

B.E. PROJECT REPORT ON

FACIAL EXPRESSION RECOGNITION USING OPENCV

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING
(COMPUTER ENGINEERING)

SUBMITTED BY

MS. PUJA DESHMUKH
MS. DIVYA DHAMANE
MR. RONIT MAKHIJA

Seat No-B190364300
Seat No-B190364304
Seat No-B190364434



Sinhgad Institutes

DEPARTMENT OF COMPUTER ENGINEERING

STES's SMT KASHIBAI NAVALE COLLEGE OF ENGINEERING

VADGAON BK, OFF SINHGAD ROAD, PUNE 411041

SAVITRIBAI PHULE PUNE UNIVERSITY
2022-2023



Sinhgad Institutes

CERTIFICATE

This is to certify that the project report entitles

Facial Expression Recognition Using OpenCV

Submitted by

Ms. Puja Deshmukh

Seat No-B190364300

Ms. Divya Dhamane

Seat No-B190364304

Mr. Ronit Makhija

Seat No-B190364434

is a bonafide work carried out by him/her under the supervision of **Prof. Mr. Suresh Shinde** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering (Computer Engineering)**. This project work has not been earlier submitted to any other Institute or University for the award of any degree.

Prof. Mr. Suresh Shinde
Guide,

Department of Computer Engineering

Prof. R. H. Borhade
Head,

Department of Computer Engineering

Dr. A. V. Deshpande
Principal,
Smt. Kashibai Navale College of Engineering Pune – 41

Place: Pune

Date:

ACKNOWLEDGMENT

It gives us great pleasure in presenting the preliminary project report on "**Facial Expression Recognition Using OpenCV**". With due respect and gratitude we would like to take this opportunity to thank our internal guide **Prof Mr. Suresh Shinde** for giving us all the help and guidance we needed. We are really grateful for her kind support. She has always encouraged us and given us the motivation to move ahead. She has put in a lot of time and effort in this project along with us and given us a lot of confidence. We are also grateful to **Prof. R. H. Borhade**, Head of Computer Engineering Department, SKN College of Engineering for his indispensable support. Also we wish to thank all the other people who have helped us in the successful completion of this project. We would also like to extend our sincere thanks to Principal Dr. A. V. Deshpande, for her dynamic and valuable guidance throughout the project and providing the necessary facilities that helped us to complete our dissertation work. We would like to thank my colleagues friends who have helped us directly or indirectly to complete this work.

Ms. Puja Deshmukh
Ms. Divya Dhamane
Mr. Ronit Makhija

ABSTRACT

Facial expression recognition is a fascinating field in computer vision that aims to interpret and understand human emotions based on their facial expressions. In recent years, with the advancements in artificial intelligence and machine learning techniques, the development of reliable facial expression recognition systems has become increasingly feasible. This project focuses on implementing a facial expression recognition system using OpenCV, an open-source computer vision library, to accurately identify and classify various facial expressions.

The project begins with an exploration of the fundamental concepts and techniques behind facial expression recognition, including the extraction of facial features, feature representation, and classification algorithms. OpenCV provides a comprehensive set of tools and functions that enable us to process and analyze facial images efficiently.

The first step in the project involves preprocessing the input facial images to enhance their quality and remove any noise or artifacts. This is followed by detecting and extracting facial landmarks using OpenCV's pre-trained face detection models. These landmarks serve as reference points for locating key facial features such as eyes, nose, and mouth.

Next, the project focuses on feature extraction from the facial region of interest. Several techniques are employed, including Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG), and facial action coding system (FACS) based features. These extracted features provide a rich representation of the facial expressions present in the image.

Once the features are extracted, a suitable classification algorithm is employed to recognize and classify the facial expressions. Popular machine learning algorithms such as Support Vector Machines (SVM), Convolutional Neural Networks (CNN), and Decision Trees are explored and compared for their performance in facial expression recognition.

Keywords: facial expression recognition, OpenCV, computer vision, emotions, artificial intelligence, machine learning, feature extraction, face detection

Contents

LIST OF FIGURES	vii
LIST OF TABLES	viii
.	
1 INTRODUCTION	1
1.1 INTRODUCTION	2
1.2 Overview	2
1.3 Motivation	2
1.4 Problem Definition	3
2 LITERATURE REVIEW	4
2.1 Literature Survey	5
3 SOFTWARE REQUIREMENTS SPECIFICATION	7
3.1 Assumptions and Dependencies	8
3.2 Functional Requirements	8
3.2.1 System Feature 1(Functional Requirement).	8
3.3 External Interface Requirements	8
3.3.1 User Interfaces	8
3.3.2 Hardware Interfaces	8
3.3.3 Software Interfaces	8
3.3.4 Communication Interfaces	9
3.4 Nonfunctional Requirements	9
3.4.1 Performance Requirements	9
3.4.2 Safety Requirements	9
3.4.3 Security Requirements	9
3.4.4 Software Quality Attributes	9
3.5 System Requirements	10
3.5.1 Database Requirements	10
3.5.2 Software Requirements(Platform Choice)	10
3.5.3 Hardware Requirements	10
3.6 Analysis Models: SDLC Model to be applied	11

3.7	System Implementation Plan	13
4	SYSTEM DESIGN	14
4.1	System Architecture	15
4.2	Data Flow Diagrams:	15
4.2.1	Level 0 data flow diagram	15
4.3	ER Diagram:	18
4.4	UML Diagrams.....	18
4.4.1	Use case Diagram.....	18
4.4.2	Class Diagram :.....	20
4.4.3	Activity Diagram:.....	21
4.4.4	Sequence Diagram:	23
4.4.5	Component Diagram	24
4.4.6	Deployment Diagram	25
5	Project Plan	26
5.1	Project Estimates	27
5.1.1	Reconciled Estimates	27
5.1.2	Project Resources	27
5.2	Risk Management.....	27
5.2.1	Risk Identification	27
5.2.2	Risk Analysis.....	28
5.3	Project Schedule	28
5.3.1	Project task set.....	28
5.3.2	Task network	29
5.3.3	Timeline Chart.....	30
5.3.4	Timeline Chart.....	31
5.4	Team Organization	31
5.4.1	Team structure	31
5.4.2	Management reporting and communication.....	32
6	Project Implementation	33
6.1	Overview of Project Modules	34
6.2	Tools and Technologies Used.....	34
6.2.1	Technology Description	34
6.2.2	Hardware Specifications	34
6.2.3	Software Specifications.....	35

7 Software Testing	36
7.1 Types Of Testings	37
7.1.1 Unit testing	37
7.1.2 Integration testing	37
7.1.3 Functional test	37
7.1.4 System Test	38
7.1.5 White Box Testing	38
7.1.6 Black Box Testing	38
7.1.7 Unit Testing:	38
7.1.8 Integration Testing	39
7.1.9 Acceptance Testing	39
7.2 Test cases and Test Results	39
8 Results	43
8.1 Screenshots.....	43
9 Results	44
10 OTHER SPECIFICATION	45
10.1 Advantages.....	46
10.2 Limitations.....	46
10.3 Applications.....	46
11 CONCLUSIONS	47
11.1 Conclusions.....	48
11.2 REFERENCES.....	49
12 Certificates	50

List of Figures

4.1	System Architecture	15
4.2	DFD Level 0	16
4.3	DFD Level 1	16
4.4	DFD Level 2	17
4.5	ER Diagram.....	18
4.6	USER USE CASE.....	19
4.7	Class Diagram.....	20
4.8	Activity Diagram.....	22
4.9	Sequence Diagram	23
4.10	Component Diagram	24
4.11	Deployment Diagram	25
5.1	Task network	29
5.2	Timeline Chart.....	30
5.3	Timeline Chart.....	31
5.4	Team structure	31

List of Tables

3.1	System Implementation Plan	13
5.1	Risk Analysis	28
5.2	Management reporting and communication	32
7.1	Test Cases.....	40
7.2	Test Cases.....	42

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

1.2 Overview

Human face detection is the most promising field of image processing that has a vast area of research oriented real life applications. In the real world the concept is widely used for the content annotation, access control, profiling and potential discrimination in the web world. There is always constructive scope of new inventions in the field of technology which is as vast as galaxy on its own. This leads to the better future. There has been a supportive development in the field of technology by the humans since the beginning of mankind. The motive was in rapid development and also in the advancement of technology to ensure the minimization of risk that is prone along with the new inventions which would make life easier, better and much faster. The main intention of face detection is to find out the human face in the given input. The Psychological process of locating the human face in the visual frame is also possible. It is also categorized as a special case of object class detection. The Eigen face approach is considered as a promising technique of face detection. In the field of marketing the facial image detection is playing a role of huge interest for the users. It has always been an issue of personal authentication that needs to be fixed for the purpose of access control of the info-security in the wider context via physical security.

Researchers found that the face detection is an issue that needs to be taken into consideration. In terms of appearance, human face has high degree of variability, making it a dynamic object of study. Application of face detection is found in crowd surveillance, video conferencing, biometrics etc. The concept of human face detection makes it difficult for computer vision. Detected face is stored with high level of secrecy and certainty. Assuring that the data is safe, is the most important aspect under discussion. The image data consists of properties associated with, such as high level of redundancy, bulk capabilities and also high correlation between the pixels.

1.3 Motivation

Facial expressions play a crucial role in human communication and are fundamental to understanding emotions and intentions. The ability to accurately recognize and interpret facial expressions has wide-ranging applications in various fields, including psychology, human-computer interaction, emotion detection, and affective computing. Motivated by the importance of facial expression analysis, this report focuses on exploring the use of OpenCV (Open Source Computer Vision Library) for facial expression recognition.

1.4 Problem Definition

Human emotions and intentions are expressed through facial expressions and deriving an efficient and effective feature is the fundamental component of facial expression system. Face recognition is important for the interpretation of facial expressions in applications such as intelligent, man-machine interface and communication, intelligent visual surveillance, teleconference and real-time animation from live motion images. The facial expressions are useful for efficient interaction. Most research and system in facial expression recognition are limited to six basic expressions (joy, sad, anger, disgust, fear, surprise). It is found that it is insufficient to describe all facial expressions and these expressions are categorized based on facial actions. Detecting face and recognizing the facial expression is a very complicated task when it is a vital to pay attention to primary components like: face configuration, orientation, location where the face is set.

CHAPTER 2

LITERATURE REVIEW

2.1 Literature Survey

Literature survey is the most important step in any kind of research. Before start developing we need to study the previous papers of our domain which we are working and on the basis of study we can predict or generate the drawback and start working with the reference of previous papers.

In this section, we briefly review the related work on Face Recognition and their different techniques. Research in the fields of face detection and tracking has been very active and there is exhaustive literature available on the same. The major challenge that the researchers face is the non-availability of spontaneous expression data. Capturing spontaneous expressions on images and video is one of the biggest challenges ahead. Many attempts have been made to recognize facial expressions. Zhang et al investigated two types of features, the geometry-based features and Gabor wavelets based features, for facial expression recognition.

Appearance based methods, feature invariant methods, knowledge based methods, Template based methods are the face detection strategies whereas Local Binary Pattern phase correlation, Haar classifier, AdaBoost, Gabor Wavelet are the expression detection strategies in related field. Face reader is the premier for automatic analysis of facial expression recognition and Emotient, Affectiva, Karios etc are some of the API's for expression recognition. Automatic facial expression recognition includes two vital aspects: facial feature representation and classifier problem.

