**Chapter 1**

**Introduction**

* 1. **Introduction:**

People tend to express their emotions, mainly by their facial expressions. Music has always been known to alter the mood of an individual. Capturing and recognizing the emotion being voiced by a person and displaying appropriate songs matching the one's mood and can increasingly calm the mind of a user and overall end up giving a pleasing effect. The project aims to capture the emotion expressed by a person through facial expressions. A music player is designed to capture human emotion through the web camera interface available on computing systems. The software captures the image of the user and then with the help of image segmentation and image processing techniques extracts features from the face of a target human being and tries to detect the emotion that the person is trying to express.

The project aims to lighten the mood of the user, by playing songs that match the requirements of the user by capturing the image of the user. Since ancient times the best form of expression analysis known to humankind is facial expression recognition. The best possible way in which people tend to analyze or conclude the emotion or the feeling or the thoughts that another person is trying to express is by facial expression. In some cases, mood alteration may also help in overcoming situations like depression and sadness. With the aid of expression analysis, many health risks can be avoided, and there can be steps taken that help brings the mood of a user to a better stage.

* 1. **Motivation:**

The primary goal of an emotion-based music player is to provide user-preferred music with emotion identification by displaying songs based on the user's feelings. In the current system, the user must manually choose the tunes because songs generated are more than one. Thus, the motivation to build such a system that could make easy task for user and give machine the ability to predict the emotions based on the features provided by the data.

* 1. **Purpose:**

Capturing and recognizing the emotion being voiced by a person and displaying appropriate songs matching the one's mood and can increasingly calm the mind of a user and overall end up giving a pleasing effect. The project aims to capture the emotion expressed by a person through facial expressions.

**1.4 Objective:**

1. To develop a computer-based system that can accurately predict human emotions based on their facial expressions.

2. Recommending the best playlist for user based on his current emotion.

3. To identify and classify facial expressions and associate them with corresponding emotions such as happiness, sadness, anger, fear, and neutral.

4. To create a system that can accurately predict emotions in a range of applications, including healthcare and entertainment.

**Chapter 2**

**Literature Survey**

**2.1 Existing System Study:**

* Existing system stated that it is very time-consuming and difficult to create and manage a large playlist.
* The current mental state of the person is provided by facial expressions and the existing system was not able to capture or recognize the gestures or hand signs.

**2.1.1 Referenced Journal/Conference Papers:**

<https://doi.org/10.1109/CSITSS.2017.8447743>

<https://doi.org/10.1109/WiSPNET.2017.8299738>

**2.1.2 Elaborate on Existing System Applications/Examples:**

In existing system, the human emotions are connected with the music player. It has thus helped in reducing the efforts of user in creating and managing playlist and providing an excellent experience to the music listeners by bringing them the most suitable song according to the user’s his/her current expression. The system uses an algorithm that gives a list of songs from the user’s playlist in accordance with the user’s emotion. The algorithm which was designed was focused on having less computational time and thus reduces the cost included in using various hardware. The main idea was to segregate the emotions into five categories i.e., Joy, sad, anger, surprise, and fear.

**2.1.3 Limitations in Existing System:**

● The system still is not able to record all the emotions correctly due to the less availability of the images in the image dataset being used.

● The image that is fed into the classifier should be taken in a well-lit atmosphere for the classifier to give accurate results.

● The quality of the image should be at least higher than 320p for the classifier to predict the emotion of the user accurately.

**2.2 Problem Statement:**

Develop a system that presents a cross-platform music player, which recommends music based on the real-time mood of the user through a web camera using Machine Learning Algorithms.

**2.3 Proposed System Study:**

* The proposed system benefits us to present interaction between the user and the music player.
* The purpose of the system is to capture the face properly with the camera. Captured images are fed into the Convolutional Neural Network which predicts the emotion. Then emotion derived from the captured image is used to get a playlist of songs.
* The main aim of our proposed system is to provide a music playlist automatically to change the user's moods, which can be happy, sad, natural, or surprised. The proposed system detects the emotions, if the topic features a negative emotion, then a selected playlist is going to be presented that contains the foremost suitable sorts of music that will enhance the mood of the person positively. Music recommendation based on facial emotion recognition contains four modules.

