A PROJECT REPORT ON

Automatic Mood and Gloom Detection Using Visual Input

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

BACHELOR OF ENGINEERING

In

COMPUTER ENGINEERING

Of

SAVITRIBAI PHULE PUNE UNIVERSITY

Aniket Gholve	**B150234259**
Om Sharma	**B150234375**
Ritik Raj	**B150234359**
Raja Kumar	**B150234351**

Under The Guidance of
Prof. S.P. BHOLANE

Sinhgad Institutes

DEPARTMENT OF COMPUTER ENGINEERING
SINHGAD COLLEGE OF ENGINEERING, PUNE-41
Accredited by NAAC

2021-22

Sinhgad Technical Education Society,
Sinhgad College of Engineering, Pune-41
Department of Computer Engineering

Sinhgad Institutes

Date -20/5/2022

CERTIFICATE

This is to certify that the project report entitled
"Automatic Mood and Gloom Detection Using Visual Input"

Submitted by

Aniket Gholve	**Seat Number: B150234259**
Om Sharma	**Seat Number: B150234375**
Ritik Raj	**Seat Number: B150234359**
Raja Kumar	**Seat Number: B150234351**

is a bonafide work carried out by Aniket Gholve, Om Sharma, Ritik Raj, Raja Kumar under the supervision of Prof. S.P Bholane and it is approved for the partial fulfillment of the requirements of Savitribai Phule Pune University, Pune for the award of the degree of Bachelor of Engineering (Computer Engineering) during the year 2021-22.

Prof. S. P. Bholane
Guide
Department of Computer Engineering

Prof. M. P. Wankhade
Head
Department of Computer Engineering

Dr. S. D. Lokhande
Principal
Sinhgad College of Engineering

Abstract

In natural psychological equilibrium, tension may be generally perceived as disturbance. If a user is unable to reconcile the expectations imposed on him/ her with user capacity to deal to them, so it generates tension and produces burden on mental health.

Gloom may be generallydescribed as psychological equilibrium disruption. One of major research fields of biomedical engineering is Gloom detection, asproper Gloom prevention could be easy.

Facial expression recognition is the process of identifying human emotion. This is both something that humans do automatically but computational methodologies have also been developed.

Several bio signals (i.e. Mri, Rgb, oxygenation, Frs etc.) are available. Which are useful in identifying levels of Mood and Gloom since these signals indicate distinctive changes in the induction of Mood and Gloom. In this project, because of the easily accessible recording, ECG is used as the primary candidate. By modifying the function number and kernel type, CNN model types have been formed.

Acknowledgments

We feel great pleasure in expressing our deepest sense of gratitude and sincere thanks to our guide **Prof. S.P. Bholane** for her valuable guidance during the Project work, without which it would have been a very difficult task. I have no words to express my sincere thanks for valuable guidance, extreme assistance and cooperation extended to all the Staff Members of our Department. This acknowledgement would be incomplete without expressing our special thanks to **Prof. M.P. Wankhade** Head of the Department (Computer Engineering) for his support during the work. We would also like to extend our heartfelt gratitude to our Principal, **Dr. S. D. Lokhande** who provided a lot of valuable support, mostly being behind the veils of college bureaucracy. Last but not least we would like to thank all the Teaching, Non- Teaching staff members of our department, our parents and colleagues those who helped us directly or indirectly for completing this Project successfully.

Aniket Gholve

Om Sharma

Ritik Raj

Raja Kumar

(B.E. Computer Engineering)

List of Figures

List of Tables

INDEX

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

Gloom and anxiety disorders are highly prevalent worldwide. Attention to the adverse effects of Gloom on patient health, as well as its associated economic burden has been warranted. To support objective Gloom assessment, the affective computing community engaged signal processing, computer vision and machine learning approaches for analyzing verbal and non-verbal behavior of Gloom patients [1] and made predictions about what patterns should be indicative of Gloom state [2] [3]. These studies have analyzed the relationship between objective measures of voice, speech, non-verbal behavior and clinical subjective ratings of severity of Gloom for the purpose of automatic Gloom assessment. Despite majoradvances have been achieved in recent years, there are still several open research directions to be solved in the study of Gloom: Audio and video features from individual only concern the paralinguistic information, such as speaking rate, facial action units (AUs), etc. rather than the linguistic information from the speaking con- tent, which can reflect the sleep status, emotional status, feeling and other life statusof the individual. It is important to explore more effective audio, visual, linguistic and other multi-modal features, and design multi-modal fusion framework for Gloom recognition.

Due to the privacy issues, only limited Gloom datasets are currently available, and there are barely pre-training models for Gloom. Moreover, these common- used Gloom datasets also lack consistency. They have different languages, different durations, different data types and different targets, which make them difficult to be combined to increase the number of samples, therefore difficult to take advantage of deep models. Adopting some data augmentation approaches to increase the number of samples are requisite to improve the model performance.

Gloom is a state of low mood and aversion to activity. From this perspective, the study of Gloom should be closely related to affective state. However, the current researches on Gloom and affective state are relatively independent. We hypothesize that combining Gloom estimation and dimensional affective analysis simultaneously would yield more powerful Gloom analysis

1.2 Motivation

The main motivation of this project is to make the man-machine interface more flexible and easier for the user. Human experts will have privileged knowledge that codes the facial features of ageing, such as smoothness, face structure, skin inflammation, lines, and under-eye bags, when determining aged. Privileged knowledge is unavailable for test images in automated age estimates. It is hypothesized that asymmetric data may be used to resolve this problem. To enhance the generalizable of the trained model, be explored and exploited.

1.3 Objective

- The main goal of this project is to Detect Gloom of Person
- The aim of the system is to continuously track human Gloomlevel and try to contain it within normal levels.

CHAPTER 2
LITERATURE SURVEY

2.1 STUDY OF RESEARCH PAPERS

A research paper is a document of a scientific article that contains relevant expertise, including substantive observations, and also references to a specific subject of philosophy and technique. Use-secondary references are reviewed in literature and no current or initial experimental work is published.

We have studied the research papers and have taken out few terminologies from it which are listed below:

CNN (Convolutional Neural Network):

The term Deep Learning or Deep Neural Network refers to Artificial Neural Networks (ANN) with multi layers. Over the last few decades, it has been considered to be one of the most powerful tools, and has become very popular in the literature as it is able to handle a huge amount of data. The interest in having deeper hidden layers has recently begun to surpass classical methods performance in different fields; especially in pattern recognition. One of the most popular deep neural networks is the Convolutional Neural Network (CNN). It take this name from mathematical linear operation between matrixes called convolution. CNN have multiple layers; including convolutional layer, non-linearity layer, pooling layer and fully connected layer. The convolutional and fully- connected layers have parameters but pooling and non-linearity layers don't have parameters. The CNN has an excellent performance in machine learning problems. Especially the applications that deal with image data, such as largest image classification data set (Image Net), computer vision, and in natural language processing (NLP) and the results achieved were very amazing. Convolutional neural network layer types mainly include three types, namely convolutional layer, pooling layer and fully-connected layer.

Convolutional Layer:

Convolutional layer is the core part of the Convolutional neural network, which has local connections and weights of shared characteristics. The aim of Convolutional layer is to learn feature representations of the inputs. As shown in above, Convolutional layer consists of several feature maps. Each neuron of the same feature map is used to extract local characteristics of different positions in the former layer, but for single neurons, its extraction is local characteristics of same positions in former different feature map. In order to obtain a new feature, the input feature maps are first convolved with a learned kernel and then the results are passed into a nonlinear activation function. We will get different feature maps by applying different kernels. The typical activation function are sigmoid, tanh and Relu .

Pooling Layer:

The sampling process is equivalent to fuzzy filtering. The pooling layer has the effect of the secondary feature extraction, it can reduce the dimensions of the feature maps and increase the robustness of feature extraction. It is usually placed between two Convolutional layers. The size of feature maps in pooling layer is determined according to the moving step of kernels. The typical pooling operations are average pooling and max pooling. We can extract the high-level characteristics of inputs by stacking several Convolutional layer and pooling layer

Fully-connected Layer:

In general, the classifier of Convolutional neural network is one or more fully-connected layers. They take all neurons in the previous layer and connect them to every single neuron of current layer. There is no spatial information preserved in fully-connected layers. The last fully-connected layer is followed by an output layer. For classification tasks, SoftMax regression is commonly used because of it generating a well-performed probability distribution of the outputs. Another commonly used method is SVM, which can be combined with CNNs to solve different classification tasks

1. **Paper Name:** Emotion Recognition Using Convolutional Neural Network (CNN

Author: Nur Alia Syahirah Badrulhisham, Nur Nabilah Abu Mangshor.

Overview: Emotion is an expression that human use in expressing their feelings. It can be express through facial expression, body language and voice tone. Humans' facial expression is a major way in conveying emotion since it is the most powerful, natural and universal signal to express humans' emotion condition. However, humans' facial expression has similar patterns, and it is very confusing in recognizing the expression using naked eye. For instance, afraid and surprised is very similar to one another. Thus, this will lead to confusion in determining the facial expression. Hence, this study aims to develop a mobile based application for emotion recognition that can recognize emotion based on facial expression in real-time. The Deep Learning based technique, Convolutional Neural Network (CNN) is implemented in this study. The Mobile Net algorithm is deployed to train the model for recognition. There are four types of facial expressions to be recognized which are happy, sad, surprise, and disgusting. As the result, this study obtained 85% recognition accuracy. In the future, the developed application could be improved by adding more face expression categories

2. Paper Name: Emotion recognition from facial expression using deep convolutional neural network.

Author: D Y Liliana

Overview: Automatic facial expression recognition is actively emerging research in Emotion Recognition. This paper extends the deep Convolutional Neural Network (CNN) approach to facial expression recognition task. This task is done by detecting the occurrence of facial Action Units (AUs) as a subpart of Facial Action Coding System (FACS) which represents human emotion. In the CNN fully-connected layers we employ a regularization method called "dropout" that proved to be very effective to reduce overfitting. This research uses the extended Cohn Kanade (CK+) dataset which is collected for facial expression recognition experiment. The system performance gain average accuracy rate of 92.81%. The system has been successfully classified eight basic emotion classes. Thus, the proposed method is proven to be effective for emotion recognition.

Keywords: facial expression recognition, action unit, deep learning, convolutional neural network,dropout

3. **Paper Name:** Convolutional Neural Network (CNN) for Image Detection and Recognition

Author: Rahul Chauhan, Kamal Kumar Ghanshala, R.C Joshi

Overview: Deep Learning algorithms are designed in such a way that they mimic the function of the human cerebral cortex. These algorithms are representations of deep neural networks i.e. neural networks with many hidden layers. Convolutional neural networks are deep learning algorithms that can train large datasets with millions of parameters, in form of 2D images as input and convolve it with filters to produce the desired outputs. In this article, CNN models are built to evaluate its performance on image recognition and detection datasets. The algorithm is implemented on MNIST and CIFAR-10 dataset and its performance are evaluated. The accuracy of models on MNIST is 99.6 %, CIFAR-10 is using real-time data augmentation and dropout on CPU unit.

Keywords- Deep Learning, Handwritten digit Recognition, Object Detection, Convolutional neural networks, MNIST, CIFAR-10, Dropout, Overfitting, Data Augmentation, Relu

4. Paper Name: Facial Emotion Recognition using Convolutional Neural Networks

Author: Akash Saravanan, Gurudutt Perichetla, Dr. K.S.Gayathri

Overview: Facial expression recognition is a topic of great interest in most fields from artificial intelligence and gaming to marketing and healthcare. The goal of this paper is to classify images of human faces into one of seven basic emotions. A number of different models were experimented with, including decision trees and neural networks before arriving at a final Convolutional Neural Network (CNN) model. CNNs work better for image recognition tasks since they are able to capture spacial features of the inputs due to their large number of filters. The proposed model consists of six convolutional layers, two max pooling layers and two fully connected layers. Upon tuning of the various hyperparameters, this model achieved a final accuracy of 0.60.

5. Paper Name: Automated depression analysis using convolutional neural networks from speech

Author: Lang Hea, Cui Caob

Overview: To help clinicians to efficiently diagnose the severity of a person's depression, the affective computing community and the artificial intelligence field have shown a growing interest in designing automated systems. The speech features have useful information for the diagnosis of depression. However, manually designing and domain knowledge are still important for the selection of the feature, which makes the process labor consuming and subjective. In recent years, deep-learned features based on neural networks have shown superior performance to hand-crafted features in various areas. In this paper, to overcome the difficulties mentioned above, we propose a combination of hand-crafted and deep-learned features which can effectively measure the severity of depression from speech. In the proposed method, Deep Convolutional Neural Networks (DCNN) are firstly built to learn deep-learned features from spectrograms and raw speech waveforms. Then we manually extract the state-of-the-art texture descriptors named median robust extended local binary patterns (MRELBP) from spectrograms. To capture the complementary information within the hand-crafted features and deep-learned features, we propose joint fine-tuning layers to combine the raw and spectrogram DCNN to boost the depression recognition performance. Moreover, to address the problems with small samples, a data augmentation method was proposed. Experiments conducted on AVEC2013 and AVEC2014 depression databases show that our approach is robust and effective for the diagnosis of depression when compared to state-of-the-art,audio-based,methods

6. Paper Name: Facial emotion recognition using convolutional neural networks (FERC)

Author: Ninad Mehendale

Overview: Facial expression for emotion detection has always been an easy task for humans, but achieving the same task with a computer algorithm is quite challenging. With the recent advancement in computer vision and machine learning, it is possible to detect emotions from images. In this paper, we propose a novel technique called facial emotion recognition using convolutional neural networks (FERC). The FERC is based on two-part convolutional neural network (CNN): The first part removes the background from the picture, and the second part concentrates on the facial feature vector extraction. In FERC model, expressional vector (EV) is used to find the five different types of regular facial expression. Supervisory data were obtained from the stored database of 10,000 images (154 persons). It was possible to correctly highlight the emotion with 96% accuracy, using a EV of length 24 values. The two-level CNN works in series, and the last layer of perceptron adjusts the weights and exponent values with each iteration. FERC difers from generally followed strategies with single-level CNN, hence improving the accuracy. Furthermore, a novel background removal procedure applied, before the generation of EV, avoids dealing with multiple problems that may occur (for example distance from the camera). FERC was extensively tested with more than 750K images using extended Cohn–Kanade expression, Caltech faces, CMU and NIST datasets. We expect the FERC emotion detection to be useful in many applications such as predictive learning of students, lie detectors, etc.

7. Paper Name: Real-Time CNN Based ST Depression Episode Detection Using Single-Lead ECG

Author: Erhan Tiryaki, Akshay Sonawane, Lakshman Tamil.

Overview: A method for real monitoring of the heart for ST Depression episodes is described here. We have developed a convolutional neural network (CNN) based machine learning algorithm for classifying ECG signals into normal or ST-depression episodes of the heart with an accuracy over 92%. Our algorithm is capable of detecting ST-depression episodes of varying duration. The algorithm is evaluated using European ST-T Database. The best results obtained here are 0.95%, 0.98%, and 0.91% respectively for accuracy, sensitivity, and specificity.

Keywords—electrocardiogram (ECG), convolutional neural network (CNN), machine learning, ST segment depression detection, real-time ST depression detection

8. Paper Name: Automatic Depression Recognition using CNN with Attention Mechanism from Videos

Author: Lang He, Jonathan Cheung-Wai Chan , Zhongmin Wang.

Overview: Artificial intelligence (AI) has incorporated various automatic systems and frameworks to diagnose the severity of depression using hand-crafted features. However, process of feature selection needs domain knowledge and is still timeconsuming and subjective. Deep learning technology has been successfully adopted for depression recognition. Most previous works pre-train the deep models on large databases followed by fine-tuning with depression databases (i.e., AVEC2013, AVEC2014). In the present paper we propose an integrated framework - Deep Local Global Attention Convolutional Neural Network (DLGACNN) for depression recognition, which adopts CNN with attention mechanism as well as weighted spatial pyramid pooling (WSPP) to learn a deep and global representation. Two branches are introduced: Local Attention based CNN (LACNN) focuses on the local patches, while Global Attention based CNN (GACNN) learns the global patterns from the entire facial region. To capture the complementary information between the two branches, Local-Global Attentionbased CNN (LGA-CNN) is proposed. After feature aggregation, WSPP is used to learn the depression patterns. Comprehensive experiments on AVEC2013 and AVEC2014 depression databases have demonstrated that the proposed method is capable of mining the underlying depression patterns of facial videos and outperforms the most of the state-of-the-art video-based depression recognition approaches. Keywords: Depression, CNN with attention mechanism, Local Attention based CNN (LA-CNN), Global Attention based CNN (GA-CNN)

9. Paper name: FACIAL EMOTION RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK

Author: Milan Tripathi

Overview: Facial expressions play a significant role in social communication since they convey a lot of information about people, such as moods, emotions, and other things. Many researchers gained an optimal accuracy in most of the popular facial recognition datasets: CK+, JAFFE, IEV, but in FER2013 the best model accuracy is about 74%. This article purpose deep learning-based models to mitigate this issue. Three models based on AlexNet, VGG19, and ResNet50 are used to train with the dataset, and the very best model among them is further analyzed. The best model is trained using various optimizers and evaluated based on its training and testing accuracy, confusion matrix, ROC Curve. The finest model gained an accuracy of 91.89504% which is better than past state of art models by at least 17% accuracy.

10. Paper Name: On the review of image and video-based depression detection using machine learning

Author: Arselan Ashraf , Teddy Surya Gunawan , Bob Subhan Riza , Edy Victor Haryanto, Zuriati Janin.

Overview: Machine learning has been introduced in the sphere of the medical field to enhance the accuracy, precision, and analysis of diagnostics while reducing laborious jobs. With the mounting evidence, machine learning has the capability to detect mental distress like depression. Since depression is the most prevalent mental disorder in our society at present, and almost the majority of the population suffers from this issue. Hence there is an extreme need for the depression detection models, which will provide a support system and early detection of depression. This review is based on the image and video-based depression detection model using machine learning techniques. This paper analyses the data acquisition techniques along with their databases. The indicators of depression are also reviewed in this paper. The evaluation of different researches, along with their performance parameters, is summarized. The paper concludes with remarks about the techniques used and the future scope of using the image and video-based depression prediction

+

CHAPTER 3

SYSTEM REQUIREMENT AND SPECIFICATIONS

3.1 Introduction

- This chapter covers the project planning and management details. It also covers system requirement Specification. SRS is considered as the base for the effort estimations and project scheduling

3.2 System Requirement Specification (SRS)

3.2.1 System Overview:

The application is a Python app which has been implemented using TensorFlow functionalities. It is consistent singular platform which caters to every user need regarding Face-based classification or analysis.

