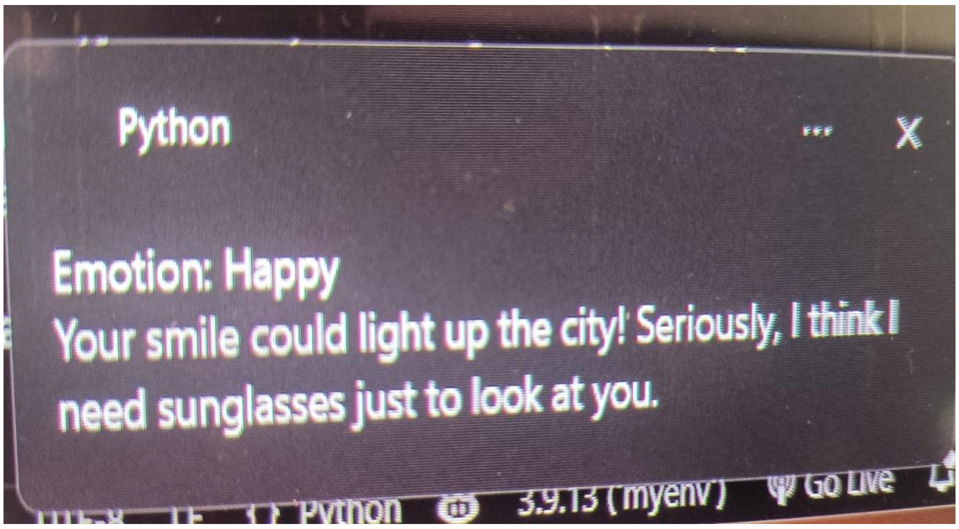


Fig 6.2.2 Bargraph

Fig 6.2.2 displays the bargraph based on the detected emotion.

Pop-up Message



Python

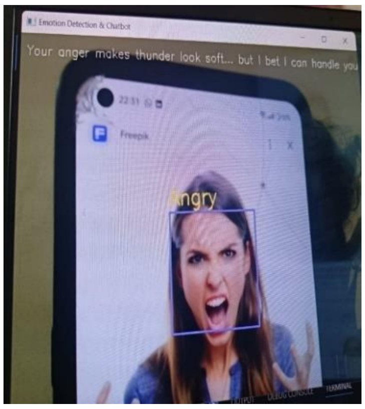
Emotion: Happy
Your smile could light up the city! Seriously, I think I need sunglasses just to look at you.

Fig. 6.2.3 Pop up message

Fig 6.2.3 displays the pop up message that makes the person more happy and also a voice based suggestion as to maintain the same.

[6.3]FACIAL EXPRESSION:ANGRY

Facial expression



Your anger makes thunder look soft... but I bet I can handle you

Fig 6.3.1 Angry exp

Fig 6.3.1 detects the emotion and display the emotion as angry.

BarGraph

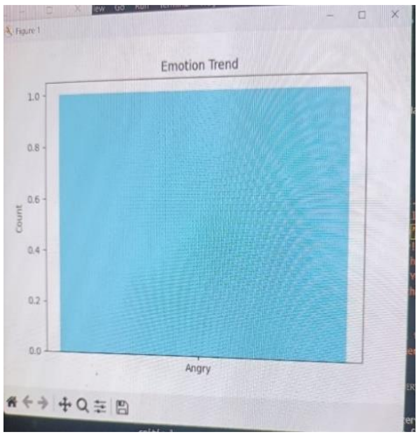
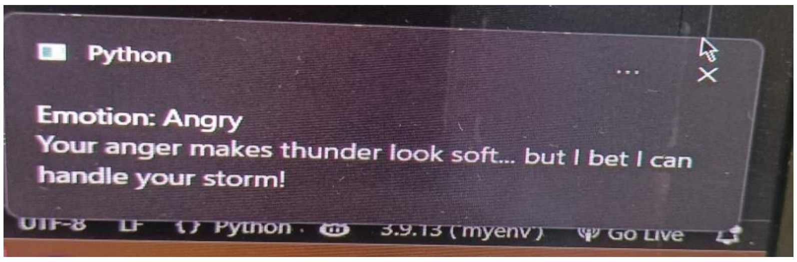


Fig 6.3.2Bargraph

Fig 6.3.2 displays the baraagraph for the detected emotion.

