A DEEP LEARNING FACIAL EXPRESSION RECOGNITION BASED SCORING SYSTEM FOR RESTAURANTS

A Project Report submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

In

ELECTRONICS AND COMMUNICATION ENGINEERING

By

A. UMA SAGAR	**18PA1A0407**
A. PRADEEP NAIDU	**18PA1A0404**
B. DHARMA NAGA SRI RAM	**19PA5A0403**
G. KRUPA RAJU	**18PA1A0452**

Under the Esteemed Guidance of

Mr. K. Ramesh Chandra

Associate Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

VISHNU INSTITUTE OF TECHNOLOGY (Autonomous)

(Accredited by NBA, NAAC, Approved by AICTE & Affiliated to JNTU Kakinada)
Vishnupur, Bhimavaram – 534202

2021 – 2022

VISHNU INSTITUTE OF TECHNOLOGY (Autonomous)

(Accredited by NBA, NAAC, Approved by AICTE & Affiliated to JNTU Kakinada)

Vishnupur, Bhimavaram – 534202

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the project entitled "***A Deep Learning Facial Expression Recognition Based Scoring System For Restaurants"*** is being submitted by **A. Uma Sagar** (18PA1A0407), **A. Pradeep Naidu**(18PA1A0404), **B. Dharma Naga Sri Ram** (19PA5A0403), and **G. Krupa Raju** (18PA1A0452) in partial fulfilment for the award of the degree of **Bachelor of Technology** in Electronics and Communication Engineering is a record of bonafide work carried out by them under my guidance and supervision during academic year 2021-2022 and it has been found worthy of acceptance according to the requirements of the university.

Project Guide **H.O.D/E.C.E**

Mr. K. Ramesh Chandra
Associate Professor

Prof. K. Srinivas
Professor & Head

External Examiner

ACKNOWLEDGEMENT

Our most sincere and grateful acknowledgement is due to this sanctum, Vishnu Institute of Technology, for giving us opportunity to fulfill my aspirations and for successful completion of Bachelor of Technology.

We express our heartfelt thanks to our Principal **Dr. D. Suryanarayana,** Vice Principal **Prof. K. Srinivas** for providing us with the necessary facilities to carry out this work.

We would like to express our sincere thanks to **K. Srinivas,** H.O.D, Electronics and Communication Engineering, Vishnu Institute of Technology for his valuable support and encouragement during the project.

It is great pleasure for us to acknowledge the guidance, encouragement and support that we have received from **Mr. K. Ramesh Chandra**, Associate Professor, Department of Electronics & Communication Engineering, Vishnu Institute of Technology. We are thankful for his continuous assistance and invaluable suggestions. He not only provides help whenever needed, but also the resources required to complete this project on time.

We would like to express our sincere thanks to **PRC** members **Dr. N. Padmavathy**, **Dr. A. Prabhakara Rao**, **Dr. Prakash Pareek** for their valuable suggestions in improvisation of the project work. We also express our sincere thanks to project coordinator **Mr. G. Prasanna Kumar** for his valuable support during the project work.

Finally, yet importantly, we would like to express heartfelt thanks to our beloved parents for their blessings, friends/classmates for their help and wishes.

A. UMA SAGAR	**18PA1A0407**
A. PRADEEP NAIDU	**18PA1A0404**
B. DHARMA NAGA SRI RAM	**19PA5A0403**
G. KRUPA RAJU	**18PA1A0452**

Abstract

Recently, the popularity of automated and unmanned restaurants has increased. Due to the absence of staff, there is no direct perception of the customers' impressions in order to find out what their experiences with the restaurant concept are like. For this purpose, this paper presents a rating system based on facial expression recognition with pre-trained convolutional neural network (CNN) models. For interactive human and computer interface (HCI) it is important that the computer understand facial expressions of human. With HCI the gap between computers and humans will reduce. The computers can interact in more appropriate way with humans by judging their expressions. There are various techniques for facial expression recognition which focuses on getting good results of human expressions and then the food is supposed to be rated. Currently, three expressions (satisfied, neutral and disappointed) are provided by the scoring system.

Keywords: Convolution Neural Network, Human Computer Interface.

LISTOF ACRONYMS

HCI	Human Computer Interface
CNN	Convolution Neural Network
FER	Facial Expression recognition
LDA	Linear Discriminant Analysis
MLP	Multilayer Perceptron
CSV	Comma Separated Values
DFD	Data Flow Diagram

List of Figures

List of Tables

INDEX

1.INTRODUCTION

Human facial expressions are extremely essential in social communication. Normally communication involves both verbal and nonverbal. Non-verbal communications are expressed through facial expressions. Face expressions are the delicate signals of the larger communication. Non-verbal communication means communication between human and animals through eye contact, gesture, facial expressions, body language, and paralanguage. Eye contact is the important phase of communication which provides the mixture of ideas. Eye contact controls the contribution, discussions and creates a link with others. Face expressions include the smile, sad, anger, disgust, surprise, and fear. A smile on human face shows their happiness and it expresses eye with a curved shape. The sad expression is the feeling of looseness which is normally expressed as rising skewed eyebrows and frown. The anger on human face is related to unpleasant and irritating conditions. The expression of anger is expressed with squeezed eyebrows, slender and stretched eyelids. The disgust expressions are expressed with pull down eyebrows and creased nose. The surprise or shock expression is expressed when some unpredicted happens. This is expressed with eye-widening and mouth gaping and this expression is an easily identified one. The expression of fear is related with surprise expression which is expressed as growing skewed eyebrows. FER has the important stage is feature extraction and classification. Feature extraction includes two types and they are geometric based and appearance based. The classification is also one of the important processes in which the above-mentioned expressions such as smile, sad, anger, disgust, surprise, and fear are categorized. The geometrically based feature extraction comprises eye, mouth, nose, eyebrow, other facial components and the appearance based feature extraction comprises the exact section of the face.

In recent Years, the rapid growth and development of information and communication technology, the Internet of Things and Artificial Intelligence has resulted in an increase in number of applications based on these technologies. Following this trend, the popularity of automated and unmanned restaurants continues to grow. Particularly in developing countries the number of automated restaurants and a large number of unmanned restaurants are successfully growing.

1.1 LITERATURE SURVEY

Huang, et al

Part Based Face Detection Models includes kernel-SVMs used as the part detector and LDA is adopted to combine the results of the previous part detector. This method of face detection achieved better performance when comparing to others when using only parts of face rather than using the whole face. It was not design to deal with faces with occlusion. Kiyoto Ichikawa, et al., have come up with a new methodology of face detection which is based on partial information that includes AdaBoost method and used to train the machine with images of partial faces and LDA. Decision tree structure is also adapted to combine the output of whole partial classifiers to extract the final result. The partial information includes features about distinctive features of face such as eyes, and lips. Even if any of these parts are occluded the performance of this method did not degrade. This approach will not be effective when applied to other type of occlusions. In surplus, Huang, et al., have proposed a component-based framework for generalized face alignment.

Yang, et al.

Have proposed a face live-ness detection method with component dependent descriptor and Zhang, et al., have proposed a face detection method based on local region sparse coding, etc. Though already existing part based face detection methods have achieved high performance, but they are not applicable to all types of occlusions [7]. Visibility estimation for face recognition plays a very dominant role when we are dealing with occlusion. All the above methods calculated visibility of every part of faces independently e and adopted hard threshold to estimate the total visibility [8].

Ji Tao and Yap-Peng Tan

Have come up with a system which automates facial detection by capturing the images from a video, arrange them in a cluster and store as sub sequences. When video is running, frame sequences are captured in the form of images and are converted into gray color images and later they will be ordered in a sequence, these face sequences should be clustered by using the graph partitioning.

Jie Chen, Shiguang Shan and Peng Yang
have implanted the Gabor Features face detection. In this process a color image is taken as input and will be pre-processed. For removing the noise attenuation, skin region of the image is extracted and converts it gray color image and then detects face location, then tries to get the face normalized and get final face locations with help of Gabor features.

Viola Jones

An algorithm is named after two computer vision researchers who proposed the method in 2001, Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features". Despite being an outdated framework, Viola-Jones is quite powerful, and its application has proven to be exceptionally notable in real-time face detection. This algorithm is painfully slow to train but can detect faces in real-time with impressive speed.

Given an image(this algorithm works on grayscale image), the algorithm looks at many smaller subregions and tries to find a face by looking for specific features in each subregion. It needs to check many different positions and scales because an image can contain many faces of various sizes. Viola and Jones used Haar-like features to detect faces in this algorithm.

The Viola Jones algorithm has four main steps, which we shall discuss in the sections to follow:

1. Selecting Haar-like features
2. Creating an integral image
3. Running AdaBoost training
4. Creating classifier cascades

1.2 Problem definition:

As there is no staff available in unmanned restaurants, it is difficult for the restaurant management to estimate how the concept and the food is experienced by the customers. Existing rating systems, such as Google and TripAdvisor, only partially solve this problem, as they only cover a part of the customer's opinions. These rating systems are only used by a subset of the customers who rate the restaurant on independent rating platforms on their own initiative. This applies mainly to customers who experience their visit as very positive or negative.