Facial feature representation is to extract a set of appropriate features from original face images for describing faces. Histogram of Oriented Gradient (HOG), SIFT, Gabor Filters and Local Binary Pattern (LBP) are the algorithms used for facial feature representation. LBP is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number. The operator labels the pixels of an image by thresholding the 3X3 neighborhood of each pixel with the center value and considering the result as a binary number. HOG was first proposed by Dalal and Triggs in 2005. HOG numerates the appearance of gradient orientation in a local path of an image.

For classifier problem we use algorithms like Machine learning, Neural Network, Support Vector Machine, Deep learning, Naive Bayes. The formation of histogram by using any of facial feature representation will use Support Vector Machine

(SVM) for expression recognition. SVM builds a hyperplane to separate the high dimensional space. An ideal separation is achieved when the distance between the hyper plane and the training data of any class is the largest.

The size of the block for the LBP feature extraction is chosen for higher recognition accuracy. The testing results indicate that by using LBP features facial expressions recognition accuracy is more than 97%. The block LBP histogram features extract local as well as global features of face image resulting higher accuracy. LBP is compatible with various classifiers, filters etc.

CHAPTER 3

**SOFTWARE REQUIREMENTS
SPECIFICATION**

3.1 Assumptions and Dependencies

- User must know English Language.
- The device has a 64 bit architecture.
- The User and the device on which the app is running must be indoors.

3.2 Functional Requirements

3.2.1 System Feature 1(Functional Requirement).

1. Face Recognition system

3.3 External Interface Requirements

3.3.1 User Interfaces

Home page

Upload Image Page

Image Prediction Page

Use Webcam Page

3.3.2 Hardware Interfaces

The entire software requires a completely equipped computer system including monitor, keyboard, and other input output devices.

3.3.3 Software Interfaces

The system can use Microsoft as the operating system platform. System also makes use of certain GUI tools. To run this application we need python and above as java platform and Apache tomcat as server. To store data we need MySQL database.

3.3.4 Communication Interfaces

Communication using python APIs

3.4 Nonfunctional Requirements

3.4.1 Performance Requirements

The performance of the system lies in the way it is handled. Every user must be given proper guidance regarding how to use the system. The other factor which affects the performance is the absence of any of the suggested requirements.

3.4.2 Safety Requirements

To ensure the safety of the system, perform regular monitoring of the system so as to trace the proper working of the system. An authenticated user is only able to access system.

3.4.3 Security Requirements

Any unauthorized user should be prevented from accessing the system. Password authentication can be introduced.

3.4.4 Software Quality Attributes

Accuracy: -

The level of accuracy in the proposed system will be higher. All operation would be done correctly and it ensures that whatever information is coming from the center is accurate. Result is organic results.

Reliability: -

The reliability of the proposed system will be high due to the above stated reasons. The reason for the increased reliability of the system is that now there would be proper storage of information and Recommending location model.

3.5 System Requirements

3.5.1 Database Requirements

MySQL Database

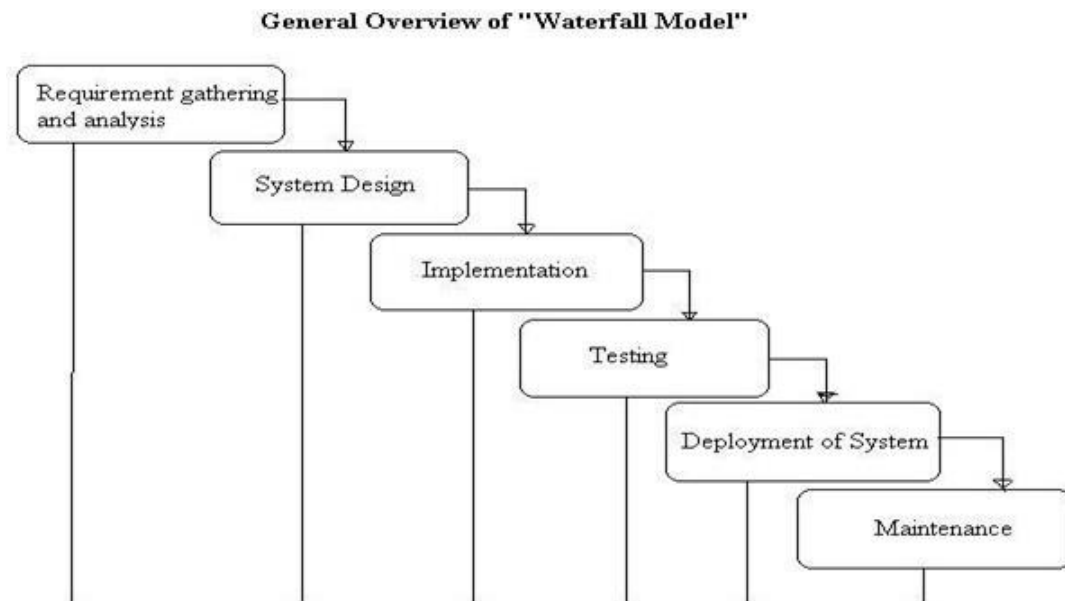
3.5.2 Software Requirements (Platform Choice)

- Operating System - Windows
- Language - Python.
- IDE - VS Code.

3.5.3 Hardware Requirements

- Processor - I3/I5/I7
- Speed - 3.1 GHz
- RAM - 4 GB
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

3.6 Analysis Models: SDLC Model to be applied



The Waterfall Model is sequential design process, often used in Software development processes, where progress is seen as flowing steadily down through the phase of conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance. This Model is also called as the classic Life cycle model as it suggests a systematic sequential approach to software developments. This one of the oldest model followed in software engineering. The process begins with the communication phase where the customer specifies the requirements and then progress through other phases like planning, modeling, construction and deployment of the software.

There are 5 Phase of water fall model:

The Waterfall Model is sequential design process, often used in Software development processes, where progress is seen as flowing steadily down through the phase of conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance. This Model is also called as the classic Life cycle model as it suggests a systematic sequential approach to software developments. This one of the oldest model followed in software engineering. The process begins with the communication phase where the customer specifies the requirements and then progress through other phases like planning, modeling, construction and deployment of the software.

There are 5 Phase of water fall model:

1. COMMUNICATION

In communication phase the major task performed is requirement gathering which helps in finding out exact need of customer. Once all the needs of the customer are gathered the next step is planning.

2. PLANNING

In planning major activities like planning for schedule, keeping tracks on the processes and the estimation related to the project are done. Planning is even used to find the types of risks involved throughout the projects. Planning describes how technical tasks are going to take place and what resources are needed and how to use them.

3. MODELING

This is one of the important phases as the architecture of the system is designed in this phase. Analysis is carried out and depending on the analysis a software model is designed. Different models for developing software are created depending on the requirements gathered in the first phase and the planning done in the second phase.

4. CONSTRUCTION

The actual coding of the software is done in this phase. This coding is done on the basis of the model designed in the modeling phase. So in this phase software is actually developed and tested.

5. DEPLOYMENT

In this last phase the product is actually rolled out or delivered installed at customer's end and support is given if required. A feedback is taken from the customer to ensure the quality of the product. From the last two decades Waterfall model has come under a lot of criticism due to its efficiency issues. So let's discuss the advantages and disadvantages of waterfall model.

3.7 System Implementation Plan

Schedule		Date	Project Activity
July	1 st Week	01/07/2022	Project Topic Searching
	2 nd Week	08/07/2022	Project Topic Selection
	3 rd Week	15/07/2022	Synopsis Submission
August	1 st Week	05/08/2022	Presentation On Project Ideas
	2 nd Week	12/08/2022	Submission Of Literature Survey
	3 rd Week	19/08/2022	Feasibility Assessment
September	1 st Week	02/09/2022	Documentation for paper publishing.
	3 rd Week	16/09/2022	Design Of Mathematical Model
	4 th Week	23/09/2022	Paper is publish.
October	1 st Week	09/10/2022	Report Preparation And Submission
December	3 rd Week	19/12/2022	1 st module presentation
	4 th Week	26/12/2022	Discussion and implementation of 2 nd module
January	1 st Week	02/01/2023	Preparation for conference
	2 nd Week	09/01/2023	Study of algorithm.
	3 rd Week	16/01/2023	Discussion about modification.
	4 th Week	23/01/2023	1 st and 2 nd module presentation
	5 th Week	30/01/2023	Discussion on flow of project and designing new module
February	1 st Week	06/02/2023	Modification of modules.
	2 nd Week	13/02/2023	Designed test cases for our module.
	3 rd Week	20/02/2023	Worked on user interface.
March	1 st Week	06/03/2023	Integration of all modules.
April	1 st Week	8/04/2023	Final Report.
May	1 st Week	10/05/2023	Final Presentation.

Table 3.1: System Implementation Plan

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture

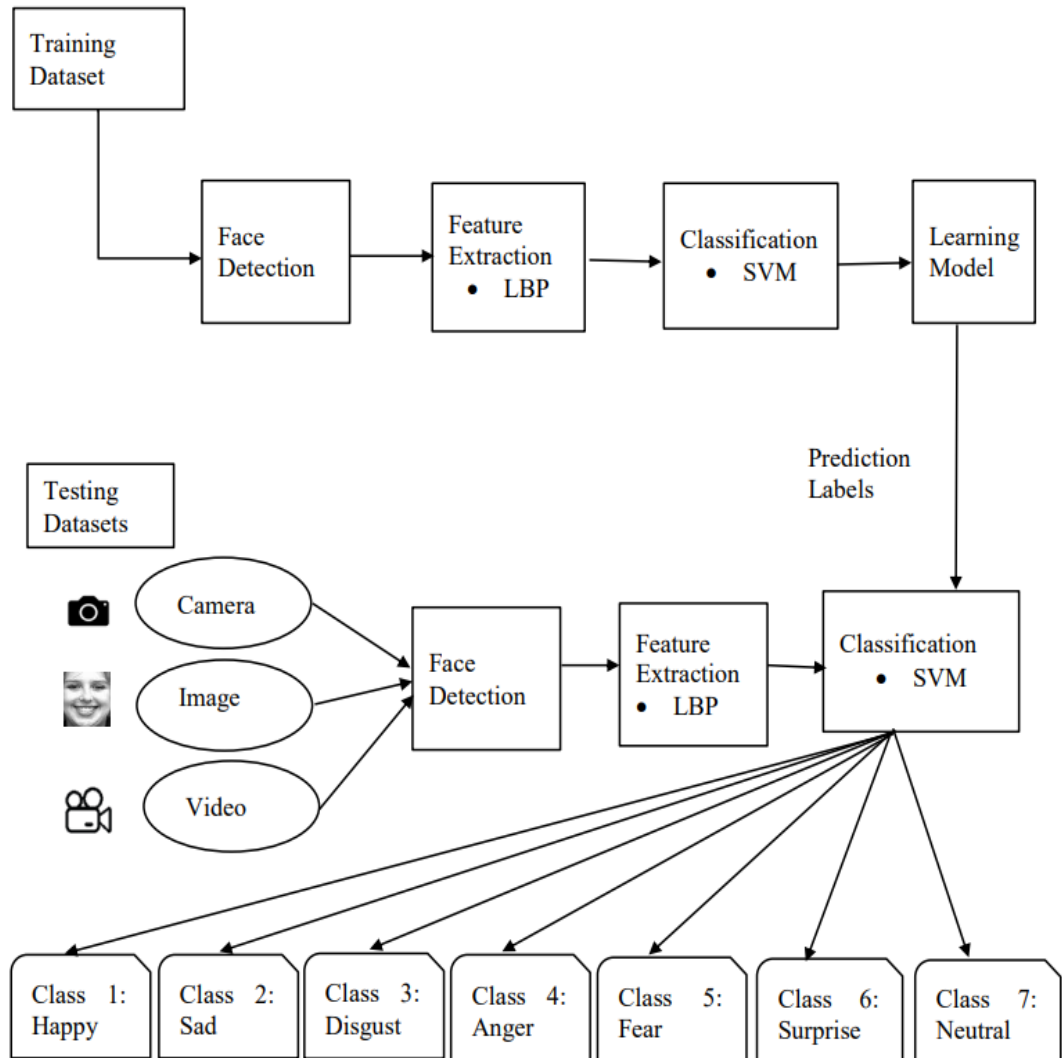


Fig. 4.1: System Architecture

4.2 Data Flow Diagrams:

4.2.1 Level 0 data flow diagram

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. Figure 4.1 shows level 0 DFD which shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