The following are the main features of the project:

1. It Extract Face from Image and extract some features like mouth eyes nose etc.
2. On Based extracted features it will show the user is Gloomed or not

3.2.2 Functional Requirements:

• Image to Text

Main flow:

User will show face on camera.

System will output the Text according to the Face gesture performed by the user.

3.2.3 Hardware Interfaces:

- RAM: 8 GB

  As we are using Machine Learning Algorithm and Various High Level Libraries Laptop RAM minimum required is 8 GB.

- Hard Disk: 40 GB

  Data Set of Stress is to be used hence minimum 40 GB Hard Disk memory is required.

  Processor: Intel i5 Processor

- PyCharm IDE that Integrated Development Environment is to be used and data loading should be fast hence Fast Processor is required

- IDE: PyCharm

  Best Integrated Development Environment as it gives possible suggestions at the time of typing code snippets that makes typing feasible and fast.

- Coding Language: Python Version 3.5

  Highly specified Programming Language for Machine Learning because of availability of High-Performance Libraries.

- Operating System: Windows 10

  Latest Operating System that supports all type of installation and development Environment

3.2.4 Software Interfaces

- Operating System: Windows 10
- IDE: PyCharm, Spyder
- Programming Language: Python

3.3 NON-FUNCTIONAL REQUIREMENT

3.3.1 Performance Requirements

The performance of the functions and every module must be well. The overall performance of the software will enable the users to work recently. Performance of encryption of data should be fast. Performance of the providing virtual environment should be fast Safety Requirement

The application is designed in modules where errors can be detected. This makes it easier to install and update new functionality if required.

3.3.2 Safety Requirement

The application is designed in modules where errors can be detected and fixed easily. This makes it easier to install and update new functionality if required.

3.3.3 Software Quality Attributes

- Our software has many quality attribute that are given below: -
- Adaptability: This software is adaptable by all users.
- Availability: This software is freely available to all users. The availability of the software is easy for everyone.
- Maintainability: After the deployment of the project if any error occurs then it can be easily maintained by the software developer.
- Reliability: The performance of the software is better which will increase the reliability of the Software.
- User Friendliness: Since, the software is a GUI application; the output generated is much user friendly in its behavior.
- Integrity: Integrity refers to the extent to which access to software or data by unauthorized persons can be controlled.
- Security: Users are authenticated using many security phases so reliable security is provided.
- Test ability: The software will be tested considering all the aspects

CHAPTER 4

ANALYSIS AND DESIGN

4.1 INTRODUCTION

This chapter covers the analysis and design of the considered system

4.2 SYSTEM ARCHITECTURE

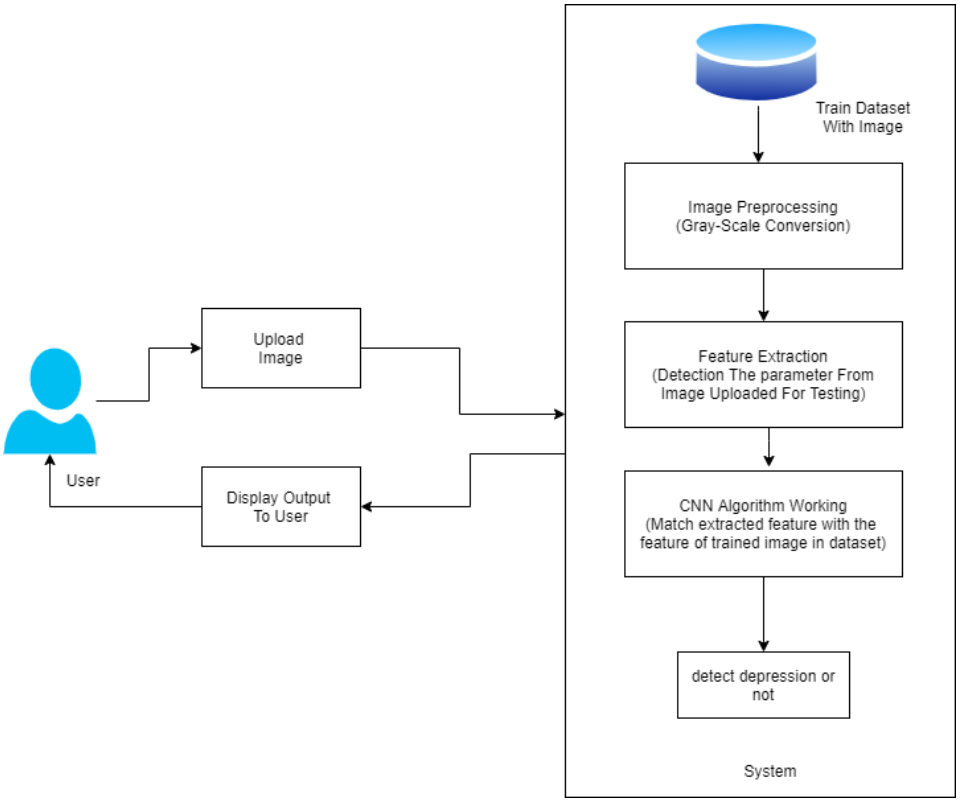


Fig 4.1: System Architecture

4.2.1 Module

- **Prepossessing:**

  In this Module Machine will be processing on given Input. In prepossessing machine will train the data-set, removing Noisy part of given input and then resize the data-set

- **Feature Extraction:**

  In this module user will give EMG, HR, ECG, RESP Seconds, etc. that attributegive to machine

- **Classification:**

  User testing value classify with train data-set using CNN Algorithm (Convolutional Neural Network Algorithm). Machine Learning will predicate given input of person is Gloom or not. to getting more accuracy, here we use machine learning with SVM (support vector Machine Algorithm).

4.2.2 Data Flow Diagram

In Data Flow Diagram, we show that flow of data in our system in DFD0 we show that base DFD in which rectangle present input as well as output and circle show our system.

In DFD1 show actual input and actual output of system input of our system is image and output are rumor detected likewise in DFD 2 we present operation of user as well as admin.

Figure 4.2: Data Flow diagram

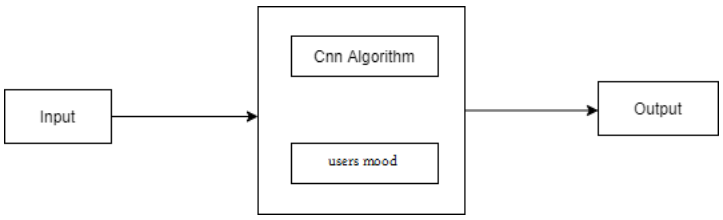


Fig 4.3: Data Flow diagram

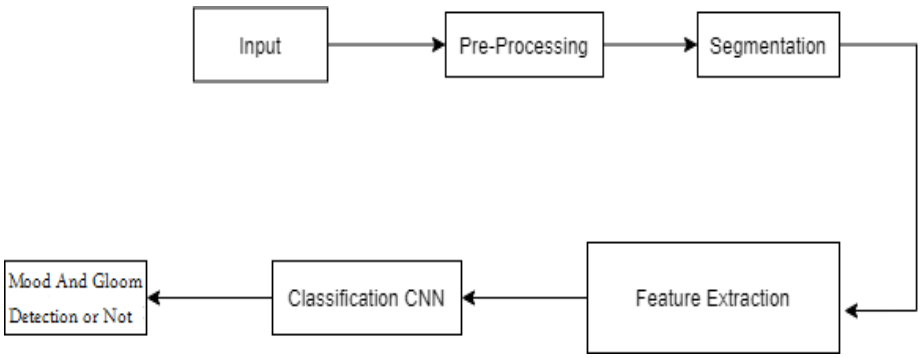


Fig 4.4: Data Flow diagram

UML DIAGRAMS

Unified Modeling Language is a standard language for writing software blueprints. The UML may be used to visualize, specify, construct and document the artifacts of a soft- ware intensive system. UML is process independent, although optimally it should be used in process that is use case driven, architecture centric, iterative, and incremental. The Number of UML Diagram is available.

Use case Diagram:

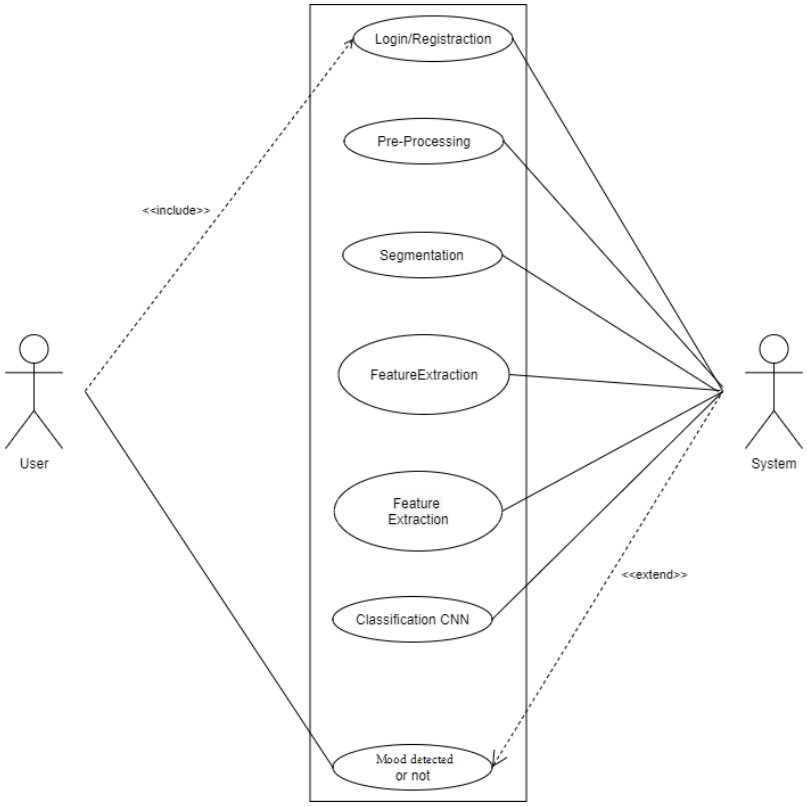


Fig 4.5: Use case Diagram

- Here we have two primary actors - User and System, who initiates the use cases.
- User login /Register and the web camera takes the recording of the user
- From recording frames are taken out and the feature extraction is done
- Using CNN classifier, the output is predicted

Activity Diagram

- First activity is show face in front of webcam

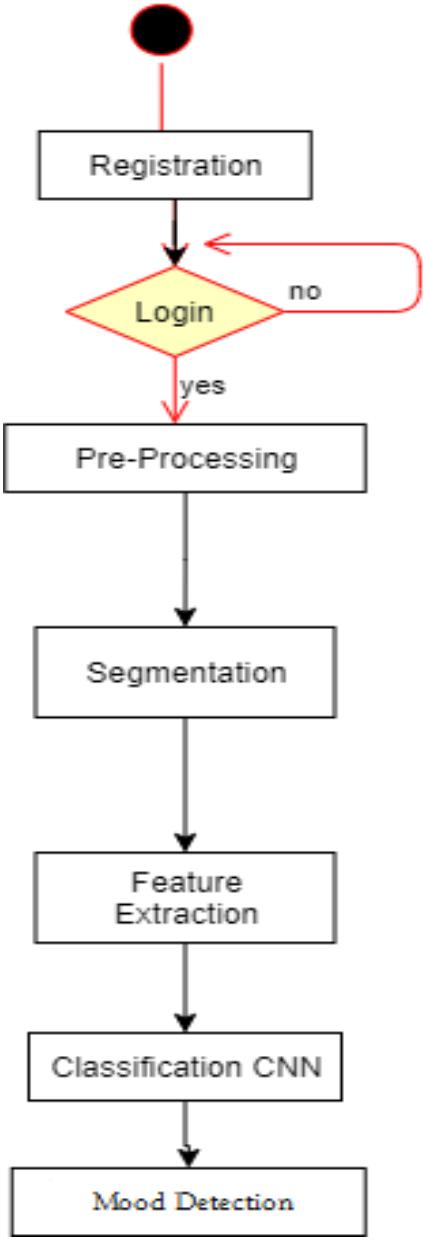


Fig 4.6: Activity Diagram

- Then the video is converted into individual frames.
- After this, the pre-processing is carried out on frames.
- The trained neural network classifies the Face Expression and displays the user is current mood and is user gloom or not

Sequence Diagram

- Signer makes hand gesture in front of the webcam

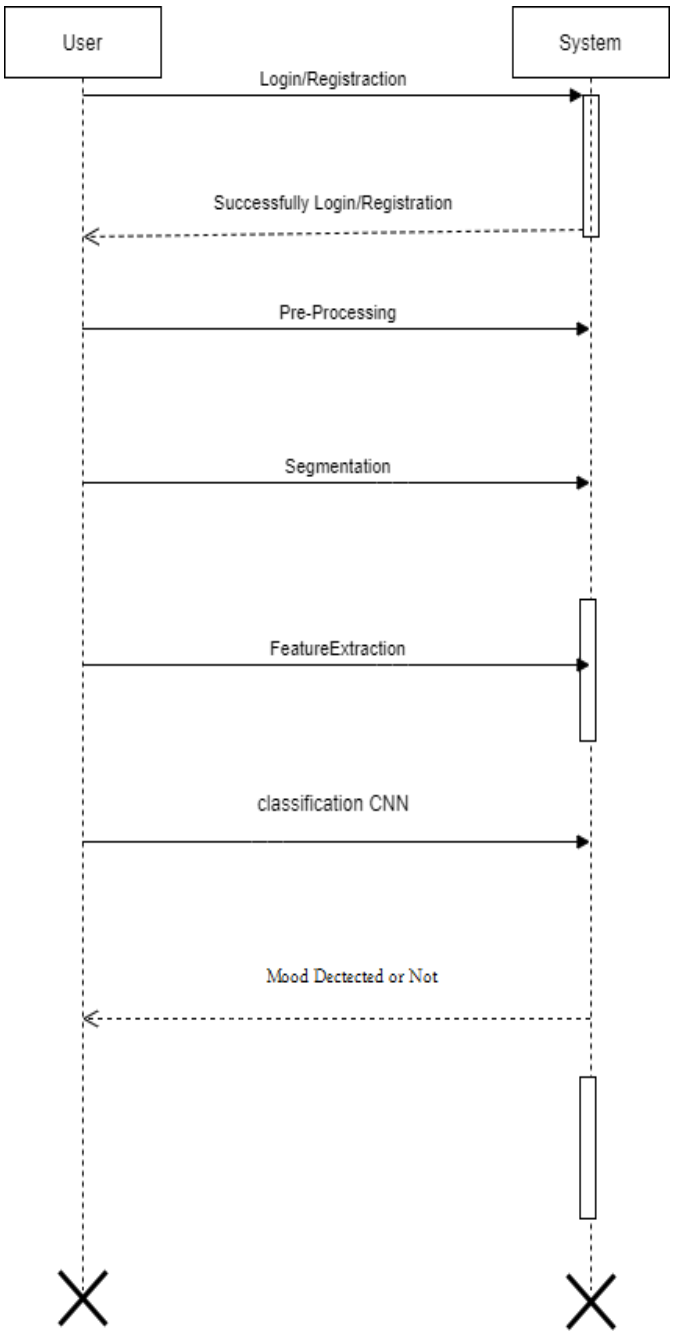


Fig 4.7: Sequence Diagram

- Then the video is converted into individual frames.
- After this, the pre-processing is carried out on frames.
- The trained neural network classifies the Face Emotions and displays it into equivalent text

ER Diagram:

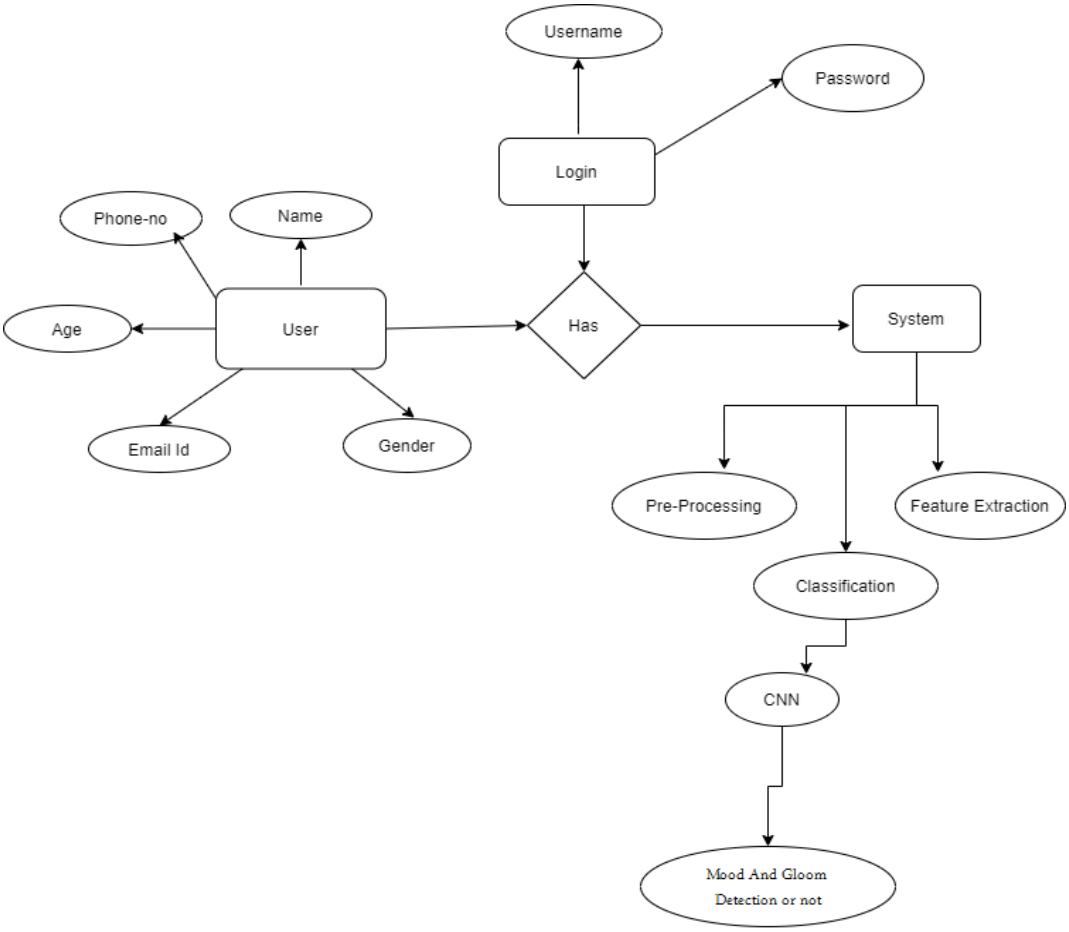


Fig 4.8: ER Diagram

Class Diagram:

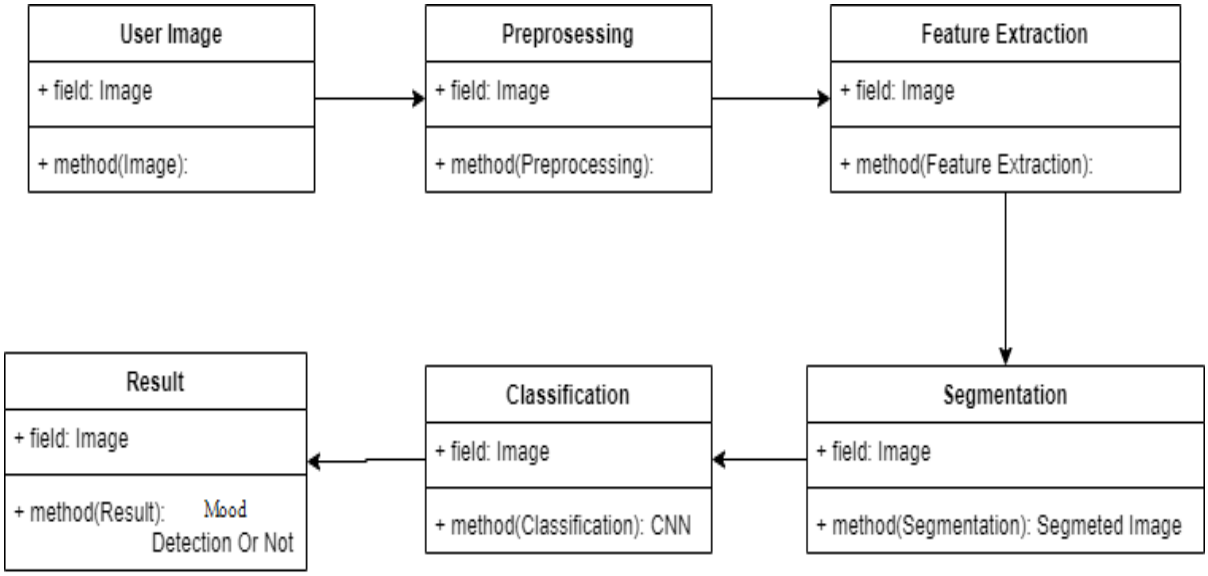


Figure 4.9: Class Diagram

- In neural network applications, class diagram is of different modules in the project
- There are five modules:
    1. User Image
    2. Preprocessing
    3. Feature Extraction
    4. Segmentation
    5. Classification
    6. Results

CHAPTER 5
IMPLEMENTATION AND CODING

5.1 Introduction:

This chapter covers the role of various subsystems along with implementation details listing of the code for the major functionalities used by the algorithm

5.2 Operational Details

5.2.1 Pipeline:

The pipeline of this project consists of 4 steps:

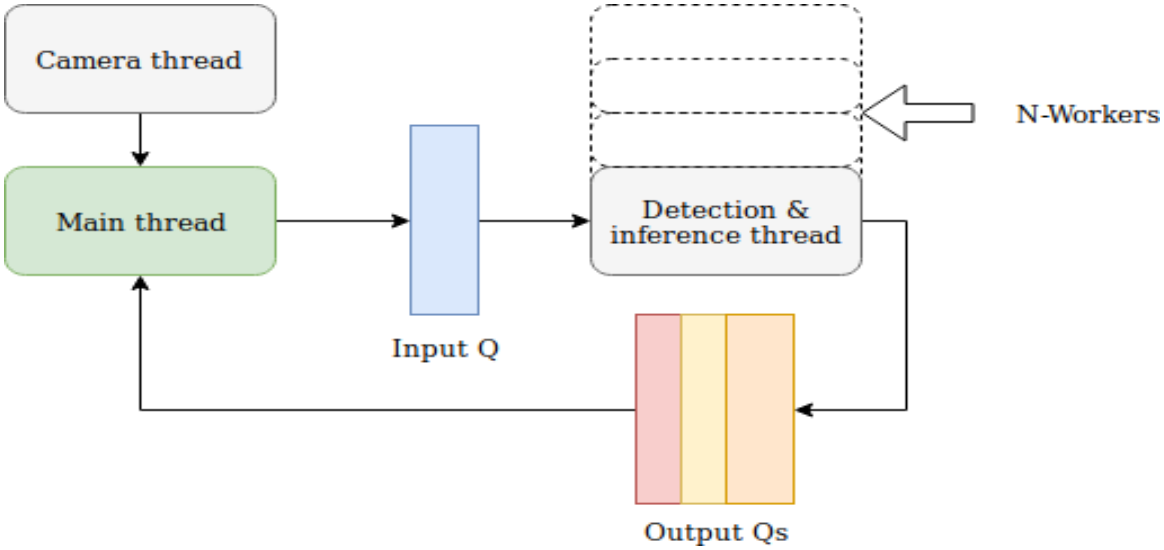


Fig 5.1: Program flow pipeline

1. **Camera Input and frame pre-processing:**

   Video input from the camera is read as frames using the open-source video processing library OpenCV. This is done with the help of read () function of OpenCV which returns Boolean for presence of frame and the frame itself.

   This frame is then converted to RGB and put into Input Queue for effective parallelization.

2. **Worker:**

   Worker is a function which is used for face detection from the frame. User can define number of workers for effective parallelization. A worker grabs theframe from the queue and pass it into the SSD. This gives us a bounding box of where the face(s) is/are and the corresponding cropped frame.

3. Inference:

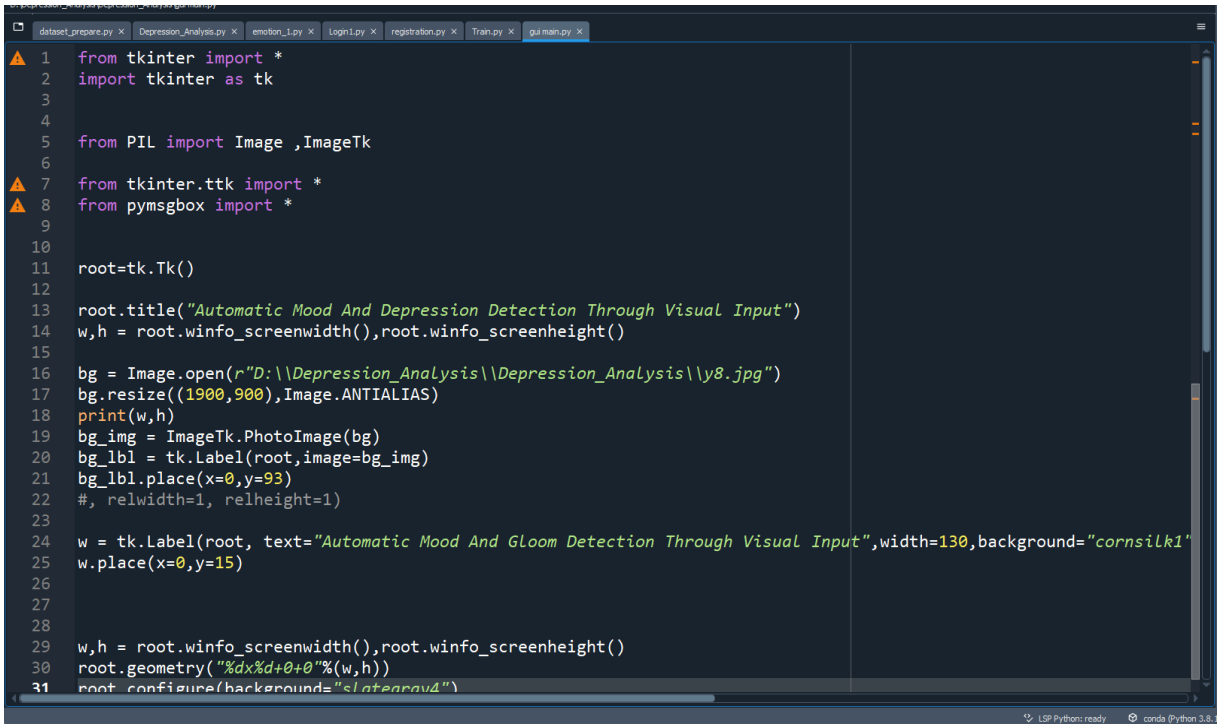
This cropped frame of the hand is then passed to the CNN, which give us a class vector output of values between 0 and 1. These values correspond to the probability of the frame to be one of the classes. The worker has finished its job and put: the frame with bounding boxdrawn on top, the cropped frame and the classes into several different queues.

4. Main Thread:

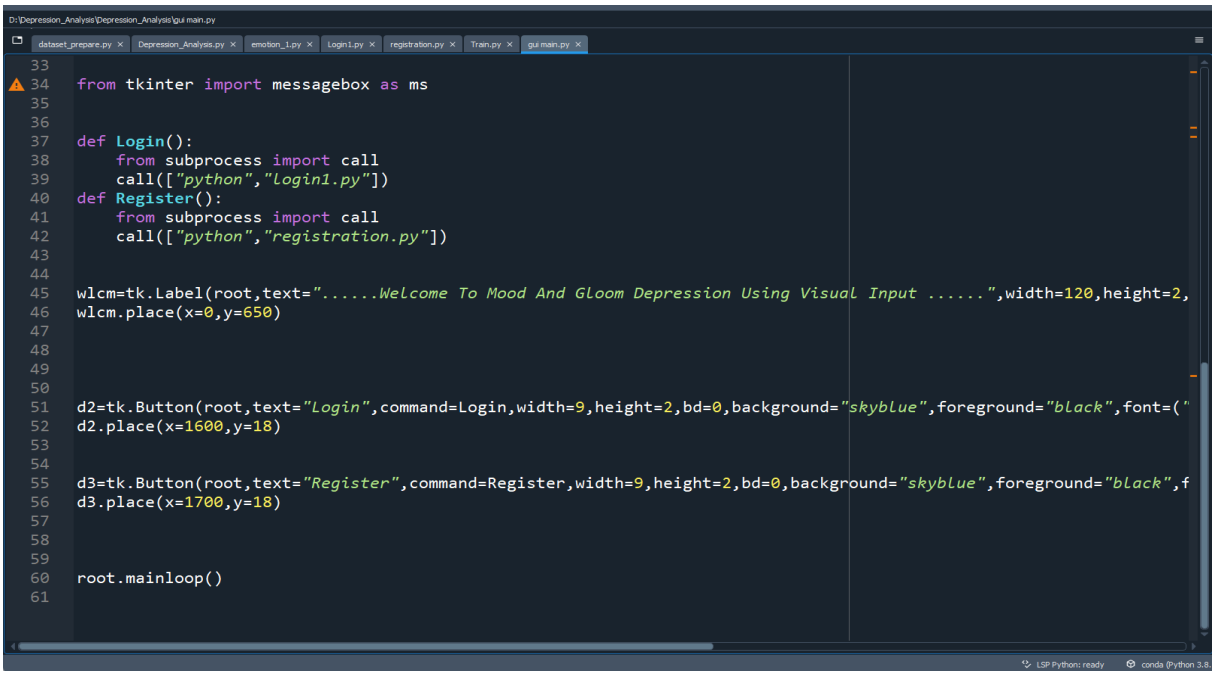
The main thread, responsible of showing the results can grab the information from the queues and display them in four different windows.

5. Code Listing

Code Snippet 1:



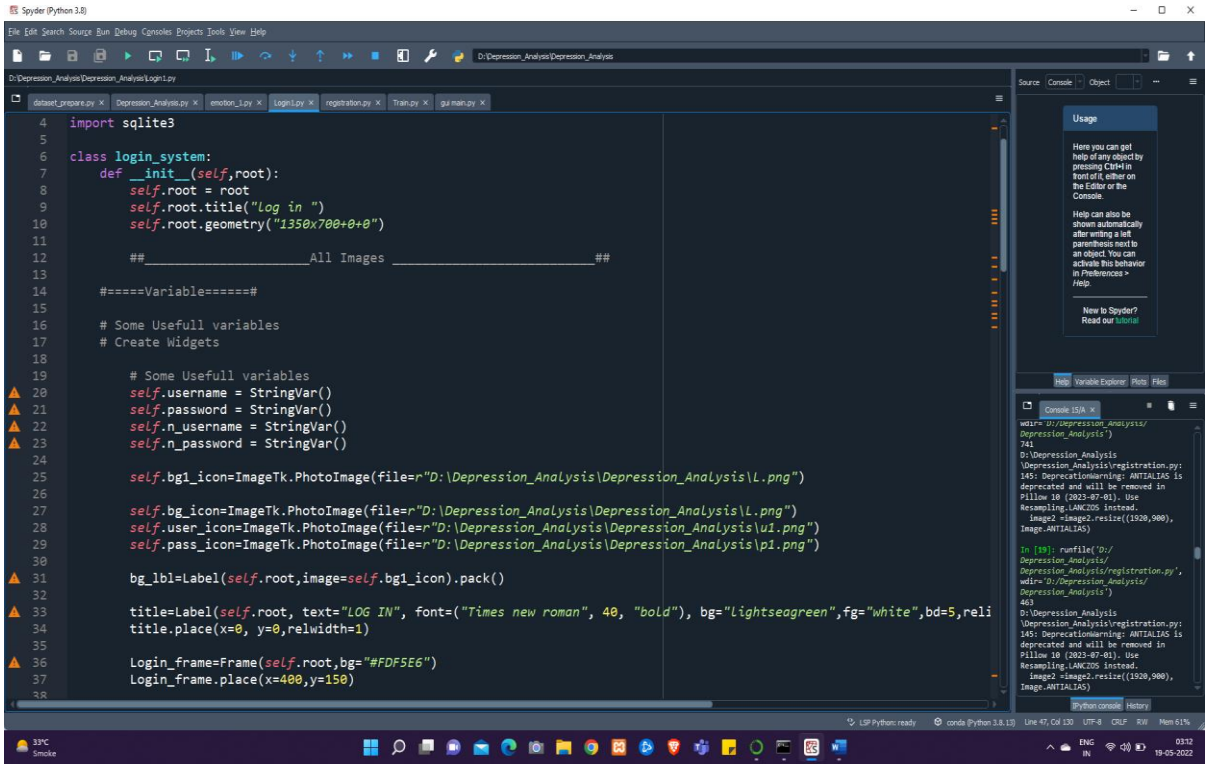
```
from tkinter import *
import tkinter as tk


from PIL import Image ,ImageTk

from tkinter.ttk import *
from pymsgbox import *


root=tk.Tk()

root.title("Automatic Mood And Depression Detection Through Visual Input")
w,h = root.winfo_screenwidth(),root.winfo_screenheight()

bg = Image.open(r"D:\\Depression_Analysis\\Depression_Analysis\\y8.jpg")
bg.resize((1900,900),Image.ANTIALIAS)
print(w,h)
bg_img = ImageTk.PhotoImage(bg)
bg_lbl = tk.Label(root,image=bg_img)
bg_lbl.place(x=0,y=93)
#, relwidth=1, relheight=1)

w = tk.Label(root, text="Automatic Mood And Gloom Detection Through Visual Input",width=130,background="cornsilk1"
w.place(x=0,y=15)



w,h = root.winfo_screenwidth(),root.winfo_screenheight()
root.geometry("%dx%d+0+0"%(w,h))
root.configure(background="slategray4")
```



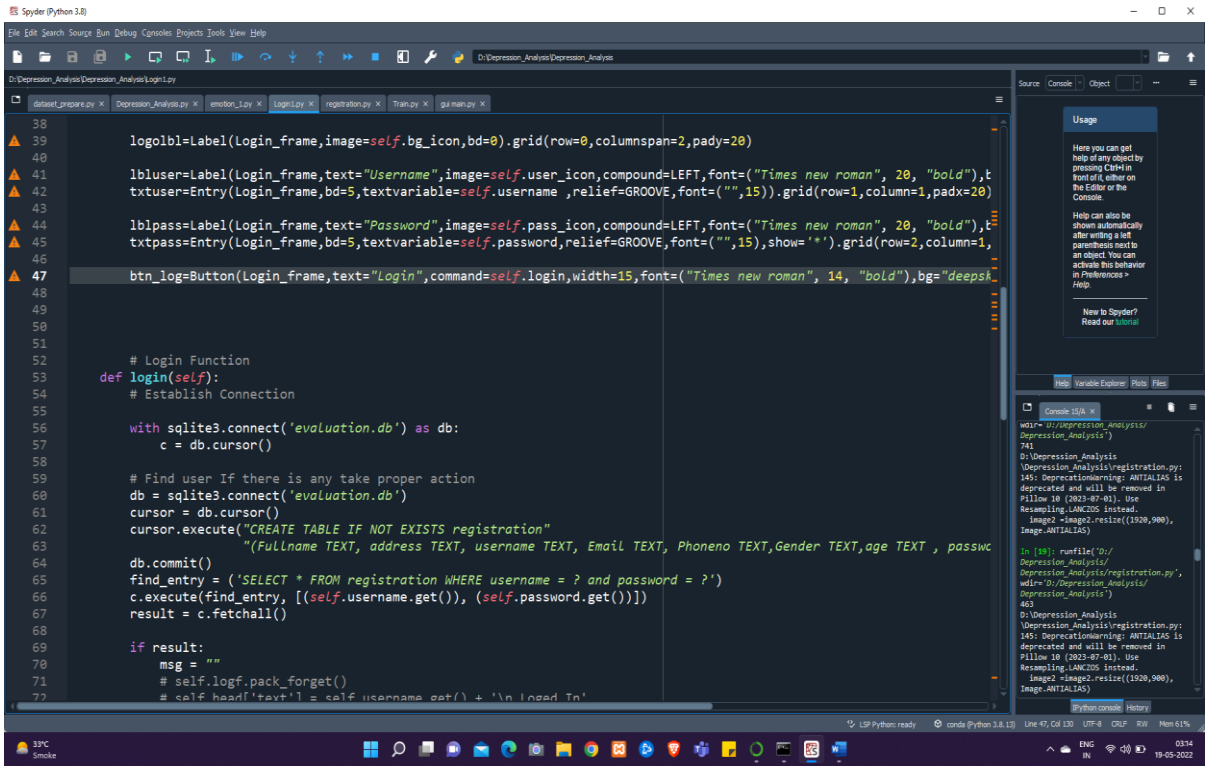
```
33
34  from tkinter import messagebox as ms
35
36
37  def Login():
38      from subprocess import call
39      call(["python","Login1.py"])
40  def Register():
41      from subprocess import call
42      call(["python","registration.py"])
43
44
45  wlcm=tk.Label(root,text="......Welcome To Mood And Gloom Depression Using Visual Input ......",width=120,height=2,
46  wlcm.place(x=0,y=650)
47
48
49
50
51  d2=tk.Button(root,text="Login",command=Login,width=9,height=2,bd=0,background="skyblue",foreground="black",font=("
52  d2.place(x=1600,y=18)
53
54
55  d3=tk.Button(root,text="Register",command=Register,width=9,height=2,bd=0,background="skyblue",foreground="black",f
56  d3.place(x=1700,y=18)
57
58
59
60  root.mainloop()
61
```