2.SYSTEM ANALYSIS

2.1 What is Waterfall Model?

Waterfall Model is a sequential model that divides software development into different phases. Each phase is designed for performing specific activity during SDLC phase. It was introduced in 1970 by Winston Royce.

Requirements:

The first phase involves understanding what needs to design and what is its function, purpose, etc. Here, the specifications of the input and output or the final product are studied and marked.

System Design:

The requirement specifications from the first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The software code to be written in the next stage is created now.

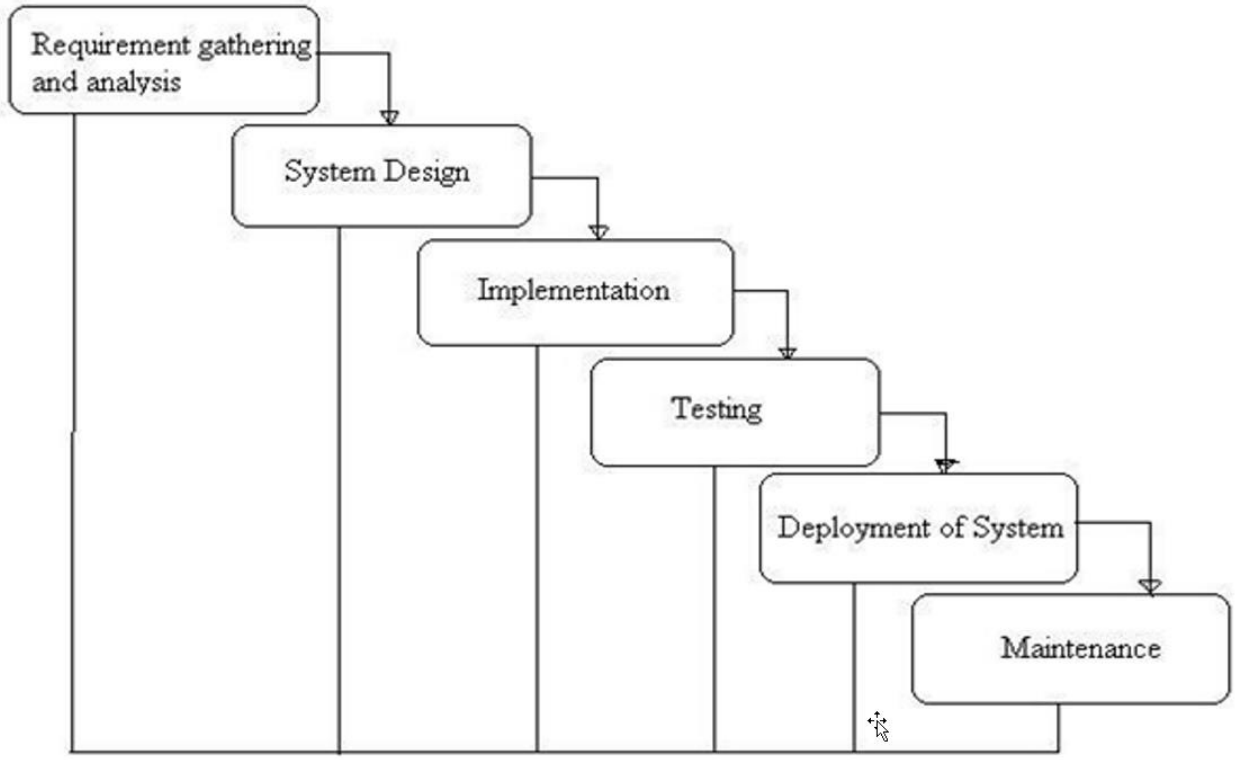


Figure 2.1: System Analysis

Implementation:
With inputs from system design, the system is first developed in small programs called units, which are integrated into the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

Integration and Testing:
All the units developed in the implementation phase are integrated into a system after testing of each unit. The software designed, needs to go through constant software testing to find out if there are any flaws or errors. Testing is done so that the client does not face any problem during the installation of the software.

Deployment of System:
Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.

Maintenance:
This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. These modifications arise either due to change requests initiated by the customer, or defects uncovered during live use of the system. The client is provided with regular maintenance and support for the developed software.

2.2 Functional requirements

- Webcam
- Image Upload
- Live on webcam
- Expressions Finding
- Restaurant Feedback

2.3 Non-functional requirements

What is Non-Functional Requirement?

NON-FUNCTIONAL REQUIREMENT (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software

system. Example of nonfunctional requirement, *"how fast does the website load?"* Failing to meet non-functional requirements can result in systems that fail to satisfy user needs.

Non-functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are > 10000. Description of non-functional requirements is just as critical as a functional requirement.

- Usability requirement
- Serviceability requirement
- Manageability requirement
- Recoverability requirement
- Security requirement
- Data Integrity requirement
- Capacity requirement
- Availability requirement
- Scalability requirement
- Interoperability requirement
- Reliability requirement
- Maintainability requirement
- Regulatory requirement
- Environmental requirement

Examples of Non-functional requirements

Here, are some examples of non-functional requirement:

1. Users must change the initially assigned login password immediately after the first successful login. Moreover, the initial should never be reused.
2. Employees never allowed to update their salary information. Such attempt should be reported to the security administrator.
3. Every unsuccessful attempt by a user to access an item of data shall be recorded on an audit trail.
4. A website should be capable enough to handle 20 million users with affecting its performance

5. The software should be portable. So moving from one OS to other OS does not create any problem.
6. Privacy of information, the export of restricted technologies, intellectual property rights, etc. should be audited.

Advantages of Non-Functional Requirement

Benefits/pros of Non-functional testing are:

- The nonfunctional requirements ensure the software system follow legal and compliance rules.
- They ensure the reliability, availability, and performance of the software system
- They ensure good user experience and ease of operating the software.
- They help in formulating security policy of the software system.

Disadvantages of Non-functional requirement

Cons/drawbacks of Non-function requirement are:

- None functional requirement may affect the various high-level software subsystem
- They require special consideration during the software architecture/high-level design phase which increases costs.
- Their implementation does not usually map to the specific software sub-system,
- It is tough to modify non-functional once you pass the architecture phase.

KEY LEARNING

- A non-functional requirement defines the performance attribute of a software system.
- Types of Non-functional requirement are Scalability Capacity, Availability, Reliability, Recoverability, Data Integrity, etc.
- Example of Non Functional Requirement is Employees never allowed to update their salary information. Such attempt should be reported to the security administrator.
- Functional Requirement is a verb while Non-Functional Requirement is an attribute
- The advantage of Non-functional requirement is that it helps you to ensure good user experience and ease of operating the software
- The biggest disadvantage of Non-functional requirement is that it may affect the various high-level software subsystems.

2.4 Existing System:

As there is no staff available in unmanned restaurants, it is difficult for the restaurant management to estimate how the concept and the food is experienced by the customers. Existing rating systems, such as Google and TripAdvisor, only partially solve this problem, as they only cover a part of the customer's opinions. These rating systems are only used by a subset of the customers who rate the restaurant on independent rating platforms on their own initiative. This applies mainly to customers who experience their visit as very positive or negative.

Figure 2.2 : Existing Review System

2.5 Proposed System:

In order to solve the above problem, all customers must be motivated to give a rating. This paper introduces an approach for a restaurant rating system that asks every customer for a rating after their visit to increase the number of ratings as much as possible. This system can be used unmanned restaurants; the scoring system is based on facial expression detection using pre-trained convolutional neural network (CNN) models. It allows the customer to rate the food by taking or capturing a picture of his face that reflects the corresponding feelings. Compared to text-based rating system, there is much less information and no individual experience reports collected. However, this simple fast and playful rating system should give a wider range of opinions about the experiences of the customers with the restaurant concept.

