

# **Facial Emotion Detection Using Convolutional Neural Network and OpenCV**

**A Project report submitted to**

**FACULTY OF**

**COMPUTER SCIENCE AND ENGINEERING**

**In partial fulfillment of the requirements for the award of the Degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**By**

**Guvvala Namitha-20HT1A0531**

**Under the esteemed guidance of**

**Mr.G. RAMACHANDRA RAO, M.Tech, (Ph.D)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CHALAPATHI INSTITUTE OF TECHNOLOGY**

**(Approved by A.I.C.T.E, Affiliated To JNTUK, Kakinada)**

**A.R.Nagar, Mothadaka, Guntur, Andhra Pradesh, India**

**2023-24**

# **CHALAPATHI INSTITUTE OF TECHNOLOGY**

**(Approved by A.I.C.T.E, Affiliated to JNTUK, Kakinada)**

A.R. Nagar, Mothadaka, Guntur-522016, Andhra Pradesh, India



## **CERTIFICATE**

This is to certify that the project work entitled as **“Facial Emotion Detection using Convolutional Neural Network and OpenCV”** submitted by Guvvala Namitha in partial fulfillment for the award of the Degree of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafied work carried out under my guidance and supervision.

### **GUIDE**

G. Ramachandra Rao  
Assistant Professor

### **HEAD OF THE DEPARTMENT**

G. Ramachandra Rao  
Assistant Professor & HOD

**Internal Examiner**

**External Examiner**

## ACKNOWLEDGMENT

I express my sincere thanks to our beloved Chairman **Sri. Y.V. ANJANEYULU** garu for providing support and stimulating environment for developing the project.

I express deep sense of reverence and profound gratitude to **Dr.K.KIRAN KUMAR, Ph.D**, Principal for providing me the great support in carrying out the project.

It plunges me in exhilaration in taking privilege in expressing our heartfelt gratitude to **Mr.G. RAMACHANDRA RAO, MTech, Ph.D**, Assistant Professor & Head of the Department for providing for his constant encouragement, suggestions and abundant support, every facility throughout my project.

It gives us immense pleasure to express a deep sense of gratitude to my guide **Mr.G. RAMACHANDRA RAO, M.Tech, Ph.D**, for whole hearted and invaluable guidance throughout the my project. Without her/his sustained and sincere effort, this project would not have taken this shape.

Finally, I would like to thank all the faculty members and the non-teaching staff of the Department of Computer Science Engineering for their direct or indirect support for helping us in completion of this project

By  
Guvvala Namitha

# INDEX

<b>TITLE</b>	<b>PAGE NO</b>
<b>ABSTRACT</b>	<b>I</b>
<b>LIST OF FIGURES</b>	<b>II</b>
<b>LIST OF TABLES</b>	<b>III</b>
<b>CHAPTER1 : INTRODUCTION</b>	<b>1</b>
<b>CHAPTER 2 : LITERATURE SURVEY</b>	<b>3</b>
<b>CHAPTER 3 : SYSTEM ANALYSIS</b>	<b>8</b>
3.1 EXISTING SYSTEM	8
3.2 PROPOSED SYSTEM	8
3.3 FUNCTIONAL REQUIREMENTS	9
3.3.1 SOFTWARE REQUIREMENTS	9
3.4 NON FUNCTIONAL REQUIREMENTS	11
3.5 PROCESS MODEL USED WITH JUSTIFICATI	11
3.5.1 SDLC(UMBRELLA MODEL)	11
3.5.2 STAGES IN SDLC	12
3.5.2.a PLANNING	13
3.5.2.b REQUIREMENT GATHERING	13
3.5.2.c ANALYSIS	15
3.5.2.d DESIGNING	16
3.5.2.e CODING	17
3.5.2.f TESTING	18
3.5.2.g MAINTENANCE	19
3.6 SOFTWARE REQUIREMENT SPECIFICATION	19
3.7 EXTERNAL INTERFACE REQUIREMENTS	21
3.7.1 SYSTEM REQUIREMENTS	21
3.7.1.a HARDWARE REQUIREMENTS	21
3.7.1.b SOFTWARE REQUIREMENTS	21
<b>CHAPTER4 : SYSTEM DESIGN</b>	<b>22</b>
4.1 CLASS	22
4.2 DATA FLOW	25
<b>CHAPTER 5 : IMPLEMENTATION</b>	<b>26</b>
5.1 PYTHON	26

5.2 CONVOLUTIONAL NEURAL NETWORK	28
5.3 SAMPLE CODE	30
<b>CHAPTER 6 : TESTING</b>	37
6.1 IMPLEMENTATION AND TESTING	37
6.1.1 IMPLEMENTATION	37
6.2 TESTING	37
6.2.1 SYSTEM TESTING	38
6.2.2 MODULE TESTING	38
6.2.3 INTEGRATION TESTING	38
6.2.4 ACCEPTANCE TESTING	39
<b>CHAPTER 7 : RESULTS</b>	40
7.1 EXECUTION	40
7.1.1 LIST OF FILES	40
7.1.2 INPUT COMMAND	41
7.1.3 OUTPUT SCREEN	42
7.2 REAL-TIME FED	43
7.2.1 HAPPY	43
7.2.2 NEUTRAL	44
7.3 FED ON IMAGE	45
7.3.1 ANGRY	46
7.3.2 FEAR	47
7.3.3 SURPRISE	48
7.3.4 DISGUST	49
7.4 FED ON VIDEO	50
7.4.1 SAD	51
7.4.2 ANGRY	52
7.4.3 HAPPY	53
7.4.4 SAD	54
7.4.5 SURPRISE	55
<b>8 : CONCLUSION</b>	56
<b>9 : REFERENCES</b>	57

## **ABSTRACT**

Image processing is a method to convert an image into digital form and perform some operations on it. This is done to enhance the image or to extract useful information from it. Facial expressions are nonverbal form of communication. There are eight universal facial expressions which include: neutral, happy, sadness, anger, contempt, disgust, fear, and surprise. So it is very important to detect these emotions on the face. A monitoring system for elderly people which is based on technology involving recognizing emotions from video image. Our proposed system includes video analysis technology which includes the data from video is adopted to realize monitoring elders' living conditions in real time. In case of emergency, the system will alert their relatives and children by sending a message.

Keywords-facial emotion recognition; Local binary. Pattern Histogram (LBPH)algorithm, Convolutional Neural Networks (CNN)

## **LIST OF FIGURES**

<b>FIGURE NO</b>	<b>FIGURE NAME</b>	<b>PAGE NO</b>
3.3.1	SYSTEM ARCHITECTURE	10
3.5.1	SDLC(UMBRELLA MODEL)	12
3.5.2.b	REQUIREMENTS GATHERING	13
3.5.2.c	ANALYSIS	15
3.5.2.d	DESIGN	16
3.5.2.e	CODING	17
3.5.2.f	TESTING	18
4.2	DATA FLOW	25
7.1.1	LIST OF FILES	40
7.1.2	INPUT COMMAND	41
7.1.3	OUTPUT SCREEN	42
7.2.1	HAPPY	43
7.2.2	NEUTRAL	44
7.3	FED ON IMAGE	45
7.3.1	ANGRY	46
7.3.2	FEAR	47
7.3.3	SURPRISE	48
7.3.4	DISGUST	49
7.4	FED ON VIDEO	50
7.4.1	SAD	51
7.4.2	ANGRY	52
7.4.3	HAPPY	53
7.4.4	SAD	54
7.4.5	SURPRISE	55

## **LIST OF TABLES**

<b>TABLE NO</b>	<b>TABLE NAME</b>	<b>PAGE NO</b>
6.2.4	Acceptance	39



# CHAPTER 1

## INTRODUCTION

The rapid development of artificial intelligence (AI) (Ustun et al., 2021), big data (Wang J. et al., 2020), and Blockchain technology (Lv et al., 2021a) has changed the social structure, talent demand, as well as the form of social education. Through traditional data acquisition methods, people need a lot of time and energy to collect data, which hinders the convergence and synchronization of art developed to a certain extent. With the rapid development of information technology, in the internet era, art information and exhibition information around the world can be known by global users in a very short time. People can easily collect landscape materials from all over the world online without leaving home. Meanwhile, the computer gave birth to new art forms and ideas. Through the computer, the scope of traditional art expression has also expanded from oil painting, traditional Chinese painting, printmaking, sculpture, watercolor, etc. to animation art, image art, photoelectric art, etc. through the sketches drawn by artists. The change in technologies has caused the innovation of the learning environment, and the intelligent learning environment with the Internet of Things (IoT) technology as the core has begun to attract extensive attention from people. In the intelligent learning environment, teachers carry out teaching activities online through the Internet, and learners can easily acquire and learn knowledge through the network. However, psychological research has shown that various emotions generated in the learning process can affect the learning effect. For example, positive emotions such as happiness and satisfaction generated in the learning process are conducive to raising learning interest, while emotions such as boredom and anxiety can hinder the cognitive process. In traditional teaching activities, face-to-face communication between teachers and students enables learners to maintain a positive interest in learning at any time. In contrast, it is difficult for teachers and students to feel each other's emotional state in time due to the constraints of time and space in the intelligent learning environment. Correspondingly, it is urgent to seek out an effective way to combine knowledge transmission with emotional communication in the current intelligent learning environment. Human emotions are complex and simple. As a smart species currently in the dominant position on the earth, humans can express emotions through various methods, such as voice, text, and facial expressions (Jain et al., 2019; Khare and Bajaj, 2020; Guanghui and Xiaoping, 2021). In the intelligent learning environment, emotion

recognition of learners' images in class hours through computers and deep learning algorithms can facilitate timely monitoring of psychological and emotional states of learners. The emotion recognition through facial expression images requires high-quality cameras to capture facial images, resulting in high implementation cost. Therefore, the speech-based human emotion recognition method has gradually become the principal method to study human-computer emotion recognition. In the process of communication and expression, speech of humans not only contains semantic information, but also implies rich information like the speaker's emotion. Therefore, the research on emotion recognition based on human speech and image through computer and intelligent algorithms of deep learning is of great significance. Speech-based emotion recognition (Liu and Fu, 2021) has been using the method of acoustic statistical features since it was proposed in the 1980s. Until the 21st century, the fast-growing computer multimedia technology and the continuous breakthrough in the field of AI technologies have made great progress in speech-based emotion recognition. The traditional machine learning algorithms based on Gaussian mixture model (Tian et al., 2020), support vector machine (SVM) (Chuan et al., 2020), and artificial neural networks (Shao et al., 2020) have achieved brilliant results in speech-based emotion recognition tasks. However, the traditional machine learning algorithms have some defects in the accuracy of emotion recognition by speech and images. Improving the accuracy of emotion recognition by speech and images based on existing technologies is a critical goal of AI and deep learning algorithm.

## **CHAPTER 2**

### **LITERATURE SURVEY**

Yadan Lv in this paper implemented human emotions using machine learning algorithm. The parsed components help to detect the various types of the feature recognition technique; therefore, we do not need to add the additional feature for removing noise or any adjusting feature. The parsed technique is one of the most important techniques and a unique technique. Mehmood et al. in this paper implemented the optimal feature selection and deep learning ensemble methods for emotional recognition from human brain EEG sensors. This paper implements the EEG feature extraction and the feature selection methods based on the optimization of the face recognition technique. Four types of emotional classifications are involved, namely, happy, calm, sad, and scared. The feature extraction is based on the optimal selected feature like the balanced one-way ANOVA technique, so it provides better accuracy in the emotional classification. Additional techniques like the arousal-valence space provide enhanced EEG recognition. Li et al. in this paper implemented the Reliable Crowd-sourcing and deep locality-preserving learning for expression recognition in the wild, for reducing the crowd-sourcing and the new locality loss layer preservation using a deep learning algorithm that is based on the RAFDB face recognition algorithm. Thus, the RAFDB expressed that the five different techniques, such as the calculation of the aged rat and the gender, the second step helps to identify the dimensional space of the image, and the third step helps to identify the two subsets. The first one contains seven types of emotions, and the second subset contains twelve types of emotions. The fourth one is identifying the accuracy, and the fifth one is classifying the images based on the input. Chen et al. in this paper implemented the Soft-max regression-based deep sparse auto-encoder network for facial emotion recognition in human-robot interaction. This paper implements the SRDSAN technique, which helps to reduce the distortion and identify the learning efficiency and dimensional complexity, whereas the DSAN helps with accurate feature extraction and the soft-max regression helps to classify the input signal. Babajee et al. in this paper implemented the identification of human expressions from facial expressions using a deep learning algorithm. This paper proposed the seven types of facial expressions recognition like happy, sad, etc., through deep learning using a convolutional neural network. Thus, this paper contains a dataset of 32,398 for collecting various types of

emotional recognition using the Facial Action Coding System (FACS). This paper only depends on the identification method not working as the optimization method. Hassouneh et al in this paper implemented the development of a real-time emotional recognition system using facial expressions and EEG based on machine learning and deep neural network methods. This paper implements the optical flow algorithm for identifying facial regression using virtual markers. Therefore, the system of the optical flow algorithm helps to physically challenge people since it recognizes less computational complexity.

Tan et al in this paper implemented the short-term emotion recognition and understanding based on the spiking neural network modeling of the spatio-temporal EEG patterns using neuro-sense. This paper implements the SNN technique for the very first time. It helps to identify the functions of the brain system. The EEG data are measured by using two techniques like arousal-valence space. The arousal-valence space consists of four types of columns, namely, high arousal, low arousal, and high and low valence space techniques. Satyanarayana et al in this paper implemented emotional recognition using deep learning and cloud access. Facial emotional recognition is one of the most important techniques in various applications. The deep learning algorithm plays a vital role in face recognition. Her thesis paper collects various types of emotions like sadness, happiness, calmness, and anger reactions. Therefore, this information goes through the python code, and then it creates its own IP address for every technique. Jayanthi et al in this paper implemented an integrated framework for emotional classifications using speech and static images with deep classifier fusion. Emotion recognition is one of the crucial techniques for identifying the stress level in a human being. The two factors that play a crucial role in the identification of the stress level in the human body, namely, emotion recognition and speech modulation.

This paper introduced the integral framework for the calculation of the stress thesis; therefore, this paper includes both emotional recognition and speech modulations in the form of static function, which helps to identify the mental state of human nature. Therefore, this result gives better accuracy when compared to the other algorithms. Li et al in this paper implemented the survey of deep facial expression recognition. Face expression recognition is considered one of the major challenges in the network system. The major challenge in facial expression recognition (FER) is based on the lack of training sets and the unrelatable expression variations. In the very

first case, the dataset is arranged in the allocation of the neural pipeline technique. This will help to reduce the challenging issues in the FER technique. Yadahalli et al in this paper implemented facial micro expression detection using a deep learning algorithm. This paper collects the eight layers of the dataset while using six types of emotions, namely, happiness, sadness, anger, fear, neutral, and surprised faces. The collected dataset contains the FER model; thus, this paper implies that the FER with a convolutional neural network enhances better accuracy, and the cleared output results in the multimodal facial expression using a single algorithm.

Yang et al in this paper implemented the three-class emotions recognition based on deep learning using a stacked auto-encoder. This paper implements that the EEG signal is measured with discrete entropy calculation methods. The auto-encoder technique in the deep learning algorithm provides better accuracy than the calculation methods in the encoding system. The emotions in this technique are evaluated as the alpha, beta, and gamma values. It gives better accuracy and the classification results by using a deep learning algorithm. Normally, the deep learning algorithm provides good results for the multiple classifications of emotional recognition.

Asaju et al in this paper implemented the temporal approach to facial emotional expression recognition. This paper implements the various types of emotional recognition in the human body through a deep learning algorithm using a convolutional neural network. The VGG-19 methods are used for feature extraction. Then the facial emotion recognition and the accurate mapping technique are carried out by using the BiL-STM architecture.