1. Real-Time Capture: In this module, the system is to capture the face of the user correctly
2. Face Recognition: Here it will take the user's face as input. The convolutional neural network is programmed to evaluate the features of the user image.
3. Emotion Detection: In this section extraction of the features of the user image is done to detect the emotion and depending on the user's emotions, the system will generate captions.
4. Music Recommendation: Song is suggested by the recommendation module to the user by mapping their emotions to the mood type of the song.

Capture Face Feature Emotion

Image Detection Extraction Detection

Input

Play the Recommend Online song Music

Song songs Database Classifier

Output

**Fig. 2.1. Block diagram of the proposed system.**

**2.4 Feasibility Study:**

**2.4.1 Technical Feasibility:**

To assess the technical feasibility, the following aspects will be considered:

a. Facial Expression Analysis: Evaluate the availability and performance of existing facial expression analysis algorithms and libraries.

b. Emotion Recognition Model: Assess the accuracy and efficiency of different machine learning models for emotion prediction.

c. Music Recommendation Techniques: Explore various recommendation algorithms and determine their suitability for the proposed system.

d. Infrastructure and Compatibility: Ensure that the required hardware and software infrastructure are available and compatible for system implementation.

**2.4.2 Financial Feasibility:**

The financial feasibility of the project will be evaluated by considering the following factors:

a. Cost Analysis: Assess the costs associated with data collection, model development, infrastructure setup, and system deployment.

b. Return on Investment (ROI): Estimate the potential revenue generation and assess whether the project can provide a satisfactory ROI.

**Chapter 3**

**Project Scope and Requirement Analysis**

**3.1 Project Scope:**

**3.1.1. Scope Description:**

There are various aspects of the application that can be modified to produce better results and a smoother overall experience for the user. Some of these that an alternative method, based on additional emotions which are excluded in our system as disgust and fear. This emotion included supporting the playing of music automatically. The scope within the system would style a mechanism that might be helpful in music therapy treatment and help the music therapist to treat the patients suffering from mental stress, anxiety, acute depression, and trauma. The current system does not perform well in extremely bad light conditions and poor camera resolution thereby provides an opportunity to add some functionality as a solution in the future.

**3.1.2. In-Scope:**

**1. Data Collection:** Collect a dataset of facial expressions captured through images or videos. Ensure the dataset represents a wide range of facial expressions relevant to music preference, such as happiness, sadness, excitement, etc.

**2. Facial Expression Recognition**: Develop or employ a facial expression recognition model to analyze the collected facial expressions. Train the model using machine learning or deep learning techniques to accurately identify different expressions.

**3. Music Recommendation Model**: Design and implement a music recommendation model that incorporates facial expression data. Determine the appropriate approach to link facial expressions to music preferences (e.g., mapping facial expressions to specific genres or emotions). Explore methods such as collaborative filtering or content-based filtering to recommend music based on facial expressions.

**4. Integration of Facial Expression Recognition and Recommendation Model:** Integrate the facial expression recognition model with the music recommendation model. Use the recognized facial expressions as input to generate personalized music recommendations.

**5. User Interface:** Create a user interface that captures facial expressions through a camera or uploaded images/videos. Display the recommended music based on the recognized facial expressions. Provide options for users to refine or customize the recommendations based on additional preferences.

**3.1.3 Out of Scope:**

**1. Hardware Development:** The project assumes the availability of hardware (e.g., cameras, image/video capture devices) to capture facial expressions. The development of custom hardware or devices is beyond the scope of this project.

**2. Real-time Facial Expression Recognition:** Real-time facial expression recognition can be a complex task requiring specialized hardware and techniques. This project focuses on analyzing pre-recorded facial expressions rather than real-time analysis.

**3. Extensive Facial Expression Dataset:** The project scope assumes the availability of a suitable dataset for facial expression recognition. The collection of a large-scale facial expression dataset is a separate task that falls outside the scope.

**4. Music Streaming Integration:** Integrating the music recommendation system with music streaming platforms or APIs (e.g., Spotify, Apple Music, You Tube) is not explicitly included in the project scope. The project assumes the availability of a music dataset for recommendation purposes.

**5. User Feedback and Rating System:** The scope does not cover the implementation of a user feedback or rating system for the recommended music. Gathering user feedback and incorporating it into the recommendation process is considered an enhancement beyond the project scope.