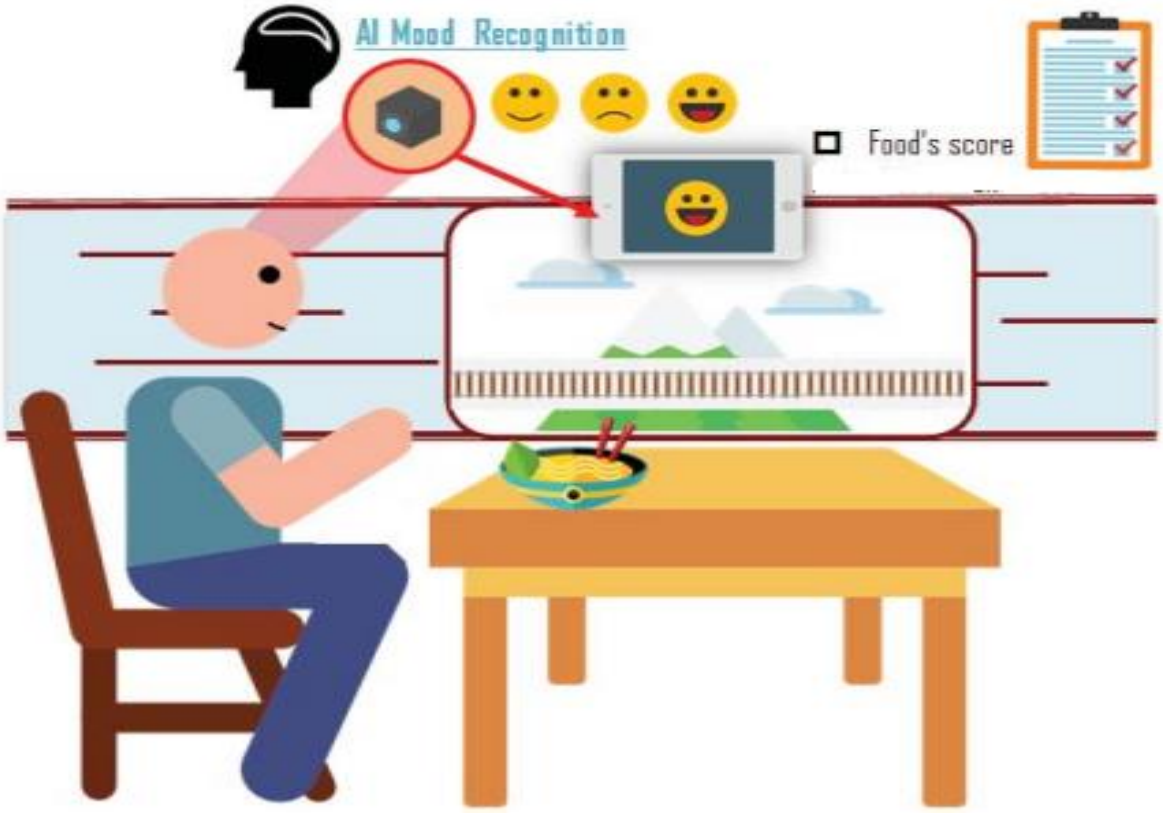


Figure 2.3 : Proposed System

3. IMPLEMENTATION

3.1 System Requirements

Hardware requirements:

Processer	:	Any Update Processer
Ram	:	Min 4 GB
Hard Disk	:	Min 100 GB

Software requirements:

Operating System	:	Windows family
Technology	:	Python 3.6
IDE	:	PyCharm

3.2 Algorithms:

Convolutional Neural Network (CNN)

```
input_image="person.jpg"

image=face_recognition.load_image_file(inputimage)

face_locations = face_recognition.face_locations(image)

for face_location in face_locations:
        top, right, bottom, left = face_location
        face_image = image[top:bottom, left:right]
        cv2.rectangle(inputimage, (left, top), (right, bottom), (0, 0, 255), 2)
'
```

3.3 Methodology

Convolution Neural Networks or covnets are a dynamic type of neural networks concepts which are most useful for image processing and natural language processing. It was unique and completely different than neural networks algorithm.

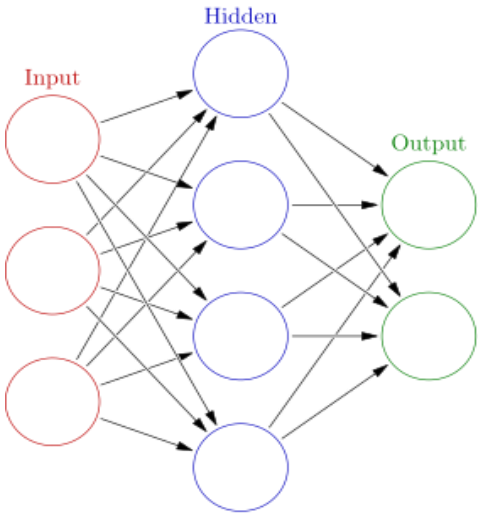


Figure 3.1 Model of Neural Networks

In normal neural networks algorithm consist 3 layers, those are input, hidden and output layer. In the input layer it will collect in input data, and in the hidden layers, a process or calculations done for the prediction, in output layer result will generate. For training and prediction we need set the features of the data in hidden features. But in the CNN Features will automatically calculate according the input data.

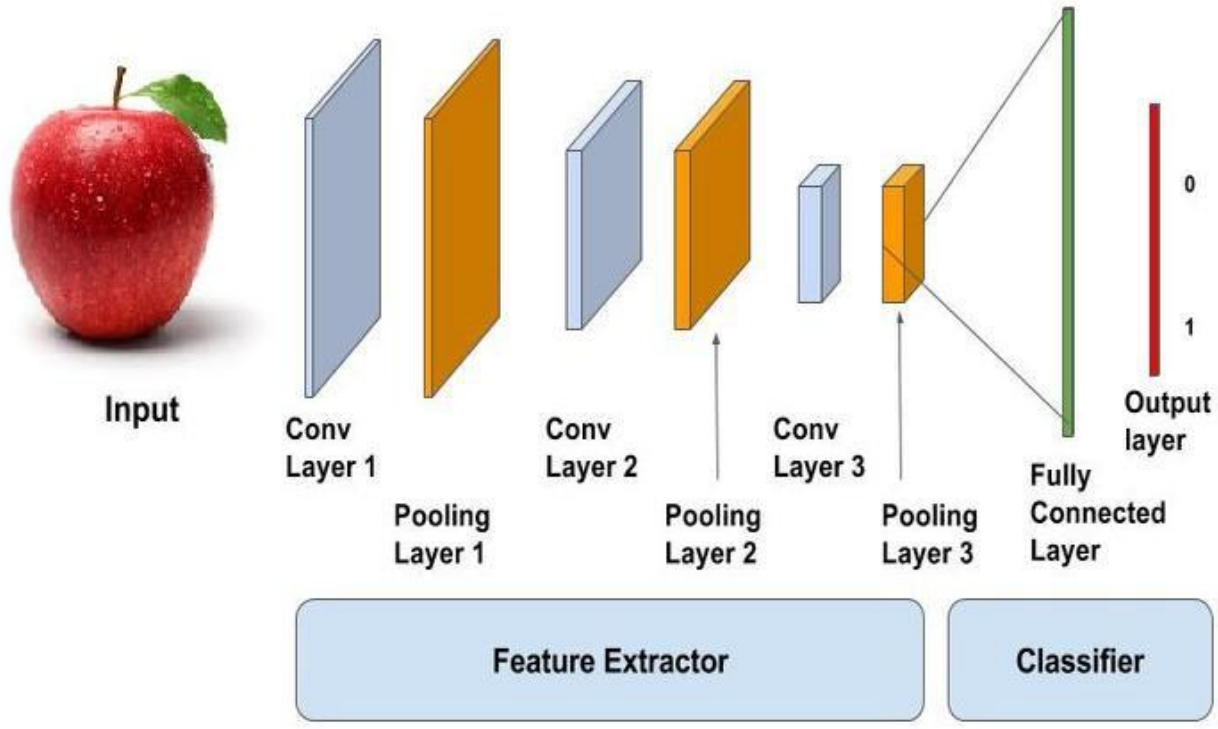


Figure 3.2: CNN block diagram

Types of layers in CNN:

- **Input Layer:**

  In this layer it holds the raw data for input.

- **Convolution Layer**

  In this layer, we can build the core block of the input data, this layer have a set of kernels means learnable filters using the small blocks of the data, and learn features of the data.

- **Activation Function Layer**

  This layer will apply element wise activation function to the output of convolution layer. Some common activation functions are RELU, Sigmoid, Tanh, Leaky RELU, etc.

- **Pool Layer**

  Pooling layers are used extract the features from the single size of the data splitting to multiple layer. We can get the features of the data and reduce computation cost.

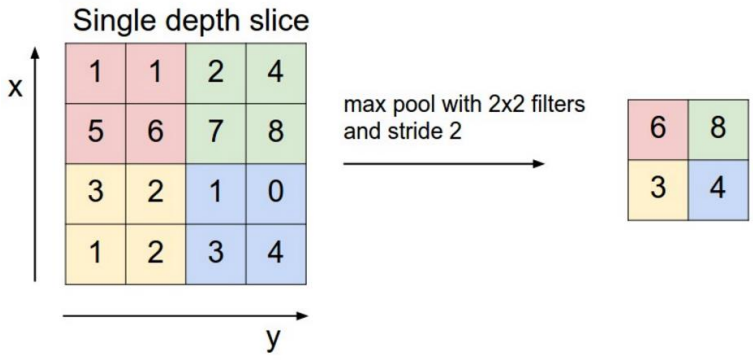


Figure 3.3: Pooling Layer

In this above example the data [1,1,5,6] convert to one set as 6, and [2,4,7,8] also convert to 8.

For input layer, we have taken 10,000 images of 10 categories of the food items.