Therefore, the CNN- BiL-STM technique is used to evaluate better accuracy and good classification results with the help of the deep learning neural network. Therefore, the Denver Intensity of the Spontaneous Facial Action (DISFA) dataset is used to detect the happy, sad, angry, and neutral faces in the techniques, and then the dataset for the effective state in the E-Environment (DAiSEE) dataset helps to detect the confusion and the irritation. Sati et al in this paper implemented face detection and recognition and face emotion recognition through NVIDIA Jetson Nano, in this paper implements both face recognition and face emotional detection, traditional to the present facial emotional identifications are one of the most challenging techniques, by adding some features in this technique helps to provide the better accuracy and the classification results. The ANN technique helps to identify and recognize facial emotions. Rajan et al in this paper implemented the novel deep learning model for

facial expression recognition based on maximum boosted CNN and LSTM and proposed a slightly different model with the boosted FER method, at very first preprocessing methods helps to reduce the noise in the given input function and reduces the dimensionality space function. And then the dual CNN technique is applied, and it becomes more and more boosted. Therefore, this paper finally shows that the combination of the LSTM and MBCNN makes it possible to produce highly accurate feature extraction and classification results. Mirsamadi et al in this paper implemented automatic speech emotion recognition using recurrent neural networks. This paper demonstrates that the RNN architecture for feature selection and the novel weighted time pooling strategy are involved in this paper, which helps to increase the salient features extraction.

The IEMOCAP is the new method being used for better classifications in emotional recognition. Therefore, the final results compare the IEMOCAP classifier with the traditional SVM-based SER using fixed design features. Domnich et al in this paper implemented the gender bias assessment in emotional recognition-based AI method, this paper implements the overview of the facial emotion recognition based on the artificial intelligence method, The SASE- FE dataset is collected based on the gender basis. Therefore, this dataset is further classified into two types depending on the male and female categories; therefore, each group consists of three neural networks. This process is carried out with the testing and the training phase because some groups are ready to work and some are not available for instant work, and then this work will be split up into three different ways, such as the whole work collection and both female and male data are individually split. Therefore, this function makes us think that the result might be accurate and perfect.

Ekundayo et al in this paper implemented the multilabel convolution neural network for facial expression recognition and ordinal intensity estimation, as there are many functions that work with the emotional recognition technique such as FER, but none of them is apt for the perfect multiclass emotional classification technique. This paper implements the multilabel convolutional neural network. This multiclass emotional classification leads to interclass variation. This problem will be overcome by using enhanced ML-CNN with the Binary Cross Entropy (BCE) loss and the loss from an island. The VGG-16 helps to overcome the fitting process in this technique; therefore, this paper implements the Multilabel Kernel Nearest Neighbor and the MLARAM for feature extraction and the classification is done using a chain

classifier. Wang et al in this paper implemented the recently advanced technique in deep learning. This paper implements the four-category model for deep learning. The first category consists of deep architectures and convolutional neural networks. The deep neural networks are majorly convinced by the deep learning model. It is one of the most important functions in the machine learning algorithm. It plays a crucial role in the data accuracy and the classification contains both linear and nonlinear specific functions. The convolutional neural network has three most crucial layers, namely, the convolutional neural layer, the pooling layer, and the fully connected layer. The convolution layer is the very first layer that adds some filters to reduce the noise and the dimensional space in the filter. The pooling layer in the CNN helps to reduce the over-fitting problem. Then the fully connected layer is arranged after the convolutional layer and the pooling layer. Therefore, it removes the inaccurate data in the function. In this paper Gnana et al implemented the literature review for the feature selection of high-dimensional data. The simplest way of the feature selection method in the data is set of all data are sent to the statistical measurement approach, therefore this helps to select the feature selection approach. There are four types of approach involved in the feature selection methods.

## **CHAPTER 3**

### **SYSTEM ANALYSIS**

#### **3.1 EXISTING SYSTEM**

Facial emotions are important factors in human communication that help to understand the intentions of others. In general, people infer the emotional state of other people, such as joy, sadness and anger, using facial expressions. Facial expressions are one of the main information channels in interpersonal communication. Therefore, it is natural that facial emotion research has gained a lot of attention over the past decade with applications in perceptual and cognitive sciences . Interest in automatic Facial Emotion Recognition (FER) has also been increasing recently with the rapid development of Artificial Intelligent (AI) techniques. They are now used in many applications and their exposure to humans is increasing. To improve Human Computer Interaction (HCI) and make it more natural, machines must be provided with the capability to understand the surrounding environment, especially the intentions of humans. Machines can capture their environment state through cameras and sensors.

##### **Disadvantages of Existing System:**

1. Less Accuracy
2. Time taken process

#### **3.2 PROPOSED SYSTEM**

In this project, we use a deep learning technique called Convolutional Neural Networks(CNNs) to establish a classification model that combines feature extraction with classification to detect the facial emotions. To develop a facial expression recognition system. To experiment machine learning algorithm in computer vision fields. To detect emotion thus facilitating Intelligent Human-Computer Interaction. With the goal to improve the process of facial sentiment analysis systems, a classification mechanism is proposed using a CNN architecture. The goals of this project are to establish a model that can classify 7 basic emotions: happy, sad, surprise, angry, disgust, neutral, and fear.

##### **Advantages of Proposed System:**

1. High picture quality improves the effectiveness of facial recognition
2. Even low-resolution photographs are usable with the suggested method



3. Higher accuracy while being more computationally efficient.

### **Modules Information:**

To implement this project author has used Convolutional Neural Network algorithm.

This project consists of following modules

1. Upload Image: Using this module, we will upload the image or video to application
2. Preprocess image or video: the Images can be pre-processed and apply the CNN algorithm and OpenCV to detect the emotion.
3. Run Algorithms: using above images or video we will train all deep learning algorithms and then detect the emotion from the human images or video

## **3.3 FUNCTIONAL REQUIREMENTS:**

### **3.3.1 SOFTWARE REQUIREMENTS:**

#### **System Attributes:**

1. filename
2. X, Y
3. Images or video
4. classifier
5. X\_train, X\_test, y\_train, y\_test

#### **Data base Requirements:**

No need

#### **Usecase:**

Use cases - Use cases describe the interaction between the system and external users that leads to achieving particular goals.

1. Upload Dataset
2. Preprocess Image or video
3. Run Algorithms

**User Stories:** A smart grid is an electricity network enabling a two-way flow of electricity and data with digital communications technology enabling to detect, react and pro-act to changes in usage and multiple issues. Smart grids have self-healing capabilities and enable electricity customers to become active participants. In some smart cities electricity consumption, opening or closing doors etc are managed by this smart grid. Some malicious user can attack this smart grid system to spread or inject

false information and smart grid will get executed based on provide false information which can cause huge financial loss

**Prototype:**

python 3.7.0flask==2.1.0

OpenCV-python==4.5.1.48

keras==2.3.1

tensorflow==1.14.0

protobuf==3.16.0

h5py==2.10.0

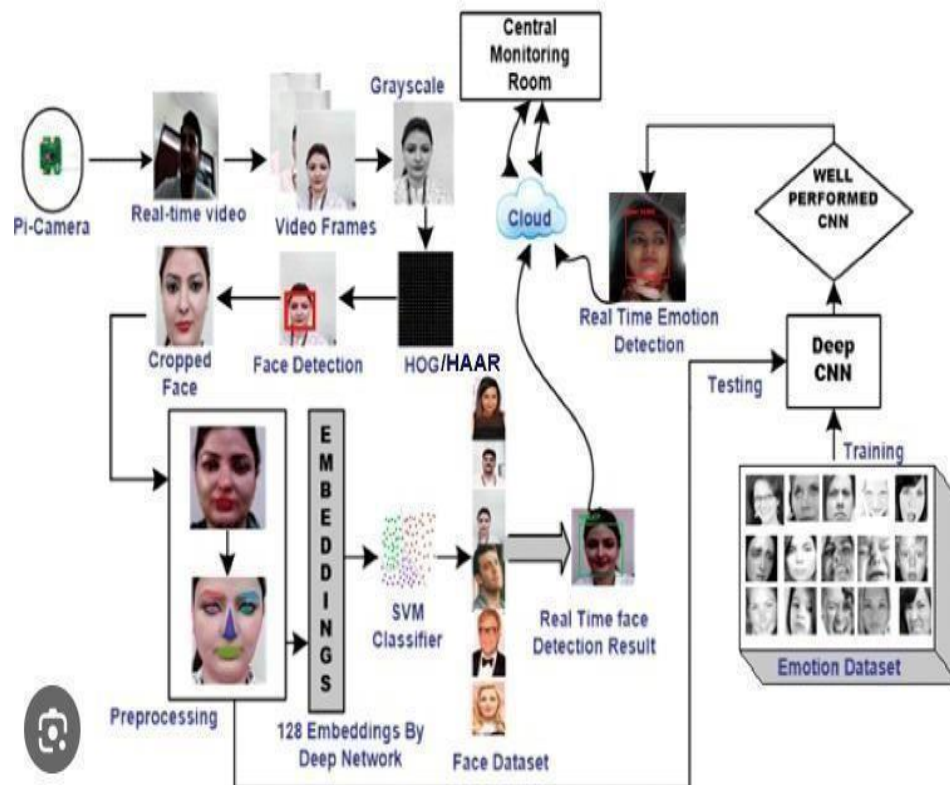
sklearn-extensions==0.0.2

scikit-learn==0.22.2.

post1Numpy

Pandas

**Models and Diagrams:**



**Fig 3.3.1 SYSTEM ARCHITECTURE**

### 3.4 NON-FUNCTIONAL REQUIREMENT

**Usability:** Usability is a quality attribute that assesses how easy user interfaces are to use. The word "usability" also refers to methods for improving ease-of-use during the design process. (how it was handle entire project easy)

**Security:** the quality or state of being secure: such as. a : freedom from danger : safety. b : freedom from fear or anxiety. c : freedom from the prospect of being laid off job security.

**Readability:** Readability is the ease with which a reader can understand a written text.

**Performance:** the execution of an action. : something accomplished : deed, feat. : the fulfillment of a claim, promise, or request : implementation. 3. : the action of representing a character in a play.

**Availability:** the quality or state of being available trying to improve the availability of affordable housing. 2 : an available person or thing.

**Scalability:** Scalability is the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processing demands.

### 3.5 PROCESS MODEL USED WITH JUSTIFICATION

#### 3.5.1 SDLC (UMBRELLA MODEL):

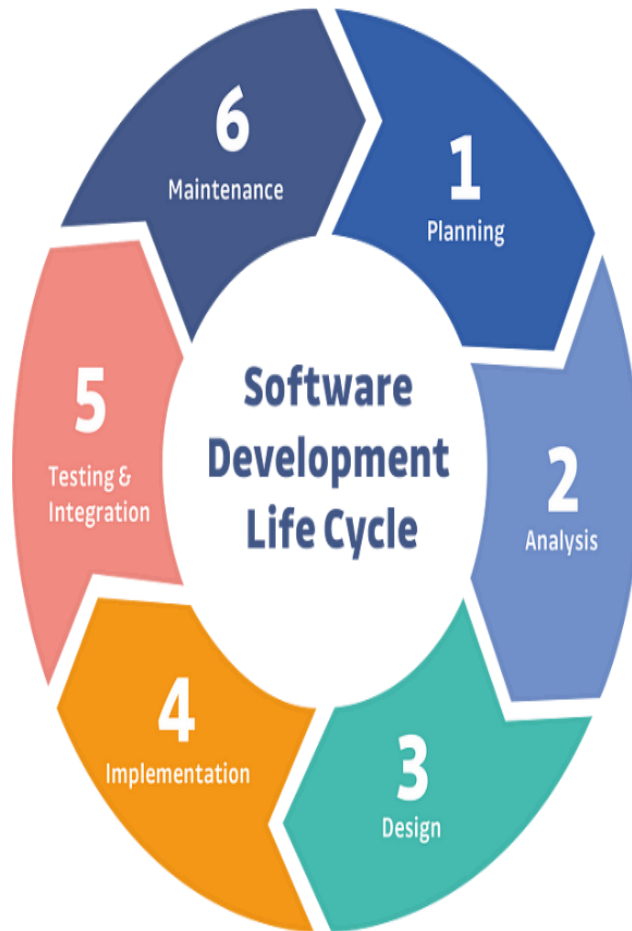
SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

The software development lifecycle (SDLC) is the cost-effective and time-efficient process that development teams use to design and build high-quality software. The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production and beyond. This methodology outlines a series of steps that divide the software development process into tasks you can assign, complete, and measure.

Software development can be challenging to manage due to changing requirements, technology upgrades, and cross-functional collaboration. The software development lifecycle (SDLC) methodology provides a systematic management framework with specific deliverables at every stage of the software development process. As a result, all stakeholders agree on software development goals and requirements upfront and also have a plan to achieve those goals.

Here are some benefits of SDLC:

- Increased visibility of the development process for all stakeholders involved
- Efficient estimation, planning, and scheduling
- Improved risk management and cost estimation
- Systematic software delivery and better customer satisfaction



**Fig 3.5.1 SDLC (UMBRELLA MODEL)**

### **3.5.2 STAGES IN SDLC**

The software development lifecycle (SDLC) outlines several tasks required to build a software application. The development process goes through several stages as developers add new features and fix bugs in the software.

A software development lifecycle (SDLC) model conceptually presents SDLC in an organized fashion to help organizations implement it. Different models arrange the SDLC phases in varying chronological order to optimize the development cycle.

The details of the SDLC process vary for different teams. However, we outline some common SDLC phases below.

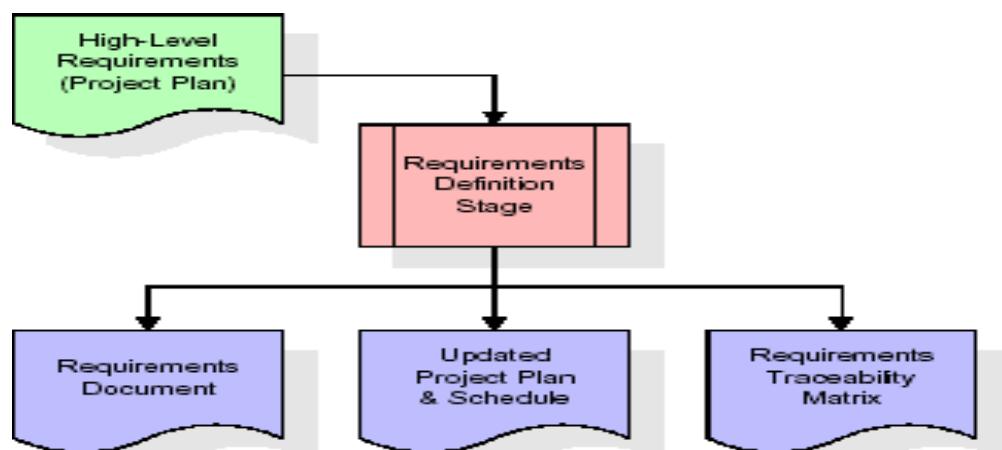
- a. Planning
- b. Requirement Gathering
- c. Analysis
- d. Designing
- e. Coding
- f. Testing
- g. Maintenance

### 3.5.2.a PLANNING

The first phase of the SDLC is the project planning stage where you are gathering business requirements from your clients . This phase is when you evaluate the feasibility of creating the product, revenue potential, the cost of production, the needs of the end-users, etc.

### 3.5.2.b. REQUIREMENTS GATHERING

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define. The initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports.definitions is termed a Requirement.