**3.1.4 Deliverables:**

Facial expression recognition algorithm integrated into the system. Music recommendation engine based on predicted emotions. User interface (UI) for interacting with the system and viewing recommended music. Integration of camera or image input for real-time facial expression analysis. Machine learning or deep learning models for continuous improvement of emotion prediction. Test reports and optimization documentation. Final software application package ready for deployment.

* 1. **Requirement Gathering and Analysis:**

**3.3.1 Software Requirements:** -

a. Operating System: Windows 11

b. Platform: PyCharm Community Edition

c. Music Source: YouTube

d. Programming Language: Python 3.7 and above.

e. Technologies: Machine Learning, Deep Learning

f. Libraries: Keras, NumPy, TensorFlow, OpenCV, Media Pipe, Stream lit

**3.3.2 Hardware Requirements:**

a. RAM: 4 GB and above.

b. Processor: intel i5

c. HDD:50 GB and above.

d. Camera: minimum resolution of 720p (1280 x 720 pixels)

e. Peripherals: Mouse, Keyboard, Monitor.

**Chapter 4**

**Project Design and Modeling Details**

**4.1 Software Requirement Specification (SRS)**

**4.1.1. Functional requirements:**

**. Emotion Recognition:** Develop a facial expression analysis module that accurately recognizes emotions such as happiness, sadness, anger, etc., from users' facial expressions in real-time.

**. Music Recommendation Engine:** Build a recommendation engine that generates music suggestions based on the predicted emotions and users' profiles, taking into account their preferred genres, listening history, and feedback.

**. User Interface:** Design an intuitive and visually appealing interface that allows users to interact with the system, view recommended music, and provide feedback.

**. Personalization:** Implement algorithms that dynamically adapt recommendations based on users' feedback and continuously refine their profiles.

**. Integration with Music Platforms**: Enable seamless integration with popular music streaming platforms, allowing users to listen to recommended music directly or provide APIs to access the system's music library.

**4.1.2 Non-Functional Requirements:**

**.**  **Accuracy and Reliability:** The emotion recognition model should have a high level of accuracy and reliability in identifying users' emotions from facial expressions.

**.**  **Real-time Processing:** The system should be able to analyze facial expressions and generate recommendations in real-time, providing a smooth and seamless user experience.

**.**  **Scalability:** The system should be capable of handling a growing user base and processing large volumes of data without compromising performance.

**. User Experience:** Focus on creating a user-friendly interface that is visually appealing, easy to navigate, and provides an enjoyable music discovery experience.

**4.1.3 Technical Requirements:**

**.**  **Facial Expression Analysis:** Utilize computer vision and deep learning techniques to develop an emotion recognition model that can accurately analyze facial expressions.

**. Machine Learning Algorithms:** Employ appropriate machine learning algorithms for recommendation generation, considering collaborative filtering, content-based filtering, or hybrid approaches.

**. APIs and Integration:** Develop APIs or integration mechanisms to seamlessly connect with music streaming platforms or access the system's music library.

**. Infrastructure:** Ensure the availability of sufficient computational resources, including servers, storage, and network infrastructure, to support real-time processing and handle the expected user load.

**4.2 System Modeling and Design**

**4.2.1 Class Diagram**

play song

pass category

Take input process and identify category

play ()

fetch song and play

PLAYER

receive input ()

identify category ()