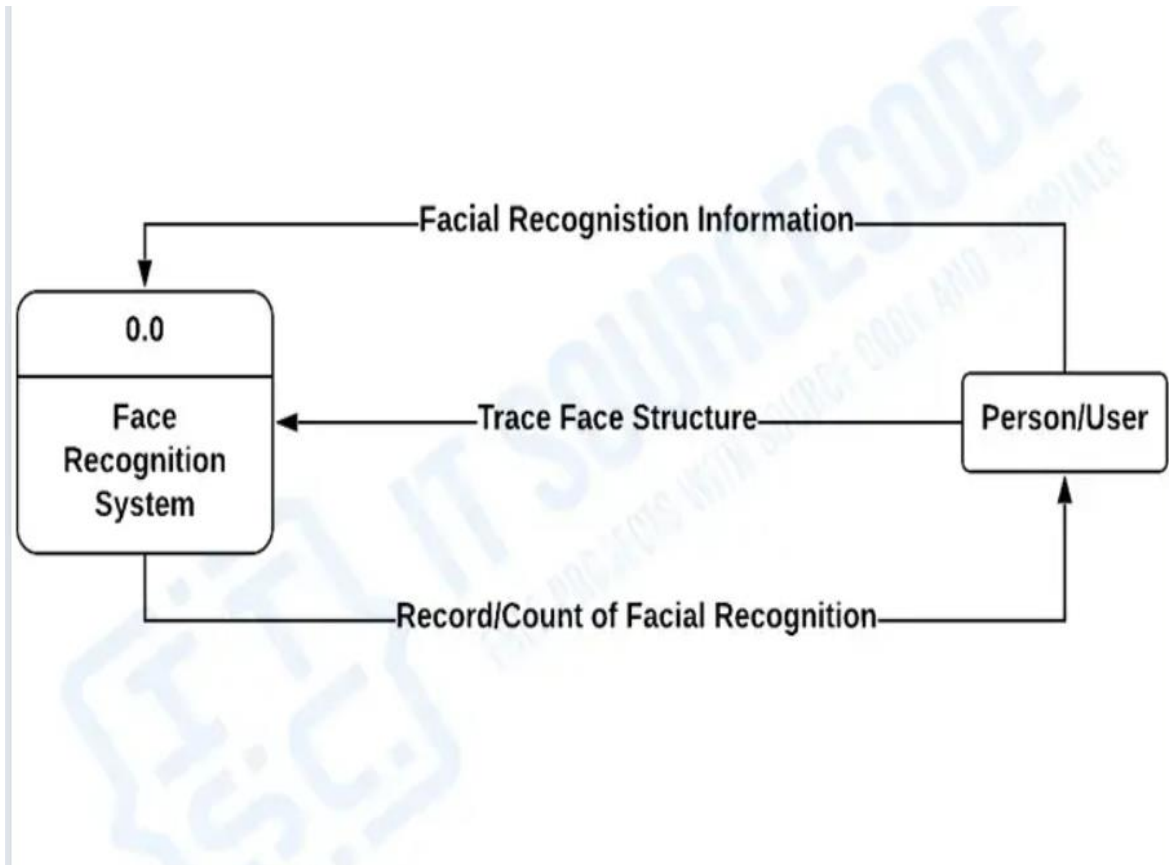


Fig. 4.2: DFD Level 0

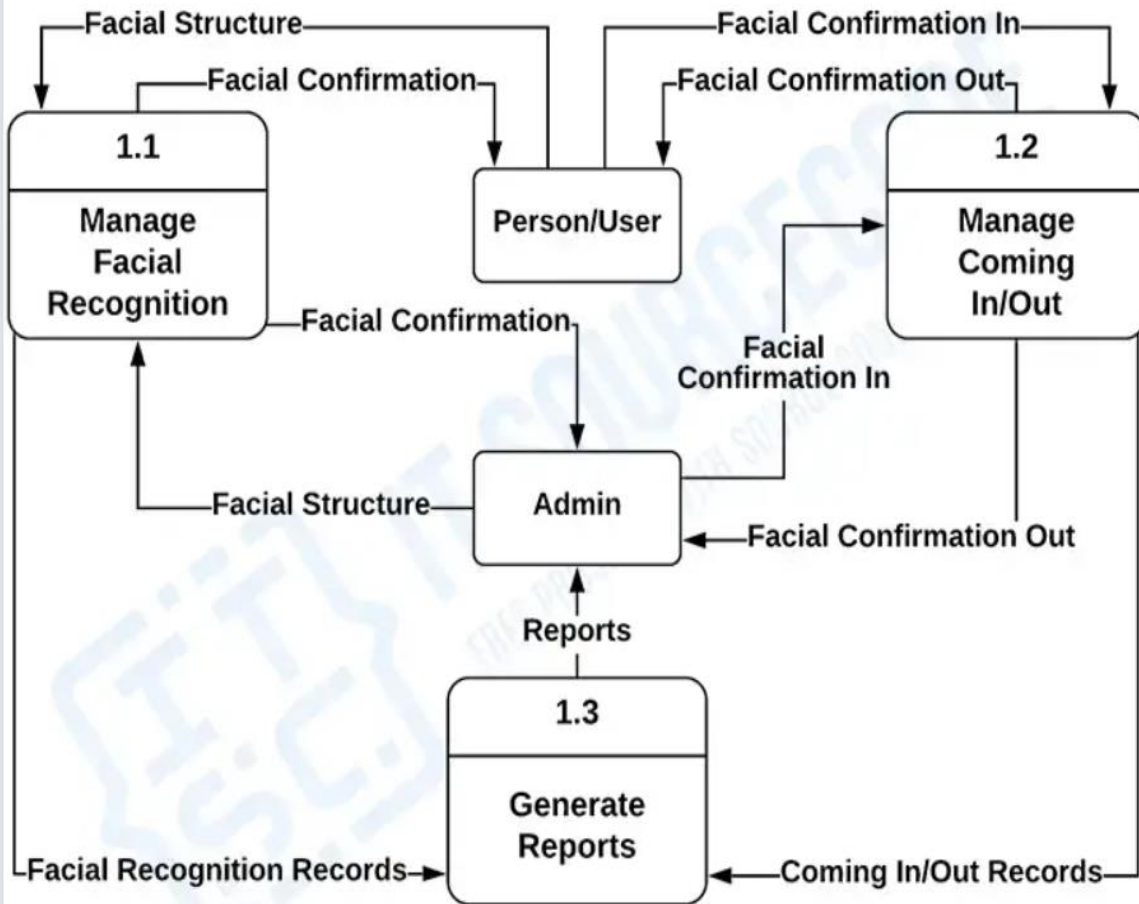


Fig. 4.3: DFD Level 1

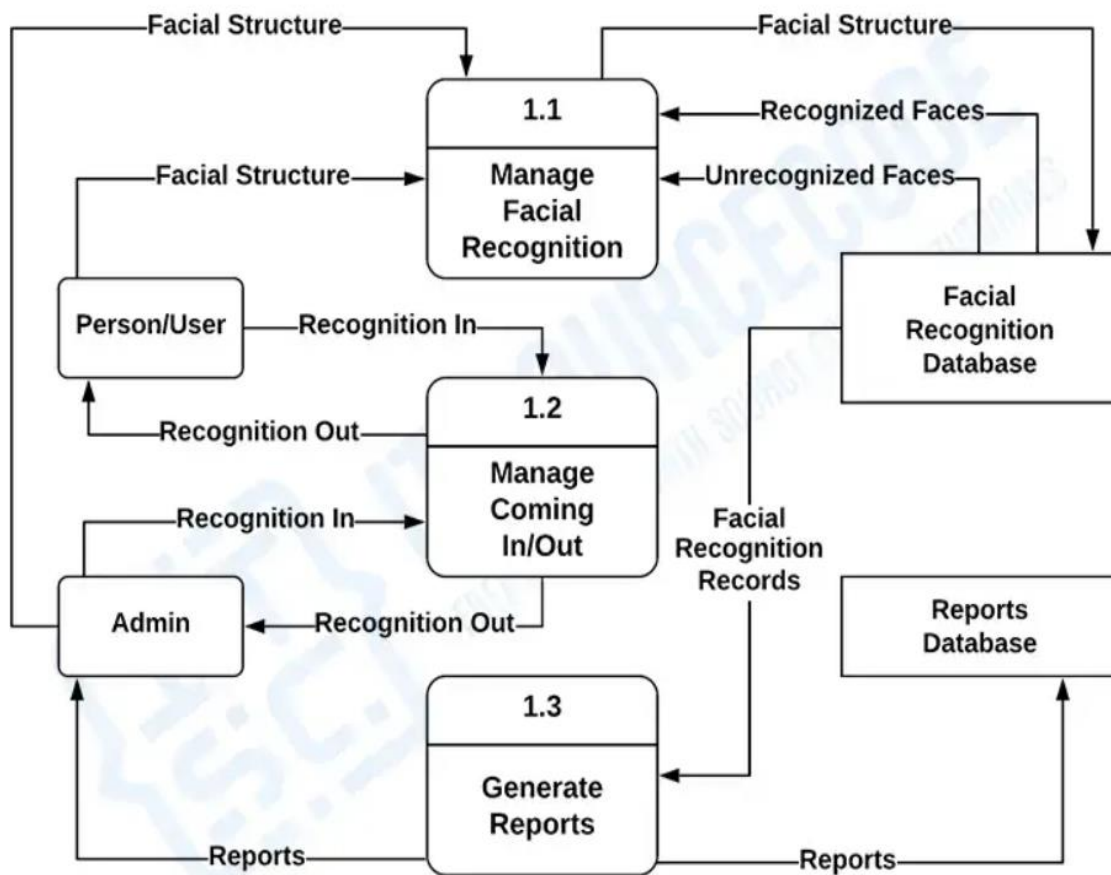


Fig. 4.4: DFD Level 2

4.3 ER Diagram:

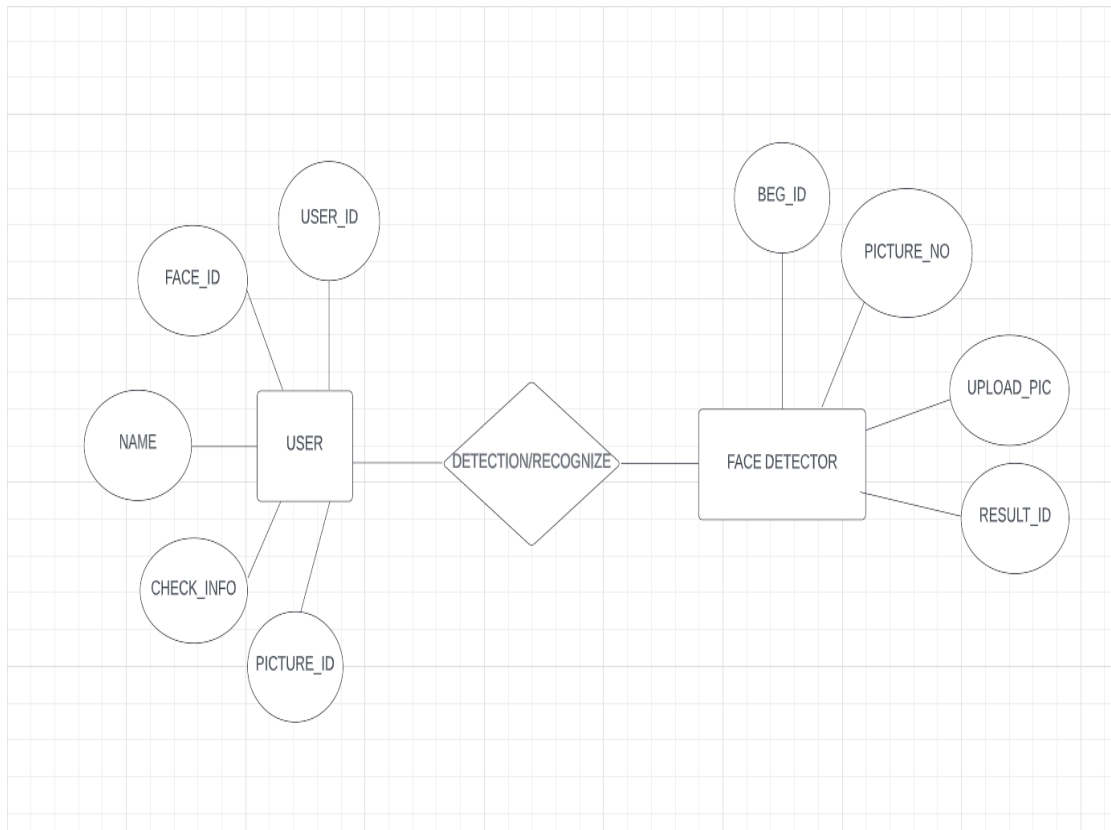


Fig. 4.5: ER Diagram

4.4 UML Diagrams

4.4.1 Use case Diagram

A use case diagram is a graphical representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can show the different types of users of a system and the various ways in which they interact with the system. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. So when a system is analyzed to gather its functionality use cases are prepared and actors are identified. The purposes of use case diagrams can be as follows:

- Used to gather requirements of a system.

- Used to get an outside view of a system.
- Identify external and internal factors influencing the system.
- Show the interaction among the actors.

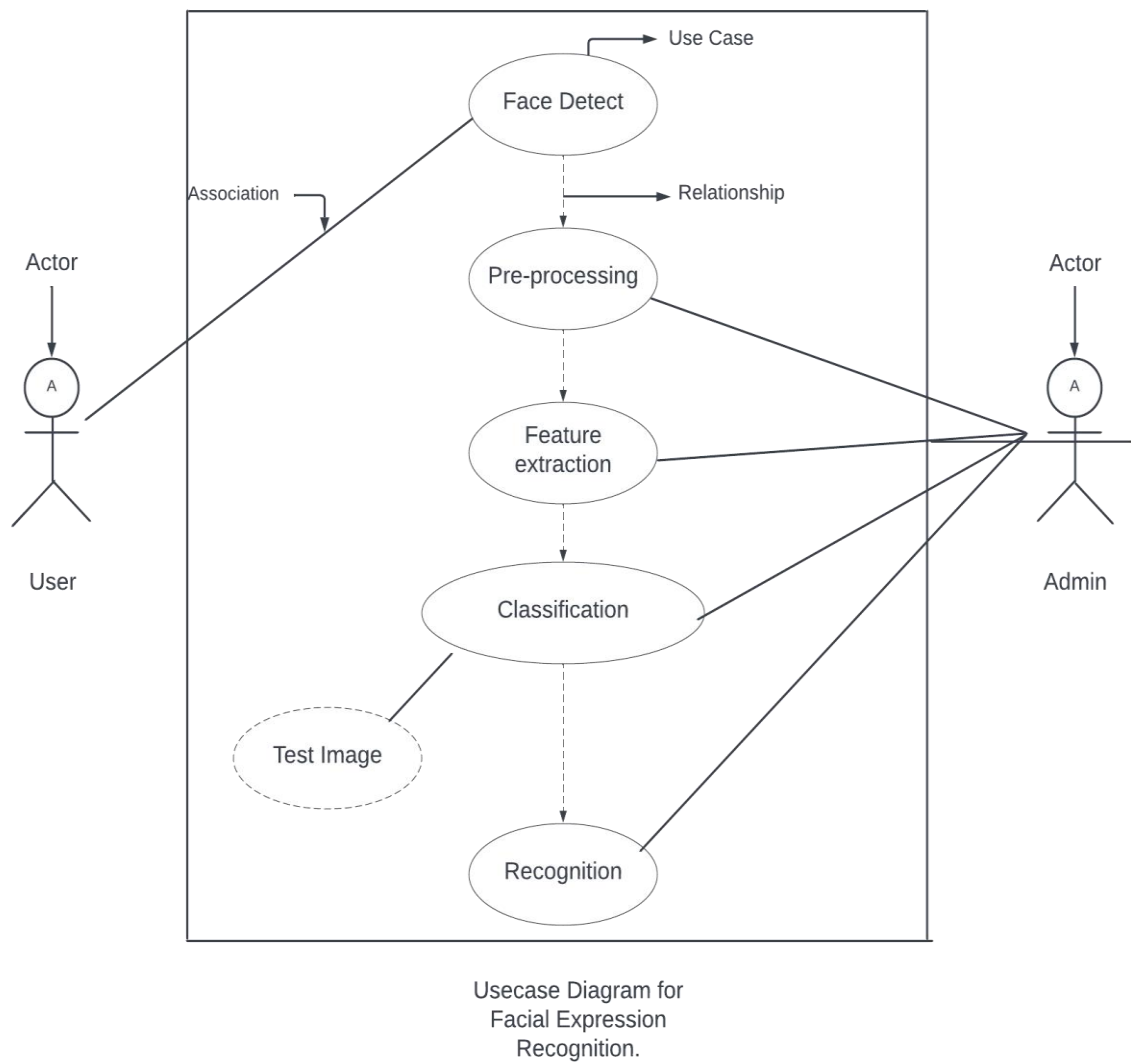


Fig. 4.6: USER USE CASE

4.4.2 Class Diagram :