**Fig 3.5.2.b REQUIREMENTS GATHERING**

Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format,

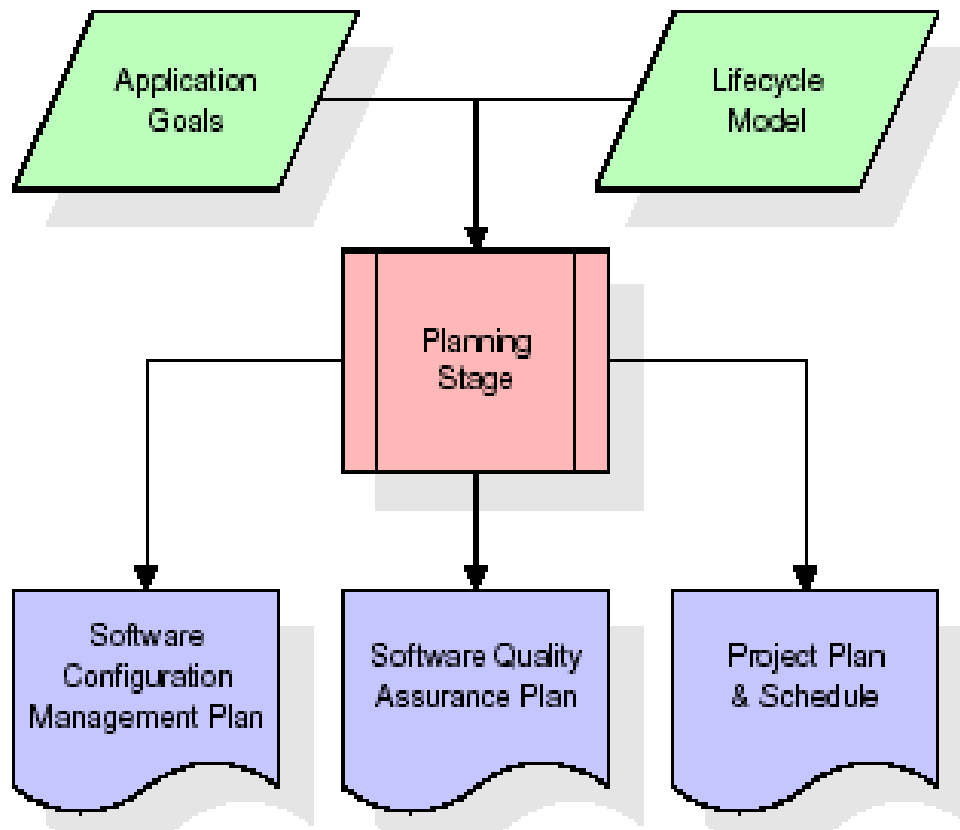
each requirement can be traced to a specific product goal, hence the term requirementstraceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

1. Feasibility study is all about identification of problems in a project.
2. No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
3. Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.
4. Note that detailed listings of database tables and fields are *not* included in the requirements document.

### 3.5.2.c ANALYSIS

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.

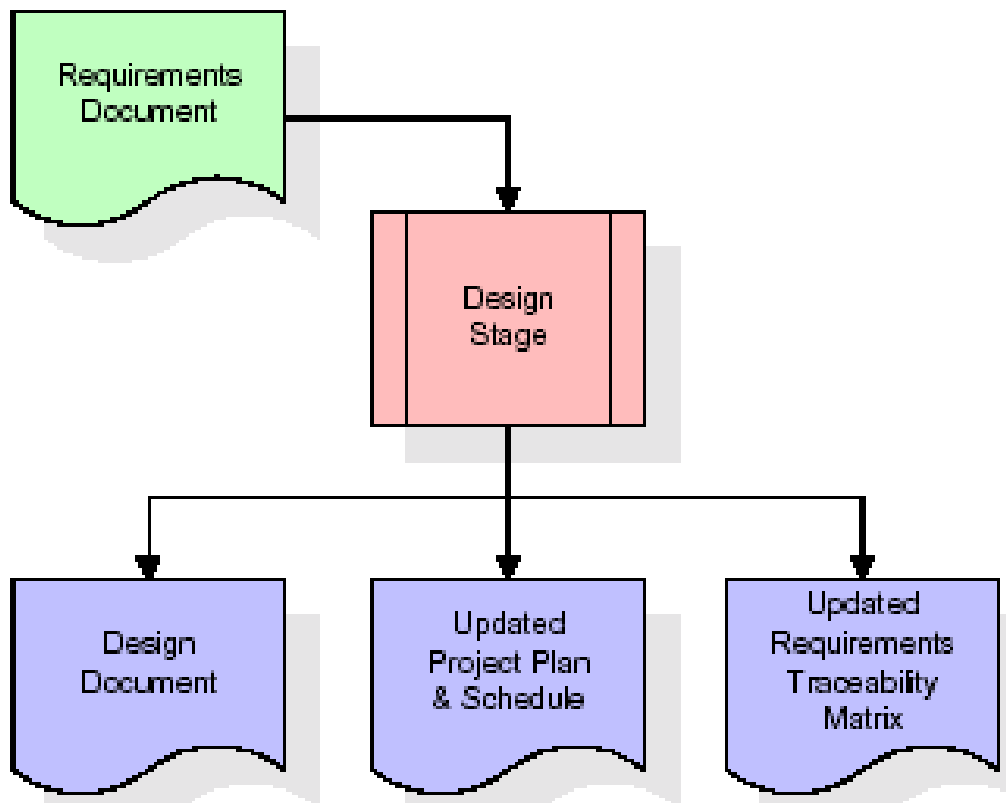


**Fig 3.5.1.c. ANALYSIS**

The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high-level estimates of effort for the out stages.

### 3.5.2.d DESIGNING

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



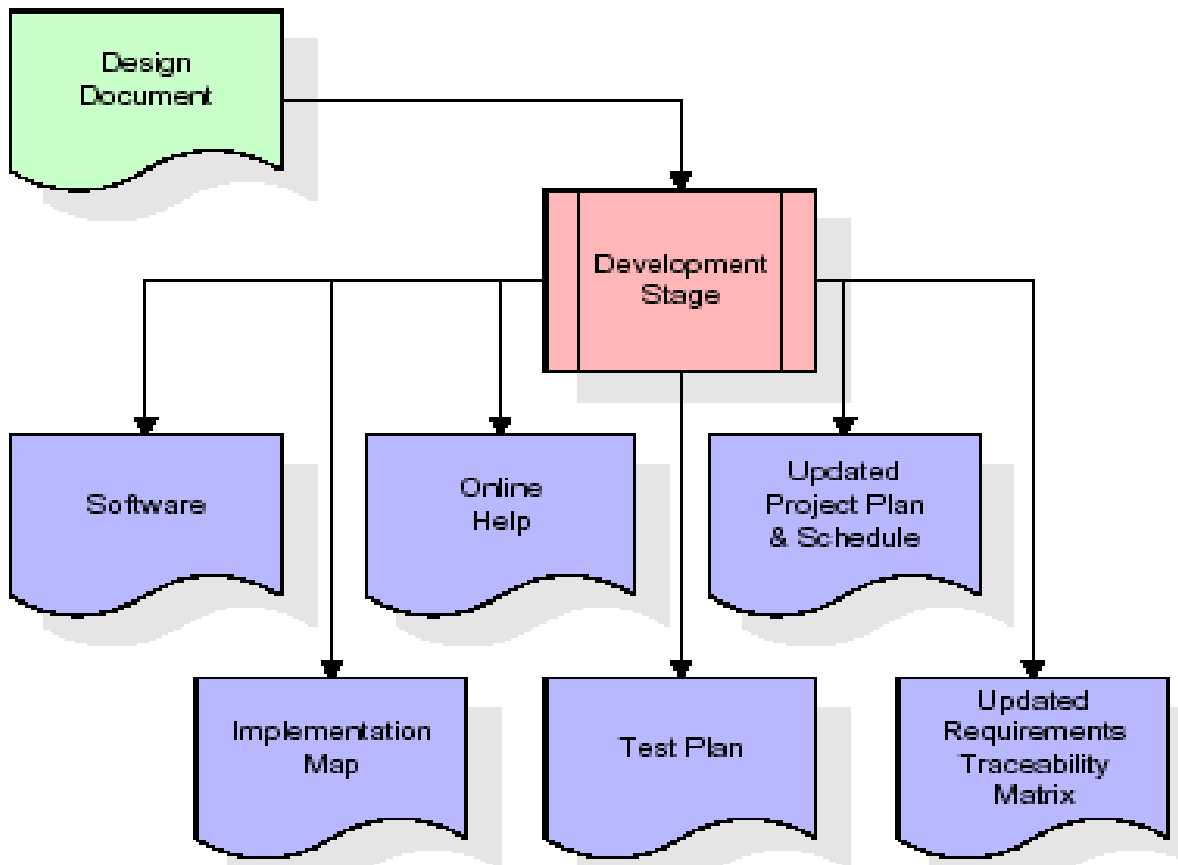
**Fig 3.5.1.d .DESIGNING**

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.



### 3.5.2.e CODING

development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



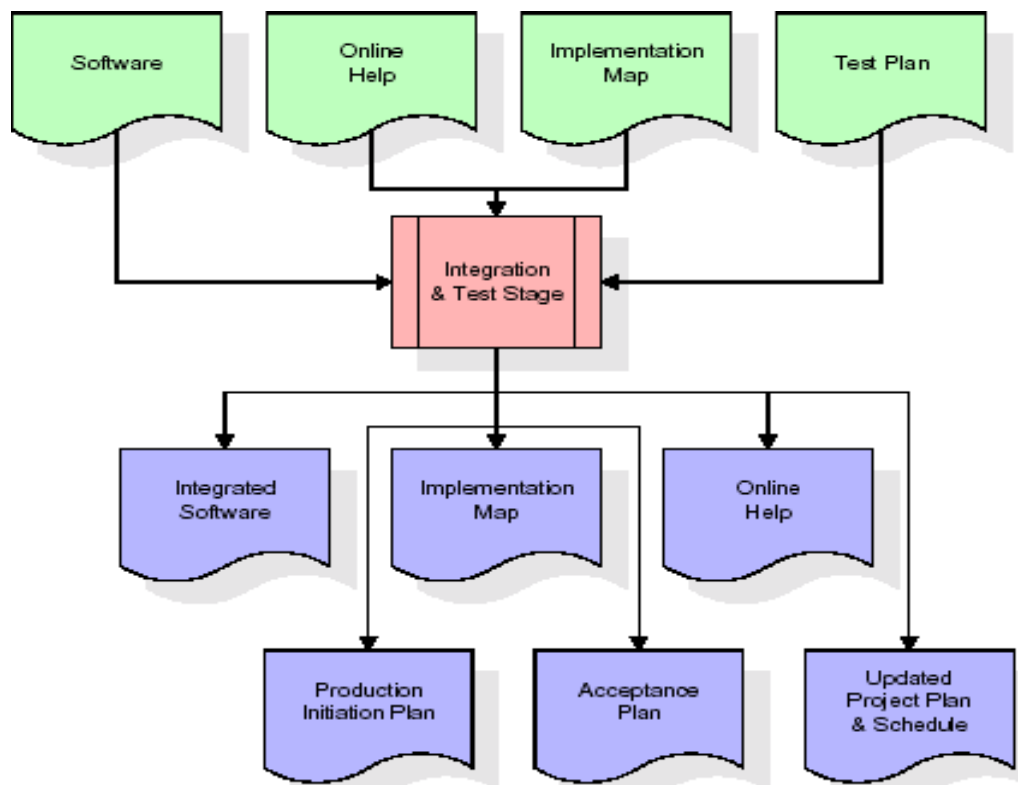
**Fig 3.5.1.e CODING**

The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the

primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

### 3.5.2.f TESTING

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



**Fig 3.5.2.f TESTING**

The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

The primary outputs of the installation and acceptance stage include a

production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

### **3.5.2.g MAINTENANCE**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

## **3.6. SOFTWARE REQUIREMENT SPECIFICATION**

### **OVERALL DESCRIPTION**

A Software Requirements Specification (SRS) – a requirements specification for a software\_system is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Nonfunctional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business\_analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

Business requirements describe in business terms what must be delivered or accomplished to provide value. Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)

Process requirements describe activities performed by the developing organization. For instance, process requirements could specify. Preliminary investigation examines

project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

### **Economic Feasibility**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economic feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, there is nominal expenditure and economic feasibility for certain.

### **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

### **Technical Feasibility**

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .The users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles

specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

### **3.7 EXTERNAL INTERFACE REQUIREMENTS**

#### **User Interface**

The user interface of this system is a user-friendly python Graphical User Interface.

#### **Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

#### **Software Interfaces**

The required software is python.

#### **3.7.1 SYSTEM REQUIREMENT:**

##### **3.7.1.a. HARDWARE REQUIREMENTS:**

- |              |   |                           |
|--------------|---|---------------------------|
| 1. Processor | - | Intel i3(min)             |
| 2. Speed     | - | 1.1 GHz                   |
| 3. RAM       | - | 4GB(min)                  |
| 4. Hard Disk | - | 500 GB                    |
| 5. Key Board | - | Standard Windows Keyboard |
| 6. Mouse     | - | Two or Three Button Mouse |
| 7. Monitor   | - | SVGA                      |

##### **3.7.1.b. SOFTWARE REQUIREMENTS:**

- |                         |   |                  |
|-------------------------|---|------------------|
| 8. Operating System     | - | Windows10(min)   |
| 9. Programming Language | - | Python           |
| 10. Front-end           | - | HTML, CSS, Flask |

# CHAPTER 4

## SYSTEM DESIGN

### 4.1 CLASS

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

1. The upper part holds the name of the class
2. The middle part contains the attributes of the class
3. The bottom part gives the methods or operations the class can take or undertake

#### Usecase

A use case at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as we

#### Sequence

A sequence is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.

#### Collaboration

A collaboration describes interactions among objects in terms of sequenced

messages. Collaboration diagrams represent a combination of information taken from

System requirements specification: A structured collection of information that embodies the requirements of a system. A business\_analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

Business requirements describe in business terms what must be delivered or accomplished to provide value. Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.

### **Component**

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.

### **Deployment**

A deployment in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

### **Activity**

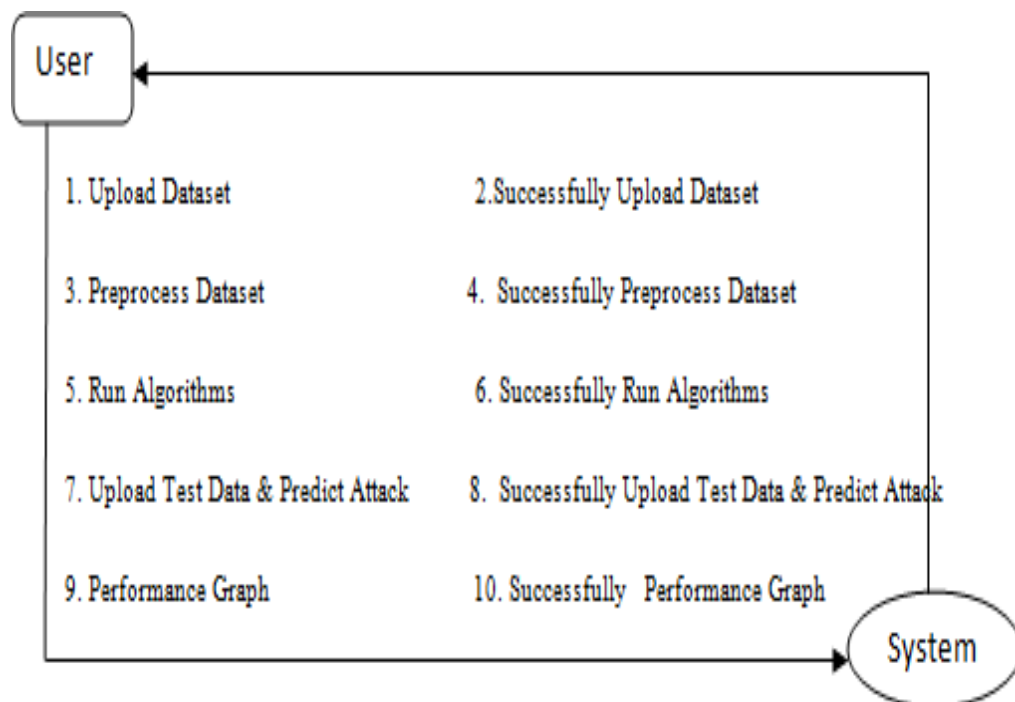
Activity is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.



## 4.2 DATA FLOW:

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.