SYSTEM

gesture image

gesture performance

gesture input

USER

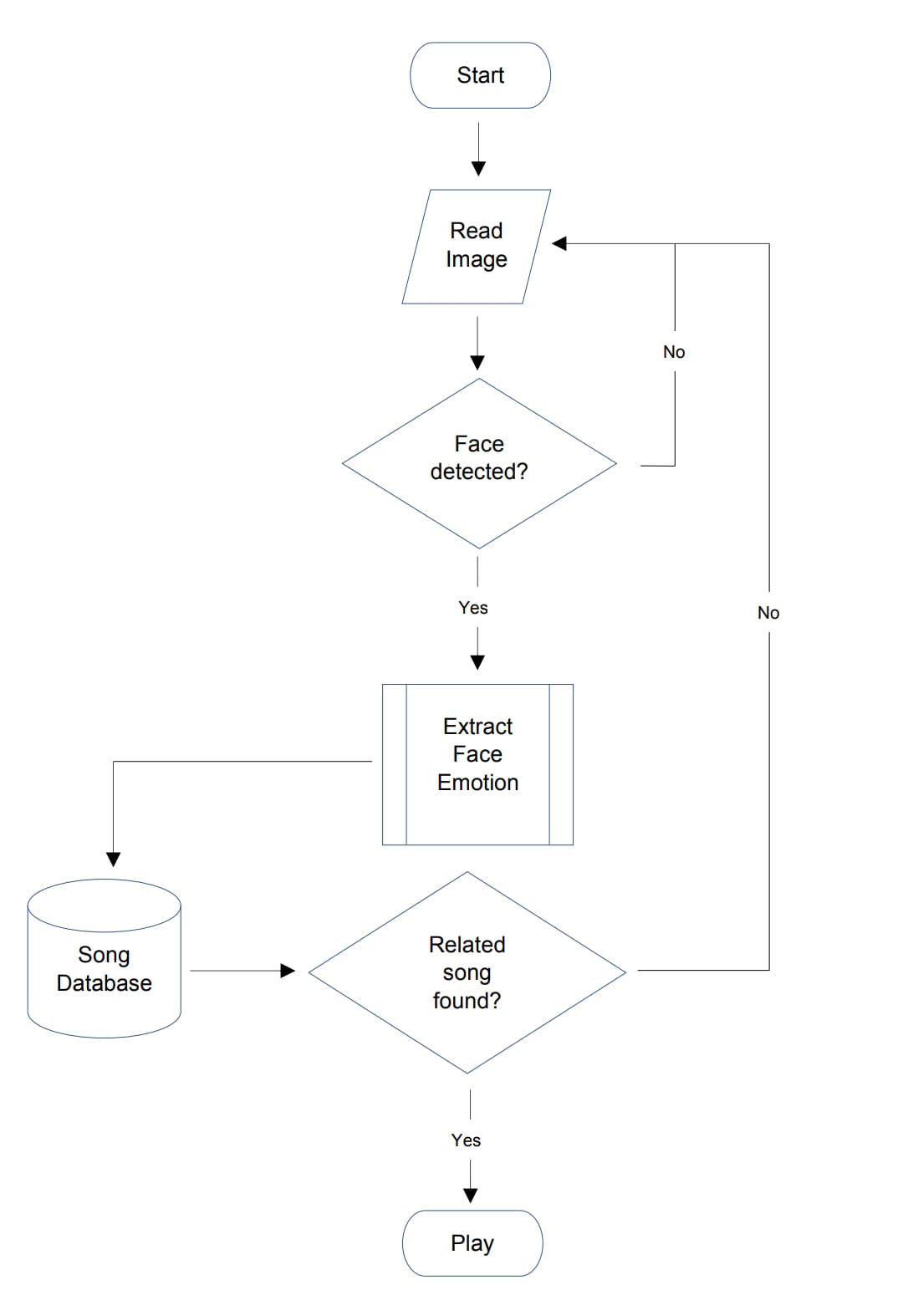
**Fig. 4.1 Class Diagram**

1) Firstly, the user class will have the parameters data that is fetched from the user input i.e language of song and the name of the singer. It has two methods gesture input which capture the face and gesture of the user and another method is gesture performance which will process the image.

2) There is another class i.e., system and it will process the image captured by the webcam. It has also the method which will identify the emotion and classify it.

3) Then after recognizing the emotion the player class will apply the algorithm and displays the song list which are best suited for the current emotion.

**4.2.5 Activity Diagram:**





**Fig 4.2. Activity Diagram**



1) Activity diagram describes the flow of control in a system. It gives an idea of how the system will work when executed.

2) In the above activity diagram there are various activities such as read image i.e. capture image, detect face, emotion detection, passing input parameters etc. which are used for the flow of data.

3) When the user will give the input for song, the song language, and his favorite singer then the system will take that input and enable webcam to capture the image.

4) The image captured by the webcam is send to system which will identify the emotion.

5) After applying algorithm, the emotion will be classified into one of the emotions like sad, happy, anger, neutral, etc.

6) After that the songs playlist will be displayed.

**4.2.4 Sequence Diagram:**

Music

Preprocessing

Face

User

1: Input ()

2: Web cam ()

3: Face Detection ()

4: Feature Extraction ()

5: Emotion Recognition ()

6: Emotion Classification ()

7: Song recommendation ()

8: Final process ()

**Fig 4.3 Sequence Diagram**

* **User:** User will give the input as the language of music and the singer’s name.
* **Face:** When the user will apply the language and name of the singer then automatically the webcam will be enabled.
* **Preprocessing:** When the webcam captures the face of user and the emotion is predicted by the machine.
* **Music**: After emotion recognition the user must press the ‘Recommend me song” button

and the playlist according to the emotion of user will be displayed.