The class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application. The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams which can be mapped directly with object oriented languages. The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as a structural diagram. The purpose of the class diagram is to model the static view of an application.

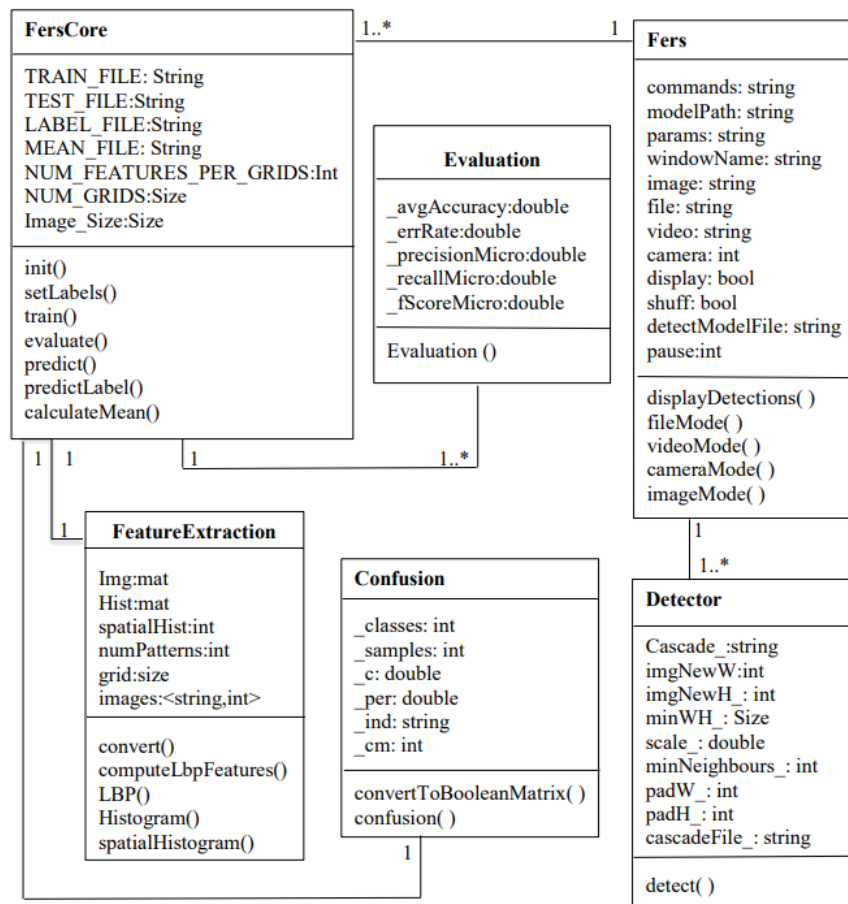


Fig. 4.7: Class Diagram

4.4.3 Activity Diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e. workflows). Activity diagrams show the overall flow of control. Activity diagrams are constructed from a limited number of shapes, connected with arrows. The most important shape types:

- Rounded rectangles represent actions;
- Diamonds represent decisions;
- Bars represent the start (split) or end (join) of concurrent activities;
- A black circle represents the start (initial state) of the workflow;
- An encircled black circle represents the end (final state).

Arrows run from the start towards the end and represent the order in which activities happen. Hence they can be regarded as a form of flowchart. Typical flowchart techniques lack constructs for expressing concurrency. However, the join and split symbols in activity diagrams only resolve this for simple cases; the meaning of the model is not clear when they are arbitrarily combined with decisions or loops.