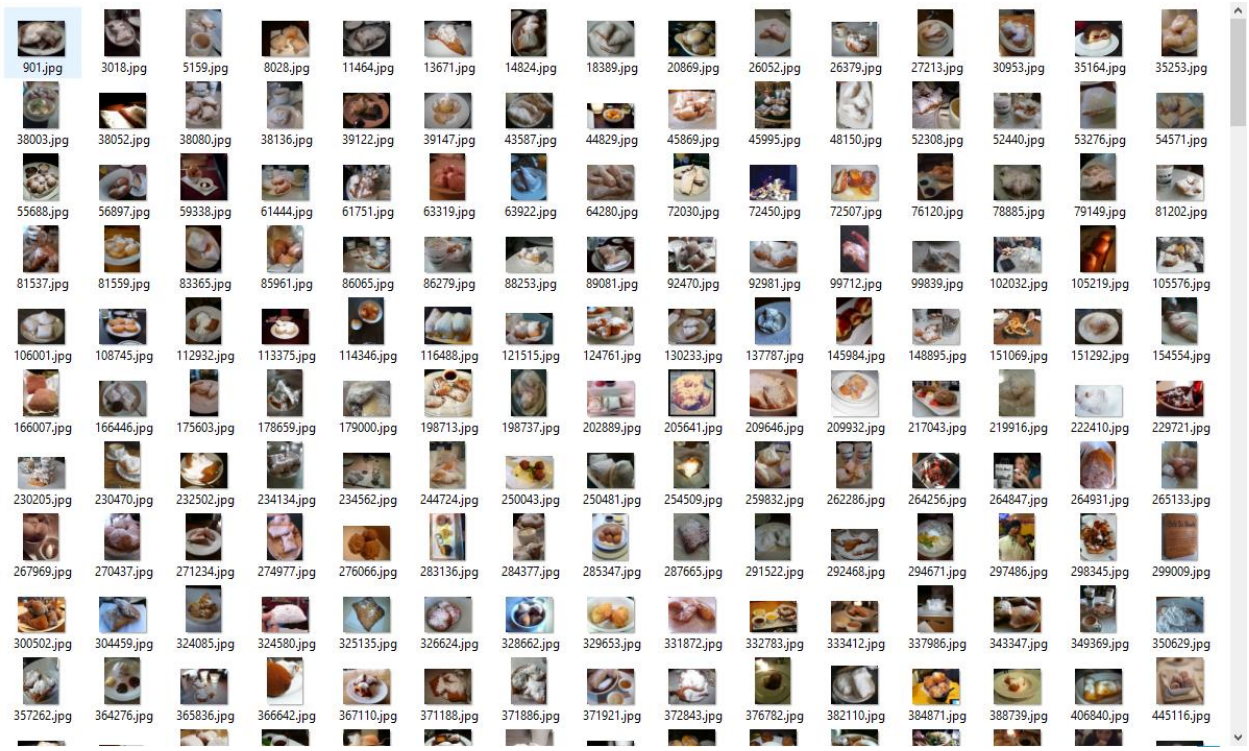


Figure 3.4: Snapshot of Images of the 'beignets' category images of training dataset.

Based on dataset we create training model using CNN

```
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.preprocessing.image import ImageDataGenerator

#Initialize the CNN
classifier = Sequential()
#Convolution and Max pooling
classifier.add(Conv2D(32, (4, 4), input_shape = (128, 128, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2,2)))
classifier.add(Conv2D(64, (4, 4), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2,2)))
classifier.add(Conv2D(128, (4, 4), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2,2)))

#Flatten
classifier.add(Flatten())

#Full connection
classifier.add(Dense(128, activation = 'relu'))
classifier.add(Dense(4, activation = 'softmax'))

#Compile classifier
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])

#Fitting CNN to the images
train_datagen = ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)
training_set = train_datagen.flow_from_directory('./training', target_size=(128, 128), batch_size=32, class_mode='categorical')
test_set = test_datagen.flow_from_directory('./testing', target_size=(128, 128), batch_size=32, class_mode='categorical')
classifier.fit_generator(training_set, steps_per_epoch=800/32, epochs=50, validation_data=test_set, validation_steps = 200/32)

#save model
classifier.save('model.h5')
classifier.save_weights('weights.h5')
```

3.4 System Architecture:

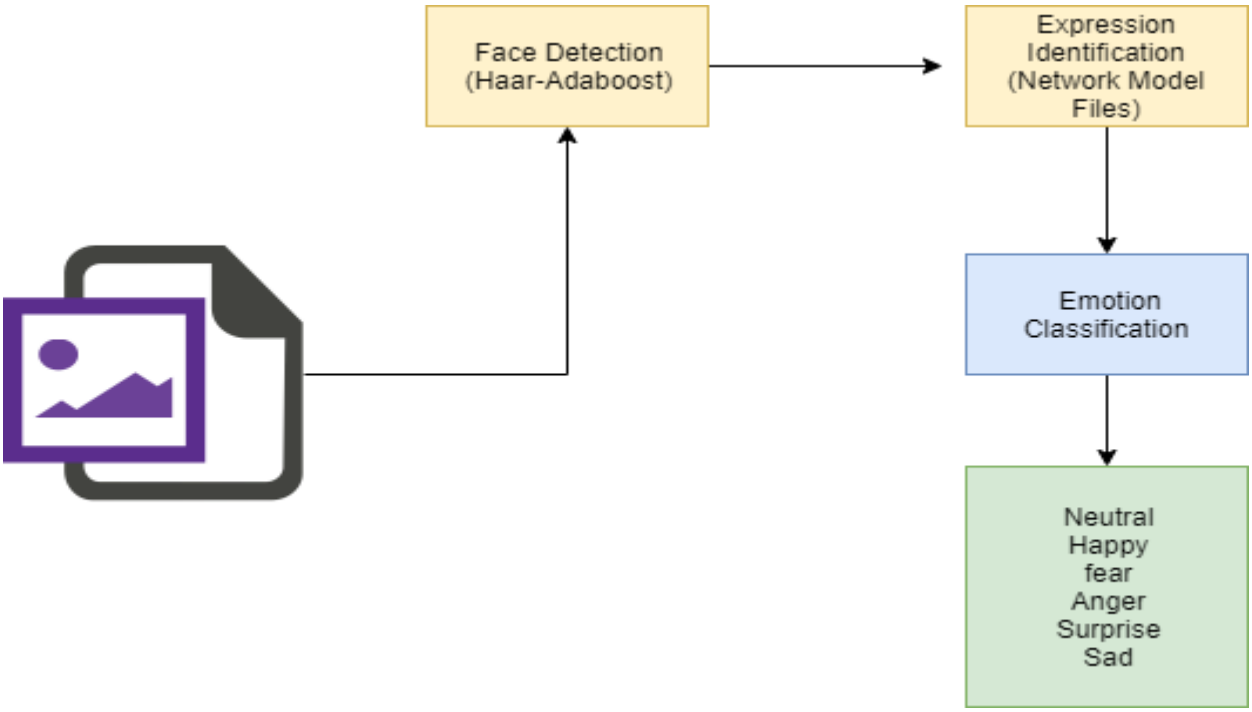


Figure:3.5 – System Architecture

3.5 Modules

Face Detection:

Face detection or localization is an important step for image classification since only the principal component of face such as nose, eyes, mouth are needed for classification. Face detection algorithms can be broadly classified into feature, knowledge, template and appearance based methods. Our proposed system uses Viola Jones object detection algorithm for face localization which comes under feature based classification. Viola Jones object detection algorithm uses Haar featurebased cascade classifiers. The Haar Cascade classifier is extremely important element of the face detection. The presence of the features in any of the input image is determined by the Haar features.

Facial Expression Recognition classification:

After learning the deep features, the final step of FER (Facial Expression Recognition) is to classify the given face into one of the basic emotion categories. Unlike the traditional methods, where the feature extraction step and the feature classification step are independent, deep networks can perform FER in an end-to-end way. Specifically, a loss layer is added to the end of the network to regulate the back-propagation error; then, the prediction probability of each sample can be directly output by the network. In CNN, softmax loss is the most common used function that minimizes the cross-entropy between the estimated class probabilities and the ground truth distribution.

Convolutional neural network (CNN):

CNN has been extensively used in diverse computer vision applications, including FER. At the beginning of the 21st century, several studies in the FER literature found that the CNN is robust to face location changes and scale variations and behaves better than the multilayer perceptron (MLP) in the case of previously unseen face pose variations, employed the CNN to address the problems of subject independence as well as translation, rotation, and scale invariance in the recognition of facial expressions.

3.6 Software overview:

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Input as CSV File

Reading data from CSV(comma separated values) is a fundamental necessity in Data Science. Often, we get data from various sources which can get exported to CSV format so that they can be used by other systems. The Panadas library provides features using which we can read the CSV file in full as well as in parts for only a selected group of columns and rows.

The CSV file is a text file in which the values in the columns are separated by a comma. Let's consider the following data present in the file named input.csv.You can create this file using windows notepad by copying and pasting this data. Save the file as input.csv using the save As All files(*.*) option in notepad.

```
import pandas as pd

data= pd.read_csv('path/input.csv')

print(data)
```

Operations using NumPy

NumPy is a Python package which stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Using NumPy, a developer can perform the following operations −

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

Key Features of Pandas

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

4.SYSTEM DESIGN

4.1 UML DIAGRAMS

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

Global Use Case Diagrams:

Identification of actors:

Actor: Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

Graphical representation:

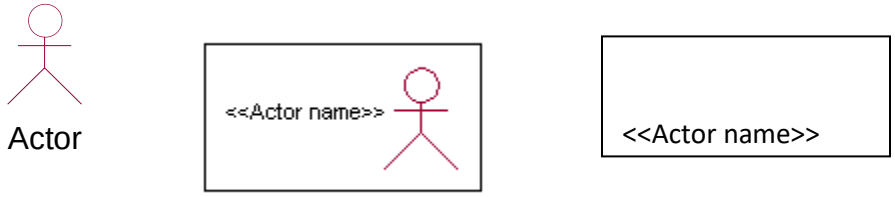


Figure:4.1 – Graphical Representation of Usecase

An actor is someone or something that:

Interacts with or uses the system.