**4.2.3 Data Flow Diagram:**

**4.2.3.1 Data flow diagram level-0:**

Music Recommendation

Dataset System Result

Using CNN

**Fig 4.4 Data flow diagram level-0**

1)It is also known as a context diagram. It is designed to be an abstraction view, showing the system as a single process with its relationship to external entities.

2)It represents the entire system as a single bubble with input as dataset and output as music indicated by incoming and outgoing arrows through Music Recommendation System using CNN.

**4.2.3.2 Data flow diagram level-1**

Initialization

Face detected

Emotion Recognition

Feature extraction

Face

Webcam

Song Recommendation

**Fig 4.5 Data flow diagram level-1**

1)In 1-level DFD, the context diagram is decomposed into multiple processes.

2)In this level, we highlight the main functions of the system and breakdown the high-level process of 0- level DFD into subprocesses.

3) The subprocesses are face detection, emotion recognition, emotion classification and song recommendation.

**4.2.2 Use Case Diagram**

**Fig 4.6 Use Case Diagram**

1)In this use case diagram there is one actor i.e., user. It has association with the webcam which is one of the use cases and before user must enter the language for song and the singer’s name.

2) System will capture the image of the user and processing will be done on the image i.e., feature extraction, segmentation and the clear image will be recognized with the emotion.

3) Features scaling is done after that we are selecting the algorithm and the emotion will be classified by the system.

4) After that the song playlist will be displayed according to the current captured emotion of the user.

**4.3 System Modules**

**Data collection:** Facial expression data would need to be collected from individuals while they listen to different types of music. This data could include images or videos of their facial expressions.

**Emotion labeling:** Experts or participants could manually label the collected data with emotions such as happiness, sadness, anger, etc. This step helps create a labeled dataset for training the system.

**Feature extraction:** Facial expression features would need to be extracted from the collected data. This process could involve techniques such as facial landmark detection, tracking facial movements, or using deep learning-based approaches like Convolutional Neural Networks (CNNs).

**Machine learning model training:** The extracted features and corresponding emotional labels would be used to train a machine learning model.

**Music recommendation:** Once the model is trained, it can be used to predict the emotional state of a person based on their facial expressions. The system can then recommend music that is known to evoke similar emotions. It is important to note that the accuracy and effectiveness of such a system would depend on the quality and diversity of the training data, the performance of the emotion prediction model, and the availability of a comprehensive music dataset with emotion labels.

**4.4 System Architecture**

Happy

Preprocessing

Sad

Feature Extraction

Input

Image

Angry

Recognition

Classification

Neutral

Image

Image Processing

Classified Label

**Fig. 4.7 Layout of System Architecture**

**Chapter 5**

**Implementation and Coding**

**5.1 Algorithms**

**5.1.1 Algorithm for Data Collection**

**Step-1**: Import the necessary libraries: mediapipe, numpy, and cv2 (OpenCV).

**Step-2**: Create a VideoCapture object to access the webcam.

**Step-3**: Prompt the user to enter a name for the data.

**Step-4**: Initialize the MediaPipe holistic and hands solutions.

**Step-5**: Create an empty list X to store the extracted data and initialize a variable data\_size to keep track of the number of data samples.

**Step-6**: Start an infinite loop to continuously read frames from the webcam.

**Step-7**: Read a frame from the webcam using cap.read() and flip it horizontally using cv2.flip().

**Step-8**: Process the frame using the holistic solution (holis.process()), converting the frame from BGR to RGB.

**Step-9:** If facial landmarks are detected (res.face\_landmarks is not None), iterate over each landmark, calculate the relative coordinates with respect to the second landmark (res.face\_landmarks.landmark[1]), and append the relative coordinates (i.x - res.face\_landmarks.landmark[1].x and i.y - res.face\_landmarks.landmark[1].y) to the lst list.

**Step-10**: If left hand landmarks are detected (res.left\_hand\_landmarks is not None), iterate over each landmark, calculate the relative coordinates with respect to the eighth landmark (res.left\_hand\_landmarks.landmark[8]), and append the relative coordinates to the lst list. Otherwise, append zeros (0.0) to lst for 42 times (indicating no hand landmarks).