**Fig 4.2 DATA FLOW**

## **CHAPTER 5**

### **IMPLEMENTATION**

#### **5.1 PYTHON**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

##### **History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

##### **Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

##### **Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

##### **Features Of Python:**

###### **A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

###### **Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software's written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

## **Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

## **Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

## **A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

## **Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQL dB library using import MySQL dB .

Standard libraries in Python are well tested and used by hundreds of people. So, you can be sure that it won't break your application.

## **Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

## **Applications of Python:**

### **1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

```
a = 2
```

```
b = 3
sum = a + b
print(sum)
```

## **2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

## **3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

## **4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

## **5.2 CONVOLUTIONAL NEURAL NETWORK**

A Convolutional Neural Network (CNN) is a type of deep learning algorithm that is particularly well-suited for image recognition and processing tasks. It is made up of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The architecture of CNNs is inspired by the visual processing in the human brain, and they are well-suited for capturing hierarchical patterns and spatial dependencies within images.

CNNs are trained using a large dataset of labeled images, where the network learns to recognize patterns and features that are associated with specific objects or classes. Proven to be highly effective in image-related tasks, achieving state-of-the-art performance in various computer vision applications. Their ability to automatically learn hierarchical representations of features makes them well-suited for tasks where the spatial relationships and patterns in the data are crucial for accurate predictions. CNNs are widely used in areas such as image classification, object detection, facial recognition, and medical image analysis.

The convolutional layers are the key component of a CNN, where filters are applied to the input image to extract features such as edges, textures, and shapes.

The output of the convolutional layers is then passed through pooling layers, which are used to down-sample the feature maps, reducing the spatial dimensions while retaining the most important information. The output of the pooling layers is then passed through one or more fully connected layers, which are used to make a prediction or classify the image.

### **Design**

The construction of a convolutional neural network is a multi-layered feed-forward neural network, made by assembling many unseen layers on top of each other in a particular order.

It is the sequential design that give permission to CNN to learn hierarchical attributes.

In CNN, some of them followed by grouping layers and hidden layers are typically convolutional layers followed by activation layers.

The pre-processing needed in a ConvNet is kindred to that of the related pattern of neurons in the human brain and was motivated by the organization of the Visual Cortex.

### **Training**

CNNs are trained using a supervised learning approach. This means that the CNN is given a set of labeled training images. The CNN then learns to map the input images to their correct labels.

The training process for a CNN involves the following steps:

- **Data Preparation:** The training images are preprocessed to ensure that they are all in the same format and size.
- **Loss Function:** A loss function is used to measure how well the CNN is performing on the training data. The loss function is typically calculated by taking the difference between the predicted labels and the actual labels of the training images.
- **Optimizer:** An optimizer is used to update the weights of the CNN in order to minimize the loss function.
- **Backpropagation:** Backpropagation is a technique used to calculate the gradients of the loss function with respect to the weights of the CNN. The gradients are then used to update the weights of the CNN using the optimizer.

### **Evaluation**

After training, CNN can be evaluated on a held-out test set. A collection of pictures that the CNN has not seen during training makes up the test set. How well the CNN

performs on the test set is a good predictor of how well it will function on actual data. The efficiency of a CNN on picture categorization tasks can be evaluated using a variety of criteria. Among the most popular metrics are:

**Accuracy:** Accuracy is the percentage of test images that the CNN correctly classifies.

**Precision:** Precision is the percentage of test images that the CNN predicts as a particular class and that are actually of that class.

**Recall:** Recall is the percentage of test images that are of a particular class and that the CNN predicts as that class.

**F1 Score:** The F1 Score is a harmonic mean of precision and recall. It is a good metric for evaluating the performance of a CNN on classes that are imbalanced.

Convolutional Neural Networks (CNNs) are a powerful tool for machine learning, especially in tasks related to computer vision. Convolutional Neural Networks, or CNNs, are a specialized class of neural networks designed to effectively process grid-like data, such as images.

### 5.3 SAMPLE CODE:

```
from flask import Flask, render_template, Response, request
from flask_socketio import SocketIO, send, emit
import base64
from PIL import Image
from io import BytesIO
import numpy as np
import tensorflow as tf
import cv2
import os
import shutil
from werkzeug.utils import secure_filename
from flask import current_app
from flask import send_file
from getmail import send_mail
import warnings
warnings.filterwarnings('ignore')
app = Flask(__name__)
app.config["SECRET_KEY"] = "secret"
app.config["DEBUG"] = True
```

```

socketio = SocketIO(app)
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'
detect_fn = tf.saved_model.load("Models/FaceDetector/saved_model")#Load the face
detector
model = tf.keras.models.load_model("Models/FEC")#Load the facial emotion
classifier
class_names = ['Angry', 'Disgust', 'Fear', 'Happy', 'Neutral', 'Sad', 'Surprise']
ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'gif', 'mp4'}
static_files = ['display.css', 'eye.png', 'Picdetectb.jpg', 'thumbsup.jpg', 'github.png',
'TU.svg', 'UI.svg', 'RT.svg', 'UV.svg', 'VU.svg', 'feedback.svg']
@app.route('/picdelete')
def picdelete():
    #When this function is called all the files that are not present in the
    #list static_files will be deleted.
    for file in os.listdir("static"):
        if file not in static_files:
            os.remove(f"static/{file}")
    return ("nothing")
@app.route('/')
def home():
    return render_template('home.html')
@app.route('/webcam')
def webcam():
    return render_template('index.html')
def allowed_file(filename):
    return ('.' in filename and
filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS)
def bound(boxes, scores, h, w):
    idxs = cv2.dnn.NMSBoxes(boxes, scores, 0.5, 1.5)
    # define a array as matrix
    signs = []
    for i in range(len(idxs))
    signs.append(i)
    height, width = h, w

```

```

# ensure at least one detection exists
if len(idxs) > 0:
    # loop over the indexes we are keeping
    for i in idxs.flatten():
        # extract the bounding box coordinates
        ymin = int((boxes[i][0] * height))
        xmin = int((boxes[i][1] * width))
        ymax = int((boxes[i][2] * height))
        xmax = int((boxes[i][3] * width))
        signs[i] = [ymin,ymax,xmin,xmax]
    return signs

def draw_bounding_box(frame, detect_fn):
    #Returns the coordinates of the bounding boxes.
    input_tensor = tf.convert_to_tensor(frame)
    input_tensor = input_tensor[tf.newaxis, ...]
    detections = detect_fn(input_tensor)
    num_detections = int(detections.pop('num_detections'))
    detections = {key: value[0, :num_detections].numpy() for key, value in
detections.items()}
    detections['num_detections'] = num_detections
    boxes = detections['detection_boxes']
    scores = detections['detection_scores']
    h, w = frame.shape[:2]
    boxes = boxes.tolist()
    scores = scores.tolist()
    coordinates = bound(boxes, scores, h, w)
    return coordinates

def detectandupdate(img):
    path = "static/" + str(img)
    image = cv2.imread(path)
    coordinates = draw_bounding_box(image, detect_fn)
    #Loop over the each bounding box.
    for (y, h, x, w) in coordinates:
        cv2.rectangle(image,(x,y),(w, h),(0, 255, 0),2)

```



```

img2 = image[y:h, x:w]#Get the face from the image with this trick.
img2 = tf.image.resize(img2, size = [128, 128])#Input for the model should have size-
(128,128)
pred = model.predict(tf.expand_dims(img2, axis=0))
pred_class = class_names[tf.argmax(pred, axis = 1).numpy()[0]]
#These conditions are just added to draw text clearly when the head is so close to the
top of the image.
if x > 20 and y > 40:
cv2.putText(image, pred_class, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,
255, 0), 2)
else:
cv2.putText(image, pred_class, (x + 10, y + 20), cv2.FONT_HERSHEY_SIMPLEX,
1, (0, 255, 0), 2)
path2 = f'static/pred{img}'
#Save as predimg_name in static.
cv2.imwrite(path2, image)
return ([img, "pred" + img])
@app.route('/detectpic', methods=['GET', 'POST'])
def detectpic():
UPLOAD_FOLDER = 'static'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
if request.method == 'POST':
file = request.files['file']
if file and allowed_file(file.filename):
filename = secure_filename(file.filename)
file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
result =detectandupdate(filename)
return render_template('showdetect.html', orig=result[0], pred=result[1])
@app.route('/picdetect')
def picdetect():
return render_template('picdetect.html')
@app.route('/feedback')
def feedback():
return render_template('feedback.html')

```

```

@app.route('/sentsafe',methods=['GET', 'POST'])
def send_sentsafe():
    if request.method == 'POST':
        email = request.form['email']
        comments = request.form['comments']
        name=request.form['name']
        comments=email+"\n "+name+"\n "+comments
        send_mail(email,comments)
        return render_template('sentfeed.html')
@socketio.on("message")
def handleMessage(input):
    input = input.split(",")[1]
    image_data = input
    #Since the input frame is in the form of string we need to convert it into array.
    im = Image.open(BytesIO(base64.b64decode(image_data)))
    im = np.asarray(im)
    #process it.
    coordinates = draw_bounding_box(im, detect_fn)
    for (y, h, x, w) in coordinates:
        cv2.rectangle(im,(x,y),(w, h),(0, 255, 0),2)
    img = im[y:h, x:w]
    img = tf.image.resize(img, size = [128, 128])
    pred = model.predict(tf.expand_dims(img, axis=0))
    pred_class = class_names[tf.argmax(pred, axis = 1).numpy()[0]]
    cv2.putText(im, pred_class, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,
255, 0), 2)
    #Convert it back into string.
    im = Image.fromarray(im)
    buff = BytesIO()
    im.save(buff, format="JPEG")
    image_data = base64.b64encode(buff.getvalue()).decode("utf-8")
    image_data = "data:image/jpeg;base64," + image_data
    emit('out-image-event', {'image_data': image_data})
def detectandupdatevideo(video):

```

```

output_path = f"static/pred{video}"
fourcc = cv2.VideoWriter_fourcc(*'h264')
out = cv2.VideoWriter(output_path, fourcc, 25.0, (640, 360))
#using Videowriter to save the processed frames as a video.
vidcap = cv2.VideoCapture(f"static/{video}");
while True:
    ret, image = vidcap.read()
    if ret == True:
        coordinates = draw_bounding_box(image, detect_fn)
        for (y, h, x, w) in coordinates:
            cv2.rectangle(image,(x,y),(w, h),(0, 255, 0),2)
            img2 = image[y:h, x:w]
            img2 = tf.image.resize(img2, size = [128, 128])
            pred = model.predict(tf.expand_dims(img2, axis=0))
            pred_class = class_names[tf.argmax(pred, axis = 1).numpy()[0]]
            if x > 20 and y > 40:
                cv2.putText(image, pred_class, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,
                255, 0), 2)
            else:
                cv2.putText(image, pred_class, (x + 10, y + 20), cv2.FONT_HERSHEY_SIMPLEX,
                1, (0, 255, 0), 2)
            output_file = cv2.resize(image, (640, 360))
            out.write(output_file)
        else:
            break
    vidcap.release()
    out.release()
    return ([video, "pred" + video])
@app.route('/detectvideo', methods=['GET', 'POST'])
def detectvideo():
    UPLOAD_FOLDER = 'static'
    app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
    if request.method == 'POST':
        file = request.files['file']

```

```
if file and allowed_file(file.filename):
    filename = secure_filename(file.filename)
    file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
    result = detectandupdatevideo(filename)
    return render_template('showvideo.html', orig=result[0], pred=result[1])
@app.route('/video')
def video():
    return render_template('video.html')
if __name__ == "__main__":
    socketio.run(app)
```

## **CHAPTER 6**

### **TESTING**

#### **6.1 IMPLEMENTATION AND TESTING**

Implementation is one of the most important tasks in project is the phase in which one has to be cautious because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

##### **6.1.1 IMPLEMENTATION**

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifying as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

##### **6.2 TESTING**

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system properly function as a unit. The test data should be chosen such that it passed through all possible condition. Actually, testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

### **6.2.1. SYSTEM TESTING**

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

### **6.2.2. MODULE TESTING**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

### **6.2.3. INTEGRATION TESTING**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are every correct. Thus the mapping of jobs with resources is done correctly by the system.

## 6.2.4. ACCEPTANCE TESTING

When that user find no major problems with its accuracy, the system passes through a final acceptance test. This test confirms that the system meets the original goals, objectives and requirements established during analysis without actual execution which eliminates wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

Test Case Id	Test Case Name	Test Case Desc.	Test Steps			Test Case Status	Test Priority
			Step	Expected	Actual		
01	Upload Dataset	Verify Dataset Uploaded or not	If Dataset may not upload	we cannot do any further operations	we can do further operations	High	High
02	Preprocess Dataset	Verify Preprocess Dataset Uploaded or not	If Preprocess Dataset File may not upload	we cannot do any further operations	we can do further operations	High	High
03	Run Algorithms	Verify Run Algorithms or not	If Algorithms May not be Run	we cannot do any further operations	we can do further operations	High	High
04	Upload Test Data & Predict Attack	Verify Upload Test Data & Predict Attack or not	If Test Data & Predict Attack not upload	We cannot run operation	We can Run the Operation	High	High
05	Performance Graph	Verify Performance Graph or not	If Performance Graph not upload	We cannot run operation	We can Run the Operation	High	High

**Fig 6.2.4 ACCEPTANCE**

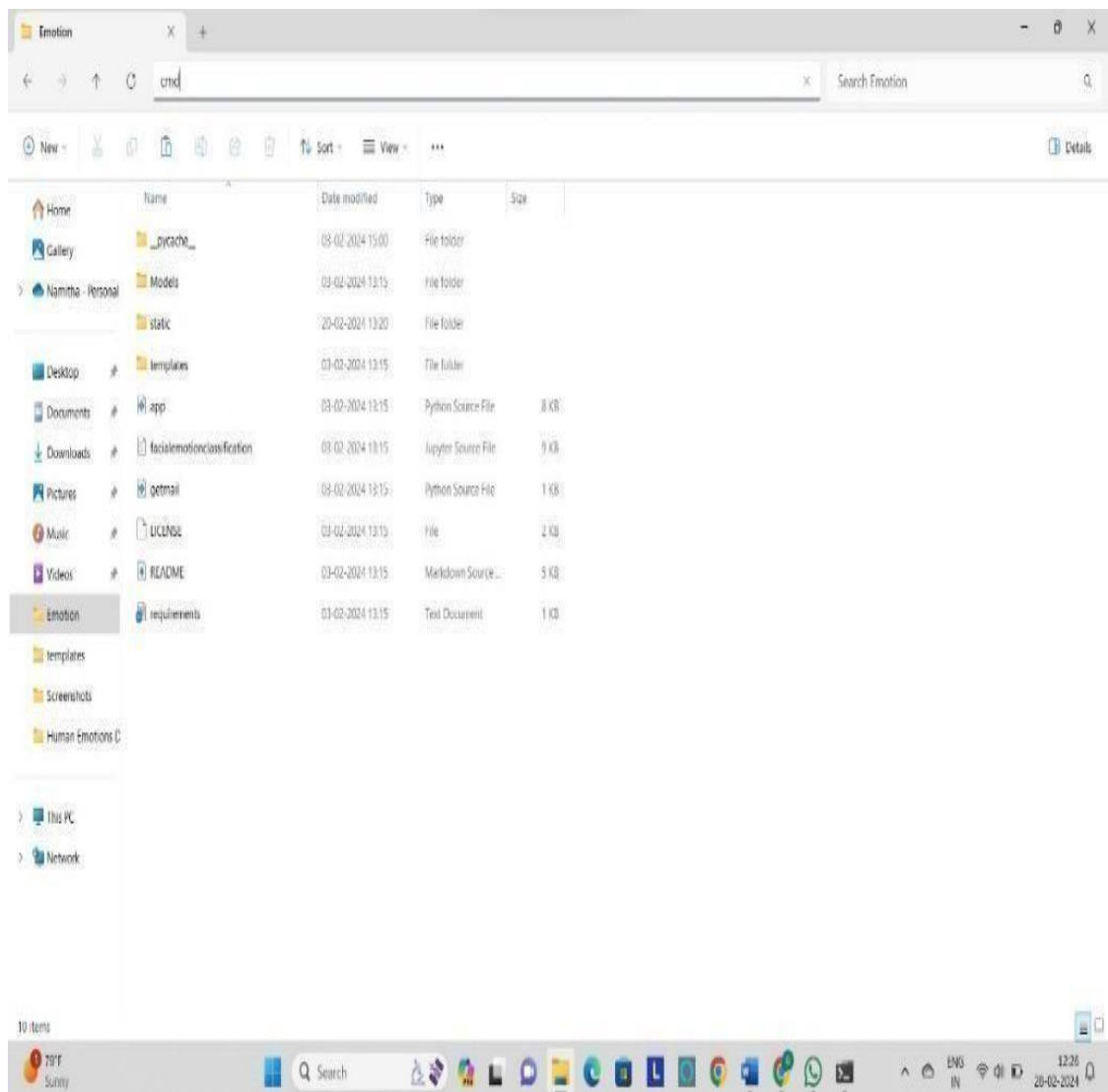
# CHAPTER 7

## RESULTS

### 7.1 EXECUTION

#### 7.1.1 LIST OF FILES

Open command prompt. Whereas you saved your project.