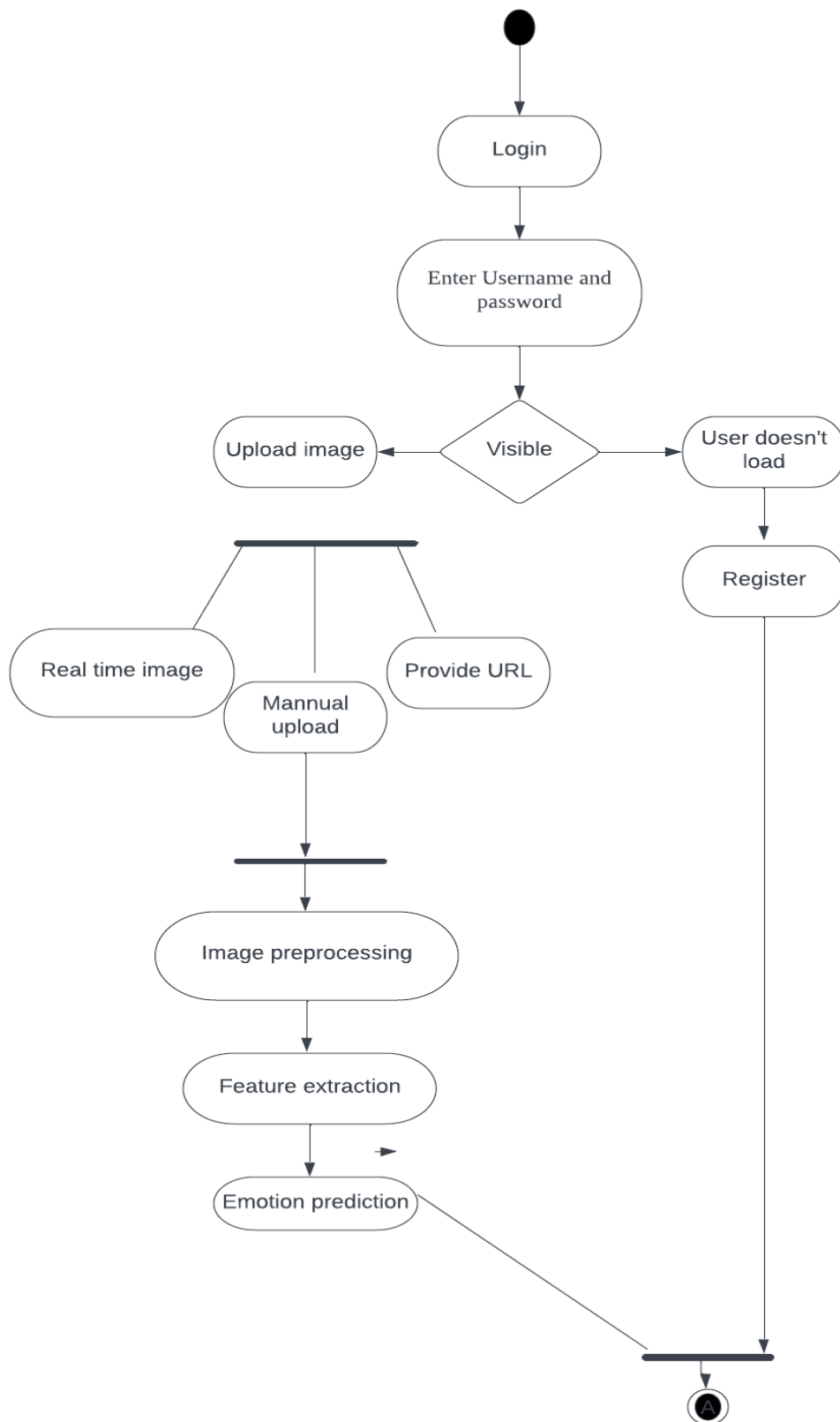


Fig. 4.8: Activity Diagram

4.4.4 Sequence Diagram:

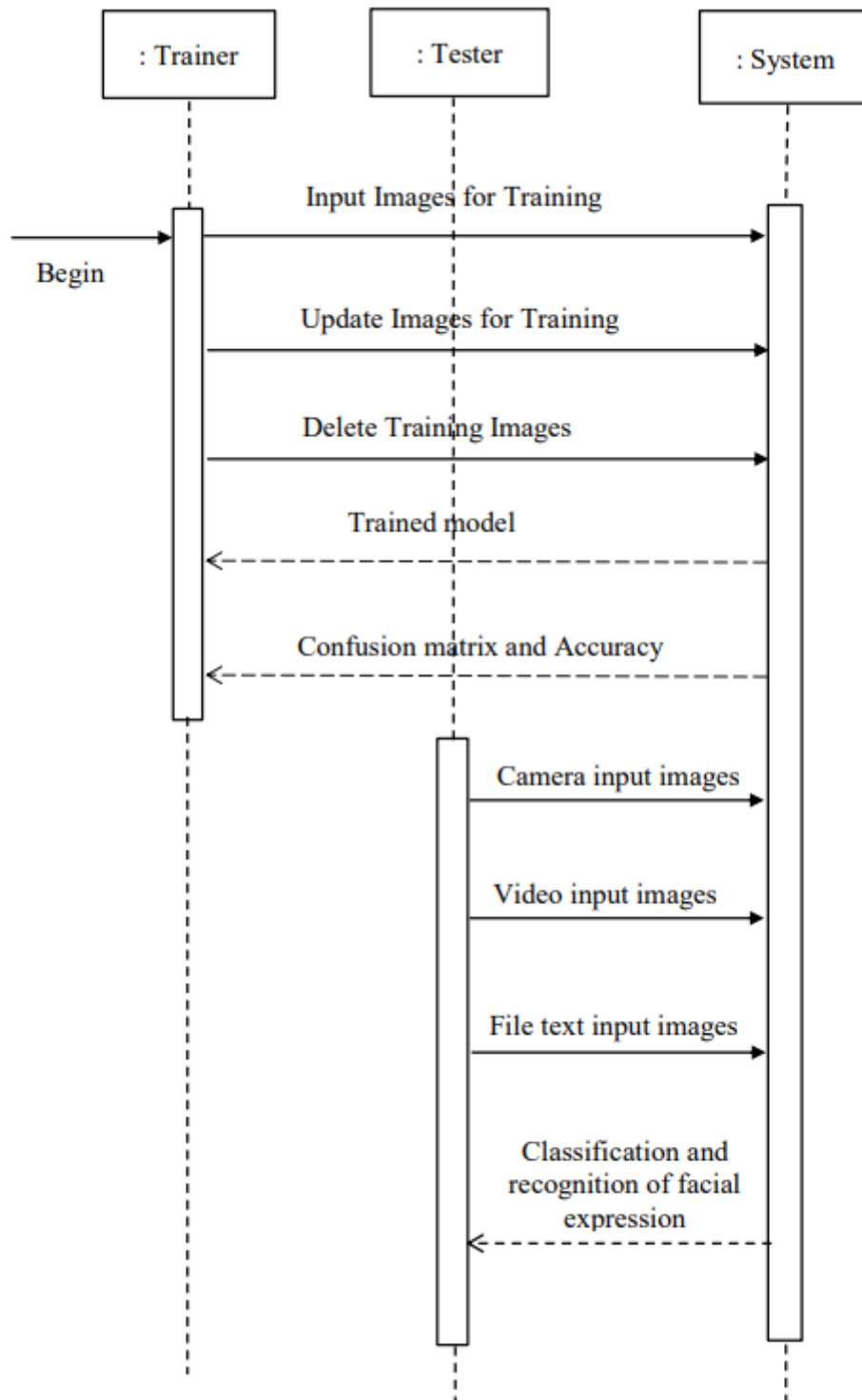


Fig. 4.9: Sequence Diagram

4.4.4 Component Diagram

A Component Diagram displays the structural relationship of components of a software system. These are mostly used when working with complex systems that have many components. Components communicate with each other using interfaces. The interfaces are linked using connectors.

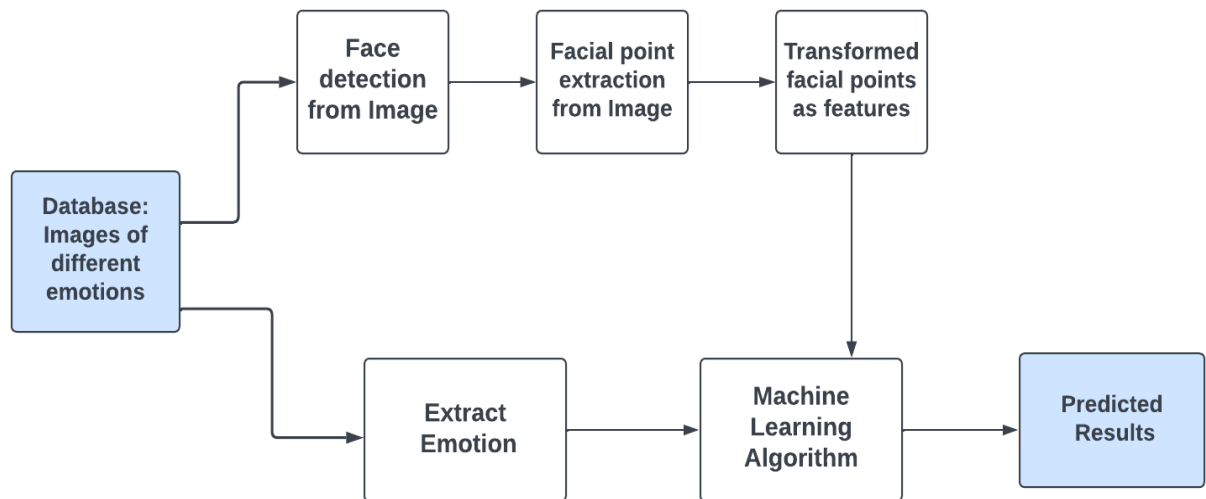


Fig. 4.10: Component Diagram

4.4.5 Deployment Diagram

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed. So deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

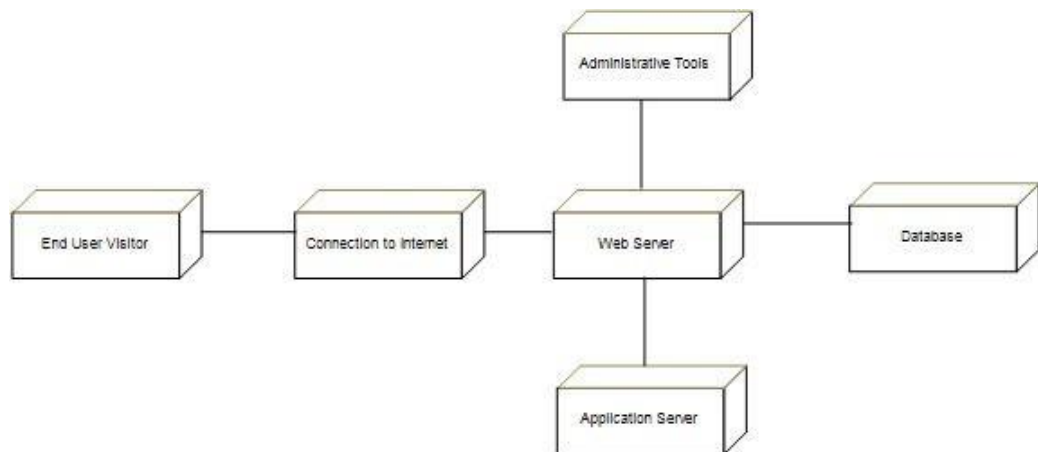


Fig. 4.11: Deployment Diagram

CHAPTER 5

PROJECT PLAN

5.1 Project Estimates

5.1.1 Reconciled Estimates

Cost Estimate

Cost will estimate after completing the project that depend on time to complete the project. Also efforts required to complete.

Time Estimates

Time will depend on modules of project. Also project plan of execution.

5.1.2 Project Resources

1. Hardware Resources Required

System: Pentium IV 2.4 GHz. Hard Disk: 40 GB. Floppy Drive: 44 Mb.
Monitor: 15 VGA Color.

2. Software Resources Required

Operating system: Windows. Coding Language: python Database: MySql 5
IDE: VS code

5.2 Risk Management

5.2.1 Risk Identification

For risks identification, review of scope document, requirements specifications and schedule is done. Answers to questionnaire revealed some risks. Each risk is categorized as per the categories mentioned in [?]. Please refer table?? for all the risks. You can refereed following risk identification questionnaire.

1. Have top software and customer managers formally committed to support the project?

Answer: Yes, Have top software and customer managers formally committed to support the project

2. Are end-users enthusiastically committed to the project and the system/product to be built?

Yse, end-users enthusiastically committed to the project and the system/product to be built

3. Are requirements fully understood by the software engineering team and

its customers?