- Provides input to and receives information from the system.
- Is external to the system and has no control over the use cases.

Actors are discovered by examining:

- Who directly uses the system?
- Who is responsible for maintaining the system?
- External hardware used by the system.
- Other systems that need to interact with the system.

Questions to identify actors:

- Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
- Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.

- Which external hardware or systems (if any) use the system to perform tasks?
- What problems does this application solve (that is, for whom)?
- And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

a. **System Administrator**
b. **Customer**
c. **Customer Care**

Identification of usecases:

Usecase: A use case can be described as a specific way of using the system from a user's (actor's) perspective.

Graphical representation:

A more detailed description might characterize a use case as:

- Pattern of behavior the system exhibits
- A sequence of related transactions performed by an actor and the system
- Delivering something of value to the actor

Use cases provide a means to:

- capture system requirements
- communicate with the end users and domain experts
- test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

- For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
- Name the use cases.

- Describe the use cases briefly by applying terms with which the user is familiar.

This makes the description less ambiguous

Questions to identify use cases:

- What are the tasks of each actor?
- Will any actor create, store, change, remove or read information in the system?
- What use case will store, change, remove or read this information?
- Will any actor need to inform the system about sudden external changes?
- Does any actor need to inform about certain occurrences in the system?
- What usecases will support and maintains the system?

Flow of Events

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

- When and how the use case starts and ends
- Use case/actor interactions
- Data needed by the use case
- Normal sequence of events for the use case
- Alternate or exceptional flows

Construction of Usecase diagrams:

Use-case diagrams graphically depict system behavior (use cases). These diagrams present a high level view of how the system is used as viewed from an outsider's (actor's) perspective. A use-case diagram may depict all or some of the use cases of a system.

A use-case diagram can contain:

- actors ("things" outside the system)
- use cases (system boundaries identifying what the system should do)
- Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

Relationships in use cases:

1. Communication:

The communication relationship of an actor in a usecase is shown by connecting the actor symbol to the usecase symbol with a solid path. The actor is said to communicate with the usecase.

2. Uses:

A Uses relationship between the usecases is shown by generalization arrow from the usecase.

3. Extends:

The extend relationship is used when we have one usecase that is similar to another usecase but does a bit more. In essence it is like subclass.

SEQUENCE DIAGRAMS

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence what happens first, what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

There are two main differences between sequence and collaboration diagrams: sequence diagrams show time-based object interaction while collaboration diagrams show how objects associate with each other. A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

Object:

An object has state, behavior, and identity. The structure and behavior of similar objects are defined in their common class. Each object in a diagram indicates some instance of a class. An object that is not named is referred to as a class instance.

The object icon is similar to a class icon except that the name is underlined:

An object's concurrency is defined by the concurrency of its class.

Message:

A message is the communication carried between two objects that trigger an event. A message carries information from the source focus of control to the destination focus of control. The synchronization of a message can be modified through the message specification. Synchronization means a message where the sending object pauses to wait for results.

Link:

A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes. The existence of a relationship between two classes

symbolizes a path of communication between instances of the classes: one object may send messages to another. The link is depicted as a straight line between objects or objects and class instances in a collaboration diagram. If an object links to itself, use the loop version of the icon.

CLASS DIAGRAM:

Identification of analysis classes:

A class is a set of objects that share a common structure and common behavior (the same attributes, operations, relationships and semantics). A class is an abstraction of real-world items.

There are 4 approaches for identifying classes:

a. Noun phrase approach:
b. Common class pattern approach.
c. Use case Driven Sequence or Collaboration approach.
d. Classes , Responsibilities and collaborators Approach

1. Noun Phrase Approach:

The guidelines for identifying the classes:

- Look for nouns and noun phrases in the usecases.
- Some classes are implicit or taken from general knowledge.
- All classes must make sense in the application domain; Avoid computer implementation classes – defer them to the design stage.
- Carefully choose and define the class names After identifying the classes we have to eliminate the following types of classes:
- Adjective classes.

2. Common class pattern approach:

The following are the patterns for finding the candidate classes:

- Concept class.
- Events class.
- Organization class
- Peoples class
- Places class
- Tangible things and devices class.

3. Use case driven approach:

We have to draw the sequence diagram or collaboration diagram. If there is need for some classes to represent some functionality then add new classes which perform those functionalities.

4. CRC approach:

The process consists of the following steps:

- Identify classes' responsibilities (and identify the classes)
- Assign the responsibilities
- Identify the collaborators.

Identification of responsibilities of each class:

The questions that should be answered to identify the attributes and methods of a class respectively are:

a. What information about an object should we keep track of?
b. What services must a class provide?

Identification of relationships among the classes:

Three types of relationships among the objects are:

Association: How objects are associated?

Super-sub structure: How are objects organized into super classes and sub classes?

Aggregation: What is the composition of the complex classes?

Association:

The **questions** that will help us to identify the associations are:

a. Is the class capable of fulfilling the required task by itself?
b. If not, what does it need?
c. From what other classes can it acquire what it needs?

Guidelines for identifying the tentative associations:

- A dependency between two or more classes may be an association. Association often corresponds to a verb or prepositional phrase.
- A reference from one class to another is an association. Some associations are implicit or taken from general knowledge.

Some common association patterns are:

Location association like part of, next to, contained in…..

Communication association like talk to, order to ……

We have to eliminate the unnecessary association like implementation associations, ternary or n-ary associations and derived associations.

Super-sub class relationships:

Super-sub class hierarchy is a relationship between classes where one class is the parent class of another class (derived class).This is based on inheritance.

Guidelines for identifying the super-sub relationship, a generalization are

1**. Top-down*:***

Look for noun phrases composed of various adjectives in a class name. Avoid excessive refinement. Specialize only when the sub classes have significant behavior.

2.**Bottom-up*:***

Look for classes with similar attributes or methods. Group them by moving the common attributes and methods to an abstract class. You may have to alter the definitions a bit.

3.**Reusability*:***

Move the attributes and methods as high as possible in the hierarchy.

4. **Multiple inheritances*:***

Avoid excessive use of multiple inheritances. One way of getting benefits of multiple inheritances is to inherit from the most appropriate class and add an object of another class as an attribute.

Aggregation or a-part-of relationship:

It represents the situation where a class consists of several component classes. A class that is composed of other classes doesn't behave like its parts. It behaves very difficultly. The major properties of this relationship are transitivity and anti symmetry.

The **questions** whose answers will determine the distinction between the part and whole relationships are:

- Does the part class belong to the problem domain?
- Is the part class within the system's responsibilities?
- Does the part class capture more than a single value?(If not then simply include it as an attribute of the whole class)
- Does it provide a useful abstraction in dealing with the problem domain?

There are three types of aggregation relationships. They are:

Assembly:

It is constructed from its parts and an assembly-partsituation physically exists.

Container:

A physical whole encompasses but is not constructed from physical parts.

Collection member:

A conceptual whole encompasses parts that may be physical or conceptual. The container and collection are represented by hollow diamonds but composition is represented by solid diamond.