The Code Snippet above is used to create the GUI of the Application and used for user interaction

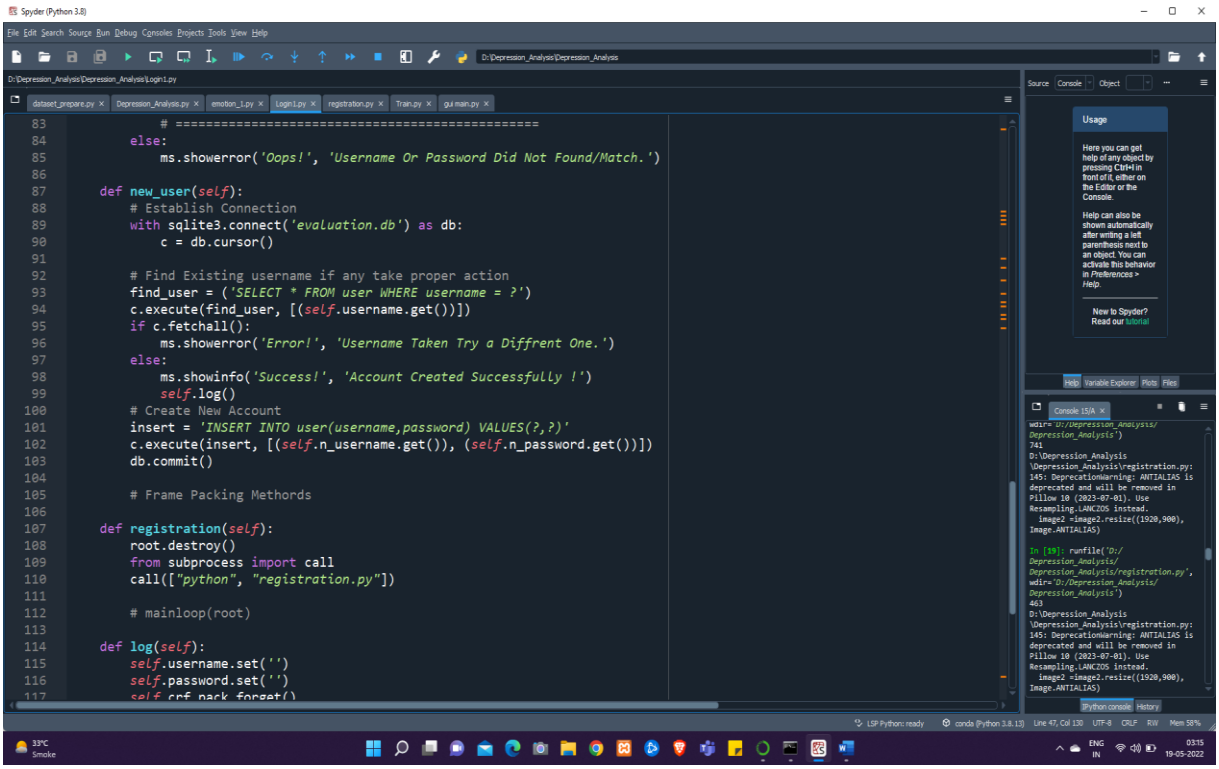
Code Snippet 2:



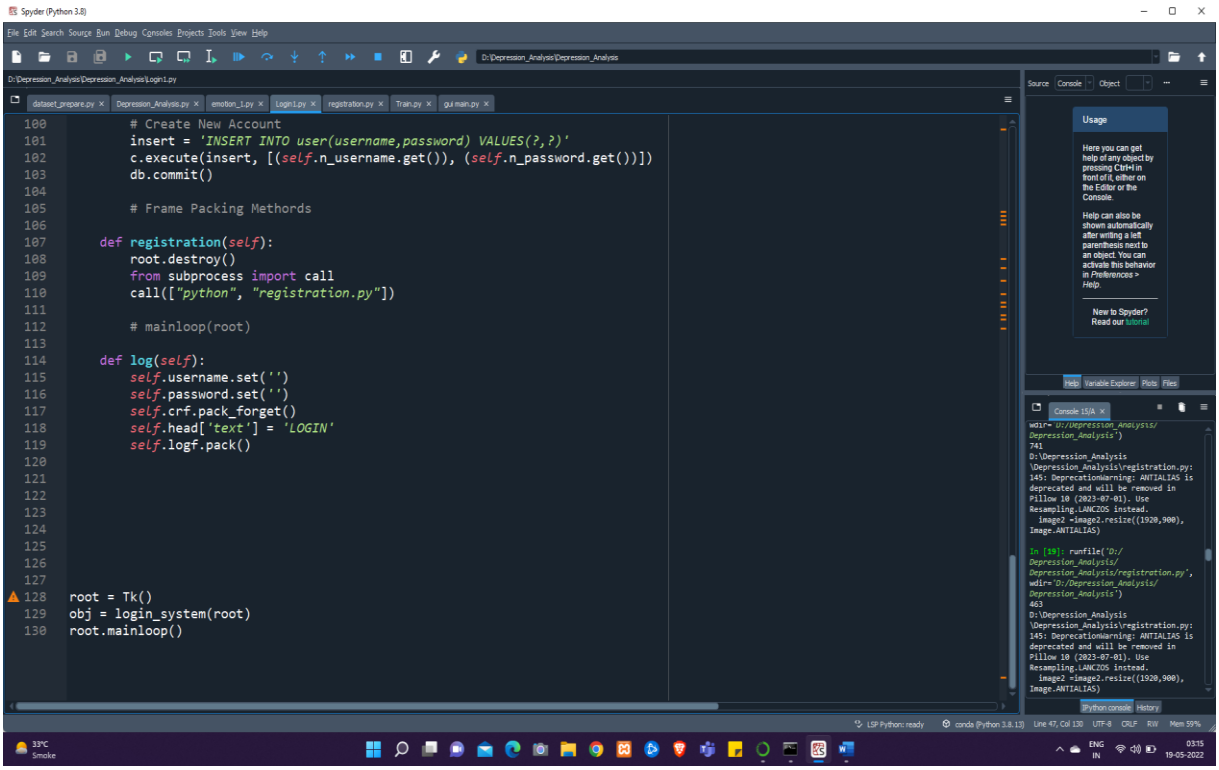
```
import sqlite3

class login_system:
    def __init__(self,root):
        self.root = root
        self.root.title("Log in ")
        self.root.geometry("1350x700+0+0")

        ##_____________________All Images _____________________##

    #=====Variable======#

    # Some Usefull variables
    # Create Widgets

        # Some Usefull variables
        self.username = StringVar()
        self.password = StringVar()
        self.n_username = StringVar()
        self.n_password = StringVar()

        self.bg1_icon=ImageTk.PhotoImage(file=r"D:\Depression_Analysis\Depression_Analysis\L.png")

        self.bg_icon=ImageTk.PhotoImage(file=r"D:\Depression_Analysis\Depression_Analysis\L.png")
        self.user_icon=ImageTk.PhotoImage(file=r"D:\Depression_Analysis\Depression_Analysis\u1.png")
        self.pass_icon=ImageTk.PhotoImage(file=r"D:\Depression_Analysis\Depression_Analysis\p1.png")

        bg_lbl=Label(self.root,image=self.bg1_icon).pack()

        title=Label(self.root, text="LOG IN", font=("Times new roman", 40, "bold"), bg="lightseagreen",fg="white",bd=5,reli
        title.place(x=0, y=0,relwidth=1)

        Login_frame=Frame(self.root,bg="#FDF5E6")
        Login_frame.place(x=400,y=150)
```



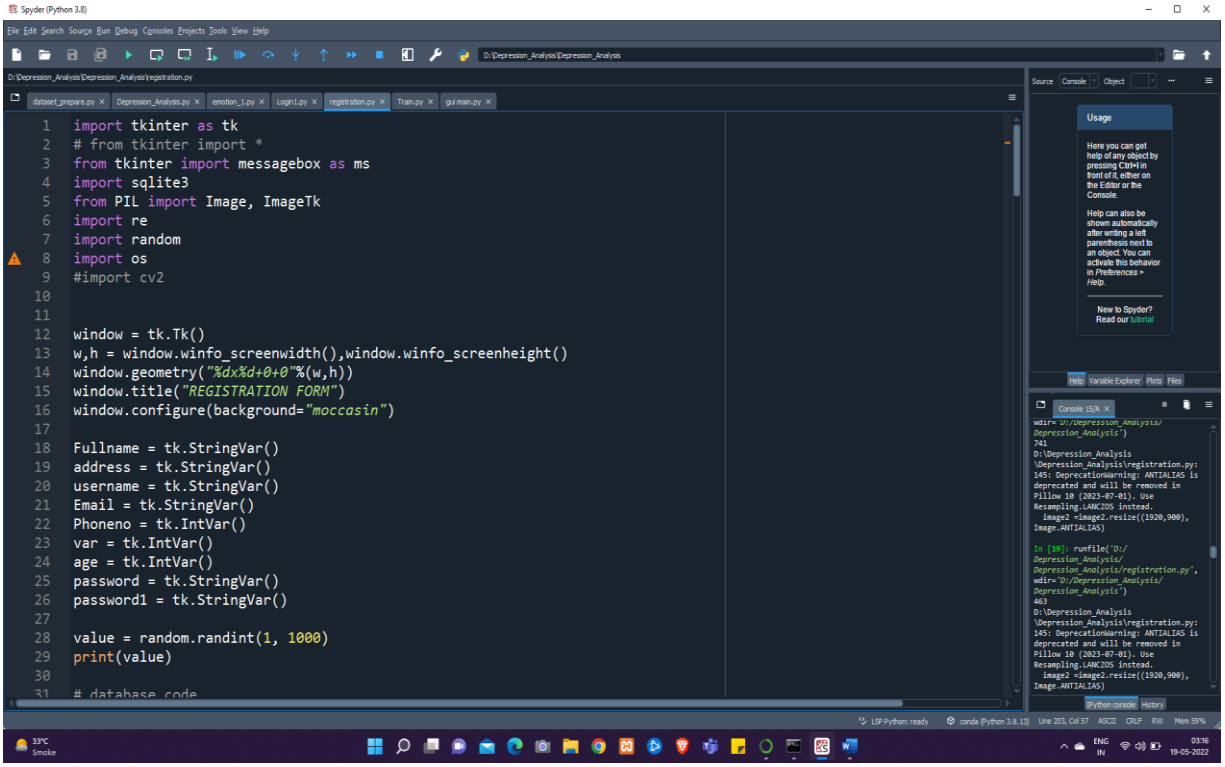
```
        logolbl=Label(Login_frame,image=self.bg_icon,bd=0).grid(row=0,columnspan=2,pady=20)

        lbluser=Label(Login_frame,text="Username",image=self.user_icon,compound=LEFT,font=("Times new roman", 20, "bold"),t
        txtuser=Entry(Login_frame,bd=5,textvariable=self.username ,relief=GROOVE,font=("",15)).grid(row=1,column=1,padx=20)

        lblpass=Label(Login_frame,text="Password",image=self.pass_icon,compound=LEFT,font=("Times new roman", 20, "bold"),t
        txtpass=Entry(Login_frame,bd=5,textvariable=self.password,relief=GROOVE,font=("",15),show='*').grid(row=2,column=1,

        btn_log=Button(Login_frame,text="Login",command=self.login,width=15,font=("Times new roman", 14, "bold"),bg="deepsk

    # Login Function
    def login(self):
        # Establish Connection

        with sqlite3.connect('evaluation.db') as db:
            c = db.cursor()

        # Find user If there is any take proper action
        db = sqlite3.connect('evaluation.db')
        cursor = db.cursor()
        cursor.execute("CREATE TABLE IF NOT EXISTS registration"
                       "(FullName TEXT, address TEXT, username TEXT, Email TEXT, Phoneno TEXT,Gender TEXT,age TEXT , passwc
        db.commit()
        find_entry = ('SELECT * FROM registration WHERE username = ? and password = ?')
        c.execute(find_entry, [(self.username.get()), (self.password.get())])
        result = c.fetchall()

        if result:
            msg = ""
            # self.logf.pack_forget()
            # self.head['text'] = self.username.get() + '\n Loged In'
```



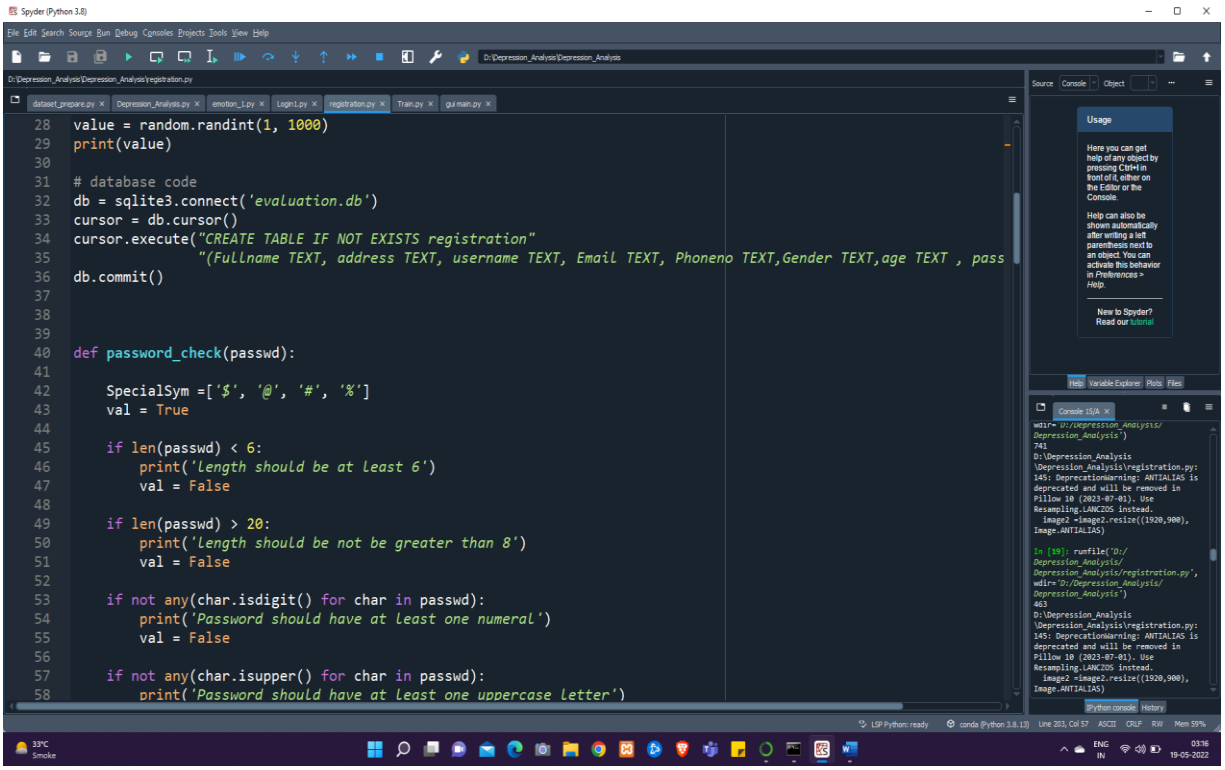
```
            # ================================================
        else:
            ms.showerror('Oops!', 'Username Or Password Did Not Found/Match.')

    def new_user(self):
        # Establish Connection
        with sqlite3.connect('evaluation.db') as db:
            c = db.cursor()

        # Find Existing username if any take proper action
        find_user = ('SELECT * FROM user WHERE username = ?')
        c.execute(find_user, [(self.username.get())])
        if c.fetchall():
            ms.showerror('Error!', 'Username Taken Try a Diffrent One.')
        else:
            ms.showinfo('Success!', 'Account Created Successfully !')
            self.log()
        # Create New Account
        insert = 'INSERT INTO user(username,password) VALUES(?,?)'
        c.execute(insert, [(self.n_username.get()), (self.n_password.get())])
        db.commit()

        # Frame Packing Methords

    def registration(self):
        root.destroy()
        from subprocess import call
        call(["python", "registration.py"])

        # mainloop(root)

    def log(self):
        self.username.set('')
        self.password.set('')
        self.crf.pack_forget()
```