Pop up message:



Python

Emotion: Angry
Your anger makes thunder look soft... but I bet I can handle your storm!

Fig 6.3.3 Pop up message

Fig 6.3.3 gives the pop up message that reminds the user and makes him/her cool and also a voice based output to take a deep breath.

[6.4]FACIAL EXPRESSION:SAD

Facial expression

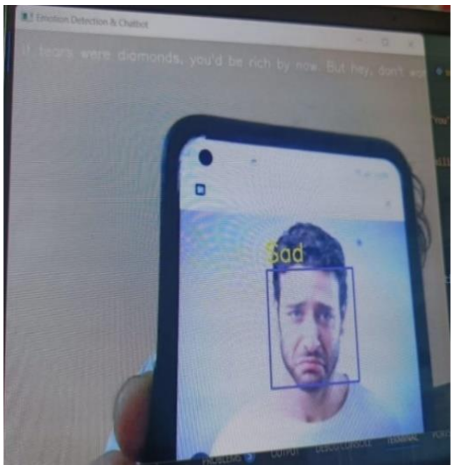

Emotion Detection & Chatbot
If tears were diamonds, you'd be rich by now. But hey, don't
Sad

Fig 6.4 1.Sad exp

Fig 6.4.1 shows that the detected emotion is sad.

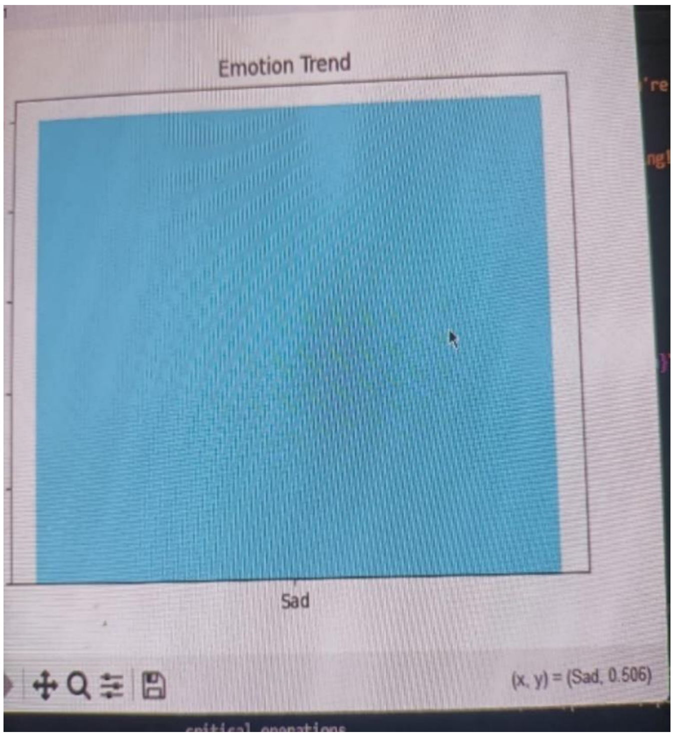
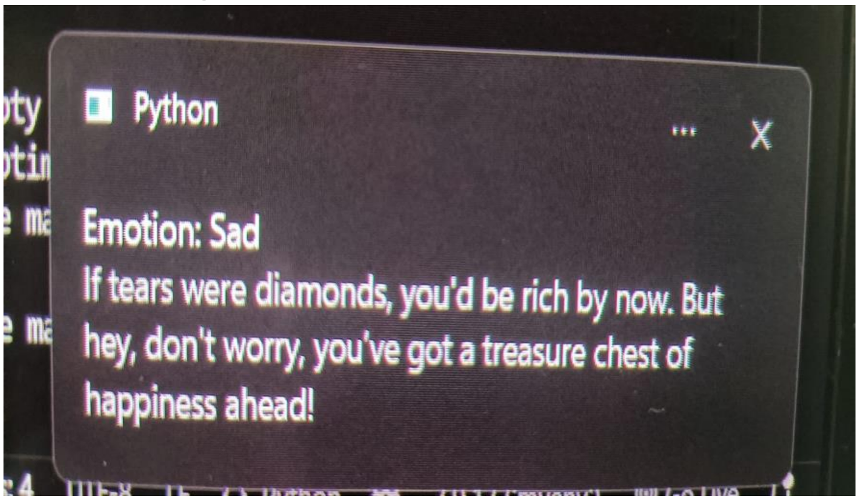


Fig 6.4.2 bargraph

Fig 6.4.2 shows that the bargraph of the detected emotion.