4.2 Use case:

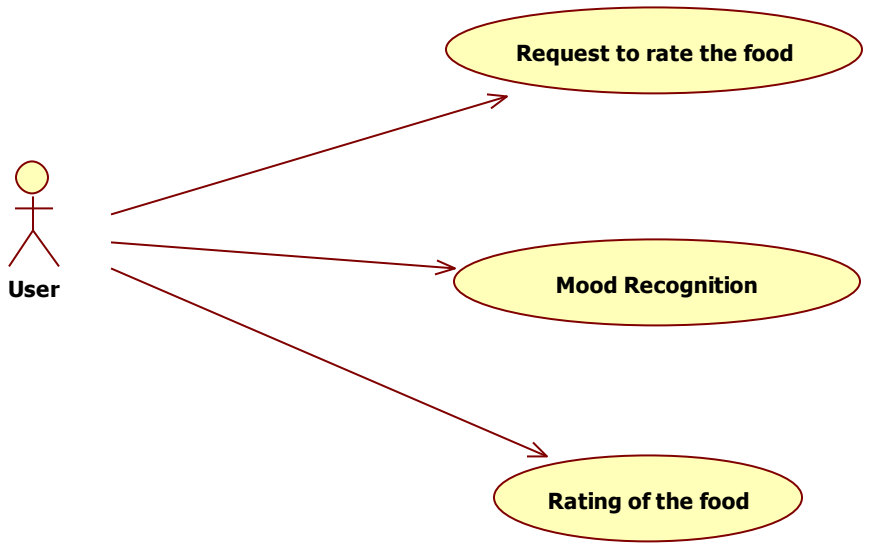


Figure:4.2 – Usecase

4.3 Sequence:

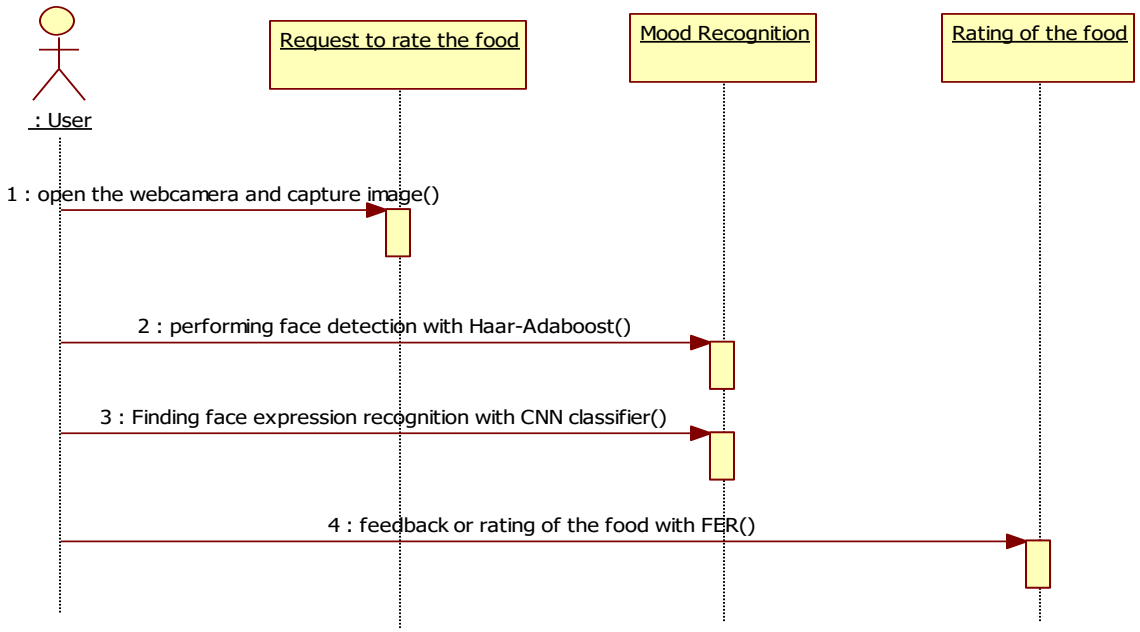


Figure 4.3 – Sequence

4.4 Collaboration:

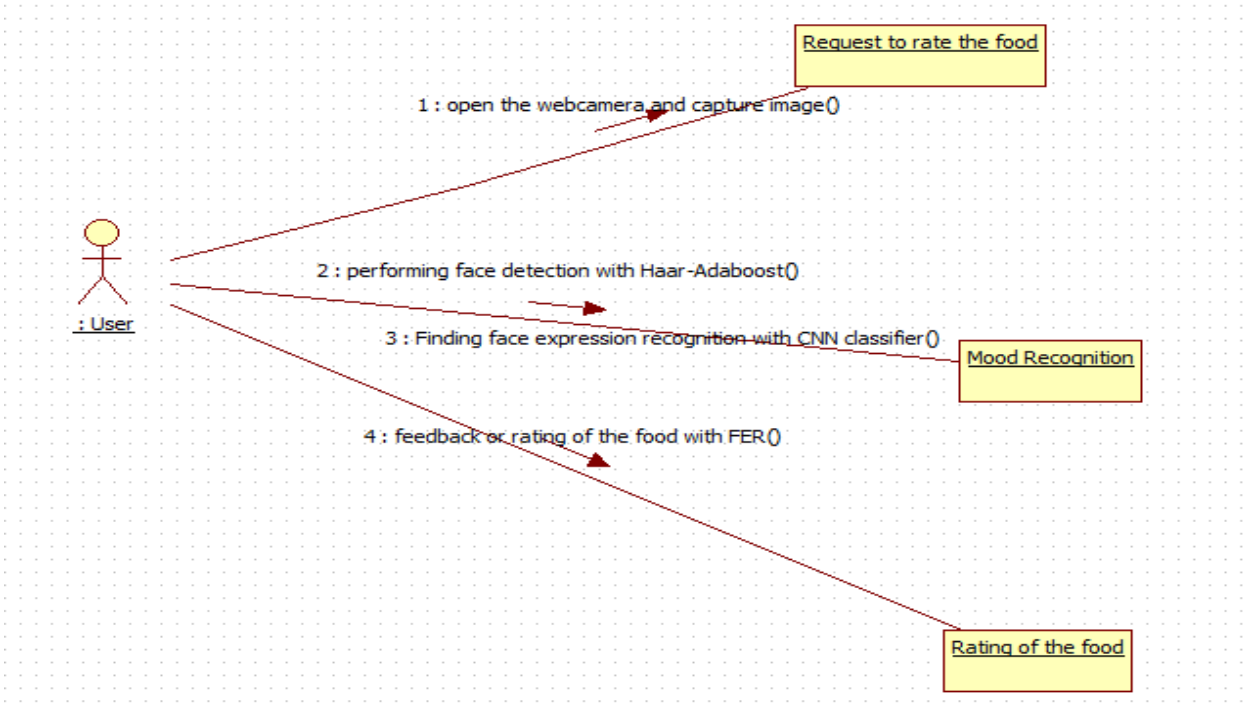


Figure 4.4 - Collaboration

4.5 State chart:

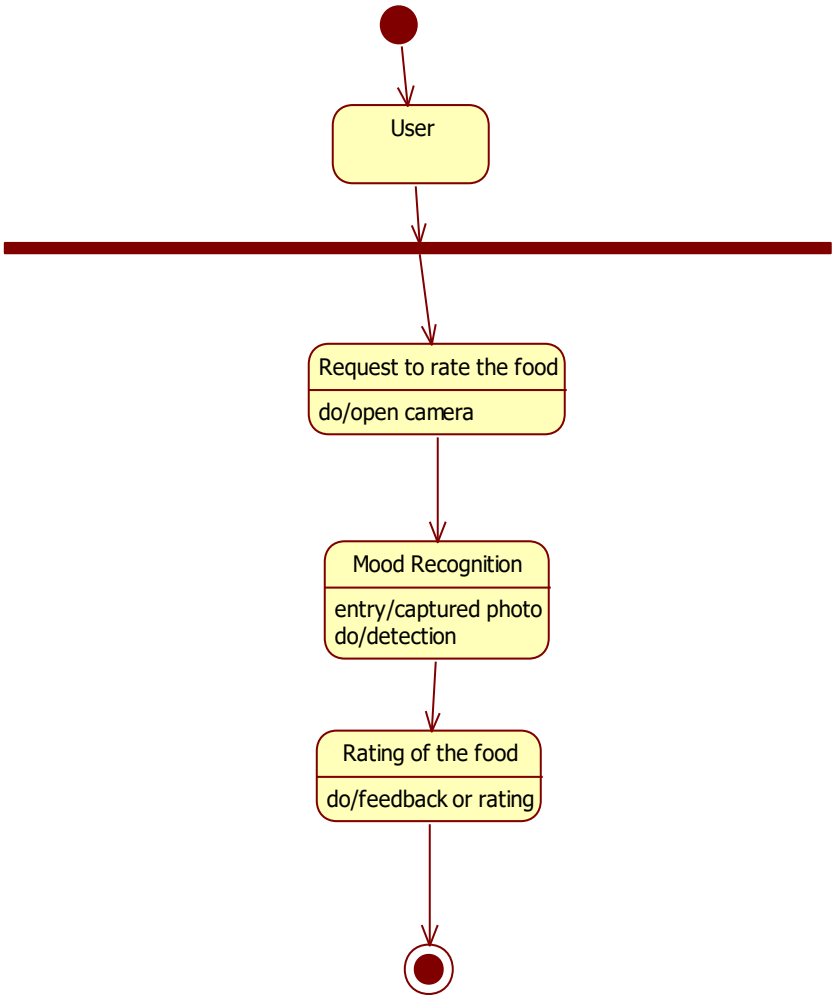


Figure 4.5 – State Chart

4.6 Activity:

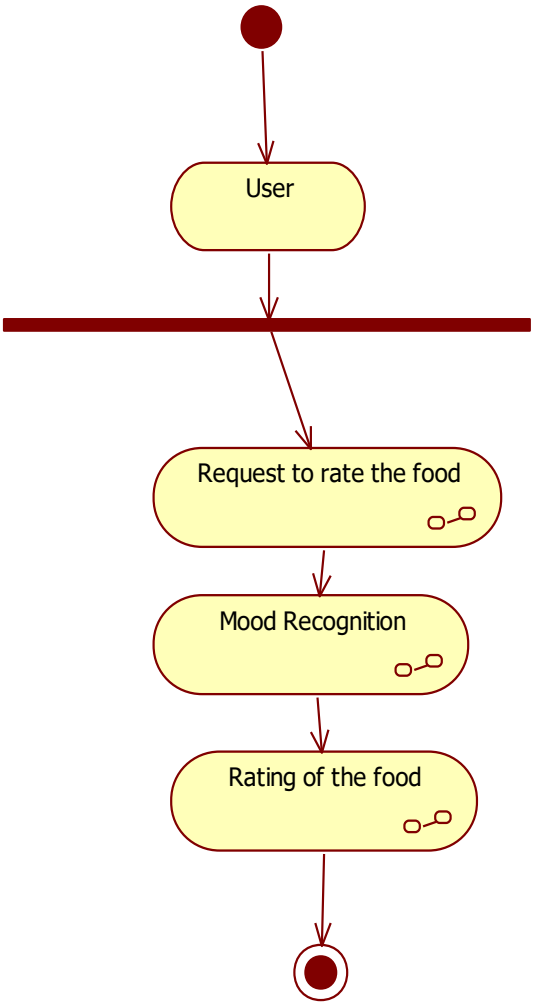


Figure 4.6 - Activity

4.7 Component:

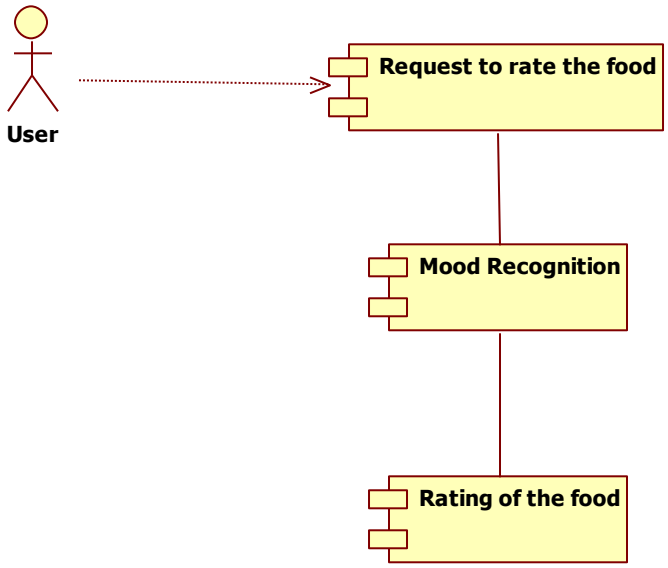


Figure 4.7 - Component

4.8 Deployment:

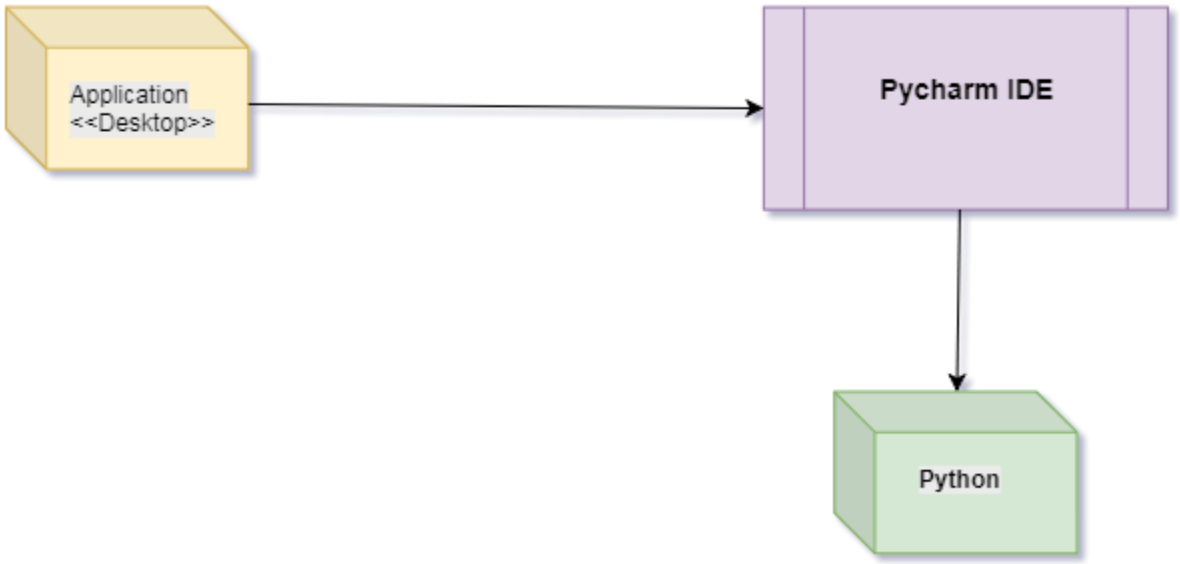


Figure 4.8 - Deployment

5.DATABASE DESIGN

5.1 About MySQL:

MySQL is a relational database management system (RDBMS)[1] that runs as a server providing multi-user access to a number of databases. The SQL phrase stands for Structured Query Language. http://en.wikipedia.org/wiki/MySQL - cite_note-3Free-software-open source projects that require a full-featured database management system often use MySQL. For commercial use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: TYPO3, Joomla, WordPress, phpBB, Drupal and other software built on the LAMP software stack. MySQL is also used in many high-profile, large-scale World Wide Web products, including Wikipedia, Google, Facebook, http://en.wikipedia.org/wiki/MySQL - cite_note-mysqlatfacebook-6 and Twitter.

MySQL is the world's most popular open source database software, with over 100 million copies of its software downloaded or distributed throughout it's history. With its superior speed, reliability, and ease of use, MySQL has become the preferred choice for Web, Web 2.0, SaaS, ISV, Telecom companies and forward-thinking corporate IT Managers because it eliminates the major problems associated with downtime, maintenance and administration for modern, online applications.

Many of the world's largest and fastest-growing organizations use MySQL to save time and money powering their high-volume Web sites, critical business systems, and packaged software — including industry leaders such as Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, Wikipedia, and Booking.com.

The flagship MySQL offering is MySQL Enterprise, a comprehensive set of production-tested software, proactive monitoring tools, and premium support services available in an affordable annual subscription.

MySQL is a key part of LAMP (Linux, Apache, MySQL, PHP / Perl / Python), the fast-growing open source enterprise software stack. More and more companies are using LAMP as an alternative to expensive proprietary software stacks because of its lower cost and freedom from platform lock-in.