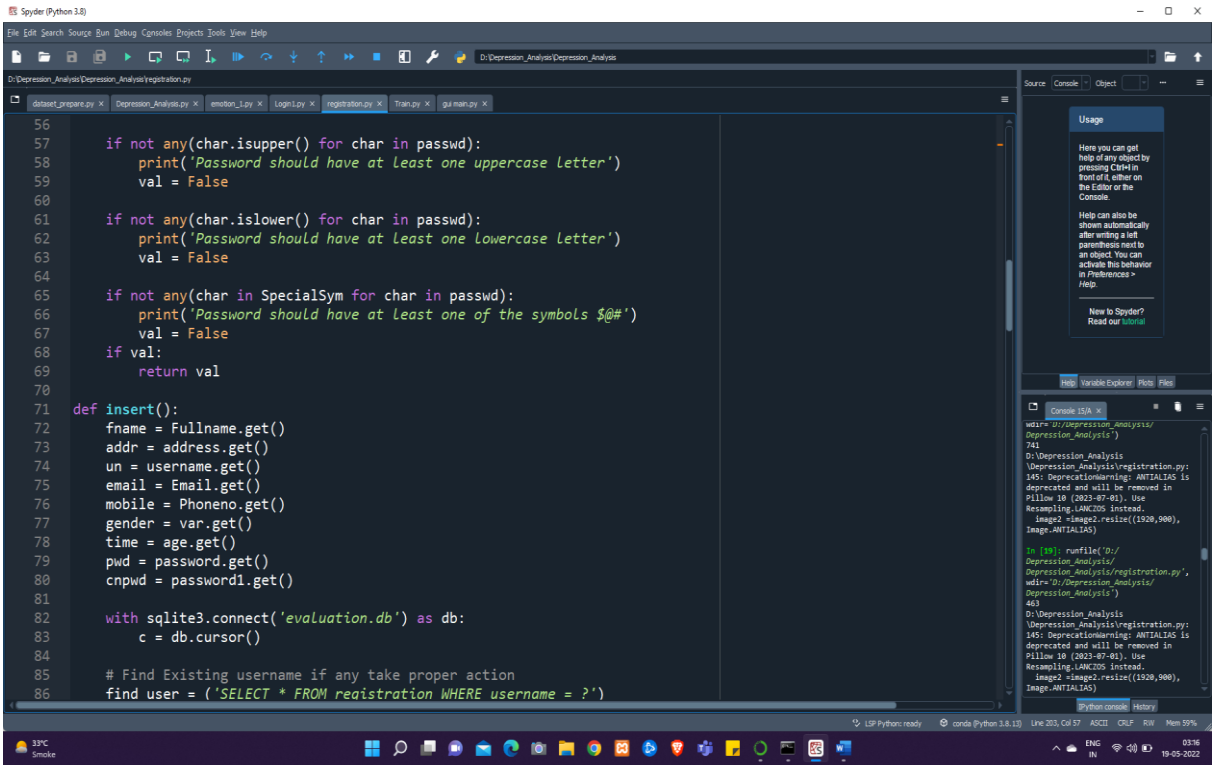
```
        # Create New Account
        insert = 'INSERT INTO user(username,password) VALUES(?,?)'
        c.execute(insert, [(self.n_username.get()), (self.n_password.get())])
        db.commit()

        # Frame Packing Methords

    def registration(self):
        root.destroy()
        from subprocess import call
        call(["python", "registration.py"])

        # mainloop(root)

    def log(self):
        self.username.set('')
        self.password.set('')
        self.crf.pack_forget()
        self.head['text'] = 'LOGIN'
        self.logf.pack()




root = Tk()
obj = login_system(root)
root.mainloop()
```



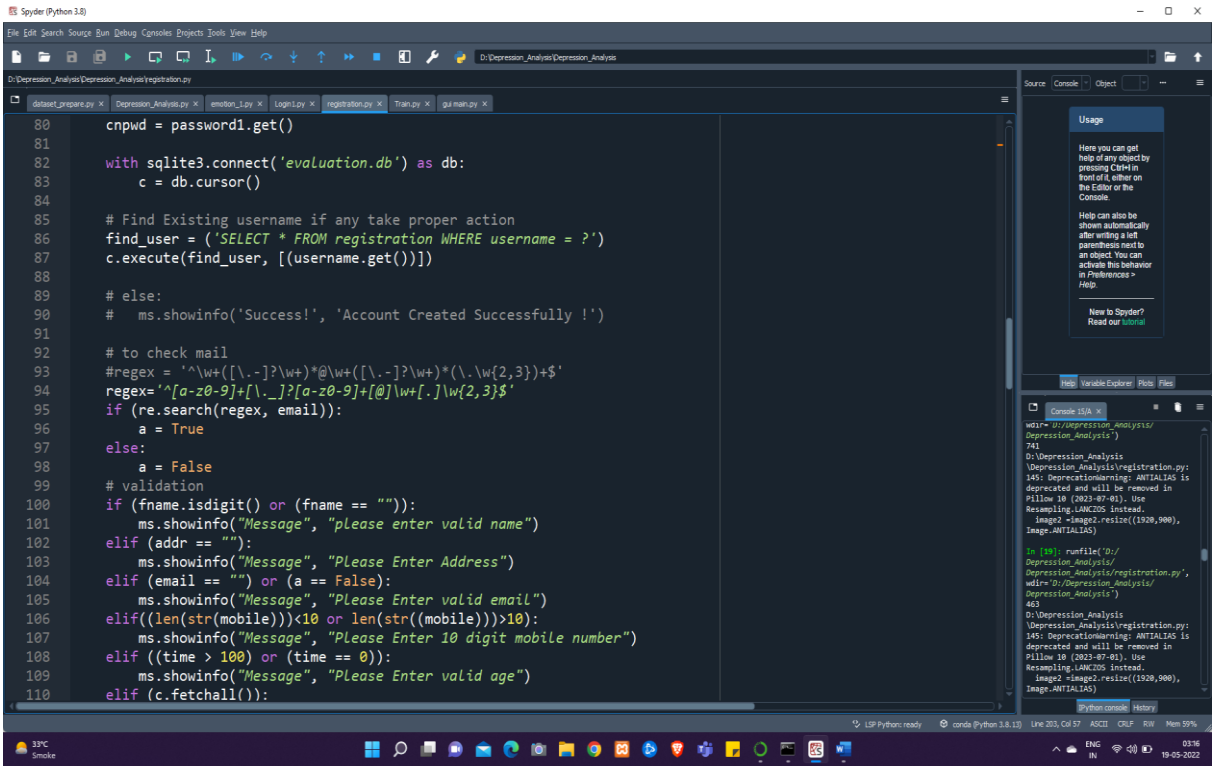
```
import tkinter as tk
# from tkinter import *
from tkinter import messagebox as ms
import sqlite3
from PIL import Image, ImageTk
import re
import random
import os
#import cv2


window = tk.Tk()
w,h = window.winfo_screenwidth(),window.winfo_screenheight()
window.geometry("%dx%d+0+0"%(w,h))
window.title("REGISTRATION FORM")
window.configure(background="moccasin")

Fullname = tk.StringVar()
address = tk.StringVar()
username = tk.StringVar()
Email = tk.StringVar()
Phoneno = tk.IntVar()
var = tk.IntVar()
age = tk.IntVar()
password = tk.StringVar()
password1 = tk.StringVar()

value = random.randint(1, 1000)
print(value)

# database code
```



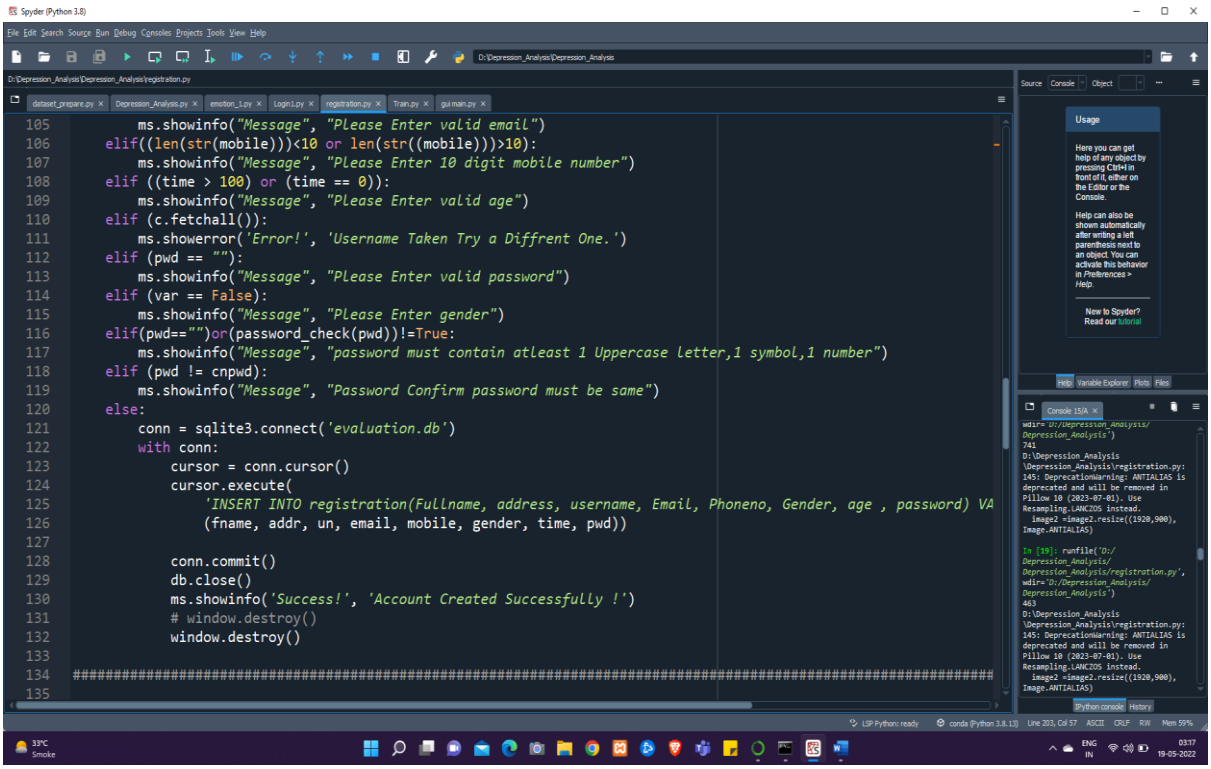
```
value = random.randint(1, 1000)
print(value)

# database code
db = sqlite3.connect('evaluation.db')
cursor = db.cursor()
cursor.execute("CREATE TABLE IF NOT EXISTS registration"
               "(Fullname TEXT, address TEXT, username TEXT, Email TEXT, Phoneno TEXT,Gender TEXT,age TEXT , pass
db.commit()



def password_check(passwd):

    SpecialSym =['$', '@', '#', '%']
    val = True

    if len(passwd) < 6:
        print('Length should be at least 6')
        val = False

    if len(passwd) > 20:
        print('Length should be not be greater than 8')
        val = False

    if not any(char.isdigit() for char in passwd):
        print('Password should have at least one numeral')
        val = False

    if not any(char.isupper() for char in passwd):
        print('Password should have at least one uppercase letter')
```



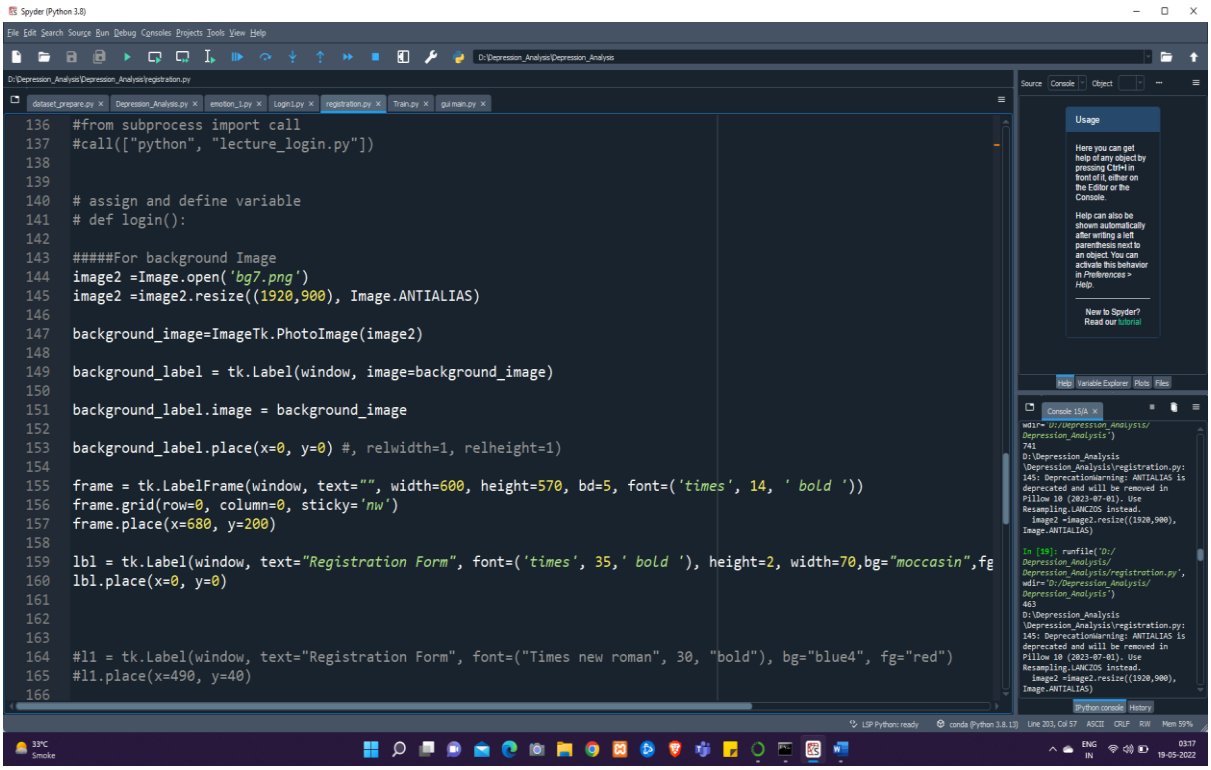
```
    if not any(char.isupper() for char in passwd):
        print('Password should have at least one uppercase letter')
        val = False

    if not any(char.islower() for char in passwd):
        print('Password should have at least one lowercase letter')
        val = False

    if not any(char in SpecialSym for char in passwd):
        print('Password should have at least one of the symbols $@#')
        val = False
    if val:
        return val

def insert():
    fname = Fullname.get()
    addr = address.get()
    un = username.get()
    email = Email.get()
    mobile = Phoneno.get()
    gender = var.get()
    time = age.get()
    pwd = password.get()
    cnpwd = password1.get()

    with sqlite3.connect('evaluation.db') as db:
        c = db.cursor()

    # Find Existing username if any take proper action
    find user = ('SELECT * FROM registration WHERE username = ?')
```



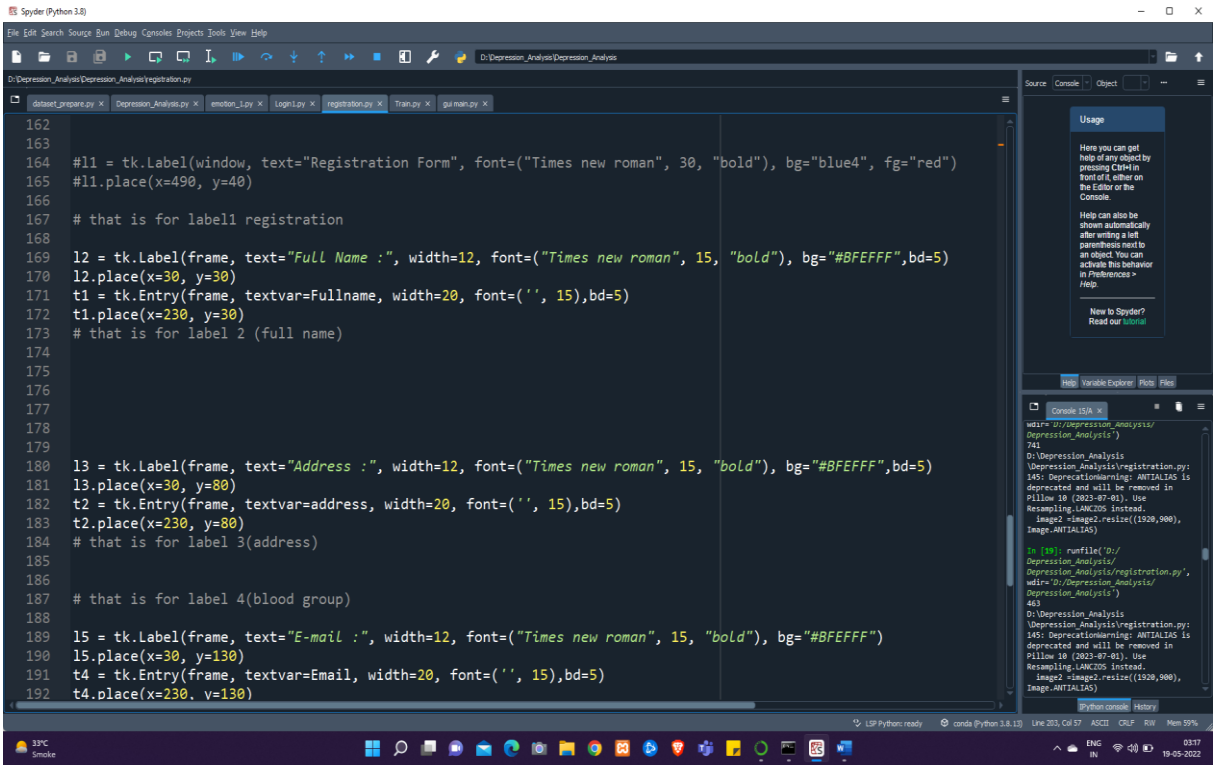
```
    cnpwd = password1.get()

    with sqlite3.connect('evaluation.db') as db:
        c = db.cursor()

    # Find Existing username if any take proper action
    find_user = ('SELECT * FROM registration WHERE username = ?')
    c.execute(find_user, [(username.get())])

    # else:
    #   ms.showinfo('Success!', 'Account Created Successfully !')

    # to check mail
    #regex = '^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$'
    regex='^[a-z0-9]+[\._]?[a-z0-9]+[@]\w+[.]\w{2,3}$'
    if (re.search(regex, email)):
        a = True
    else:
        a = False
    # validation
    if (fname.isdigit() or (fname == "")):
        ms.showinfo("Message", "please enter valid name")
    elif (addr == ""):
        ms.showinfo("Message", "PLease Enter Address")
    elif (email == "") or (a == False):
        ms.showinfo("Message", "PLease Enter valid email")
    elif((len(str(mobile)))<10 or len(str((mobile)))>10):
        ms.showinfo("Message", "PLease Enter 10 digit mobile number")
    elif ((time > 100) or (time == 0)):
        ms.showinfo("Message", "PLease Enter valid age")
    elif (c.fetchall()):
```