Pop-up Message



Python

Emotion: Sad

If tears were diamonds, you'd be rich by now. But hey, don't worry, you've got a treasure chest of happiness ahead!

Fig 6.4.3 Pop up message

Fig 6.4.3 gives the pop up message that the person is in sad mood this pop up message makes the person happy and also a voice based output

[6.5] FACIAL EXPRESSION:FEAR

Facial Expression

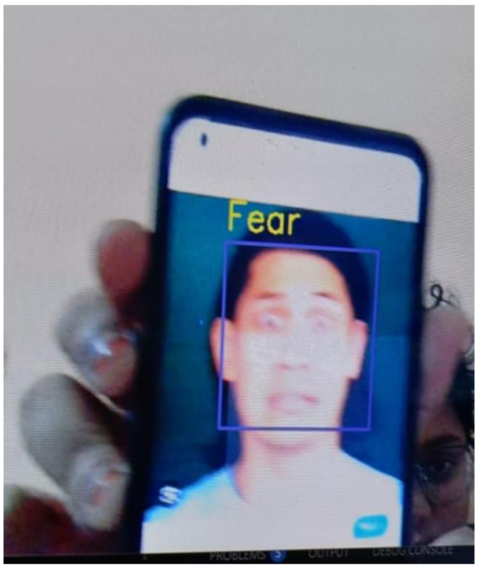


Fig 6.5.1 Fear exp

Fig 6.5.1 shows that the detected emotion is fear

Bargraph

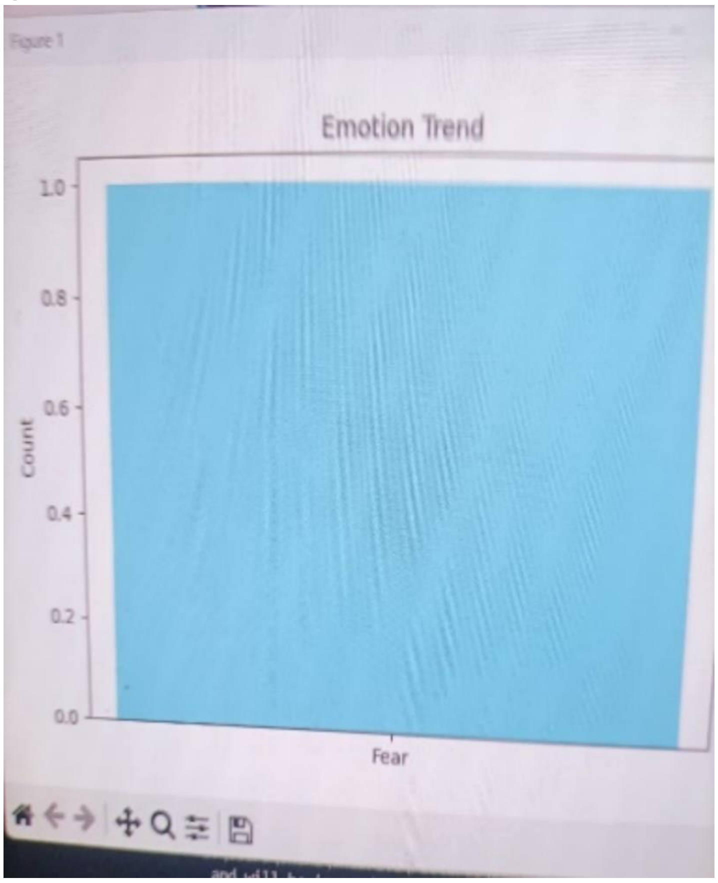
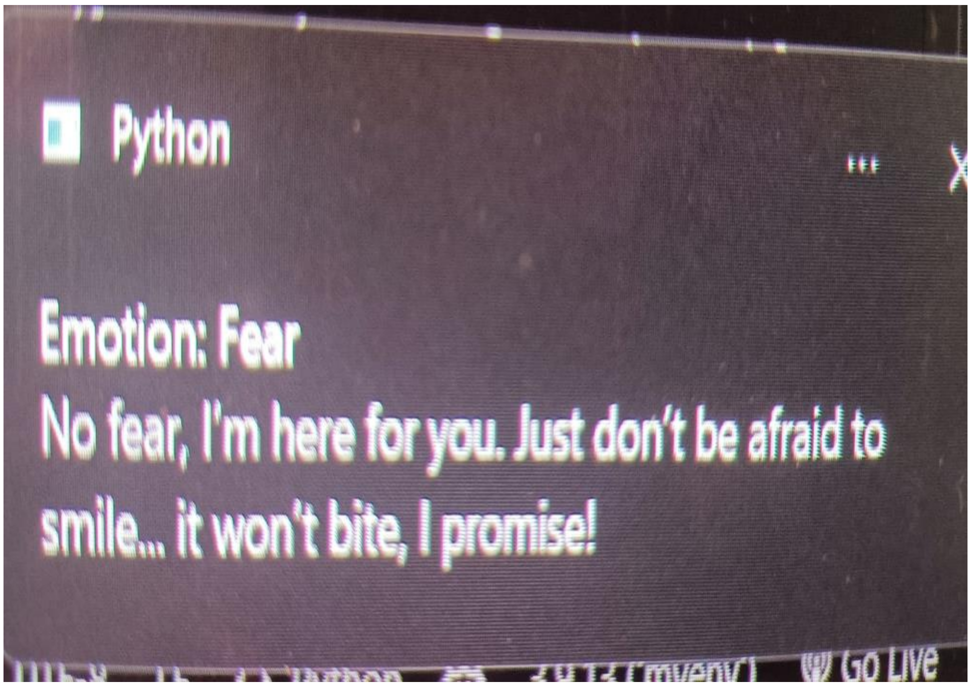


Fig 6.5.2 Bargraph

Fig 6.5.2 shows that the bargraph for the detected emotion.


Python

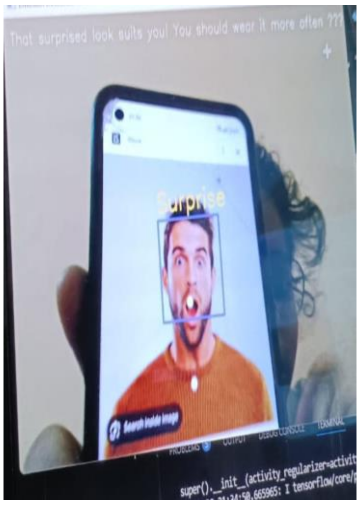
Emotion: Fear
No fear, I'm here for you. Just don't be afraid to smile... it won't bite, I promise!

Fig 6.5.3 Pop up message

Fig 6.5.3 gives the pop up message that displays that the person is frightened and boosts up the person with the voice based output.

[6] FACIAL EXPRESSION:SURPRISE

Facial Expression


That surprised look suits you! You should wear it more often ???
Surprise
Search inside image

```
super().__init__(activity_regularizer=activit
                  tensorflow/core/p
```