**Step-11**: If right hand landmarks are detected (res.right\_hand\_landmarks is not None), iterate over each landmark, calculate the relative coordinates with respect to the eighth landmark (res.right\_hand\_landmarks.landmark[8]), and append the relative coordinates to the lst list. Otherwise, append zeros (0.0) to lst for 42 times (indicating no hand landmarks).

**Step-12**: Append the lst list to the X list and increment data\_size by 1.

**Step-1:** Use the drawing utility to draw the facial landmarks, left hand landmarks, and right hand landmarks on the frame.

**Step-13**: Display the frame with cv2.imshow() and draw the data\_size on the frame using cv2.putText().

**Step-14**: If the user presses the Esc key (key code 27) or the number of data samples (data\_size) exceeds 99, exit the loop.

**Step-15**: Close all OpenCV windows and release the webcam.

**Step-16**: Save the X list as a NumPy array in a file with the provided name using np.save().

**Step-17**: Print the shape of the NumPy array to indicate the number of samples and features.

**5.1.2 Algorithm for Data Training**

**Step-1:** Import the necessary libraries:

os: Provides functions for interacting with the operating system.

numpy (np alias): A library for numerical computing with arrays and matrices.

cv2: OpenCV library for image processing.

tensorflow.keras.utils: Utilities from Keras for converting labels to categorical format.

keras.layers: Layers from Keras for building the model.

keras.models : Model class from Keras for creating the neural network model.

**Step-2:** Initialize some variables:

is\_init: A flag to indicate if the initialization has been done.

size: The initial size of the data.

label: A list to store the labels.

dictionary: A dictionary to map labels to integers.

c: A counter variable.

**Step-3:** Iterate through the files in the current directory:

**Step-4**: Check if the file has a ".npy" extension and is not a "labels.npy" file.

**Step-5**: If it is the first file encountered, load it into X and initialize size and y.

**Step-6:** For subsequent files, concatenate the data to X and update y with the corresponding labels.

**Step-7**: Add the label to the label list and update the d

..ictionary with the label mapping.

**Step-8**: Increment the counter c.

**Step-9**: Convert the labels in y to integers using the dictionary.

**5.1.3 Algorithm for Testing**

**Step-1:** Import the necessary libraries: OpenCV (cv2), NumPy, Mediapipe (mp), and Keras.

**Step-2:** Load the trained model from the "model.h5" file using load\_model function from Keras.

**Step-3**: Load the labels from the "labels.npy" file using NumPy's load function and assign it to the variable label.

**Step-4**: Create instances of the holistic and hands classes from Mediapipe.

**Step-5:** Initialize the video capture using cv2.VideoCapture(0) to capture video from the default camera.

**Step-6**: Start an infinite loop to continuously process frames from the video.

**Step-7:** Read a frame from the video capture using cap.read().

**Step-8:** Flip the frame horizontally using cv2.flip to correct the orientation.

**Step-9:** Convert the frame from BGR to RGB color space using cv2.cvtColor.

**Step-10:** Process the frame using the holistic instance to extract face landmarks and hand landmarks.

**Step-11:** If face landmarks are detected (res.face\_landmarks is not None):

**Step-12:** Iterate over each face landmark and calculate the normalized x and y distances from the second face landmark.

**Step-13:** Append the calculated distances to the lst list.

**Step-14:** If left hand landmarks are detected (res.left\_hand\_landmarks is not None):

**Step-15:** Iterate over each left hand landmark and calculate the normalized x and y distances from the eighth landmark.

**Step-16:** Append the calculated distances to the lst list.

**Step-17:** If left hand landmarks are not detected, append zeros to the lst list.

**Step-18:** If right hand landmarks are detected (res.right\_hand\_landmarks is not None):

**Step-19:** Iterate over each right hand landmark and calculate the normalized x and y distances from the eighth landmark.

**Step-20:** Append the calculated distances to the lst list.

**Step-21:** If right hand landmarks are not detected, append zeros to the lst list.

**Step-22:** Reshape the lst list into a 2D NumPy array with shape (1, -1).

**Step-23:** Use the loaded model to predict the label by passing the lst array through model.predict.

**Step-24:** Get the predicted label by finding the index of the maximum value in the predicted array and indexing the label array with it.