MySQL was originally founded and developed in Sweden by two Swedes and a Finn: David Axmark, Allan Larsson and Michael "Monty" Widenius, who had worked together since the 1980's. More historical information on MySQL is

5.2 Database Tables

```
CREATE DATABASE /*!32312 IF NOT EXISTS*/`resturantfb` /*!40100 DEFAULT
CHARACTER SET latin1 */;
USE `resturantfb`;
/*Table structure for table `feedback` */
DROP TABLE IF EXISTS `feedback`;
CREATE TABLE `feedback` (
  `rname` varchar(500) DEFAULT NULL,
  `rating` varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*Data for the table `feedback` */
insert  into `feedback`(`rname`,`rating`) values
('Almas','1'),('Almas','2'),('Spicy','3'),('Rayalaseema Ruchulu','2'),('Rayalaseema
Ruchulu','3'),('Spicy','2'),('Paradise','3'),('Paradise','3');
/*Table structure for table `ratings` */
DROP TABLE IF EXISTS `ratings`;
CREATE TABLE `ratings` (
  `rname` varchar(500) DEFAULT NULL,
  `rating` varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*Data for the table `ratings` */
insert  into `ratings`(`rname`,`rating`) values ('Almas','1'),('Spicy','2'),('Rayalaseema
Ruchulu','2'),('Paradise','2');
```

5.3 Data flow diagrams

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Game and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

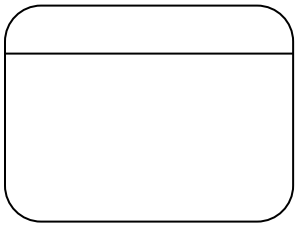
Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

A DFD is also known as a "bubble Chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

DFD Symbols:

In the DFD, there are four symbols

1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data



Process that transforms data flow.

Source or Destination of data

Data flow

Data Store

Figure:5.1 – DFD Symbols

Constructing a DFD:

Several rules of thumb are used in drawing DFD'S:

Process should be named and numbered for an easy reference. Each name should be representative of the process. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal. When a process is exploded into lower level details, they are numbered. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized. A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out. Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

Silent Feature of DFD's

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

Data Flow:

1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.
2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
3) A data flow cannot go directly back to the same process it leads. There must be at least one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
4) A Data flow to a data store means update (delete or change).

5) A data Flow from a data store means retrieve or use. A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package.