Fig 6.6.1 surprise exp

Fig 6.6.1 detects the emotion as surprise

Bargraph

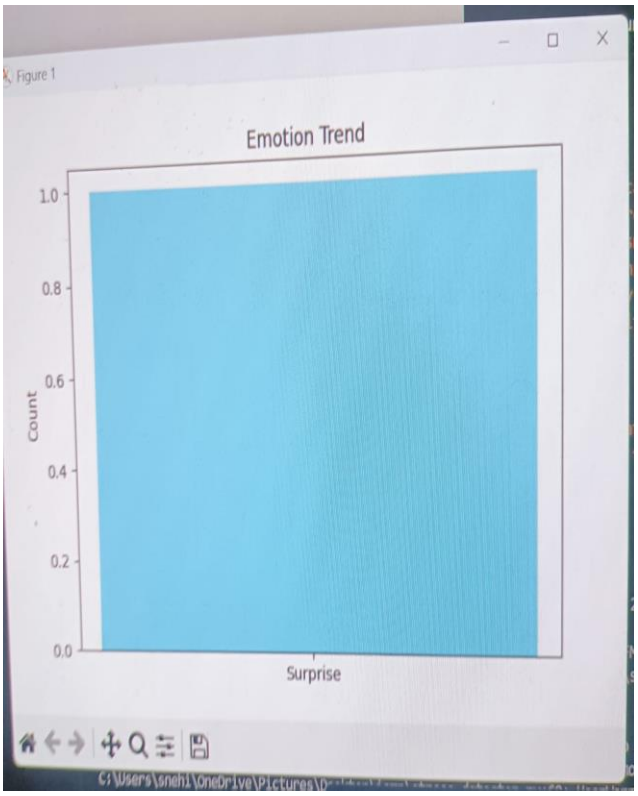
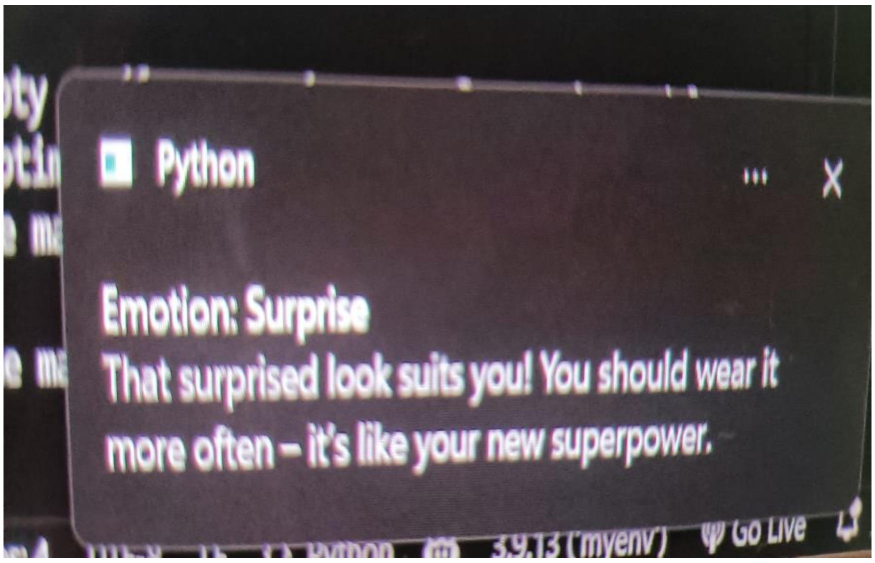


Fig 6.6.2 Bargraph

Fig 6.6.2 shows that the bargraph of the detected emotion surprise

Pop up message



Python

Emotion: Surprise
That surprised look suits you! You should wear it more often – it's like your new superpower.

Fig 6.6.3 Pop up message

Fig 6.6.3 displays a pop up message and a voice based suggestion for the surprise mood