**Step-25:** Print the predicted label.

**Step-26:** Draw the face landmarks, left hand landmarks, and right hand landmarks on the frame using drawing.draw\_landmarks.

**Step-27:** Display the frame in a window named "window" using cv2.imshow.

**Step-28:** If the 'Esc' key is pressed (key code 27), destroy all windows, release the video capture, and break the loop.

**5.1.3 Algorithm for Music Recommendation System**

**Step-1:** Import the necessary libraries:

streamlit: The Streamlit framework for creating web applications.

streamlit\_webrtc: A Streamlit extension for WebRTC video streaming.

av: The PyAV library for handling video frames.

cv2: OpenCV library for image processing.

numpy: A library for numerical computing with arrays and matrices.

mediapipe: The MediaPipe library for hand and facial landmark detection.

keras.models: Model class from Keras for loading a pre-trained model.

webbrowser: A module for opening URLs in a web browser.

**Step-2**: Load the pre-trained Keras model from "model.h5" and the labels from "labels.npy".

**Step-3:** Create an instance of the holistic class from MediaPipe for holistic (face and hand) landmark detection.

**Step-4**: Set up the Streamlit application header and check if an emotion is already saved in the "emotion.npy" file.

**Step-5**: Define a class EmotionProcessor that receives frames from the video stream.

**Step-6**: Inside the recv method of EmotionProcessor, perform the following steps:

**Step-7**: Flip the frame horizontally.

**Step-8**: Process the frame using the holistic model to detect facial and hand landmarks.

**Step-9**: Extract the landmark coordinates and normalize them.

**Step-10**: If face landmarks are detected, perform the following:

**Step-11**: For each face landmark, calculate the relative position to the second landmark and append the coordinates to a list.

**Step-12**: If left hand landmarks are detected, calculate the relative position to the eighth landmark and append the coordinates to the list.

**Step-13**: If no left hand landmarks are detected, append zeros to the list to maintain consistency.

**Step-14**: If right hand landmarks are detected, calculate the relative position to the eighth landmark and append the coordinates to the list.

**Step-15**: If no right hand landmarks are detected, append zeros to the list to maintain consistency.

**Step-16**: Convert the list to a numpy array and reshape it to match the input shape of the model.

**Step-17**: Perform inference using the loaded model on the input data and retrieve the predicted emotion label.

**Step-18**: Draw the predicted emotion label on the frame using OpenCV.

**Step-19**: Save the predicted emotion label to "emotion.npy" for later use.

**Step-20**: Draw the detected landmarks on the frame using MediaPipe drawing utilities.

**Step-21:** Create an instance of the EmotionProcessor class as the video processor for the WebRTC streamer.

**Step-22**: Get the user input for language and singer.

**Step-23**: If both language and singer are provided and the emotion is not already detected, start the WebRTC streamer with the EmotionProcessor as the video processor.

**Step-24**: Create a button labeled "Recommend me songs".

**Step-25**: If the button is clicked, check if an emotion is already detected. If not, display a warning message to capture the emotion first.

**Step-26:** If an emotion is detected, open a web browser with a YouTube search query based on the language, emotion, and singer inputs.

**Step-27**: Save an empty string to "emotion.npy" to reset the emotion detection.

**Step-28**: Update the session state to indicate that the emotion detection process is complete.

**Chapter 6**

**Testing Specification**

**6.Test Specification:**

**6.1 Fundamentals of Testing**

**Define the Test Goals**: Clearly identify the objectives of your testing. Determine what aspects of the music recommendation system you want to evaluate, such as the accuracy of emotion prediction, the effectiveness of music recommendations based on emotions, or the user experience.

**Test Data Collection**: Gather a diverse and representative dataset of facial expressions labeled with corresponding emotions. This dataset should cover a broad range of emotions, including happiness, sadness, anger, surprise, etc. Ensure that the dataset is balanced and contains enough samples for each emotion.

**Preprocessing** **and Feature Extraction**: Preprocess the facial expression data to enhance its quality and remove noise. Extract relevant features from the facial expressions, such as facial landmarks, facial action units, or statistical representations like histograms or feature vectors.

**Model Training**: Train a machine learning or deep learning model to predict emotions from facial expressions. Utilize appropriate algorithms such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or ensemble models. Split the dataset into training and validation sets and fine-tune the model to achieve optimal performance.