Yes, Are requirements fully understood by the software engineering team and its customers

5. Have customers been involved fully in the definition of requirements?

Yes, customers been involved fully in the definition of requirements

6. Are project requirements stable?

Answer: all project requirements are stable

9. Is the number of people on the project team adequate to do the job?

Yes, the number of people on the project team adequate to do the job

5.2.2 Risk Analysis

DESCRIPTION	LOW	High
Login detail	no	yes
Internet connection slow	yes	no

Table 5.1: Risk Analysis

Following are the details for each risk.

Risk ID 1

Risk Description Description 1

1.While login to system validation is there so user must follow rules and proper entries

Risk ID 2

1.While registering to system internet connection should be there for registration to get users data online

5.3 Project Schedule

5.3.1 Project task set

Major Tasks in the Project stages are:

- Task 1: Requirement Analysis (Base Paper Explanation).
- Task 2: Project Specification (Paper Work).
- Task 3: Technology Study and Design.

-
- Task 4: Coding and Implementation (Module Development).

5.3.2 Task network

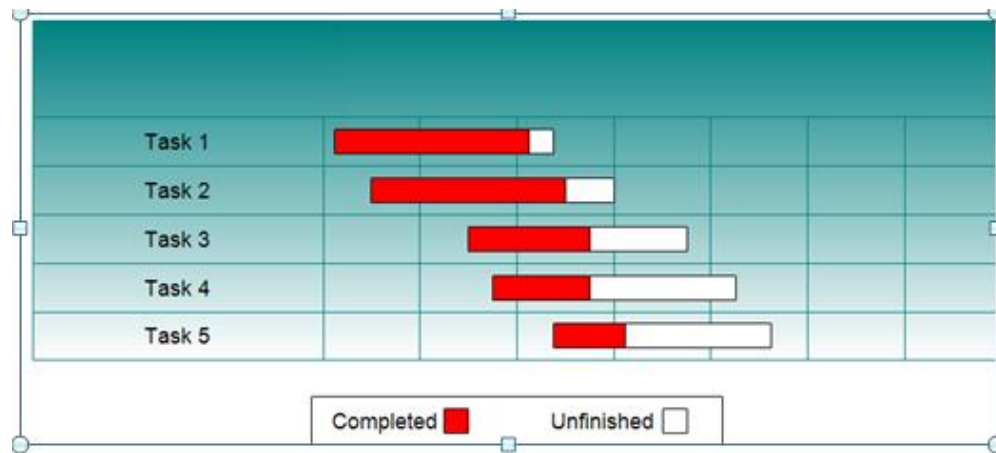


Fig. 5.1: Task network

5.3.3 Timeline Chart

Activity	I week	II week	III week	IV week	V Week	VI week	VII week	VIII week	IX week
	Aug 4	Aug 11	Aug 18	Aug 25	Sept 1	Sept 8	Sept 15	Sept 22	Sept 29
Initiate the project									
Communication									
Literature survey									
Define scope									
Develop SRS									
Plan the project									
Design mathematical model									
Feasibility Analysis									
Develop work breakdown structure									
Planning project schedule									
Design UML and other diagrams									
Design test plan									
Design risk management plan									

Fig. 5.2: Timeline Chart

5.3.3 Timeline Chart

Activity	XI week	XII week	XII I week	XIV week	XV week	XVI week	XVI I week	XVI II week	XIX week	XX week	XXI week	XXII week
	Jan 5	Jan 15	Jan 19	Jan 26	Feb 2	Feb 9	Feb 16	Feb 23	Mar 2	Mar 9	Mar 16	April 25
Execute the project												
Build and test basic functional unit												
Build and test database with login and session maintenance facility												
Build and test Bluetooth mode												
Build and test security features												

Fig. 5.3: Timeline Chart

5.4 Team Organization

5.4.1 Team structure

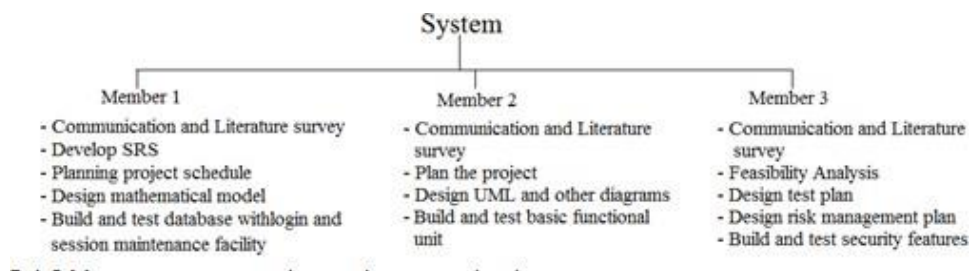



Fig. 5.4: Team structure

5.4.2 Management reporting and communication



Sr No.	Month	Description
1	June	Discussion with guide regarding domain. Searching for IEEE paper for domain.
2	July	Short listing of IEEE papers within domain. Selection of IEEE paper.
3	August	Deciding Project name. Submission of Synopsis.
4	September	Requirement analysis. Designing of models.
5	October	Report preparation. Stage-I report submission.




Table 5.2: Management reporting and communication

CHAPTER 6

PROJECT IMPLEMENTATION

6.1 Overview of Project Modules

1. Data Collection
2. Preprocessing
3. Feature Extraction
4. Classification

6.2 Tools and Technologies Used

6.2.1 Technology Description

In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes the machine language of the Java Virtual Machine¹ (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

6.2.2 Hardware Specifications

- Processor - I3,I5
- Speed - 3.8 GHz
- RAM - 4GB
- Hard Disk - 1 TB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - LCD(Liquid Crystal Display)

6.2.3 Software Specifications

- Operating System: Windows 10
- Programming Language: Python
- Backend: Mysql 5.0
- IDE : Visual Studio Code
- Tool: OpenCV Framework

CHAPTER 7

SOFTWARE TESTING

7.1 Types of Testings

7.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.1.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.1.5 White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

7.1.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a blackbox .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

7.1.7 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

7.1.8 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

7.1.9 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements. Test Results: All the test cases mentioned above passed successfully. No defects encountered.

7.1 Test cases and Test Results

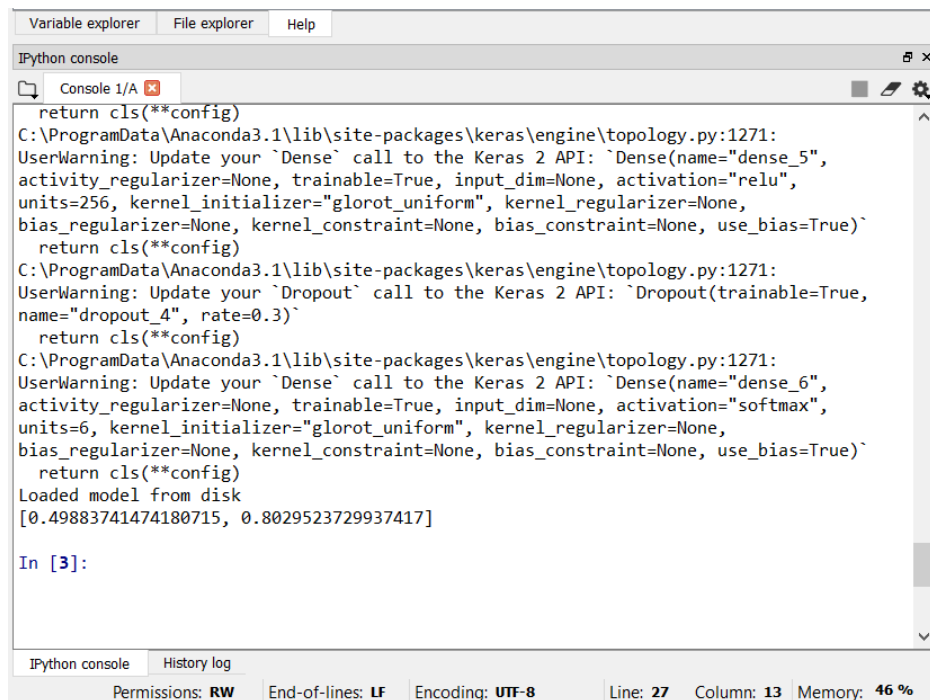
Testing of project problem statement using generated test data (using mathematical models, GUI, Function testing principles, if any) selection and appropriate use of testing tools, testing of UML diagram's reliability.

Test case ID.	Step Description	Expected Result	Actual Result	Pass/Fail
1	User captures the image to be uploaded	Image not taken Properly or Wrong image	Recapture image	Pass
2	User selects a image	The selected image should be opened	The selected document is opened	Pass
3	User selects image of very high resolution or Noisy image.	Prompt the user to enter another image of smaller resolution	Message displaying this is not a Human Face or this image is not proper to find an emotion	Pass
4	User selects image of person with mixed emotions	Best Case Scenerio is taken	Selects single Emotions which is the closest	Pass
5	User selects image displaying correct emotion of the person	Correct Emotion is Displayed	Display the Emotion Box message with the correct gesture.	Pass

Table 7.1: Test Cases

CHAPTER 8

RESULTS

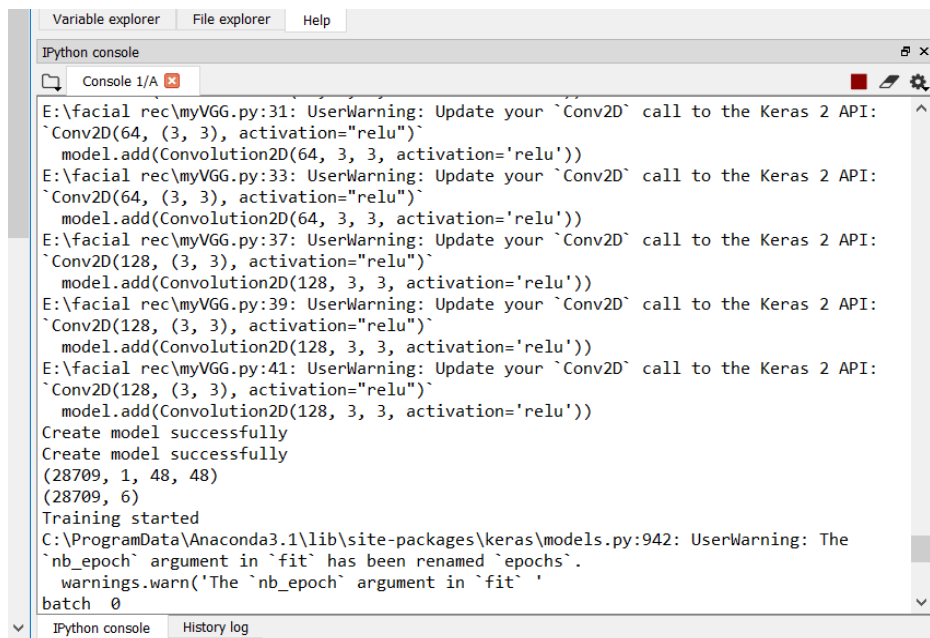


```
Variable explorer  File explorer  Help
IPython console
Console 1/A
return cls(**config)
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\engine\topology.py:1271:
UserWarning: Update your `Dense` call to the Keras 2 API: `Dense(name="dense_5",
activity_regularizer=None, trainable=True, input_dim=None, activation="relu",
units=256, kernel_initializer="glorot_uniform", kernel_regularizer=None,
bias_regularizer=None, kernel_constraint=None, bias_constraint=None, use_bias=True)`
return cls(**config)
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\engine\topology.py:1271:
UserWarning: Update your `Dropout` call to the Keras 2 API: `Dropout(trainable=True,
name="dropout_4", rate=0.3)`
return cls(**config)
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\engine\topology.py:1271:
UserWarning: Update your `Dense` call to the Keras 2 API: `Dense(name="dense_6",
activity_regularizer=None, trainable=True, input_dim=None, activation="softmax",
units=6, kernel_initializer="glorot_uniform", kernel_regularizer=None,
bias_regularizer=None, kernel_constraint=None, bias_constraint=None, use_bias=True)`
return cls(**config)
Loaded model from disk
[0.49883741474180715, 0.8029523729937417]

In [3]:

IPython console  History log
Permissions: RW  End-of-lines: LF  Encoding: UTF-8  Line: 27  Column: 13  Memory: 46 %
```