[6.7] OVERALL RESULT

6.7.1 .When the happy and neutral is greater than all the other emotions

Bargraph

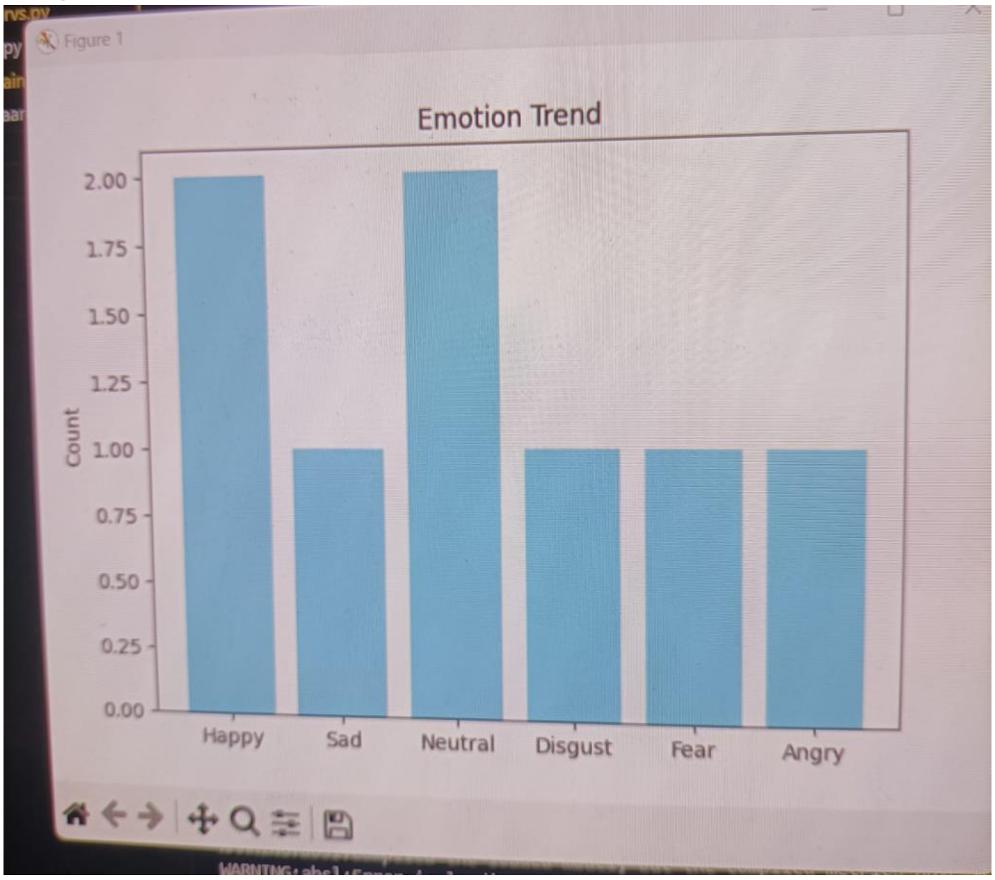
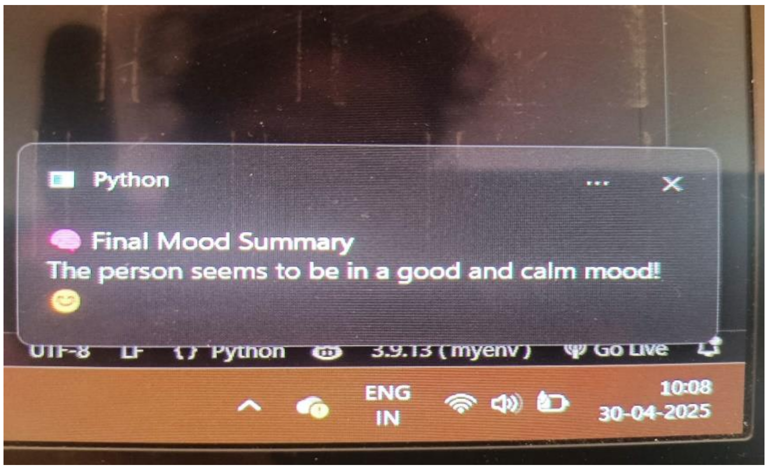


Fig 6.7.1 Bargraph of all the emotions

Fig 6.7.1 gives the bargraph of all the emotions that is neutral and happy dominates all other emotions.

Final mood Summary:



Python

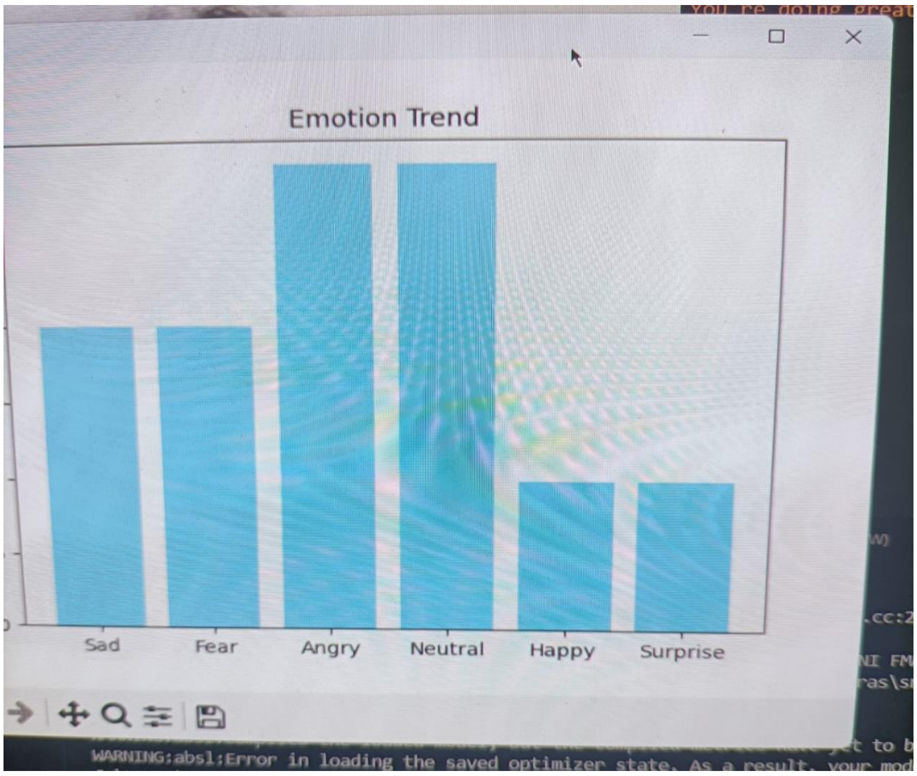
Final Mood Summary
The person seems to be in a good and calm mood!

Fig 6.7.2 Final Mood Summary

Fig 6.7.2 gives the final mood summary that the person is in a good mood and gives the voice based output to maintain the same.

6.8. When the other emotions like sad,angry,etc.. dominates the happy emotion

Bargraph

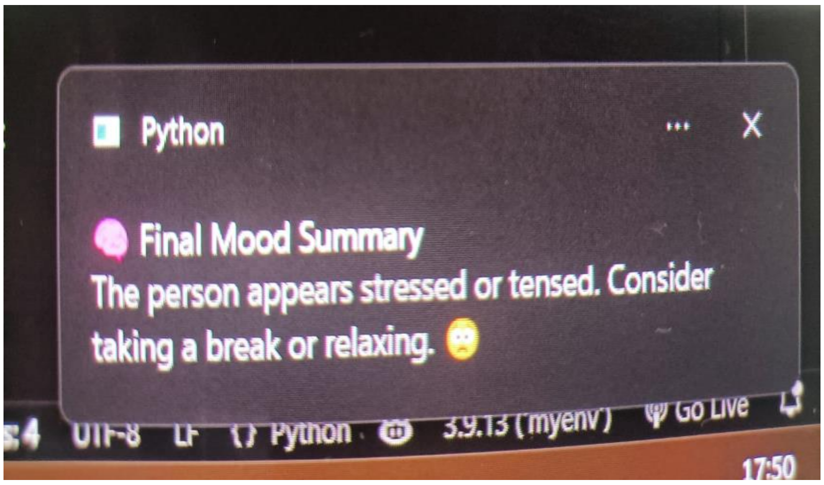


```
WARNING:absl:Error in loading the saved optimizer state. As a result, your mod
```

Fig 6.8.1 Bargraph of all emotions

Fig 6.8.1 shows that the bargraph of all the emotions and angry emotion dominates the happy emotion.

Final Mood Summary:


Python

Final Mood Summary
The person appears stressed or tensed. Consider taking a break or relaxing. 🙁

Fig 6.8.2 Final Mood Summary

Fig 6.8.2 gives the final mood summary that gives the person looks tensed and gives a voice based suggestion to take a break.

7. CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSION

The developed system efficiently detects user facial expressions in real-time and provides intelligent feedback through multiple interactive modes such as a live bar graph, voice responses, and pop-up notifications. These features not only enhance user engagement but also deliver a dynamic and intuitive experience. The inclusion of an exit GUI ensures user-friendly navigation and smooth session termination, while the live graph visualization enables continuous monitoring of emotional trends, giving users insight into their moment-to-moment emotional shifts.