**6.2 Test plan of the project:**

**6.2.1 Facial Expression Recognition:**

Test Case 1: Verify that the system accurately recognizes facial expressions from live video input.

Test Case 2: Test the system's accuracy in different lighting conditions and angles.

Test Case 3: Evaluate the system's ability to recognize facial expressions of different individuals.

**6.2.2 Emotion Prediction:**

Test Case 4: Verify that the system correctly predicts the user's emotion based on the recognized facial expression.

Test Case 5: Test the system's performance with various facial expressions and emotions.

Test Case 6: Evaluate the system's ability to handle ambiguous or mixed emotions.

**6.2.3 Music Recommendation:**

Test Case 7: Verify that the system recommends music that aligns with the predicted emotion.

Test Case 8: Test the system's ability to recommend a variety of music genres based on different emotions.

Test Case 9: Evaluate the system's responsiveness in recommendingmusic in real-time.

**6.2.4 System Performance:**

Test Case 10: Evaluate the system's performance under high user load and concurrent requests.

Test Case 11: Measure the system's response time for facial expression recognition and emotion prediction.

Test Case 12: Assess the system's reliability and stability over an extended period.

**Chapter 7**

**Conclusion and Future Scope**

**7.1 Conclusion:**

Through review of the literature tells that there are many approaches to implement Music Recommender System. A study of methods proposed by previous scientists and developers was done. Based on the findings, the objectives of our system were fixed. As the power and advantages of AI-powered applications are trending, our project will be a state-of-the-art trending technology utilization. In this system, we provide an overview of how music can affect the user's mood and how to choose the right music tracks to improve the user's moods. The implemented system can detect the user's emotions. The emotions that the system can detect were happy, sad, angry, neutral, or surprised. After determining the user’s emotion, the proposed system provided the user with a playlist that contains music matches that detected the mood. Processing a huge dataset is memory as well as CPU intensive. This will make development more challenging and attractive. The motive is to create this application in the cheapest possible way and to create it under a standardized device. Our music recommendation system based on facial emotion recognition will reduce the efforts of users in creating and managing playlists.

**7.2 Future Scope:**

This system, although completely functioning, does have scope for improvement in the future. There are various aspects of the application that can be modified to produce better results and a smoother overall experience for the user. Some of these that an alternative method, based on additional emotions which are excluded in our system as disgust and fear. This emotion included supporting the playing of music automatically. The future scope within the system would style a mechanism that might be helpful in music therapy treatment and help the music therapist to treat the patients suffering from mental stress, anxiety, acute depression, and trauma. The current system does not perform well in extremely bad light conditions and poor camera resolution thereby provides an opportunity to add some functionality as a solution in the future.

**References**

**Journal Papers:**

1. Shloka Gilda, Husain Zafar, Chintan Soni, Kshitija Waghurdekar, Smart music player integrating facial emotion recognition and music mood recommendation, Department of Computer Engineering, Pune Institute of Computer Technology, Pune, India, (IEEE),2017.
2. Vivek JD, Gokilavani A, Kavitha S, Lakshmanan S and Karthik S 2017 A novel emotion recognition-based mind and soul-relaxing system In2017 International Conference on Innovations in Information, Embedded and Communication Systems 1-5 IEEE.
3. Preema J.S, Rajashree, Sahana M, Savitri H, Review on facial expression-based music player, International Journal of Engineering Research & Technology (IJERT), ISSN-2278-0181, Volume 6, Issue 15, 2018.
4. Ramya Ramanathan, Radha Kumaran, Ram Rohan R, Rajat Gupta, and Vishalakshi Prabhu, an intelligent music player based on emotion recognition, 2nd IEEE International Conference on Computational Systems and Information Technology for Sustainable Solutions 2017.
5. Gupte A, NagaNarayanan A and Krishnan M Emotion Based Music Player-Beats International Journal of Advanced Engineering Research and Science 3236854.

**Web References:**

* 1. <https://www.ijraset.com/research-paper/music-recommendation-system-using-facial-expression-recognition>.
  2. <https://doi.org/10.1109/CSITSS.2017.8447743>.

**Books:**

1. Object-orientated Modelling & Design: - James Rambaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen. (PHI)
2. The Unified Modelling Language User Guide: - Grady Booch, James Rambaugh, Lvar Jacobson.