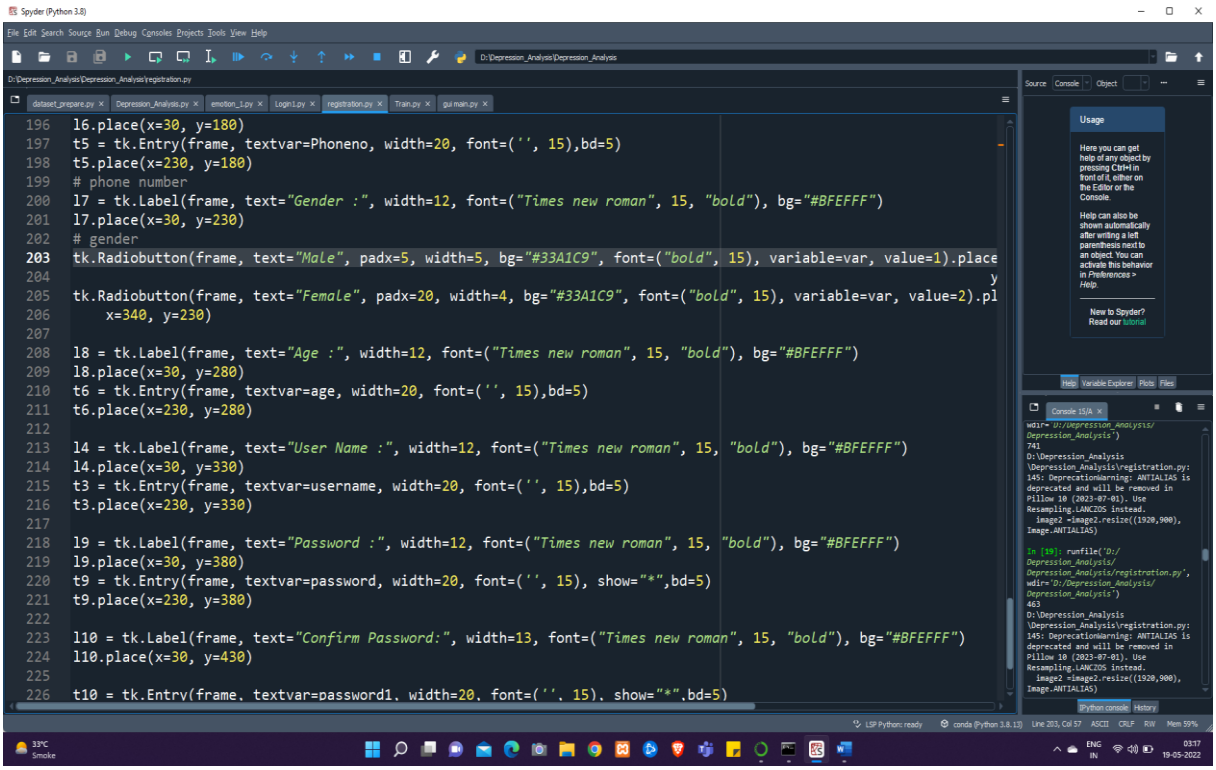
```
        ms.showinfo("Message", "Please Enter valid email")
    elif((len(str(mobile)))<10 or len(str((mobile)))>10):
        ms.showinfo("Message", "Please Enter 10 digit mobile number")
    elif ((time > 100) or (time == 0)):
        ms.showinfo("Message", "Please Enter valid age")
    elif (c.fetchall()):
        ms.showerror('Error!', 'Username Taken Try a Diffrent One.')
    elif (pwd == ""):
        ms.showinfo("Message", "Please Enter valid password")
    elif (var == False):
        ms.showinfo("Message", "Please Enter gender")
    elif(pwd==""or(password_check(pwd))!=True:
        ms.showinfo("Message", "password must contain atleast 1 Uppercase letter,1 symbol,1 number")
    elif (pwd != cnpwd):
        ms.showinfo("Message", "Password Confirm password must be same")
    else:
        conn = sqlite3.connect('evaluation.db')
        with conn:
            cursor = conn.cursor()
            cursor.execute(
                'INSERT INTO registration(Fullname, address, username, Email, Phoneno, Gender, age , password) VA
                (fname, addr, un, email, mobile, gender, time, pwd))

            conn.commit()
            db.close()
            ms.showinfo('Success!', 'Account Created Successfully !')
            # window.destroy()
            window.destroy()

###########################################################################################################################
```



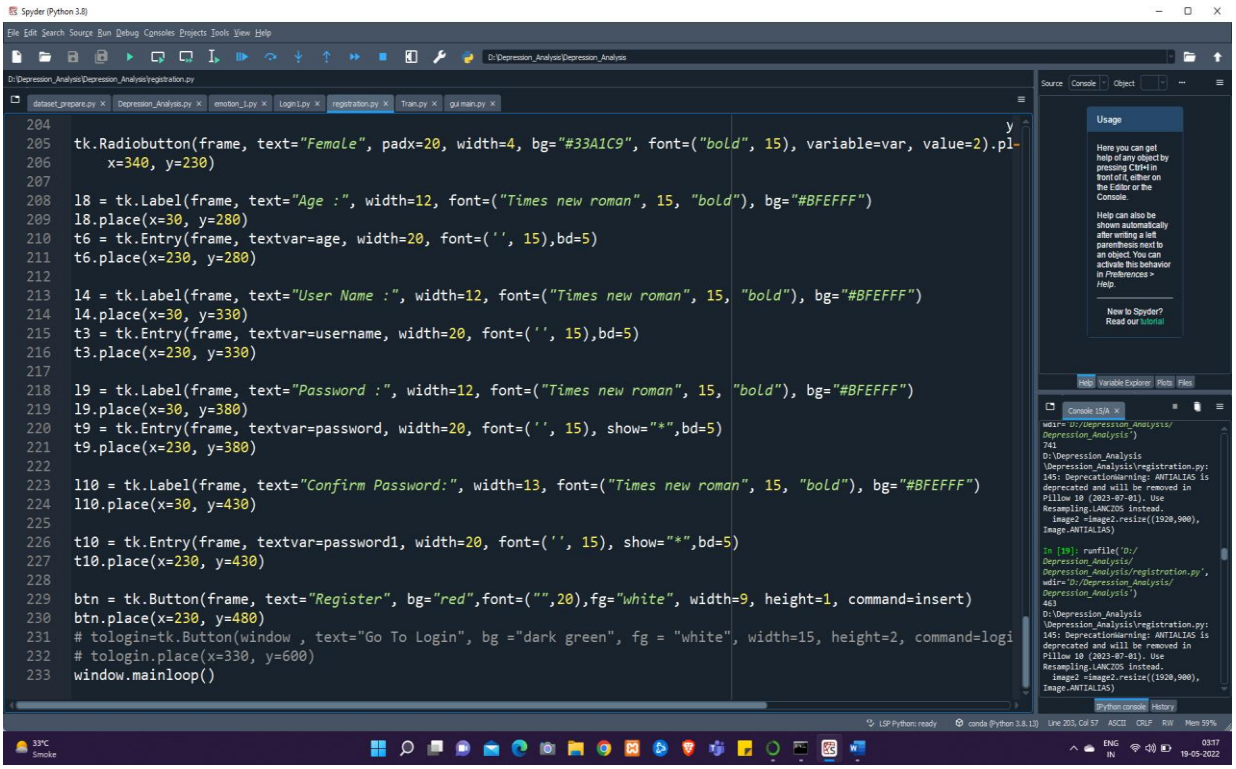
```
#from subprocess import call
#call(["python", "lecture_login.py"])


# assign and define variable
# def login():

#####For background Image
image2 =Image.open('bg7.png')
image2 =image2.resize((1920,900), Image.ANTIALIAS)

background_image=ImageTk.PhotoImage(image2)

background_label = tk.Label(window, image=background_image)

background_label.image = background_image

background_label.place(x=0, y=0) #, relwidth=1, relheight=1)

frame = tk.LabelFrame(window, text="", width=600, height=570, bd=5, font=('times', 14, ' bold '))
frame.grid(row=0, column=0, sticky='nw')
frame.place(x=680, y=200)

lbl = tk.Label(window, text="Registration Form", font=('times', 35,' bold '), height=2, width=70,bg="moccasin",fg
lbl.place(x=0, y=0)



#l1 = tk.Label(window, text="Registration Form", font=("Times new roman", 30, "bold"), bg="blue4", fg="red")
#l1.place(x=490, y=40)
```



```
#l1 = tk.Label(window, text="Registration Form", font=("Times new roman", 30, "bold"), bg="blue4", fg="red")
#l1.place(x=490, y=40)

# that is for label1 registration

l2 = tk.Label(frame, text="Full Name :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF",bd=5)
l2.place(x=30, y=30)
t1 = tk.Entry(frame, textvar=Fullname, width=20, font=('', 15),bd=5)
t1.place(x=230, y=30)
# that is for label 2 (full name)




l3 = tk.Label(frame, text="Address :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF",bd=5)
l3.place(x=30, y=80)
t2 = tk.Entry(frame, textvar=address, width=20, font=('', 15),bd=5)
t2.place(x=230, y=80)
# that is for label 3(address)


# that is for label 4(blood group)

l5 = tk.Label(frame, text="E-mail :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l5.place(x=30, y=130)
t4 = tk.Entry(frame, textvar=Email, width=20, font=('', 15),bd=5)
t4.place(x=230, y=130)
```



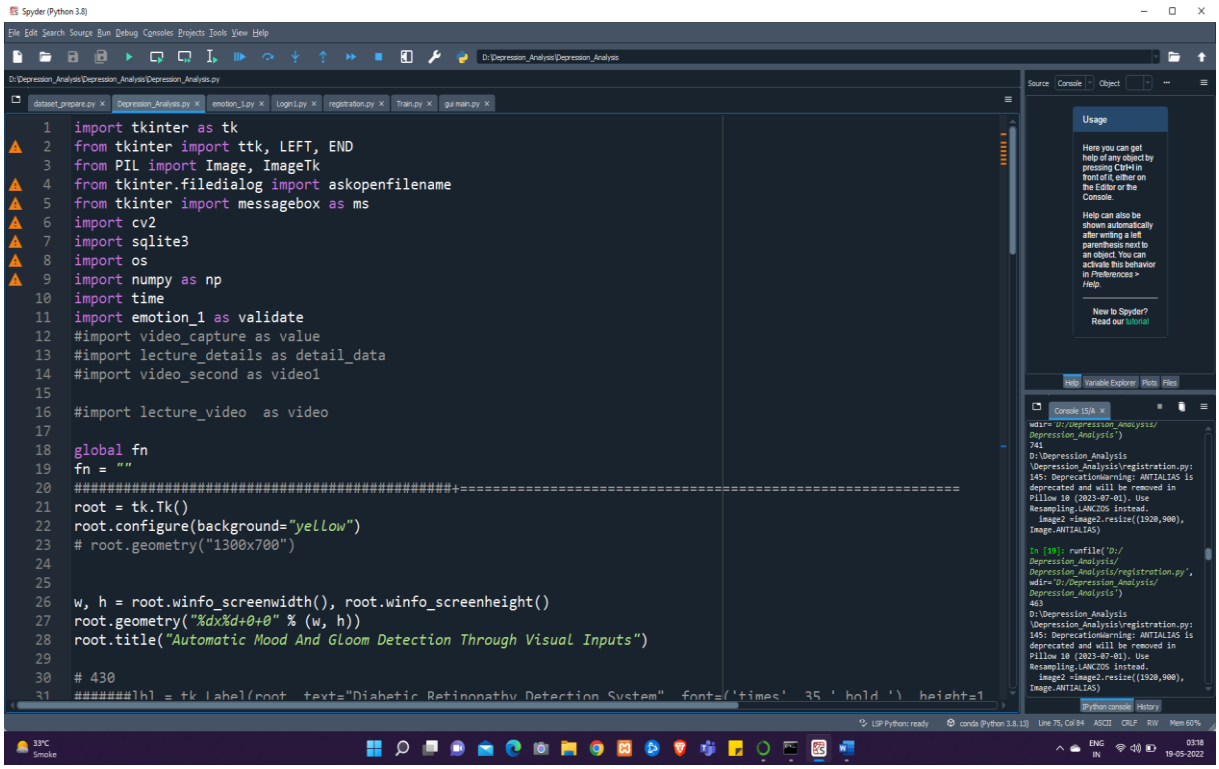
```
l6.place(x=30, y=180)
t5 = tk.Entry(frame, textvar=Phoneno, width=20, font=('', 15),bd=5)
t5.place(x=230, y=180)
# phone number
l7 = tk.Label(frame, text="Gender :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l7.place(x=30, y=230)
# gender
tk.Radiobutton(frame, text="Male", padx=5, width=5, bg="#33A1C9", font=("bold", 15), variable=var, value=1).place
                                                                                                              y
tk.Radiobutton(frame, text="Female", padx=20, width=4, bg="#33A1C9", font=("bold", 15), variable=var, value=2).pl
    x=340, y=230)

l8 = tk.Label(frame, text="Age :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l8.place(x=30, y=280)
t6 = tk.Entry(frame, textvar=age, width=20, font=('', 15),bd=5)
t6.place(x=230, y=280)

l4 = tk.Label(frame, text="User Name :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l4.place(x=30, y=330)
t3 = tk.Entry(frame, textvar=username, width=20, font=('', 15),bd=5)
t3.place(x=230, y=330)

l9 = tk.Label(frame, text="Password :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l9.place(x=30, y=380)
t9 = tk.Entry(frame, textvar=password, width=20, font=('', 15), show="*",bd=5)
t9.place(x=230, y=380)

l10 = tk.Label(frame, text="Confirm Password:", width=13, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l10.place(x=30, y=430)

t10 = tk.Entry(frame, textvar=password1, width=20, font=('', 15), show="*",bd=5)
```



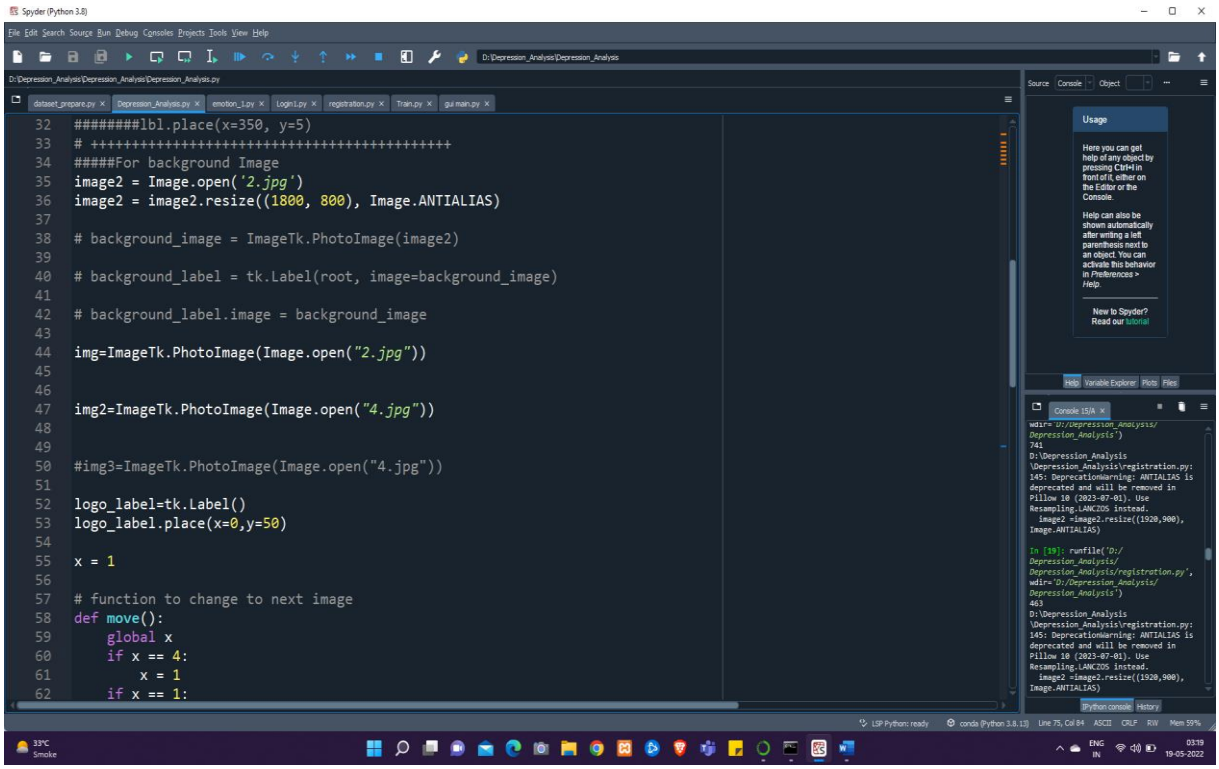
```
tk.Radiobutton(frame, text="Female", padx=20, width=4, bg="#33A1C9", font=("bold", 15), variable=var, value=2).pl-
    x=340, y=230)

l8 = tk.Label(frame, text="Age :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l8.place(x=30, y=280)
t6 = tk.Entry(frame, textvar=age, width=20, font=('', 15),bd=5)
t6.place(x=230, y=280)

l4 = tk.Label(frame, text="User Name :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l4.place(x=30, y=330)
t3 = tk.Entry(frame, textvar=username, width=20, font=('', 15),bd=5)
t3.place(x=230, y=330)

l9 = tk.Label(frame, text="Password :", width=12, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l9.place(x=30, y=380)
t9 = tk.Entry(frame, textvar=password, width=20, font=('', 15), show="*",bd=5)
t9.place(x=230, y=380)

l10 = tk.Label(frame, text="Confirm Password:", width=13, font=("Times new roman", 15, "bold"), bg="#BFEFFF")
l10.place(x=30, y=430)

t10 = tk.Entry(frame, textvar=password1, width=20, font=('', 15), show="*",bd=5)
t10.place(x=230, y=430)

btn = tk.Button(frame, text="Register", bg="red",font=("",20),fg="white", width=9, height=1, command=insert)
btn.place(x=230, y=480)
# tologin=tk.Button(window , text="Go To Login", bg ="dark green", fg = "white", width=15, height=2, command=logi
# tologin.place(x=330, y=600)
window.mainloop()
```

The Above Code is used to create GUI of the application and used to login and register them self-user will enter the data in the given field and the data will be saved in the database once the data is saved in the database the user can anytime login