a) Model Trained



```
Variable explorer  File explorer  Help
IPython console
Console 1/A
E:\facial_rec\myVGG.py:31: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(64, (3, 3), activation="relu")`
model.add(Convolution2D(64, 3, 3, activation='relu'))
E:\facial_rec\myVGG.py:33: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(64, (3, 3), activation="relu")`
model.add(Convolution2D(64, 3, 3, activation='relu'))
E:\facial_rec\myVGG.py:37: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
model.add(Convolution2D(128, 3, 3, activation='relu'))
E:\facial_rec\myVGG.py:39: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
model.add(Convolution2D(128, 3, 3, activation='relu'))
E:\facial_rec\myVGG.py:41: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
model.add(Convolution2D(128, 3, 3, activation='relu'))
Create model successfully
Create model successfully
(28709, 1, 48, 48)
(28709, 6)
Training started
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\models.py:942: UserWarning: The
`nb_epoch` argument in `fit` has been renamed `epochs`.
warnings.warn('The `nb_epoch` argument in `fit` '
batch 0

IPython console  History log
```

b) JSON Model Trained

```
Variable explorer | File explorer | Help
IPython console
Console 1/A
Training set for ['Angry', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral', 'Disgust']:
(28709, 3)
Disgust classified as Angry
C:\ProgramData\Anaconda3.1\lib\site-packages\pandas\core\indexing.py:194:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
0: Angry with 4431 samples
1: Fear with 4097 samples
2: Happy with 7215 samples
3: Sad with 4830 samples
4: Surprise with 3171 samples
5: Neutral with 4965 samples
{'Angry': (0, 4431), 'Fear': (1, 4097), 'Happy': (2, 7215), 'Sad': (3, 4830),
'Surprise': (4, 3171), 'Neutral': (5, 4965)}
Saving...
(28709, 1, 48, 48)
(28709, 6)
Done!

In [6]:
```

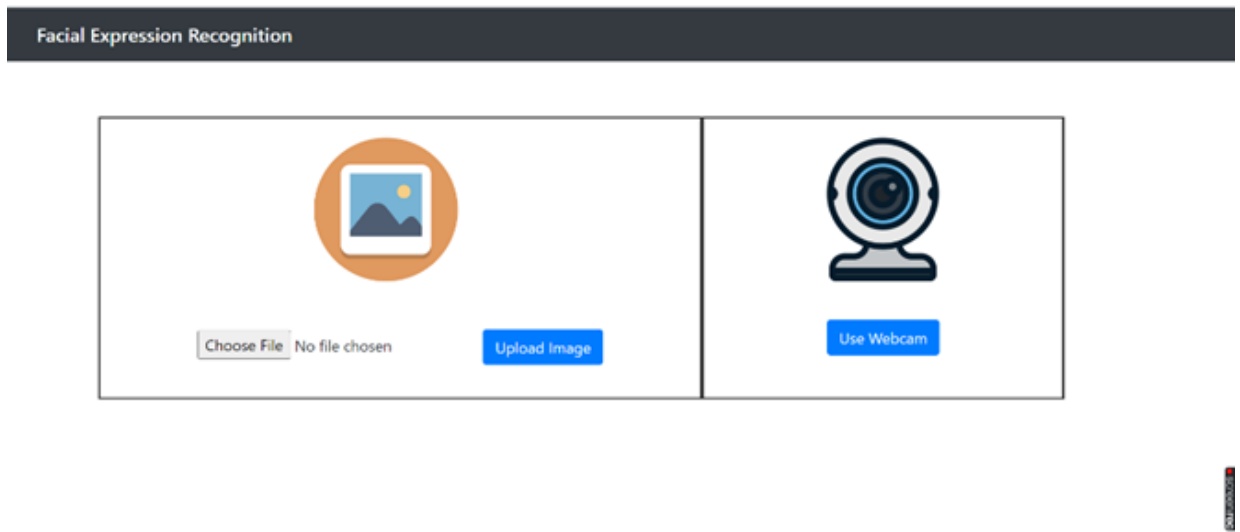
Permissions: **RW** | End-of-lines: **LF** | Encoding: **UTF-8** | Line: **65** | Column: **15** | Memory: **51 %**