Moreover, the system concludes each session with a final mood analysis that summarizes the user’s dominant emotional state, offering a meaningful overview of their emotional journey. This makes the tool highly applicable in areas like stress detection, mental wellness monitoring, and emotionally aware user interfaces. Its real-time capabilities and multi-modal feedback make it a valuable solution for mental health applications, personalized user interaction systems, and environments requiring adaptive emotional responses.

Overall, the system is intuitive, lightweight, and user-friendly—catering to individuals who may lack access to expensive health tracking devices or professional interventions.

7.2 FUTURE ENHANCEMENTS

To further enhance the functionality and user experience of the real-time facial emotion detection system, the following improvements can be considered:

1. **AddingEmotionIntensity**
   Utilize the model's output probabilities to display the strength or confidence level of each detected emotion. This will allow users to understand not just which emotion is present, but how intensely it is being experienced.

2. **Emotion-BasedRecommendations**

   Integrate an intelligent recommendation system that suggests music playlists, videos, or relaxation activities tailored to the user’s current emotional state, thereby enhancing emotional well-being.

3. **ExitGUIImprovements**

   Enhance the exit interface by including options such as saving session logs, exporting emotion graphs, or restarting a new session, giving users more control and flexibility.

4. **GraphSavingOptions**

   Implement automatic saving of emotion trend graphs at the end of each session, enabling users to track their emotional history over time for reflection or therapeutic review.

These enhancements will not only increase the usability and interactivity of the system but also make it a more engaging and personalized tool for emotional awareness and mental health support.

REFERENCES

[1] Barhoumi, C., & BenAyed, Y. Real-time face emotion recognition using deep learning and data augmentation. Artificial Intelligence Review, 58, 49 (2025).

[2] T. Winyangkun, N. Vanitchanant, V. Chouvatut, & B. Panyangam, "Real-Time Detection and Classification of Facial Emotions," 2023 15th International Conference on Knowledge and Smart Technology (KST), Phuket, Thailand, 2023, pp. 1–6. doi: 10.1109/KST57286.2023.10086866.

[3] M. Bhanupriya, N. Kirubakaran, & P. Jegadeeshwari, "EmotionTracker: Real-time Facial Emotion Detection with OpenCV and DeepFace," 2023 International Conference on Data Science, Agents & Artificial Intelligence (ICDSAAI), Chennai, India, 2023, pp. 1–4. doi: 10.1109/ICDSAAI59313.2023.10452452.

[4] D. Joshi, A. Dhok, A. Khandelwal, S. Kulkarni, & S. Mangrulkar, "Real Time Emotion Analysis (RTEA)," 2021 International Conference on Artificial Intelligence and Machine Vision (AIMV), Gandhinagar, India, 2021.

[5] Li, S., & Deng, W. (2020). Deep Facial Expression Recognition: A Survey. IEEE Transactions on Affective Computing, 13(3), 1195–1215.

[6] M. A. Ozdemir, B. Elagoz, A. Alaybeyoglu, R. Sadighzadeh, & A. Akan, "Real Time Emotion Recognition from Facial Expressions Using CNN Architecture," 2020 Medical Technologies Congress (TIPTEKNO), Izmir, Turkey, 2020.

[7] Lopes, A. T., de Aguiar, E., De Souza, A. F., & Oliveira-Santos, T. (2020). Facial Expression Recognition with CNNs: Coping with Few Data and the Training Sample Order. Pattern Recognition, 61, 610–628. doi: 10.1016/j.patcog.2016.07.026.

[8] Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2020). Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks. IEEE Signal Processing Letters 23(10), 1499–1503. doi: 10.1109/LSP.2016.2603342.

[9] Khorrami, P., Paine, T., & Huang, T. S. (2020). Do Deep Neural Networks Learn Facial Action Units When Doing Expression Recognition? In Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW), 19–27.

[10] Raut, R., Suryawanshi, S., & Choudhary, D. (2022). Real-Time Emotion Detection Using CNN and FER2013 Dataset. In International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE), Vol. 10, Issue 6.