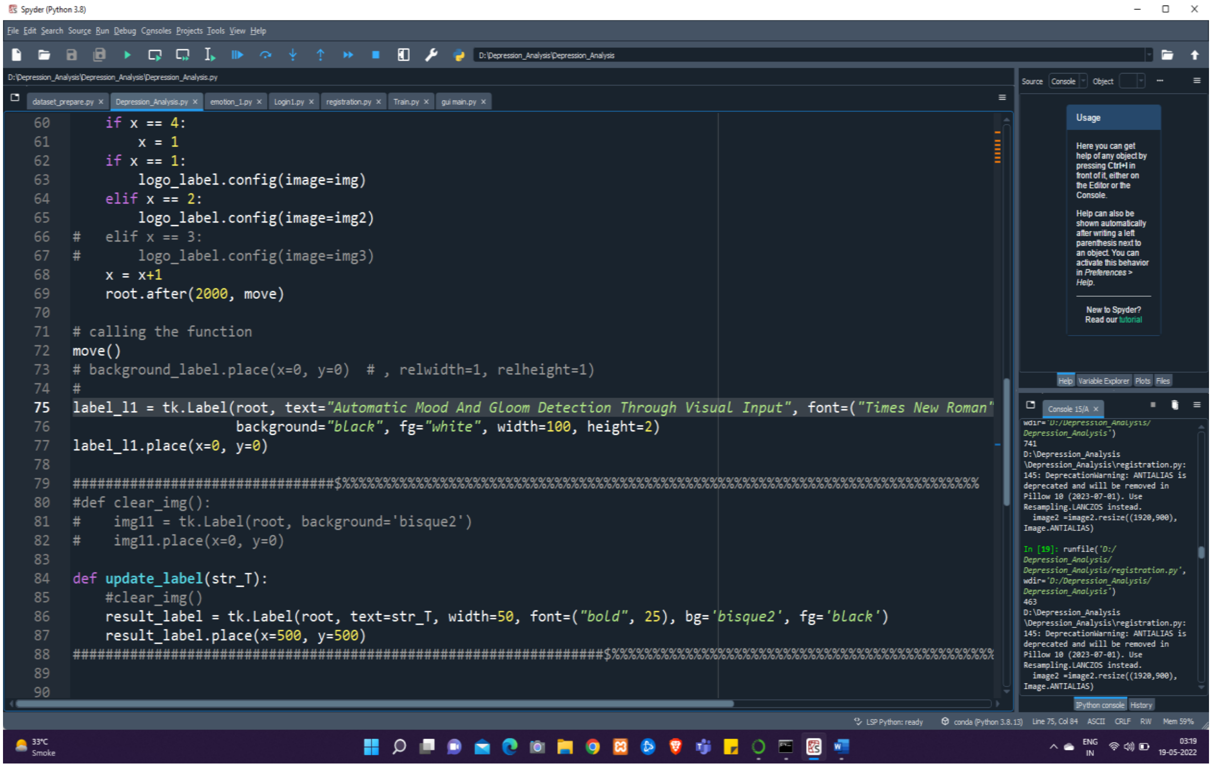
Code Snippet 3:



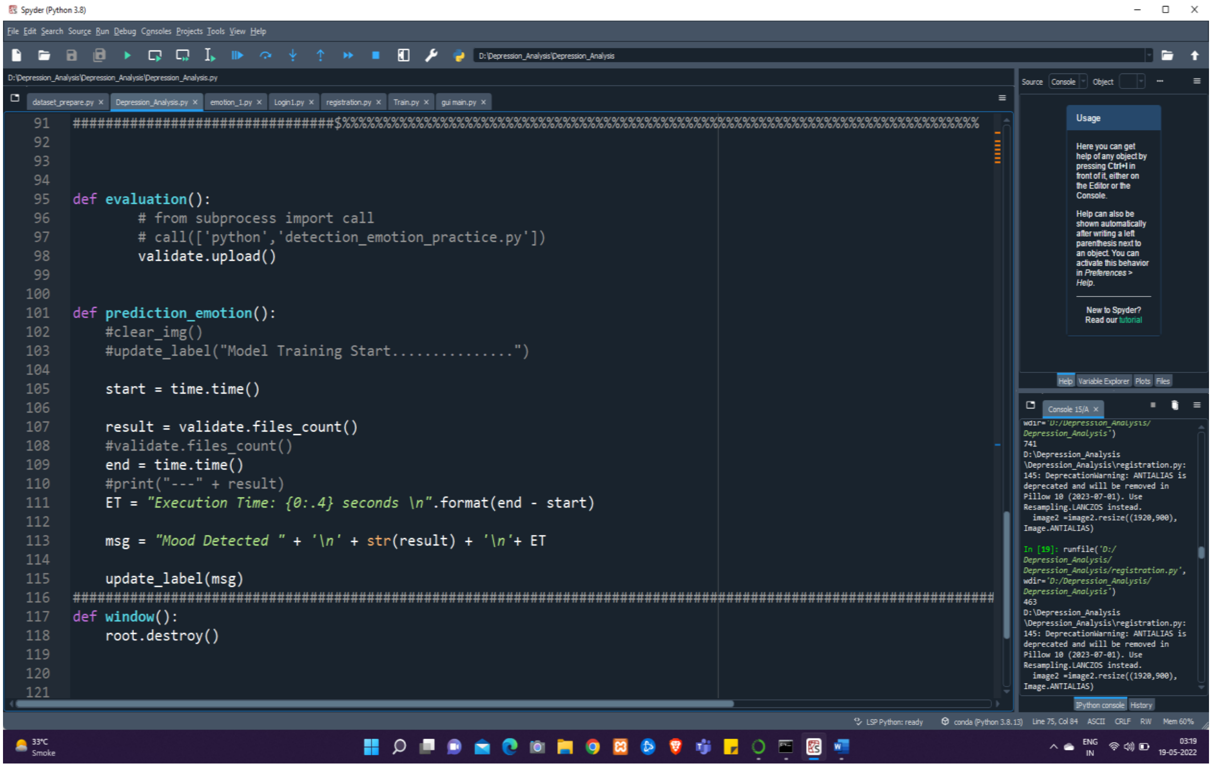
```
import tkinter as tk
from tkinter import ttk, LEFT, END
from PIL import Image, ImageTk
from tkinter.filedialog import askopenfilename
from tkinter import messagebox as ms
import cv2
import sqlite3
import os
import numpy as np
import time
import emotion_1 as validate
#import video_capture as value
#import lecture_details as detail_data
#import video_second as video1

#import lecture_video  as video

global fn
fn = ""
##########################################+=============================================================
root = tk.Tk()
root.configure(background="yellow")
# root.geometry("1300x700")


w, h = root.winfo_screenwidth(), root.winfo_screenheight()
root.geometry("%dx%d+0+0" % (w, h))
root.title("Automatic Mood And Gloom Detection Through Visual Inputs")

# 430
#######lbl = tk.Label(root, text="Diabetic Retinopathy Detection System", font=('times', 35, 'bold'), height=1
```



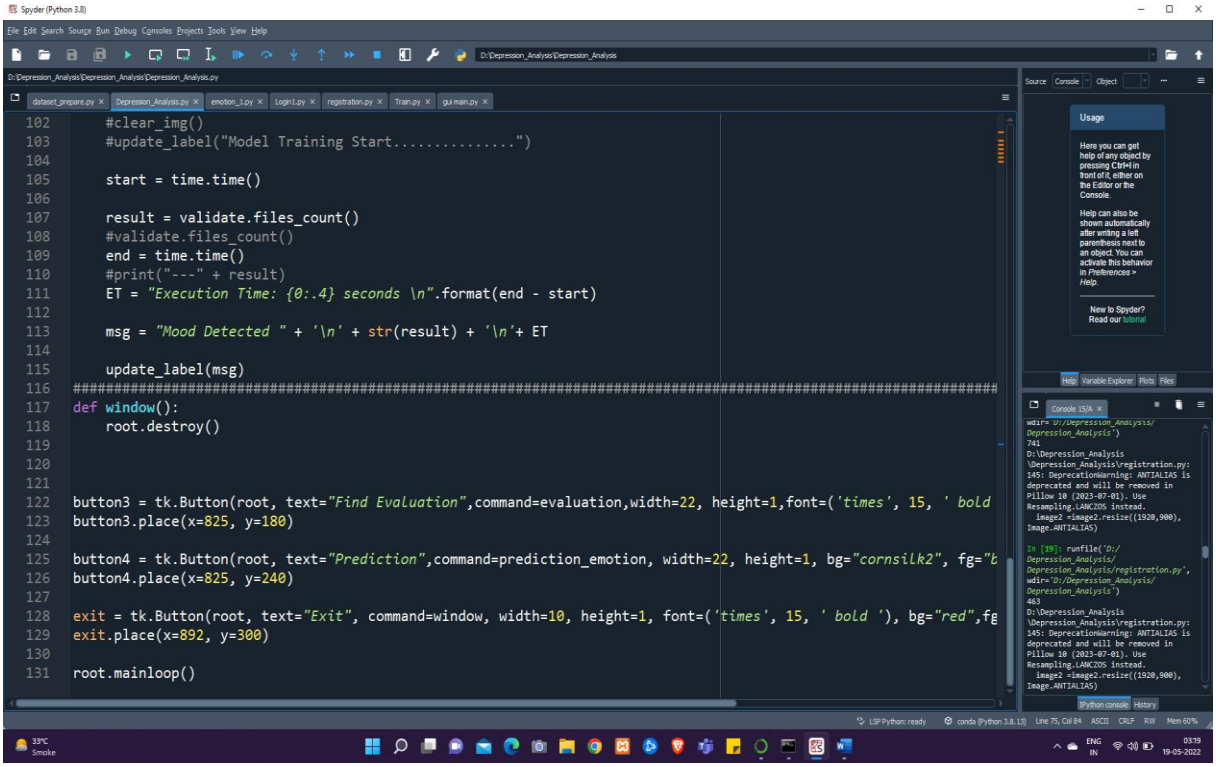
```
########lbl.place(x=350, y=5)
# ++++++++++++++++++++++++++++++++++++++++++++
#####For background Image
image2 = Image.open('2.jpg')
image2 = image2.resize((1800, 800), Image.ANTIALIAS)

# background_image = ImageTk.PhotoImage(image2)

# background_label = tk.Label(root, image=background_image)

# background_label.image = background_image

img=ImageTk.PhotoImage(Image.open("2.jpg"))


img2=ImageTk.PhotoImage(Image.open("4.jpg"))


#img3=ImageTk.PhotoImage(Image.open("4.jpg"))

logo_label=tk.Label()
logo_label.place(x=0,y=50)

x = 1

# function to change to next image
def move():
    global x
    if x == 4:
        x = 1
    if x == 1:
```



```
    if x == 4:
        x = 1
    if x == 1:
        logo_label.config(image=img)
    elif x == 2:
        logo_label.config(image=img2)
#   elif x == 3:
#       logo_label.config(image=img3)
    x = x+1
    root.after(2000, move)

# calling the function
move()
# background_label.place(x=0, y=0)  # , relwidth=1, relheight=1)
#
label_l1 = tk.Label(root, text="Automatic Mood And Gloom Detection Through Visual Input", font=("Times New Roman"
                    background="black", fg="white", width=100, height=2)
label_l1.place(x=0, y=0)

################################$%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
#def clear_img():
#    img11 = tk.Label(root, background='bisque2')
#    img11.place(x=0, y=0)

def update_label(str_T):
    #clear_img()
    result_label = tk.Label(root, text=str_T, width=50, font=("bold", 25), bg='bisque2', fg='black')
    result_label.place(x=500, y=500)
################################################################$%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```



```
################################$%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



def evaluation():
        # from subprocess import call
        # call(['python','detection_emotion_practice.py'])
        validate.upload()

def prediction_emotion():
    #clear_img()
    #update_label("Model Training Start..............")

    start = time.time()

    result = validate.files_count()
    #validate.files_count()
    end = time.time()
    #print("---" + result)
    ET = "Execution Time: {0:.4} seconds \n".format(end - start)

    msg = "Mood Detected " + '\n' + str(result) + '\n'+ ET

    update_label(msg)
###############################################################################################################
def window():
    root.destroy()
```

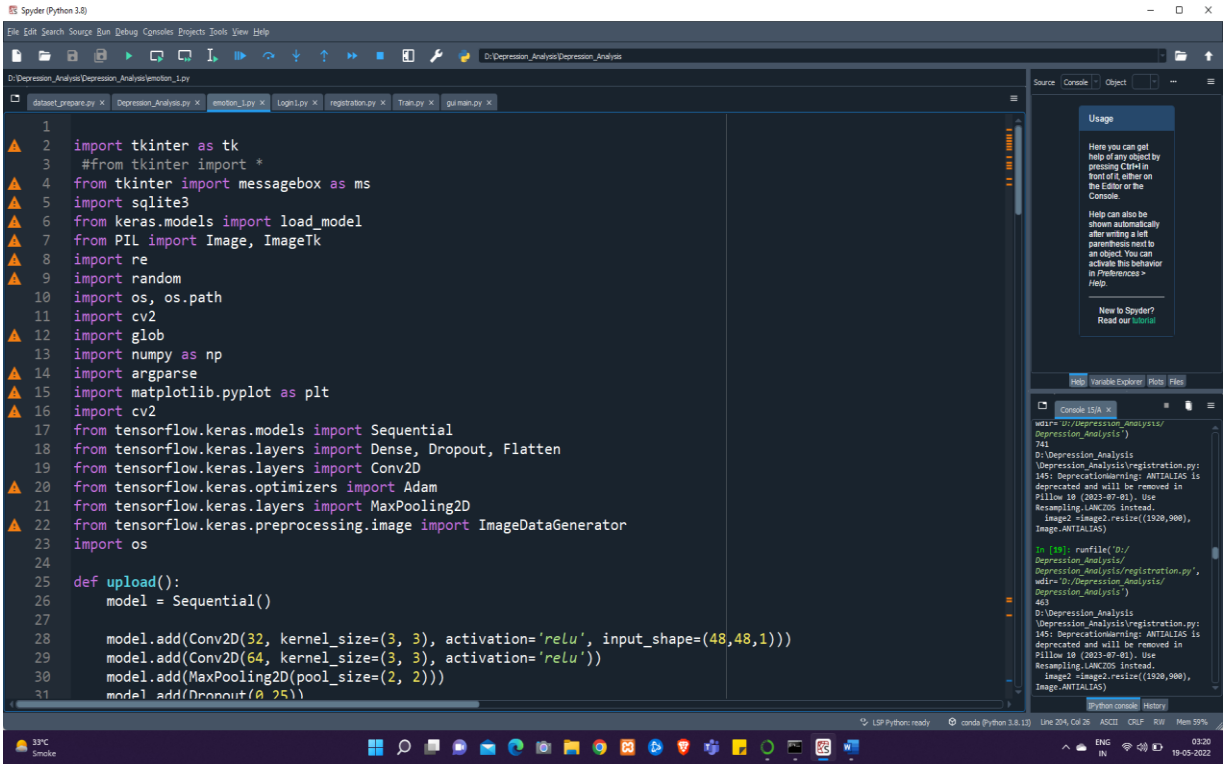


```
    #clear_img()
    #update_label("Model Training Start.............")

    start = time.time()

    result = validate.files_count()
    #validate.files_count()
    end = time.time()
    #print("---" + result)
    ET = "Execution Time: {0:.4} seconds \n".format(end - start)

    msg = "Mood Detected " + '\n' + str(result) + '\n'+ ET

    update_label(msg)
##################################################################################################################
def window():
    root.destroy()



button3 = tk.Button(root, text="Find Evaluation",command=evaluation,width=22, height=1,font=('times', 15, ' bold
button3.place(x=825, y=180)

button4 = tk.Button(root, text="Prediction",command=prediction_emotion, width=22, height=1, bg="cornsilk2", fg="b
button4.place(x=825, y=240)

exit = tk.Button(root, text="Exit", command=window, width=10, height=1, font=('times', 15, ' bold '), bg="red",fg
exit.place(x=892, y=300)