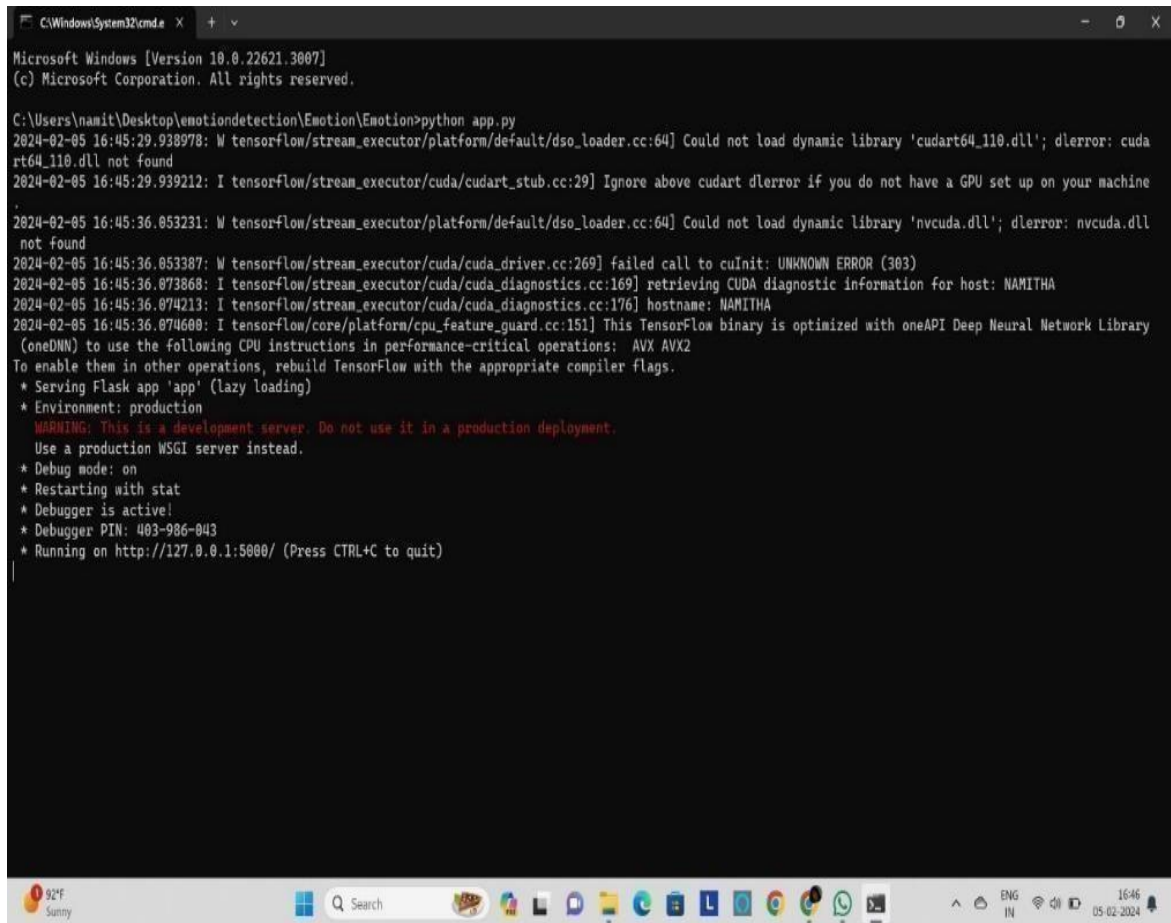


**Fig 7.1.1 LIST OF FILES**



When you open the command prompt and enter a **python app.py** (which is saving your code as app.py and writing code in python).