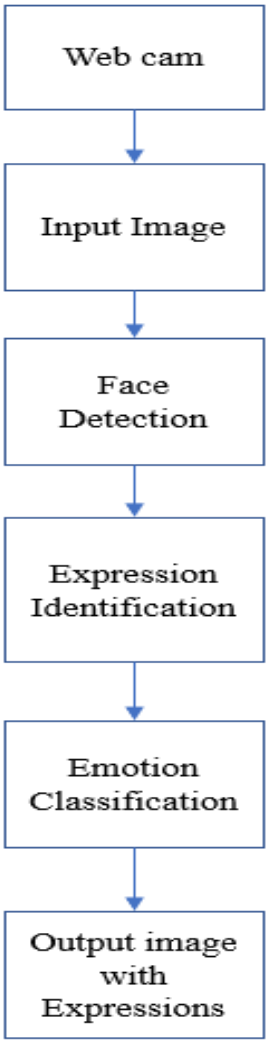


Figure 5.2: Data Flow

5.1 Manual test case

SNO	Test case	Action	Result
1	Home	Click on the Button	Main Page will open
2	Select Food Items	NULL	Validation Message
3	Add	Click	Save
4	Submit	Click	Result
5	Dedication	Click on cam	On webcam

Tables 5.1 Manual test case

6. RESULTS AND DISCUSSION

6.1- Dialogue Box for selecting food item

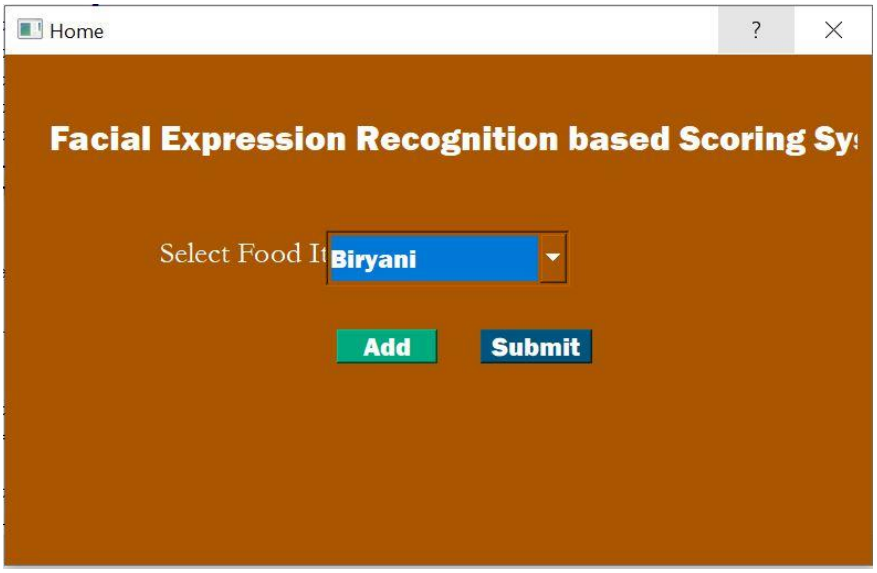


Figure 6.1: Dialogue Box for Selecting Food

The dialogue box helps us to select different food items which are being served by the restaurant for the review.

6.2- Camera dialogue

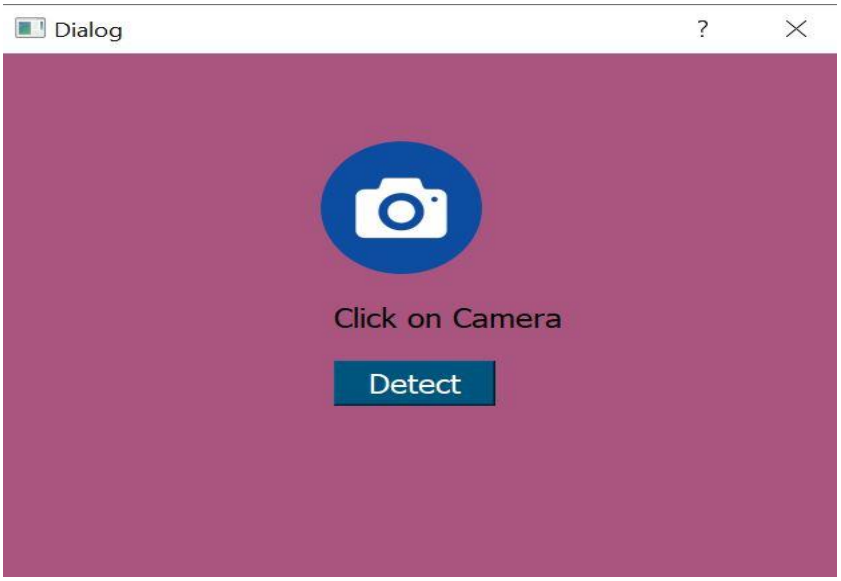


Figure 6.2: Camera dialogue

On selecting the food items when we click on submit it will navigate an another dialogue box which helps to open the camera and captures the individual image

6.3 - Capturing facial expression

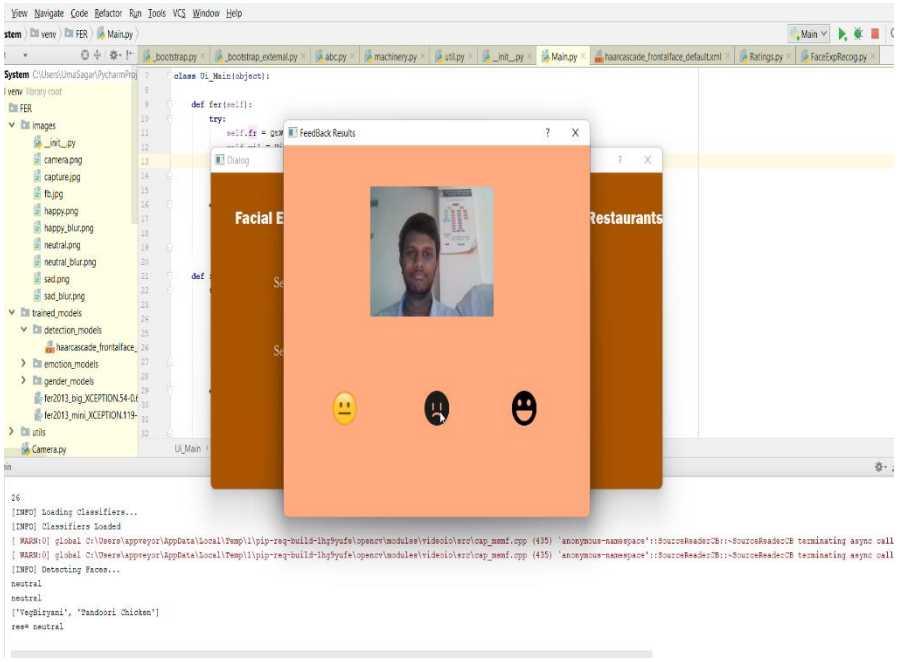


Figure 6.3: Capturing Facial Expression

Here an emoji is being classified based upon the humans expression,

6.4 Result

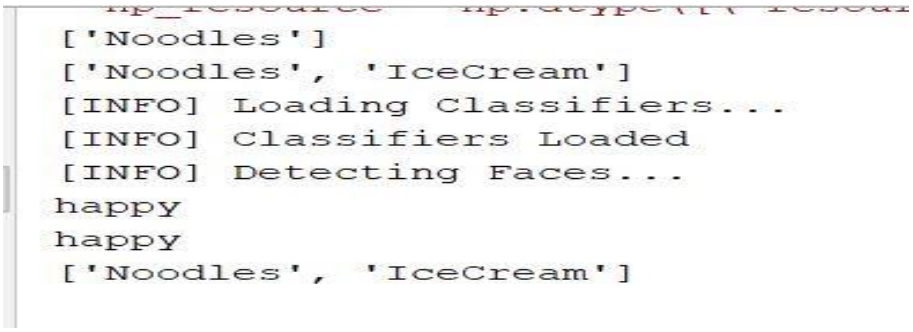


Figure 6.4 : Result

The compiler finally shows us the result of the human feelings of the food they have been served to them.

7.CONCLUSION AND FUTURE EXTENSION

7.1 Conclusion

In Our Application we are using to find the human expressions from live webcam, this application can be used in every office and police team in any meeting we can know all the expressing of the human like neutral, sad, happy, anger that will help in know the feeling of the people around the meetings or functions happing at important locations

7.2 Future Scope

The process involved in the above project requires some human involvement while capturing the face of the individual. So, a complete virtual environment is not being created. Thus taking into consideration of the above mentioned case, a capturing of an individual image can also be taken through the security cameras, live monitoring cameras which are being used by the restaurant management for monitoring purpose can be used

REFERENCES

1. Mehrabian A.: Communication Without Words. Psychology Today, Issue 2, 53--56 (1968)

2. Ekman, P., & Friesen, W. V. (1971). Constants across cultures in the face and emo-tion. Journal of Personality and Social Psychology, 17(2), 124-129.

3. P. T. Yap, R. Paramesran and Seng-HuatOng, "Image analysis by Krawtchouk moments," in IEEE Transactions on Image Processing, vol. 12, no. 11, pp. 1367-1377, Nov. 2003.

4. A. Koutlas and D. I. Fotiadis, "An automatic region-based methodology for facial expression recognition," 2008 IEEEInternational Conference on Systems, Man and Cybernetics, Singapore, 2008, pp. 662-666.

5. Velusamy S., Kannan H., Balasubramanian A., SharmaA., Navathe B..: A Method to Infer Emotions from Facial Action Units. IEEE ICASSP, 2028--2031 (2011)

6. Saaidia M., Zermi M., Ramdani M..: Facial Expression Recognition Using Neural Network Trained with Zernike Moments. IEEE Computer Society, 187--192 (2014)

7. Mandal M., Poddar S., Das A.: Comparison of Humanand Machine based Facial Ex-pression Classification. IEEEICCCA, 1198--1203 (2015)

8. Gupta A., Garg M. L.: A Human Emotion Recognition System Using Supervised Self-Organizing Maps. IEEE, 654--659 (2014)

9. Dixit B.A., Gaikwad A. N.: Statistical Moments Based Facial Expression Analysis. IEEE, 552--557 (2015)

10. M. Owayjan., R. Achkar. and M. Iskandar.: Face Detection with Expression Recognition using Artificial Neural Networks. 3rd Middle East Conference on Bi-omedical Engineering(MECBME), Beirut. IEEE MECBME, 115—119
(2016).