c)Accuracy check

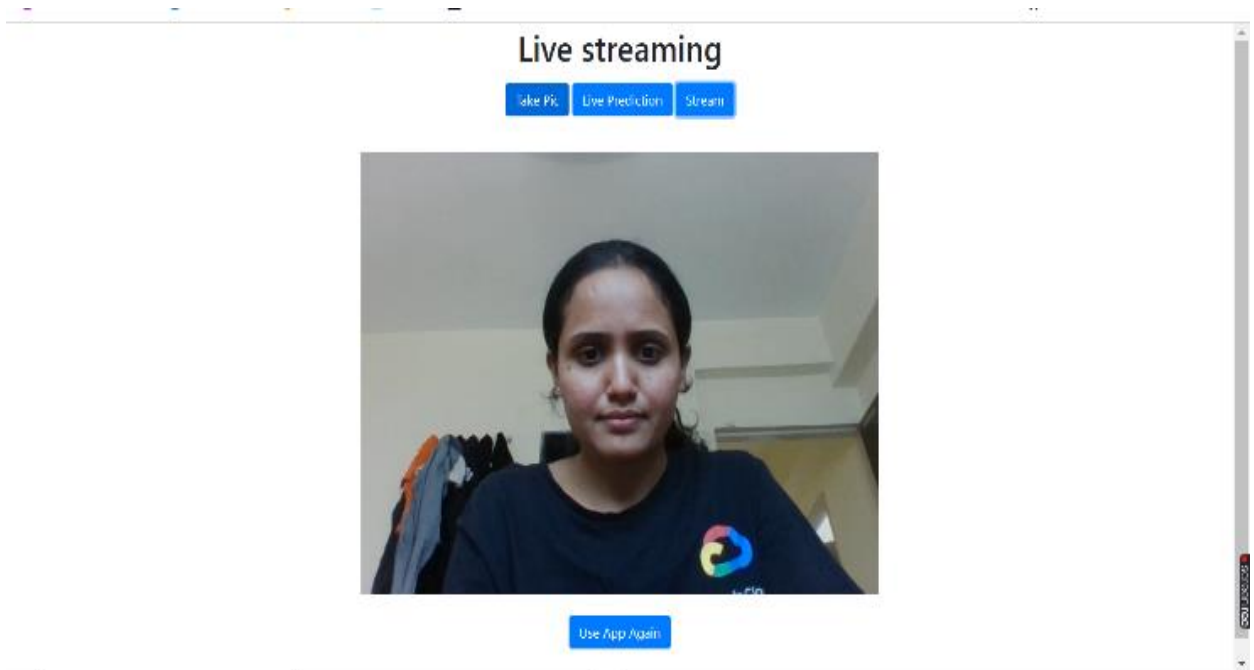
CHAPTER 9

RESULTS

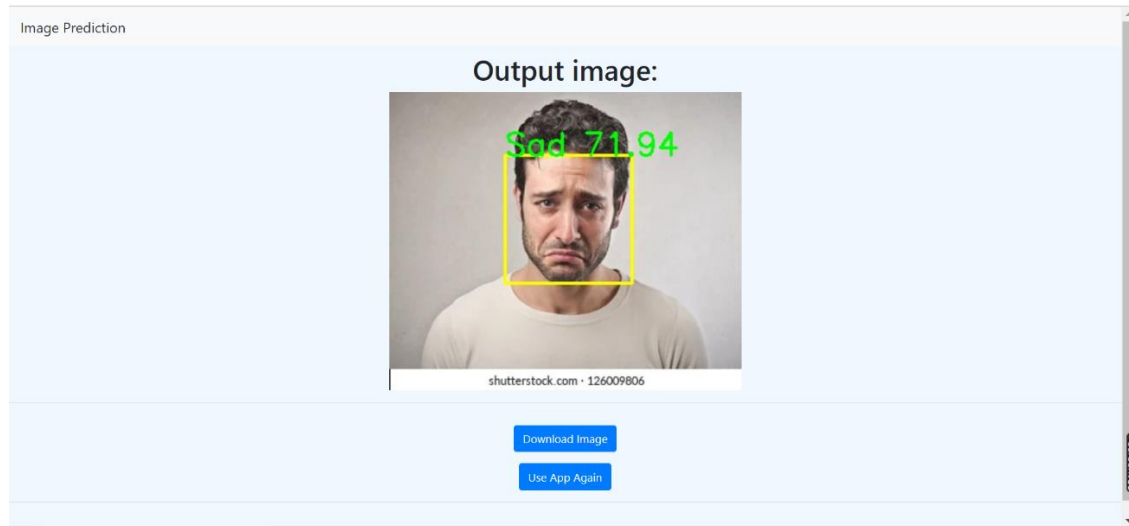
8.1 Screenshots



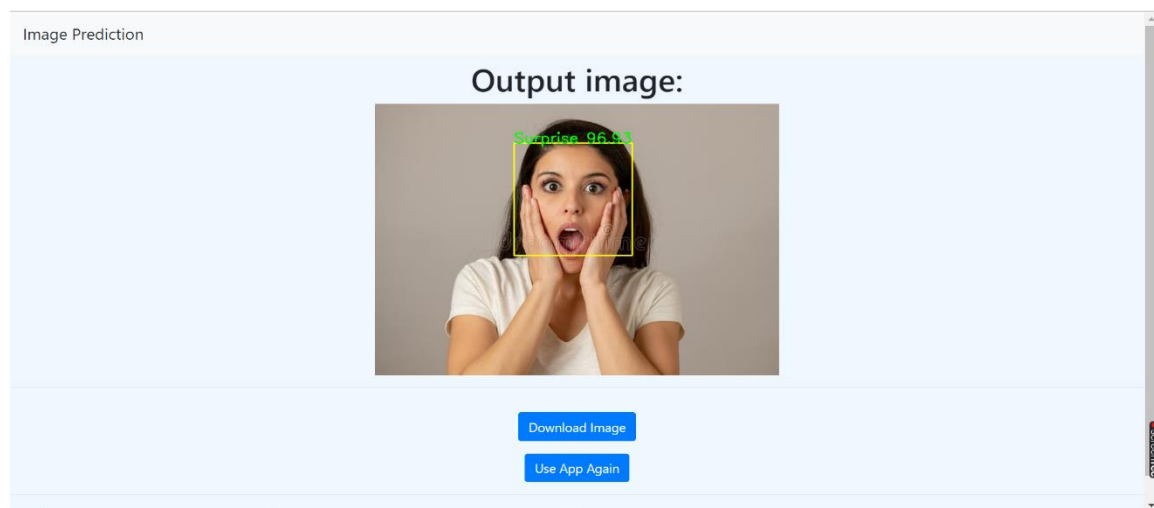
(a) Home page



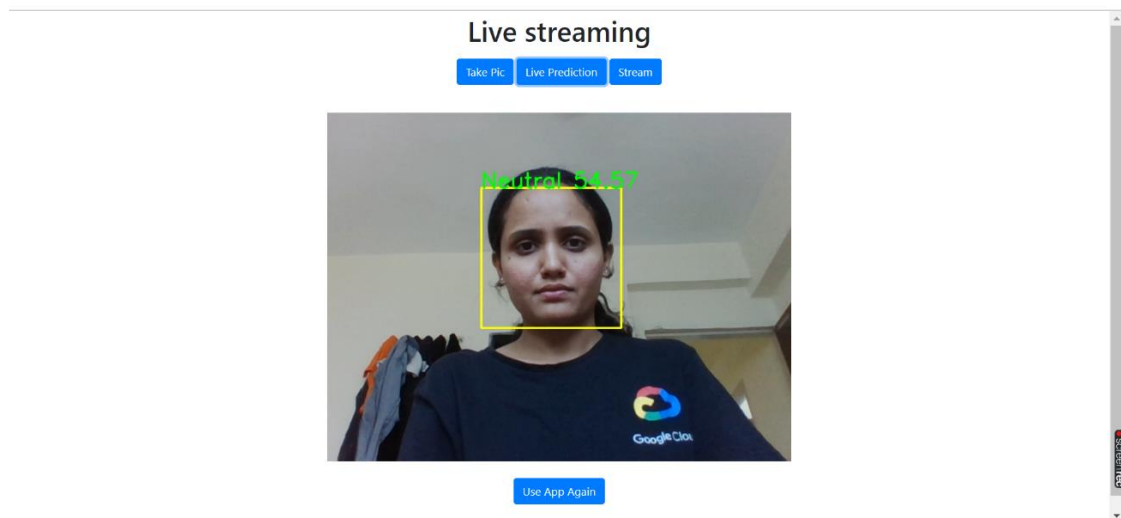
(b) Use Webcam



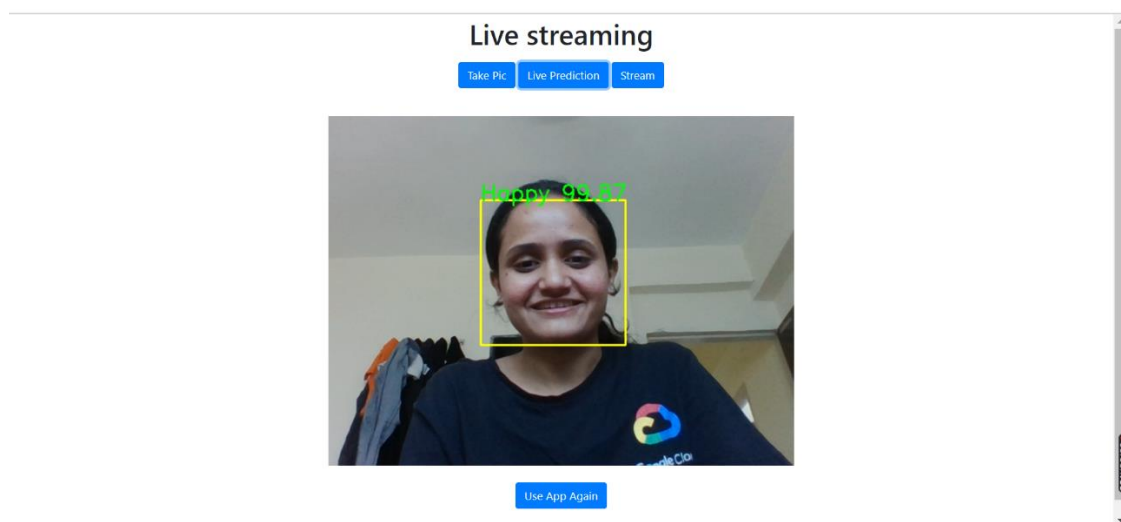
(c) Upload Image – Prediction I



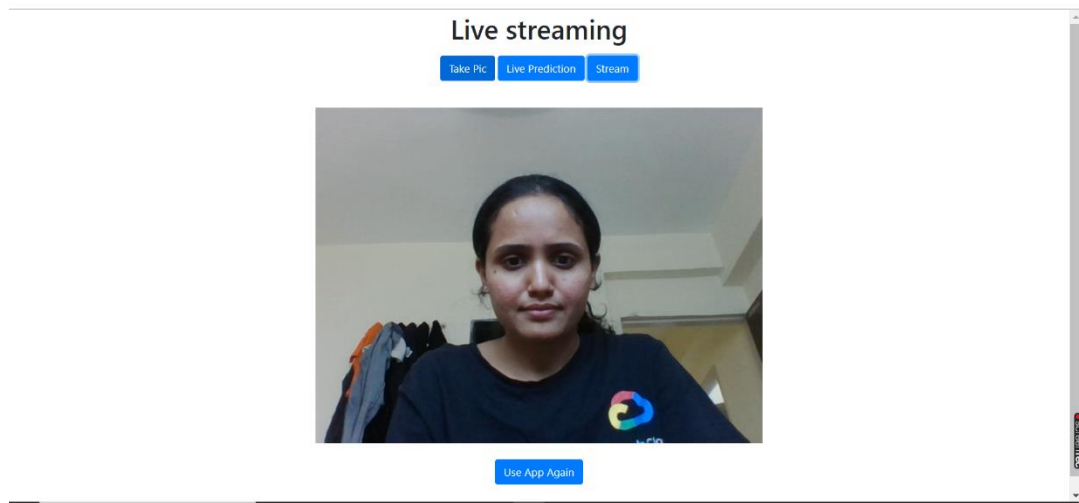
(d) Upload Image – Prediction II



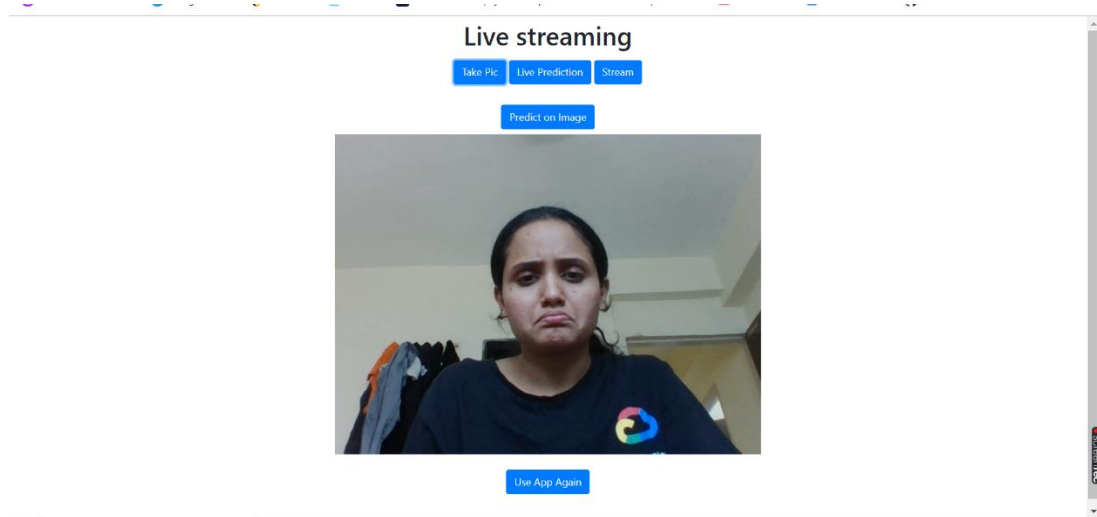
(e) Live Prediction



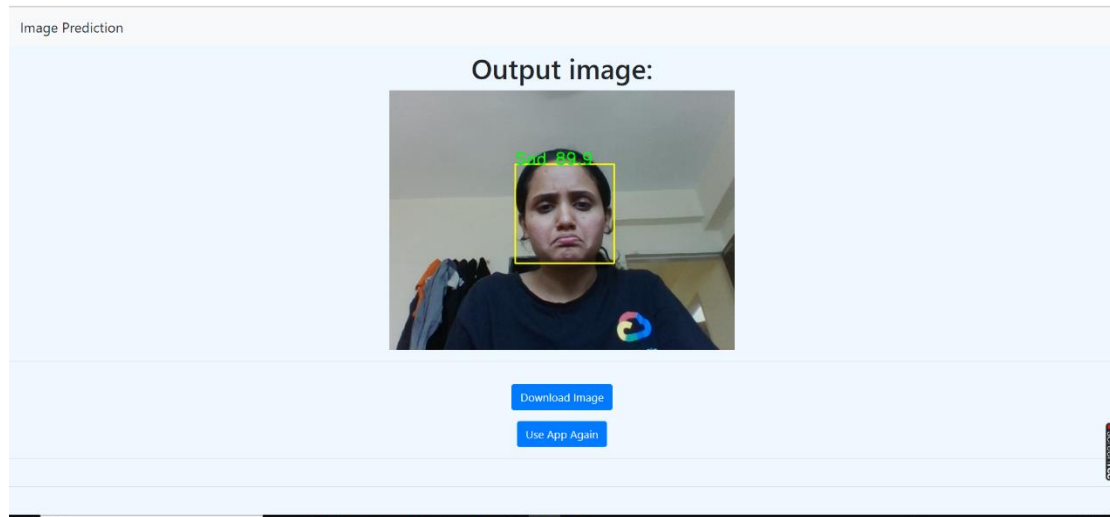
(f) Live Prediction



(g) Streaming



(h) Predict on Image



(i)Predict on Image- Result

CHAPTER 10

OTHER SPECIFICATION

10.1 Advantages

- Early Detection of Person Behavior.
- Easy to Implement.
- Cost Effective.
- Easy to identify the conditions of the person.

10.2 Limitations

- Camera should be accurate.
- Person face should capture properly.
- Light conditions must be maintained.

10.3 Applications

- robotics vision
- video surveillance
- digital cameras
- human-computer interaction
- Security Applications

CHAPTER 11

CONCLUSIONS

11.1 Conclusions

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 vision and enhancing the advanced feature extraction and classification in face 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 Support Vector Machines.

11.2 REFERENCES

- [1] Marek Kowalski, Jacek Naruniec, Tomasz Trzcinski, Deep Alignment Network: A convolutional neural network for robust face alignment”
- [2] Ali Mollahosseini, Behzad Hasani, and Mohammad H Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. Transactions on Affective Computing, 2017.
- [3] Kai Wang, Xiaojiang Peng, Jianfei Yang, Debin Meng, “Region Attention Networks for Pose and Occlusion Robust Facial Expression Recognition”
- [4] Ivona Tautke, Tomasz Trzcinski, Adam Bielski, “I Know how You Feel: Emotion Recognition with Facial Landmarks”
- [5] B. Hasani and M. H. Mahoor, “Facial expression recognition using enhanced deep 3d convolutional neural networks,” in Proceedings of CVPRW. IEEE, 2017.
- [6] Daniel Llatas Spiers, “Facial Emotion Detection Using Deep Learning”
- [7] Sivo Prasad Raju, Saumya A and Dr. Romi Murthy, “Facial Expression Detection using Different CNN Architecture Hybrid Vehicle Driving”, Centre for Communications, International Institute of Information Technology.
- [8] Deepesh Lekhak, “Facial Expression Recognition System using Convolutional Neural Network”, Tribhuvan University Institute of Engineering.
- [9] Jie Hu, Li Shen, and Gang Sun “Squeeze-and-excitation networks”, in Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7132–7141, 2018

CHAPTER 12

CERTIFICATES