## 7.1.2 INPUT COMMAND



```
Microsoft Windows [Version 10.0.22621.3807]
(c) Microsoft Corporation. All rights reserved.

C:\Users\namit\Desktop\emotiondetection\Emotion\Emotion>python app.py
2024-02-05 16:45:29.938978: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cuda
rt64_110.dll' not found
2024-02-05 16:45:29.939212: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine
2024-02-05 16:45:36.053231: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'nvcuda.dll'; dlerror: nvcuda.dll
not found
2024-02-05 16:45:36.053387: W tensorflow/stream_executor/cuda/cuda_driver.cc:269] failed call to cuInit: UNKNOWN ERROR (303)
2024-02-05 16:45:36.073868: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: NAMITHA
2024-02-05 16:45:36.074213: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: NAMITHA
2024-02-05 16:45:36.074608: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library
(oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 403-986-043
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

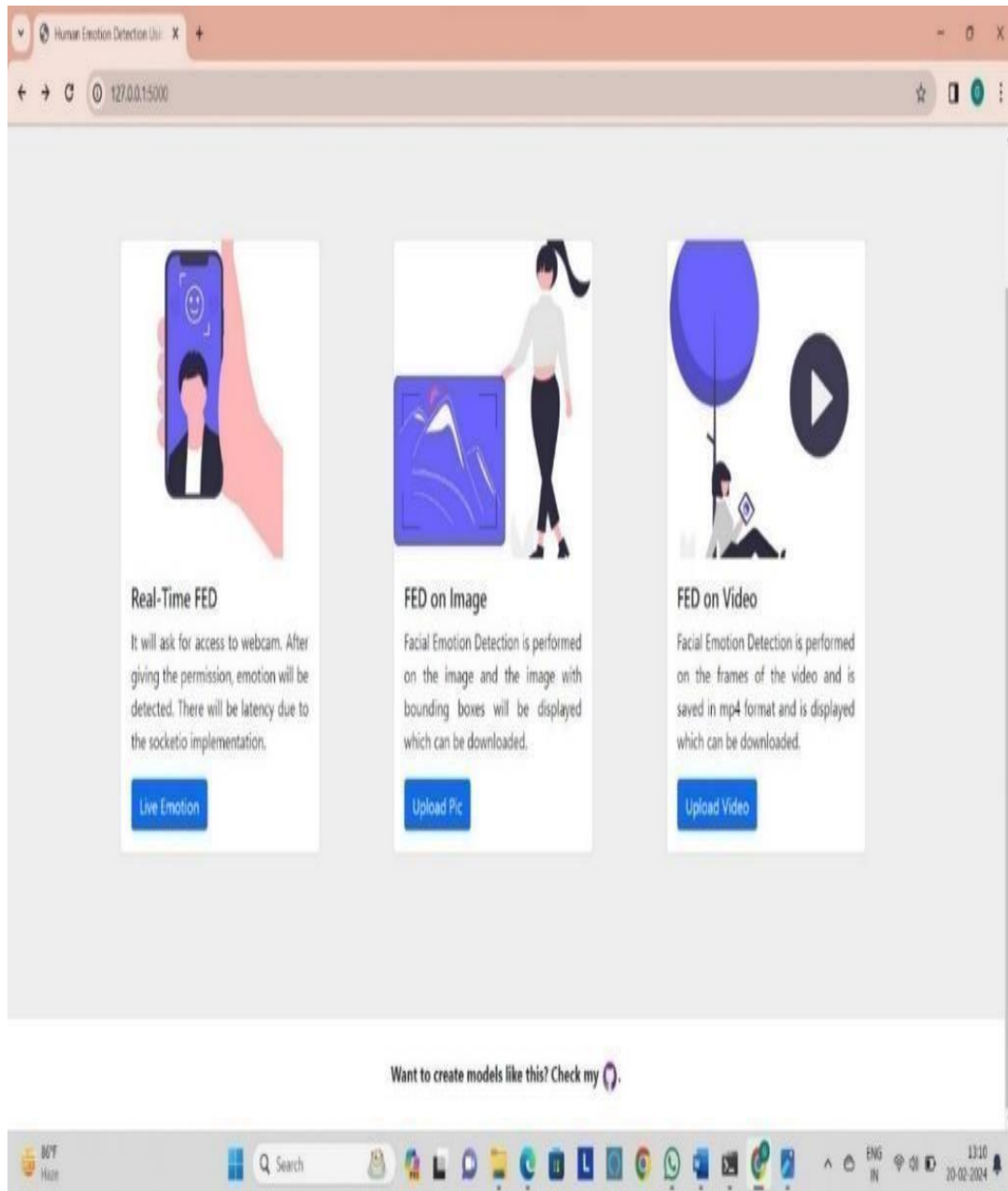
**Fig 7.1.2 INPUT COMMAND**

After entering `python app.py` you can get the website such as <http://127.0.0.1:5000/>.

Then copy that link and paste in chrome you can get the output.

### 7.1.3 OUTPUT SCREEN

When open the chrome you can view the several types of fed's. Such as Real time FED, FED on Image, FED on Video.

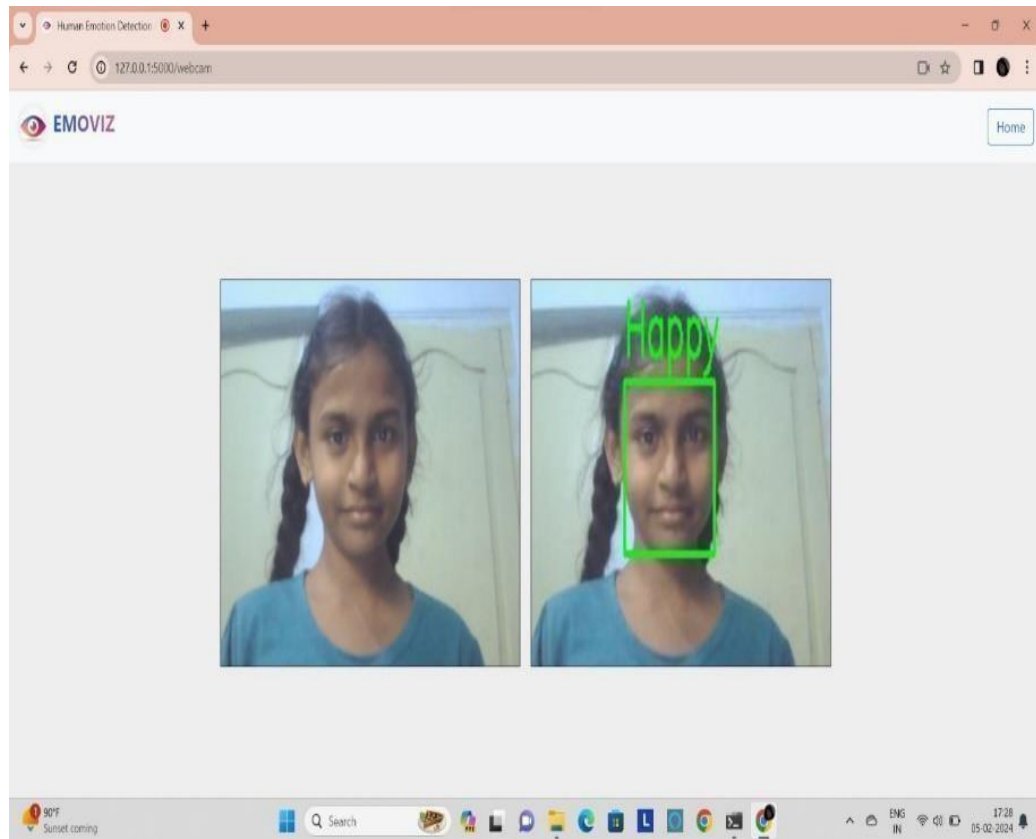


**Fig 7.1.3 OUTPUT SCREEN**

## 7.2 REAL-TIME FED

It will ask for access to webcam. After giving the permission, emotion will be detected. There will be latency due to the socketio implementation.

### 7.2.1 HAPPY

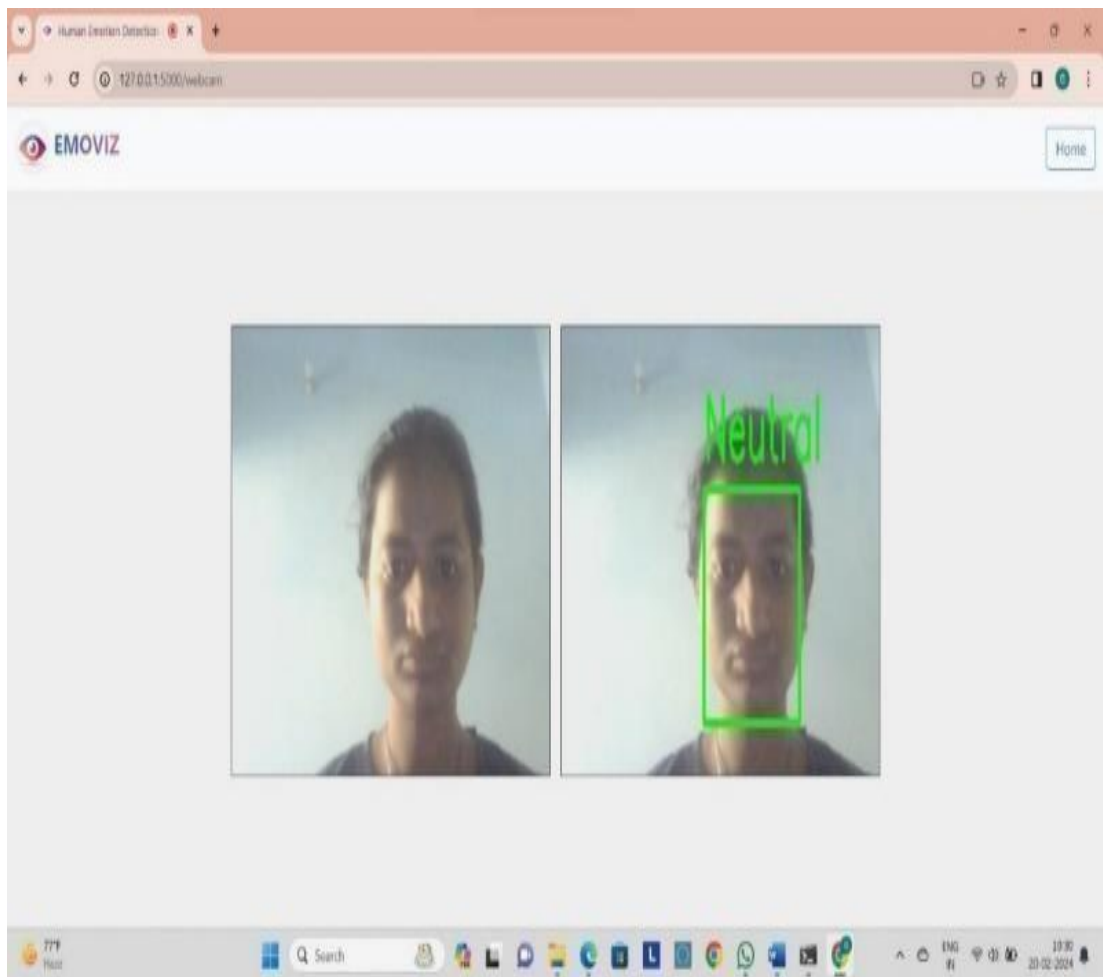


**Fig 7.2.1 HAPPY**

In the above figure shows the live emotion about happy. Happiness is often defined as a pleasant emotional state that is characterized by feelings of contentment, joy, gratification, satisfaction, and well-being.

Cheerful delighted ecstatic elated enraptured exultant glad gleeful jolly joyful joyous jubilant merry mirthful overjoyed thrilled up upbeat. Strong matches. blessed blissful blithe content contented convivial gratified peaceful pleasant pleased satisfied sparkling sunny tickled.

### 7.2.2 NEUTRAL



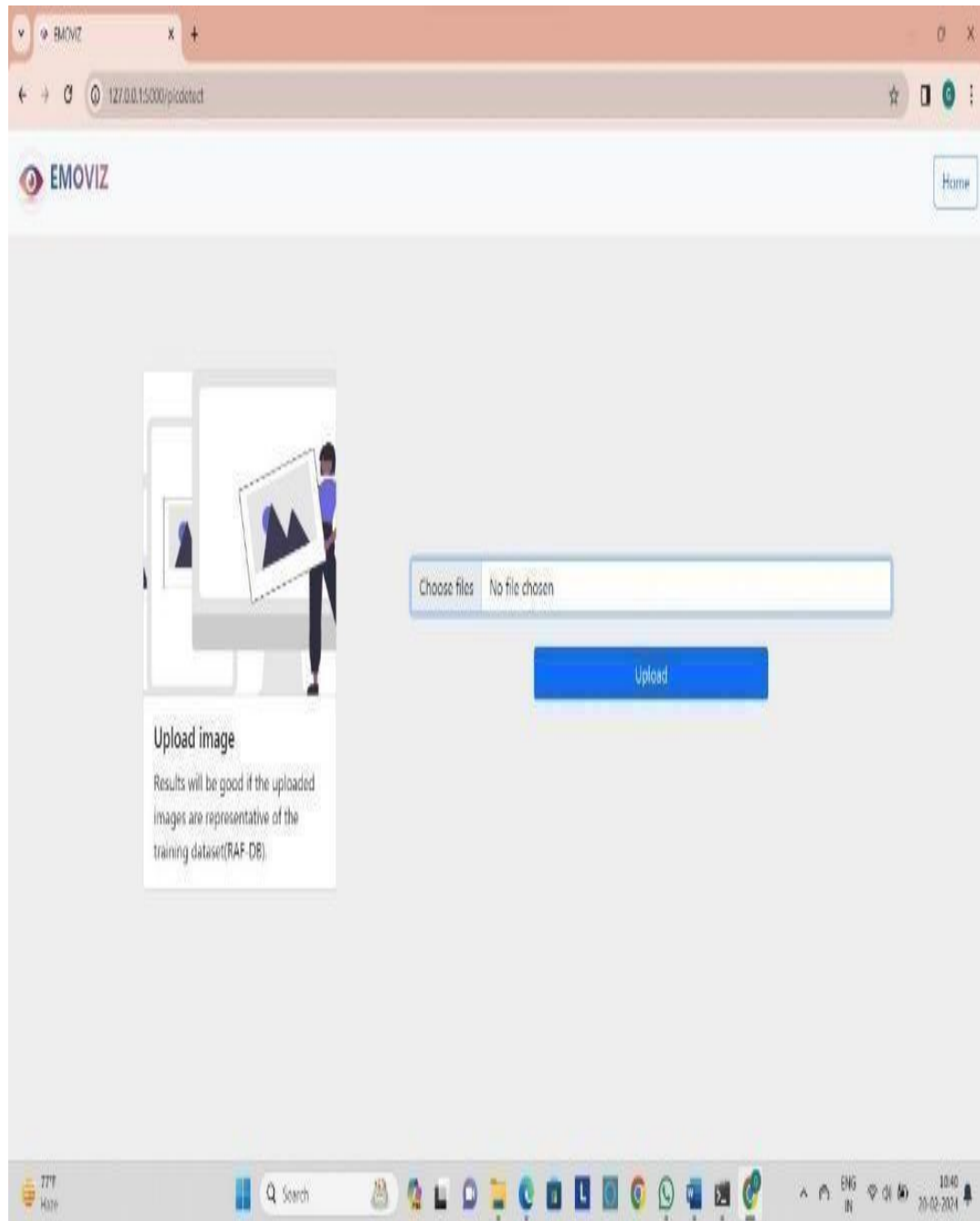
**Fig 7.2.2 NEUTRAL**

In the above figure shows about neutral. The neutral emotion refers to a state of emotional neutrality or indifference, where an individual does not experience strong feelings of either positivity or negativity. In this state, one may feel calm, relaxed, and emotionally balanced.

Neutrality is not an affective state, because affect must be positively or negatively valenced. Neutral affect is unimportant because it does not influence cognition or behavior

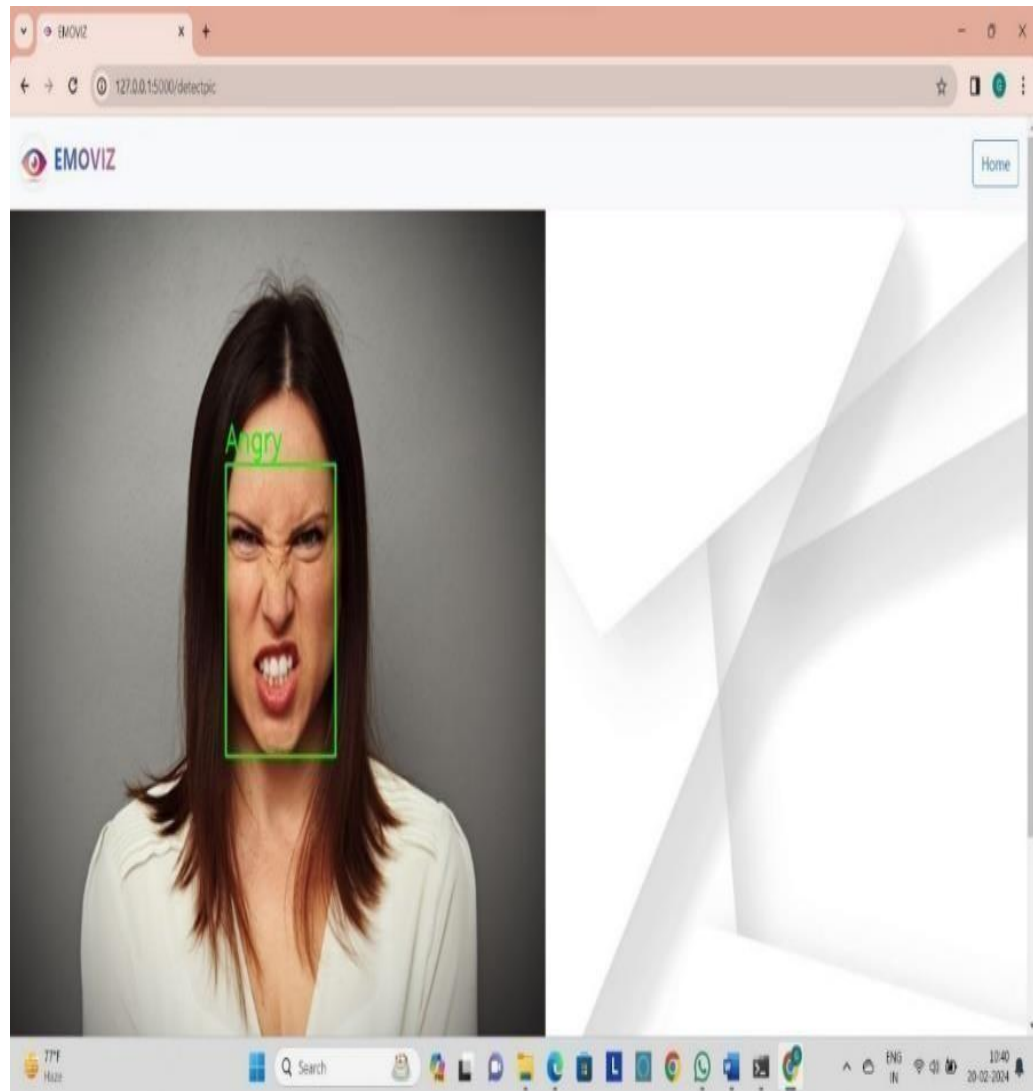
## 7.3 FED ON IMAGE

Facial Emotion Detection is performed on the image and the image with bounding boxes will be displayed which can be downloaded.



**Fig 7.3 FED ON IMAGE**

### 7.3.1 ANGRY

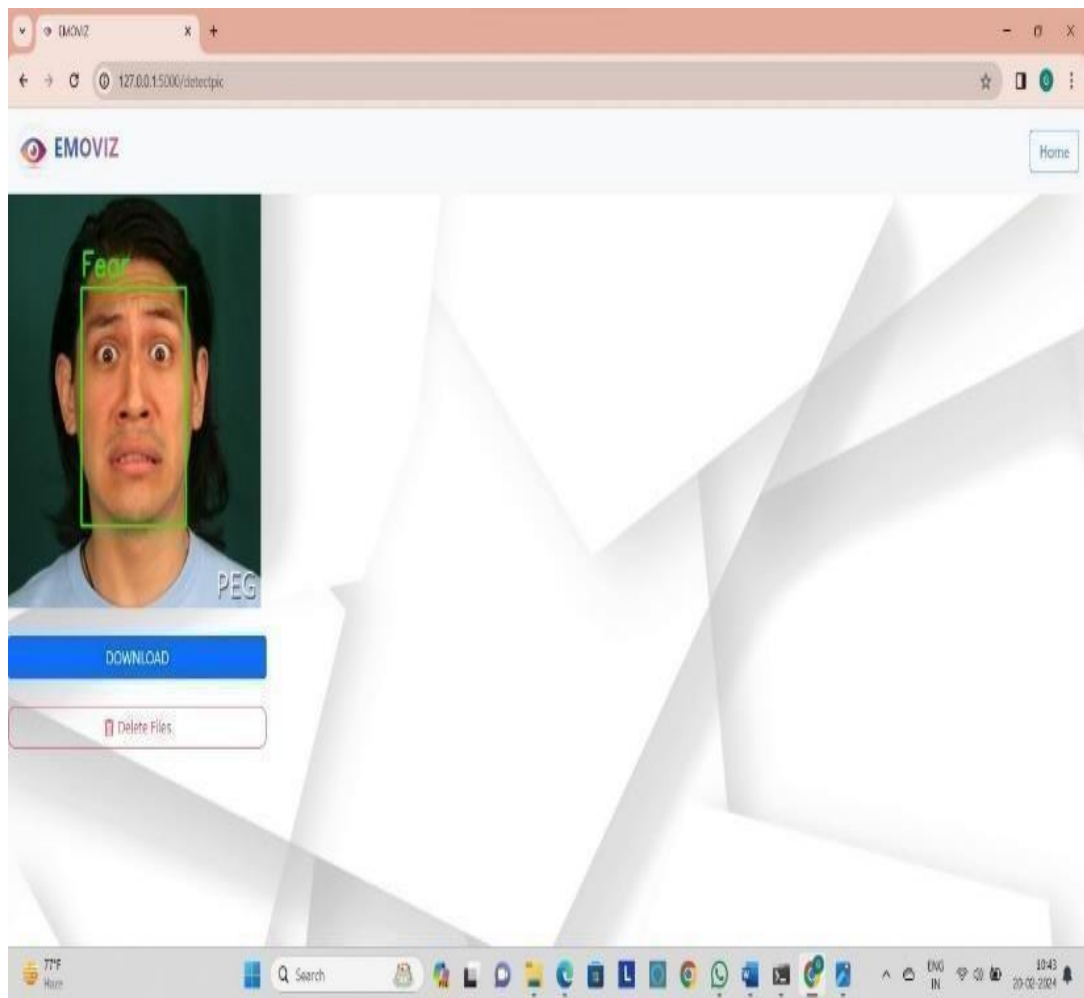


**Fig 7.3.1 ANGRY**

In the above figure shows angry. Angry emotion characterized by feelings of hostility, agitation, frustration, and antagonism towards others. Like fear, anger can play a part in your body's fight or flight response.

Anger is caused by an inconsistency between a desired, or expected, situation and the actually perceived situation, and triggers responses, such as aggressive behavior, with the expected consequence of reducing the inconsistency.

### 7.3.2 FEAR

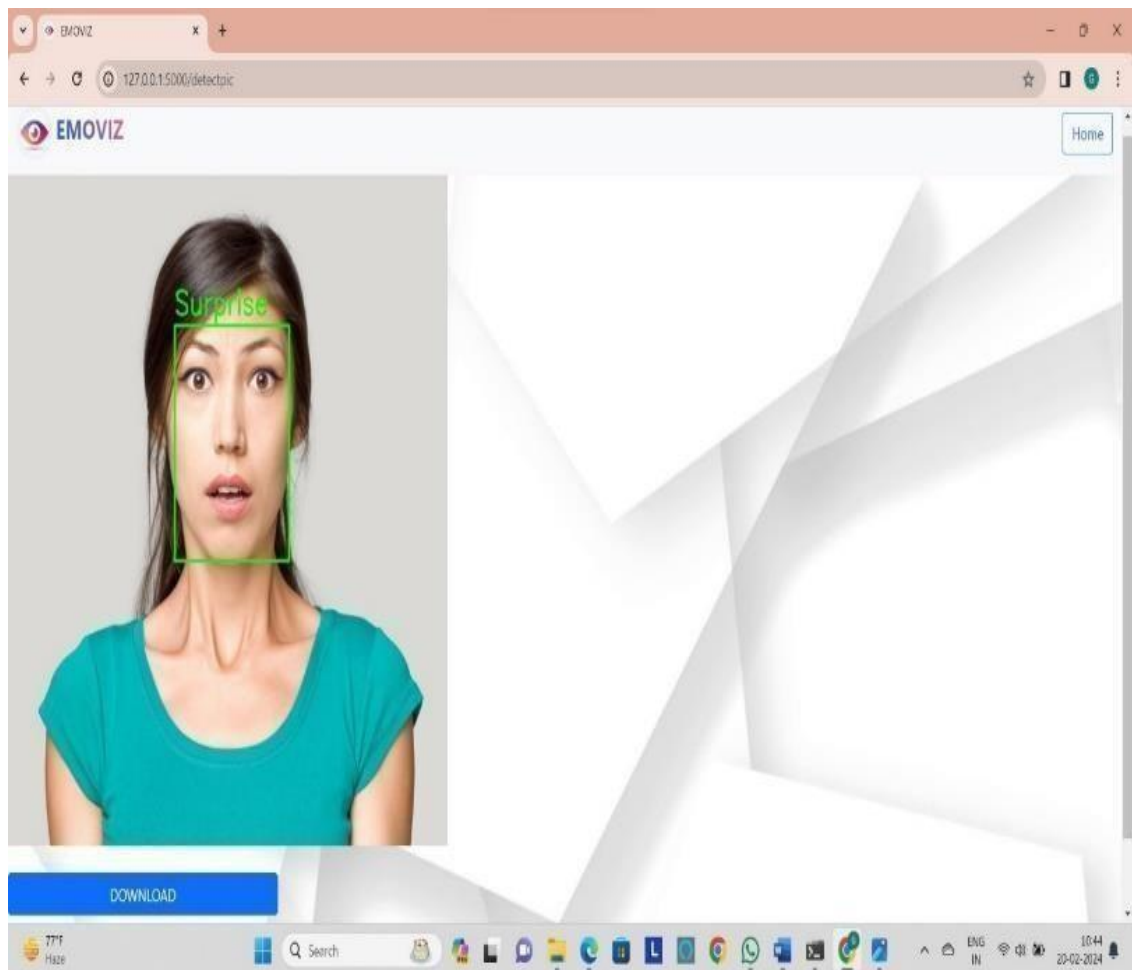


**Fig 7.3.2 FEAR**

The above figure shows fear. Fear emotion that can also play an important role in survival. Your muscles become tense, your heart rate and respiration increase, and your mind becomes more alert, priming your body to either run from the danger or stand and fight.

Fear is an important human emotion that can help protect you from danger and prepare you to take action, but it can also lead to longer-lasting feelings of anxiety. Finding ways to control your fear can help you better cope with these feelings and prevent anxiety from taking hold.

### 7.3.3 SURPRISE



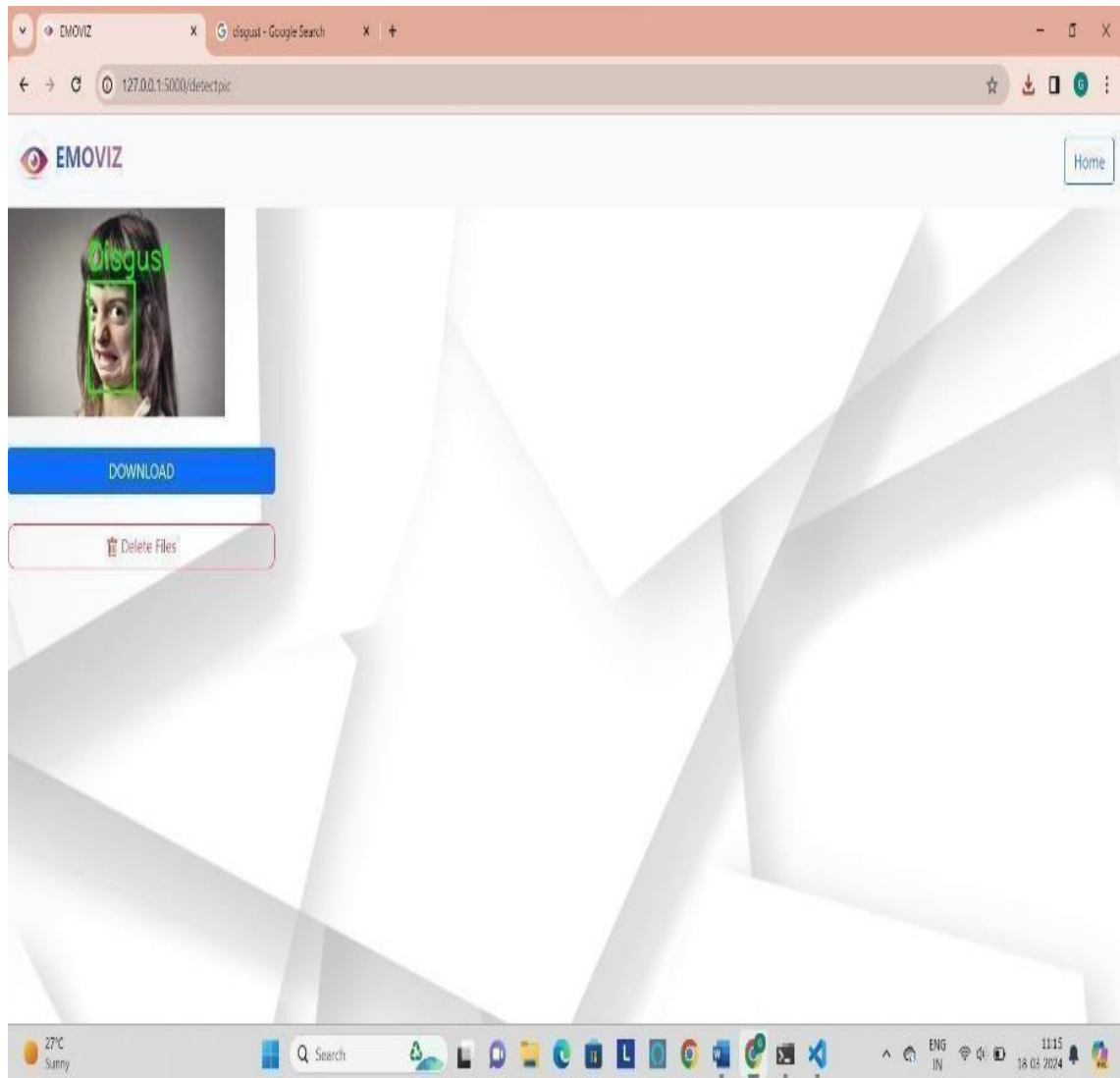
**Fig 7.3.3 SURPRISE**

Above figure shows surprise. Surprise is usually quite brief and is characterized by a physiological startle response following something unexpected.

The range of surprises also varies as some events can give you happiness and others can give you fear if that unexpected event is a threat to us. Physiological changes increased heart rate, breathing are some of the common reactions when encountering such emotions.



### 7.4.3 DISGUST



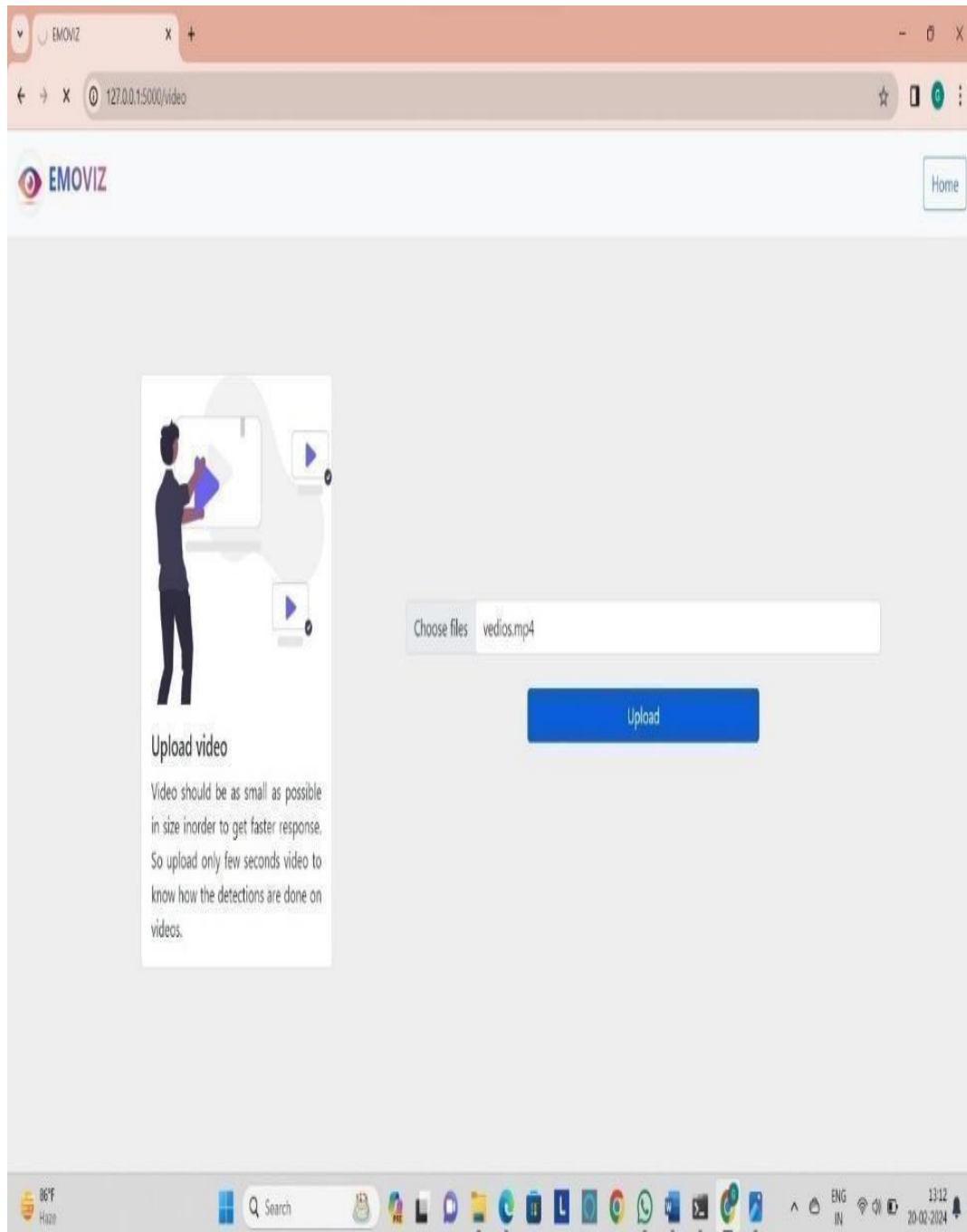
**Fig 7.3.4 DISGUST**

Above figure shows disgust. Disgust causes someone to have a strong feeling of dislike for something especially because it has a very unpleasant appearance, taste, smell, etc.

Disgust is usually thought to be a mechanism to reject foods that are potentially harmful to the organism. Included in this domain are spoiled foods, dirty foods, and foods not normally eaten in a particular culture

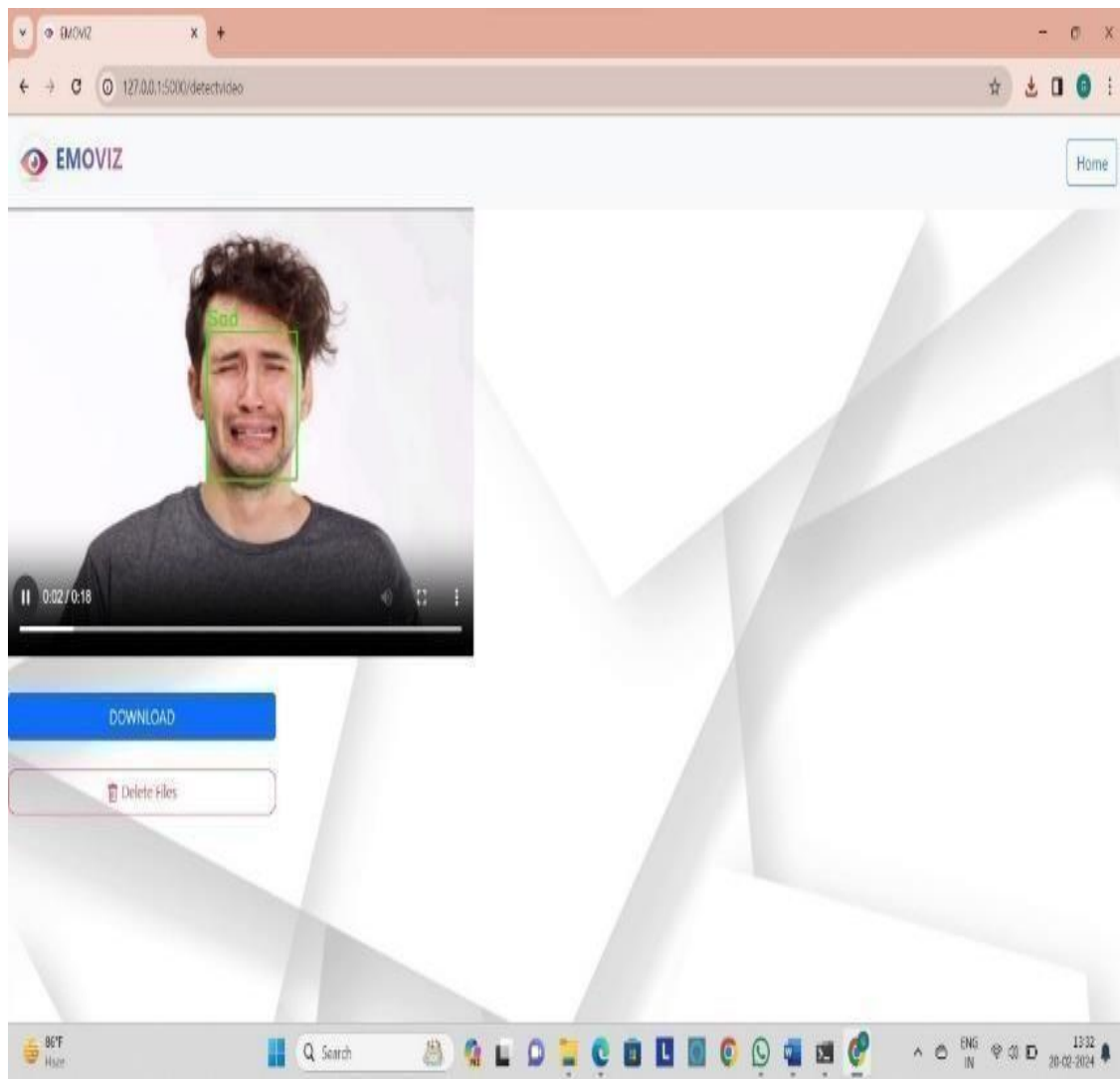
## 7.5 FED ON VIDEO

Facial Emotion Detection is performed on the frames of the video and is saved in mp4 format and is displayed which can be downloaded.



**Fig 7.4 FED ON VIDEO**

### 7.4.1 SAD

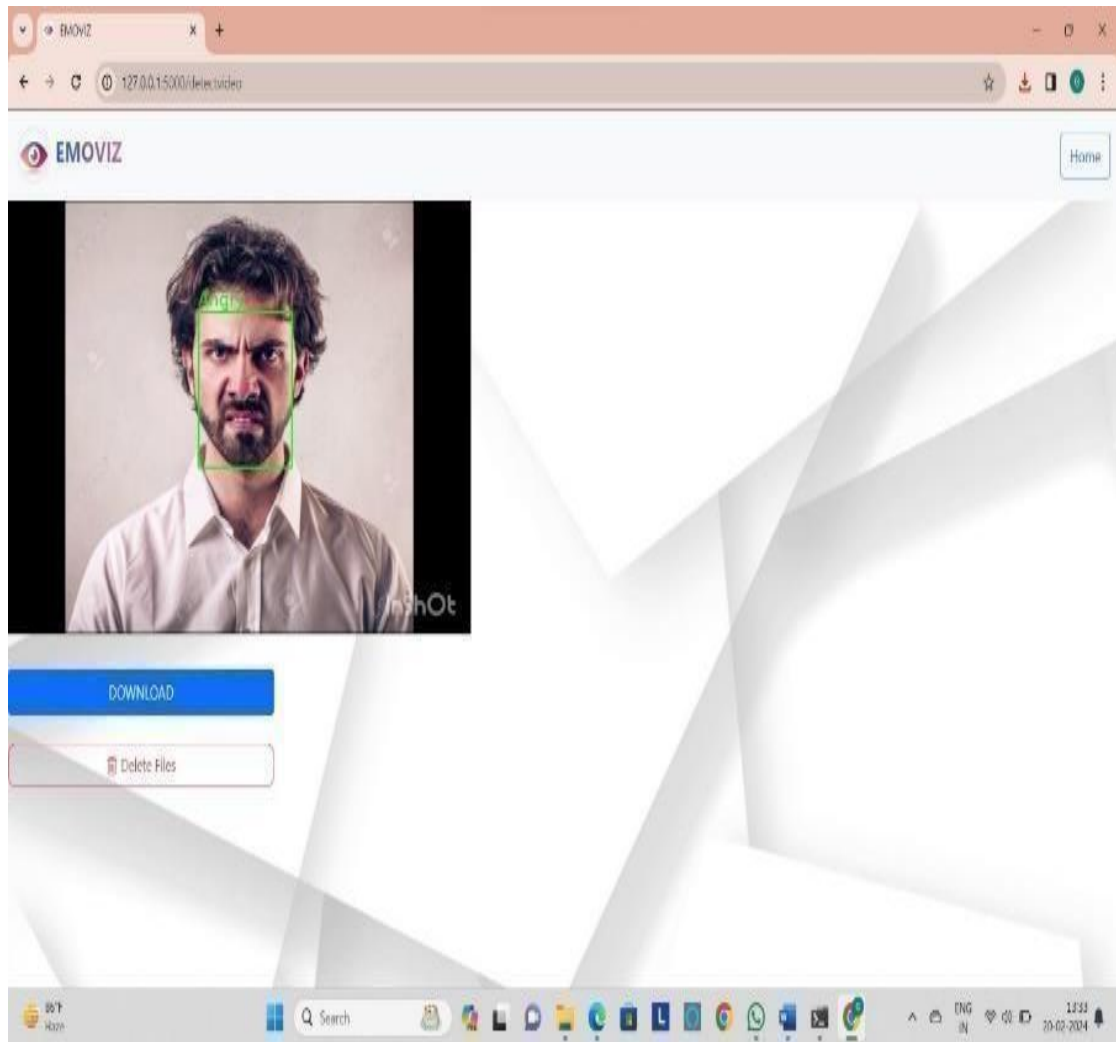


**Fig 7.4.1 SAD**

In the above figure shows sad. Sad emotional state characterized by feelings of disappointment, hopelessness, disinterest, and dampened mood. Sadness is something that all people experience from time to time.

In some cases, people can experience prolonged and severe periods of sadness that can turn into depression. Sadness can be expressed in a number of ways including crying, mood off.

## 7.4.2 ANGRY

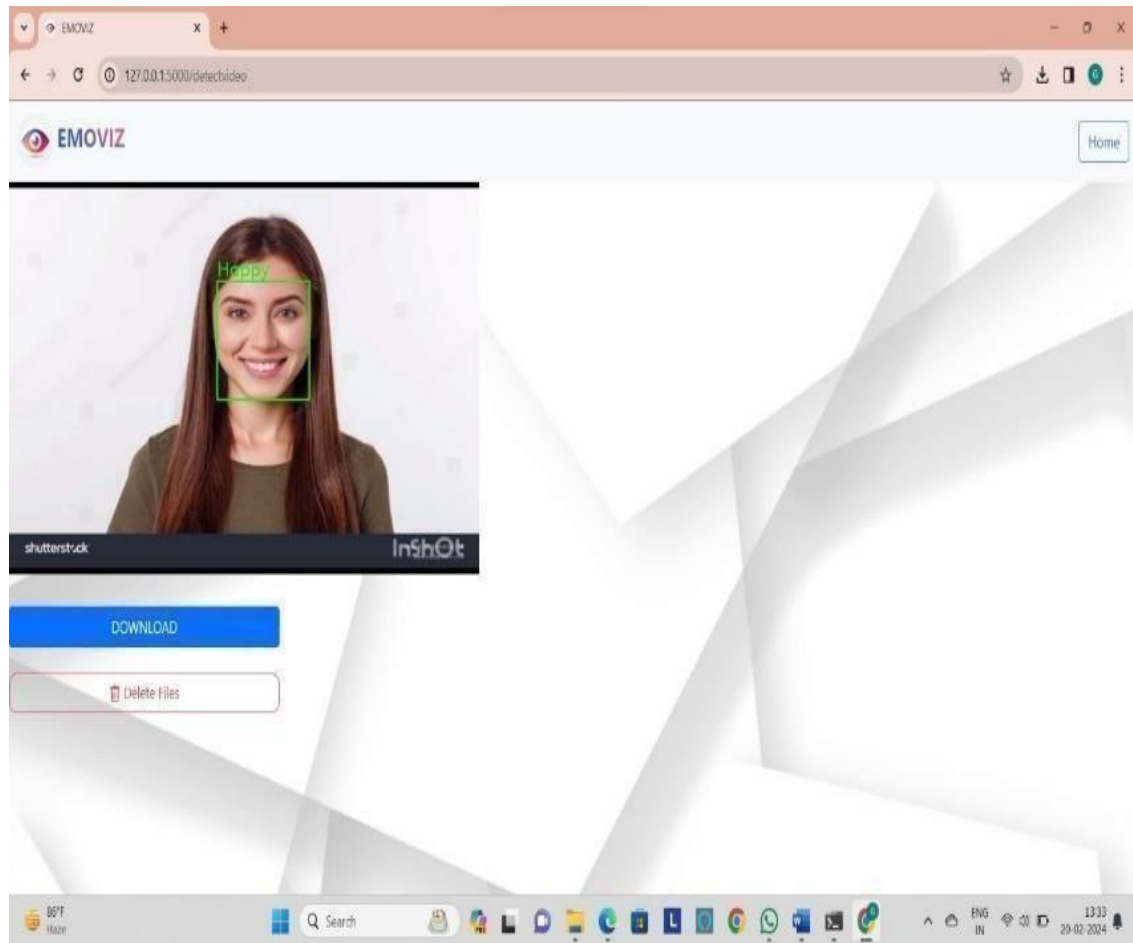


**Fig 7.4.2 ANGRY**

In the above figure shows angry. Angry emotion characterized by feelings of hostility, agitation, frustration, and antagonism towards others. Like fear, anger can play a part in your body's fight or flight response.

Anger is caused by an inconsistency between a desired, or expected, situation and the actually perceived situation, and triggers responses, such as aggressive behavior, with the expected consequence of reducing the inconsistency.

### 7.4.3 HAPPY

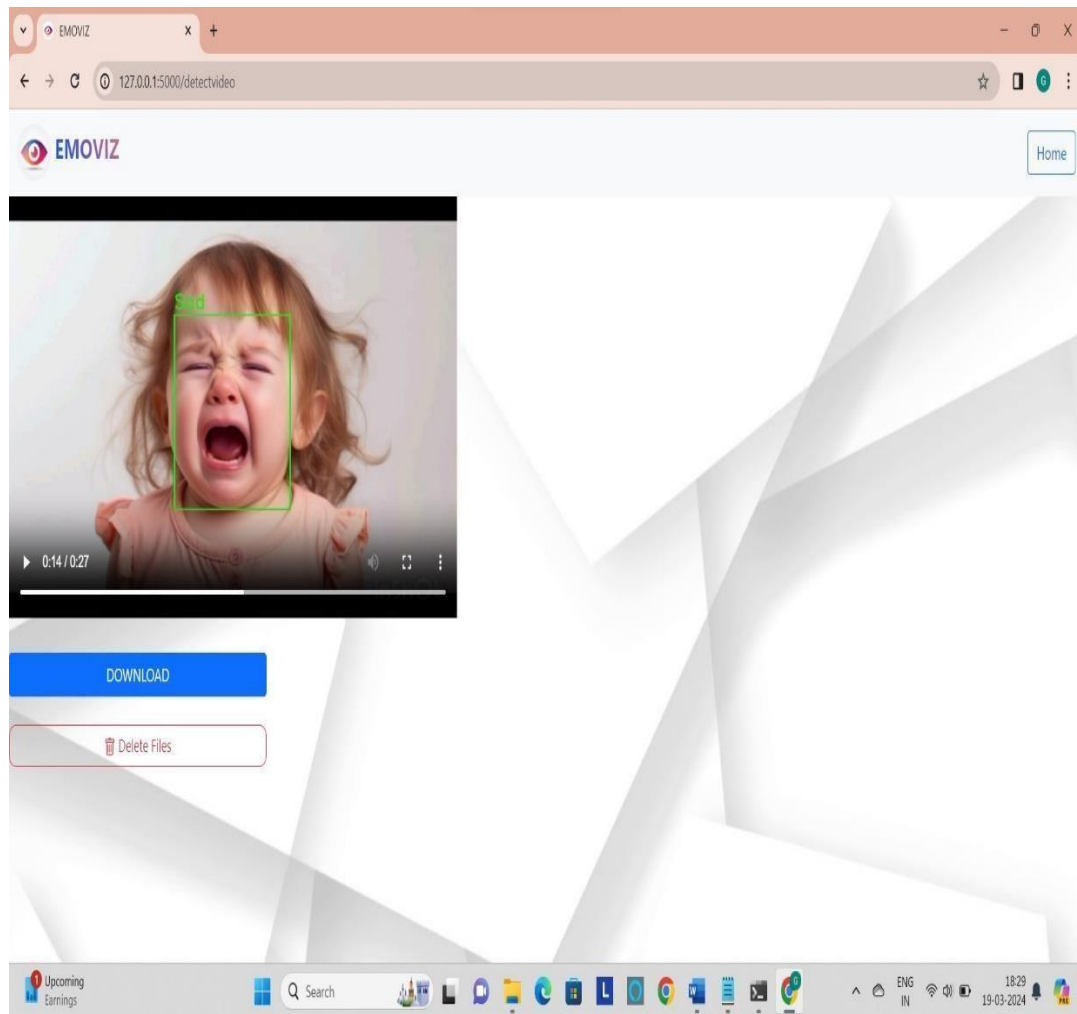


**Fig 7.4.3 HAPPY**

In the above figure shows the live emotion about happy. Happiness is often defined as a pleasant emotional state that is characterized by feelings of contentment, joy, gratification, satisfaction, and well-being.

Cheerful delighted ecstatic elated enraptured exultant glad gleeful jolly joyful joyous jubilant merry mirthful overjoyed thrilled up upbeat. Strong matches. blessed blissful blithe content contented convivial gratified peaceful pleasant pleased satisfied sparkling sunny tickled

#### 7.4.4 SAD

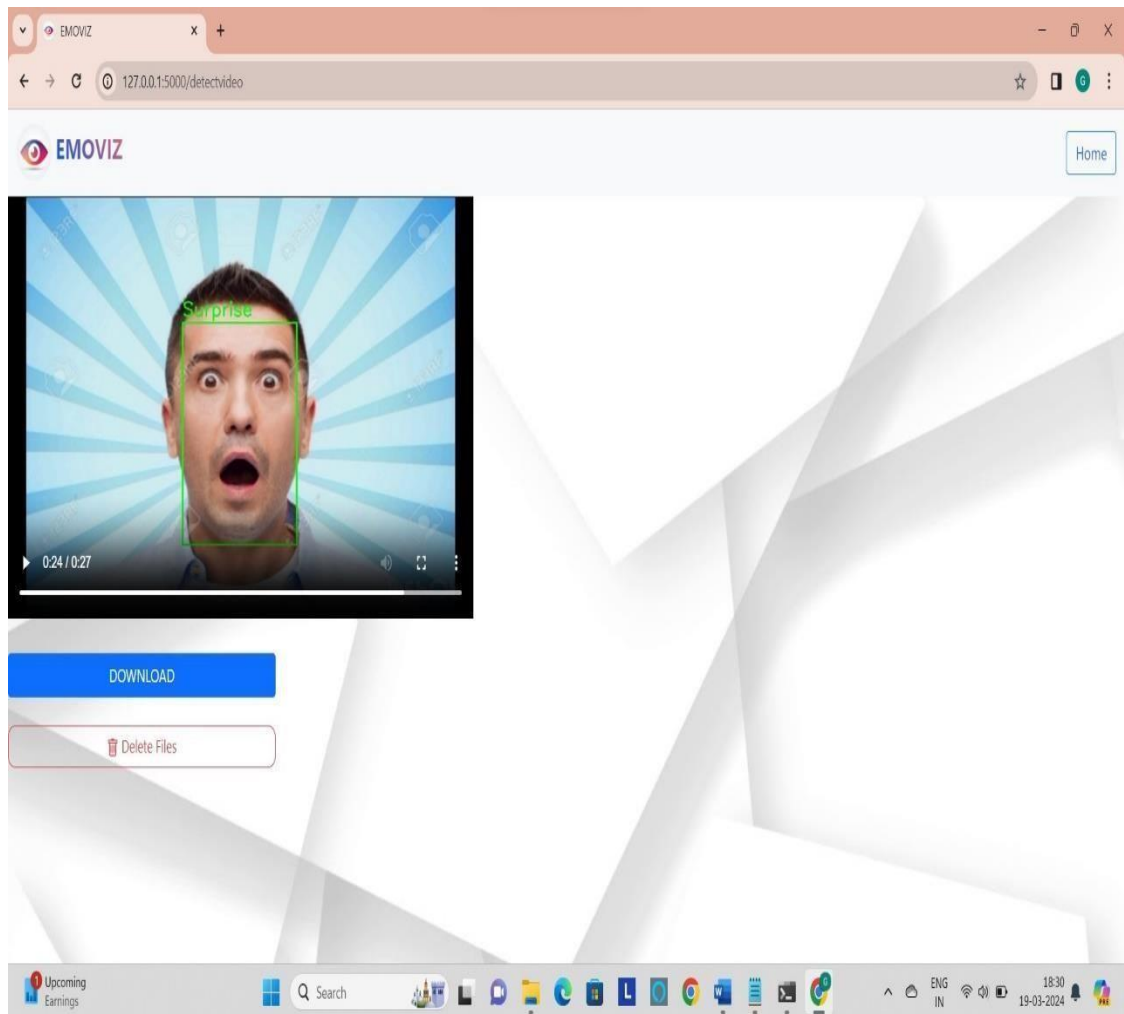


**Fig 7.4.4 SAD**

In the above figure shows sad. Sad emotional state characterized by feelings of disappointment, hopelessness, disinterest, and dampened mood. Sadness is something that all people experience from time to time.

In some cases, people can experience prolonged and severe periods of sadness that can turn into depression. Sadness can be expressed in a number of ways including crying, mood off.

## 7.4.5 SURPRISE



**Fig 7.4.5 SURPRISE**

Above figure shows surprise. Surprise is usually quite brief and is characterized by a physiological startle response following something unexpected.

The range of surprises also varies as some events can give you happiness and others can give you fear if that unexpected event is a threat to us. Physiological changes increased heart rate, breathing are some of the common reactions when encountering such emotions

## **8.CONCLUSION**

This project proposes an approach for recognizing the category of facial expressions. Face Detection and Extraction of expressions from facial images is useful in many applications, such as robotics vision, video surveillance, digital cameras, security and human-computer interaction. This project's objective was to develop a facial expression recognition system implementing the computer visions and enhancing the advanced feature extraction and classification in face expression recognition. In this project, seven different facial expressions of different persons' images from different datasets have been analyzed. This project involves facial expression preprocessing of captured facial images followed by feature extraction using feature extraction using Local Binary Patterns and classification of facial expressions based on training of datasets of facial images based on CNN.



## 9.REFERENCES

- K. K. Kumar, S. G. B. Kumar, S. G. R. Rao and S. S. J. Sydulu, "Safe and high secured ranked keyword search over an outsourced cloud data," 2017 International Conference on Inventive Computing and Informatics (ICICI), Coimbatore, India, 2017, pp. 20-25, doi: 10.1109/ICICI.2017.8365348.
- Tzirakis, George Trigeorgis, Mihalakis A. Nicolaou, Panagiotis, Björn W. Schuller, and Stefanos Zafeiriou. "End-to-end multi-modal emotion recognition using neural networks."
  - IEEE Journal of Topics in Signal Processing 11, no. 8: 1301-1309, 2017
- Gampala, V., Kumar, M.S., Sushama, C. and Raj, E.F.I., 2020. Deep learning-based image processing approaches for image deblurring. Materials Today: Proceedings.
- K. K. Kommineni and A. Prasad, "A Review on Privacy and Security Improvement Mechanisms in MANETs", Int J Intel Syst Appl Eng, vol. 12, no. 2, pp. 90–99, Dec. 2023.
- Natarajan, V.A., Kumar, M.S., Patan, R., Kallam, S. and Mohamed, M.Y.N., 2020, September. Segmentation of Nuclei in Histopathology images using Fully Convolutional Deep Neural Architecture. In 2020 International Conference on Computing and Information Technology (ICCIT1441)
- J. Nicholson, K. Takahashi, and R. Nakatsu. Emotion recognition in speech using neural networks. Neural computing applications, 9(4): 290-296, 2000.
- B. Fasel and J. Letting. Automatic facial expression analysis: a survey. Pattern recognition, 36(1):259-275, 2003.
- Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images, 2009.
- J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 248-255. IEEE, 2009.
- Kiran Kumar Kommineni, Ratna Babu Pilli, K. Tejaswi, P. Venkata Siva, Attention-based Bayesian inferential imagery captioning maker, Materials Today: Proceedings, 2023, , ISSN 2214-7853, <https://doi.org/10.1016/j.matpr.2023.05.231>.
- Y. Lv, Z. Feng, and C. Xu. Facial expression recognition via deep learning. In Smart Computing (SMARTCOMP), 2014 International Conference on, pages 303-308. IEEE, 2014.