root.mainloop()
```

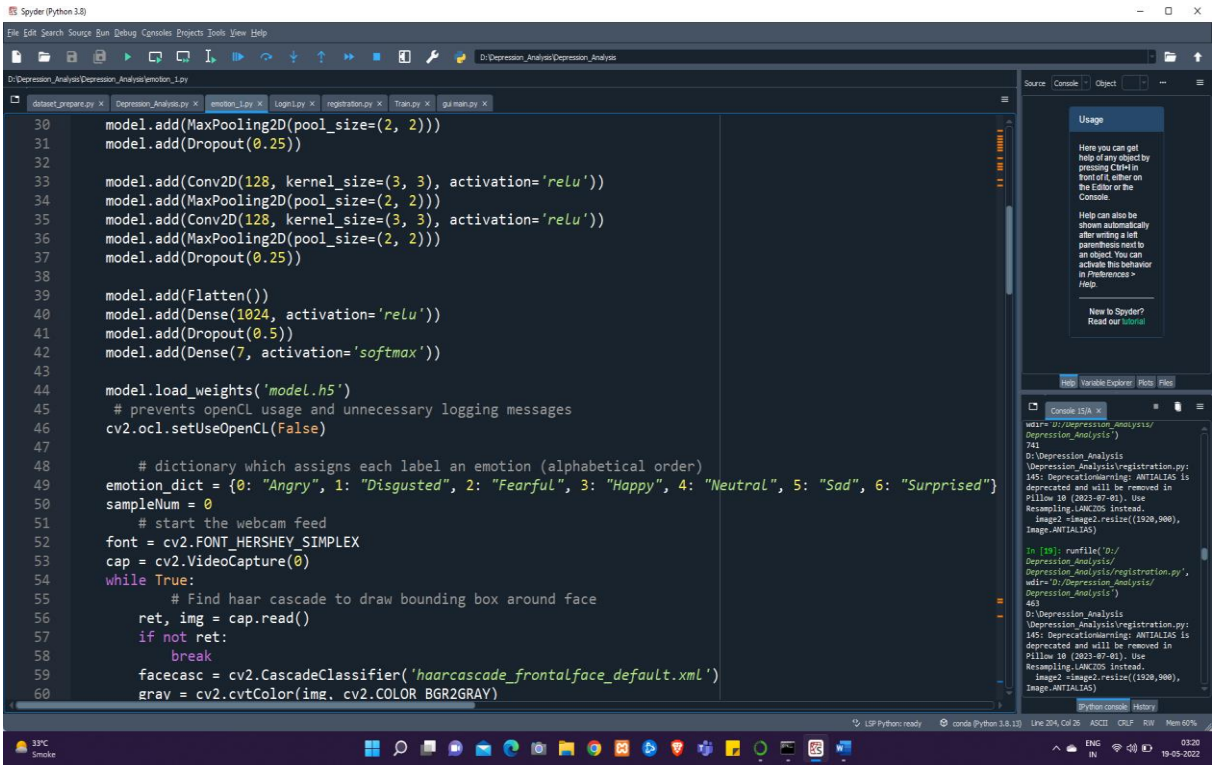
This code is used to form the GUI of the after the login it consist of three buttons

1. Find Evaluation
2. Prediction
3. Exit

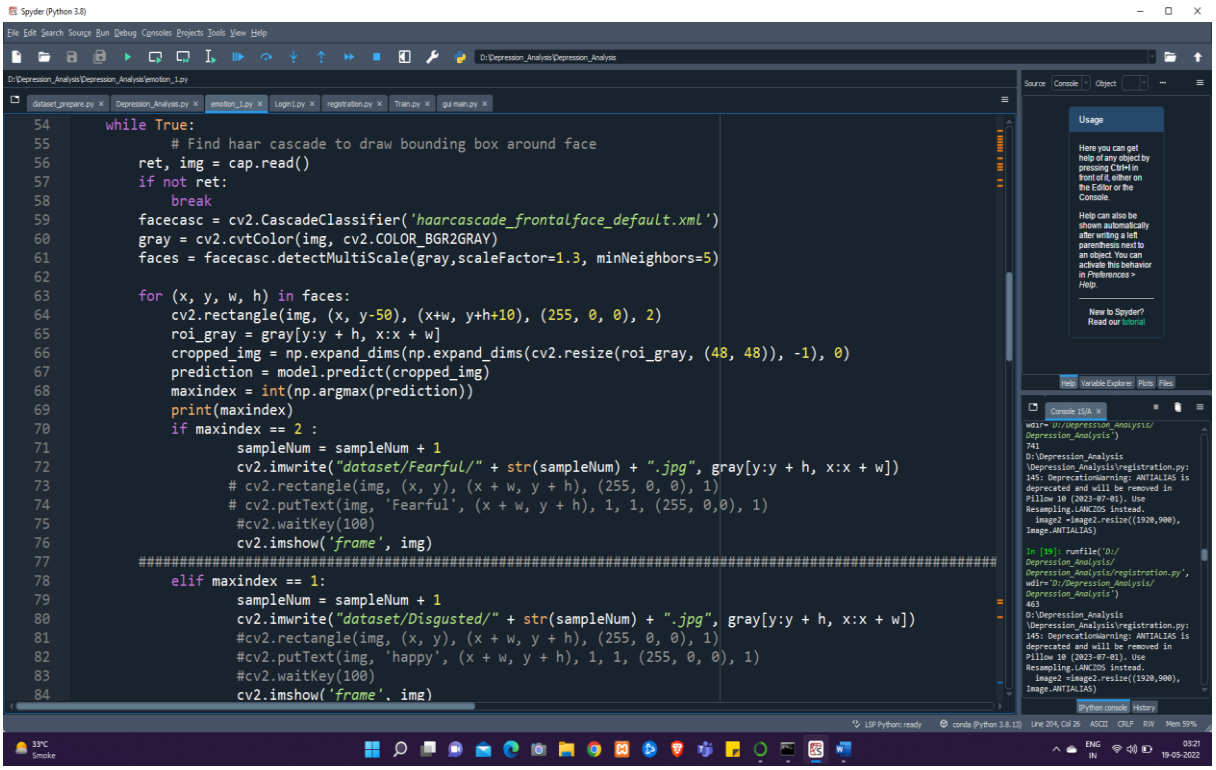
Code Snippet 4:



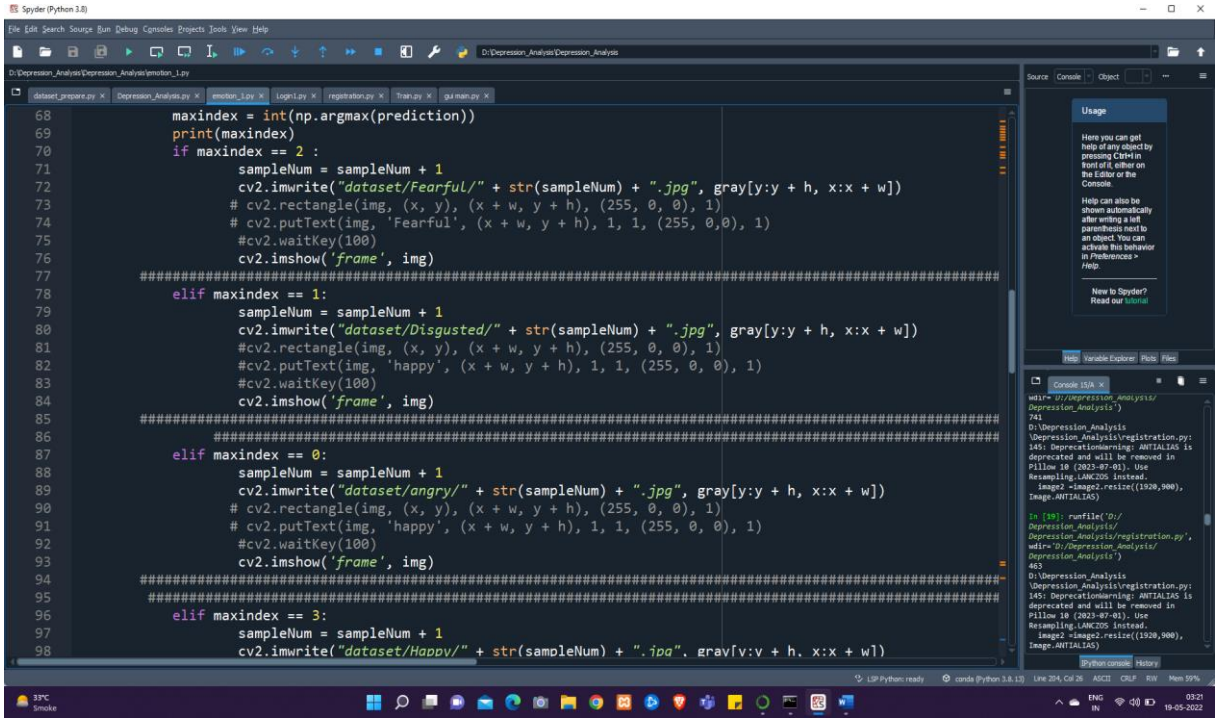
```
import tkinter as tk
 #from tkinter import *
from tkinter import messagebox as ms
import sqlite3
from keras.models import load_model
from PIL import Image, ImageTk
import re
import random
import os, os.path
import cv2
import glob
import numpy as np
import argparse
import matplotlib.pyplot as plt
import cv2
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import os

def upload():
    model = Sequential()

    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48,48,1)))
    model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Dropout(0.25))
```



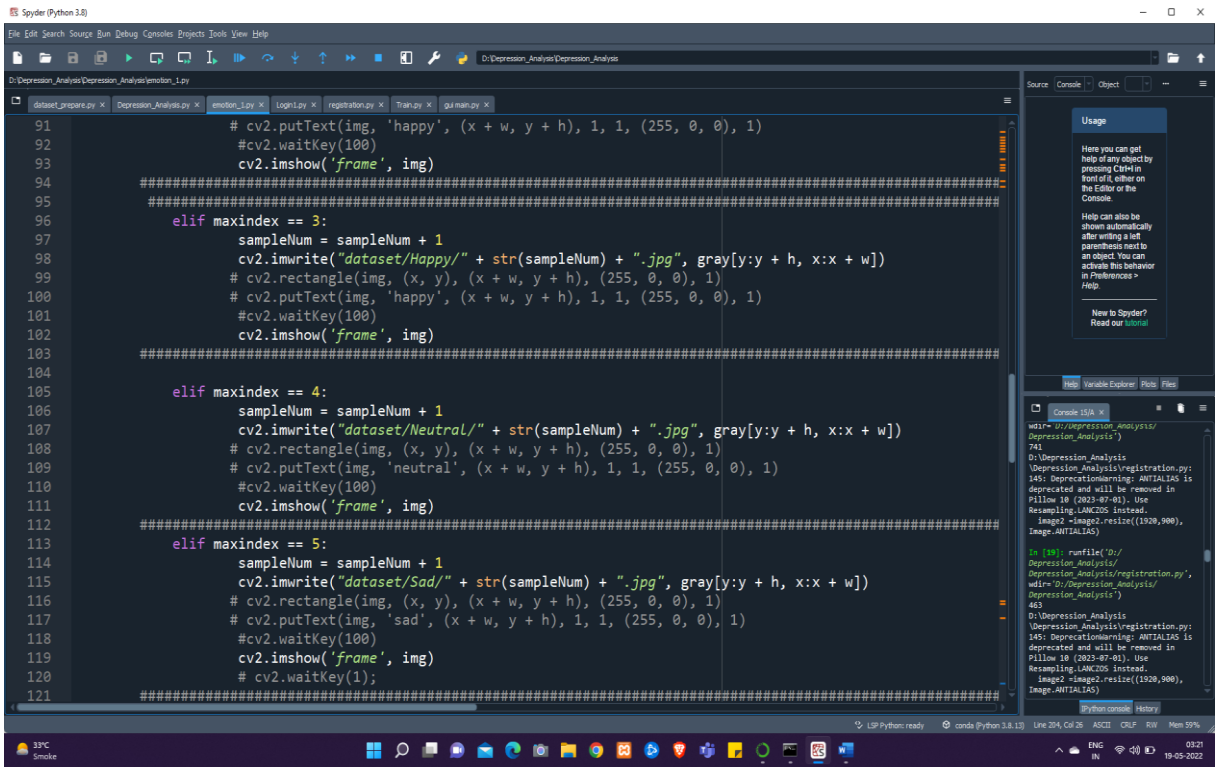
```
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(1024, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(7, activation='softmax'))

model.load_weights('model.h5')
 # prevents openCL usage and unnecessary logging messages
cv2.ocl.setUseOpenCL(False)

    # dictionary which assigns each label an emotion (alphabetical order)
emotion_dict = {0: "Angry", 1: "Disgusted", 2: "Fearful", 3: "Happy", 4: "Neutral", 5: "Sad", 6: "Surprised"}
sampleNum = 0
    # start the webcam feed
font = cv2.FONT_HERSHEY_SIMPLEX
cap = cv2.VideoCapture(0)
while True:
        # Find haar cascade to draw bounding box around face
    ret, img = cap.read()
    if not ret:
        break
    facecasc = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```



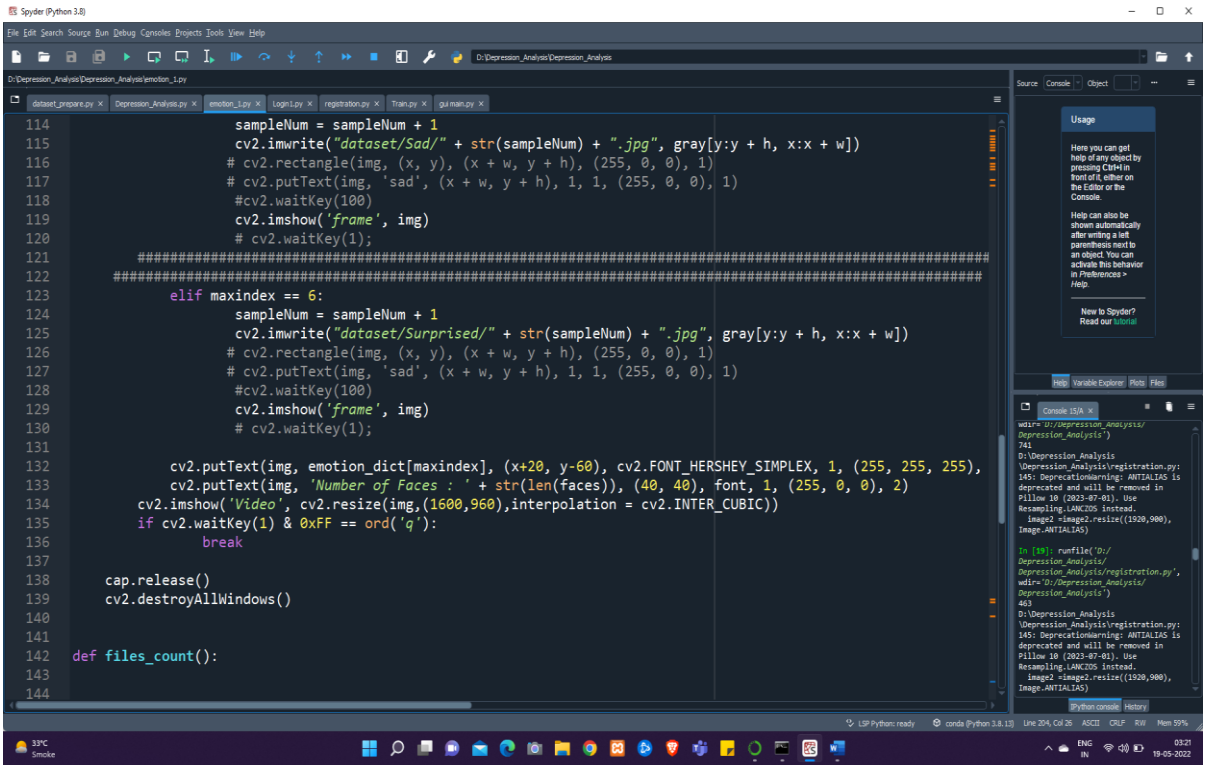
```
while True:
        # Find haar cascade to draw bounding box around face
    ret, img = cap.read()
    if not ret:
        break
    facecasc = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = facecasc.detectMultiScale(gray,scaleFactor=1.3, minNeighbors=5)

    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y-50), (x+w, y+h+10), (255, 0, 0), 2)
        roi_gray = gray[y:y + h, x:x + w]
        cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray, (48, 48)), -1), 0)
        prediction = model.predict(cropped_img)
        maxindex = int(np.argmax(prediction))
        print(maxindex)
        if maxindex == 2 :
                sampleNum = sampleNum + 1
                cv2.imwrite("dataset/Fearful/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
               # cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
               # cv2.putText(img, 'Fearful', (x + w, y + h), 1, 1, (255, 0,0), 1)
                #cv2.waitKey(100)
                cv2.imshow('frame', img)
        ##########################################################################################################
        elif maxindex == 1:
                sampleNum = sampleNum + 1
                cv2.imwrite("dataset/Disgusted/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                #cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                #cv2.putText(img, 'happy', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                #cv2.waitKey(100)
                cv2.imshow('frame', img)
```



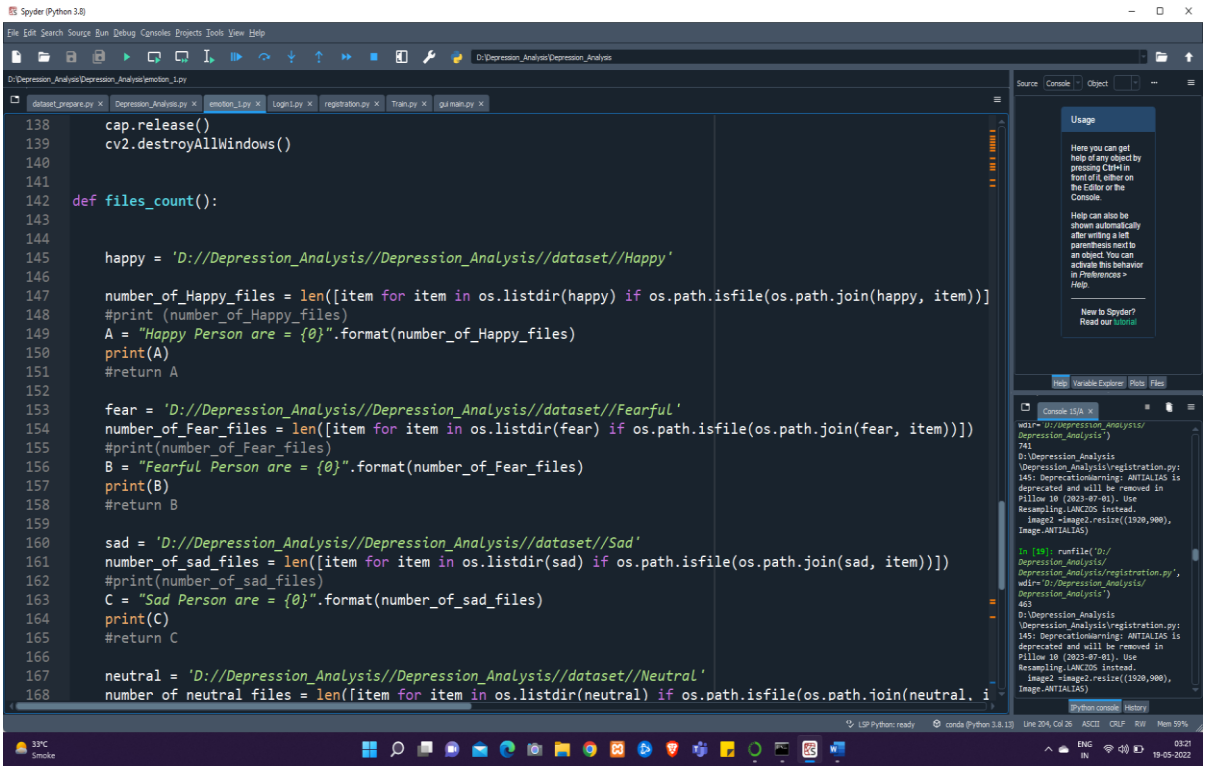
```
            maxindex = int(np.argmax(prediction))
            print(maxindex)
            if maxindex == 2 :
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/Fearful/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                   # cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                   # cv2.putText(img, 'Fearful', (x + w, y + h), 1, 1, (255, 0,0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
        ##########################################################################################################
            elif maxindex == 1:
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/Disgusted/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                    #cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                    #cv2.putText(img, 'happy', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
        ##########################################################################################################
                  ##########################################################################################################
            elif maxindex == 0:
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/angry/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                   # cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                   # cv2.putText(img, 'happy', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
        ##########################################################################################################
         ##########################################################################################################
            elif maxindex == 3:
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/Happy/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
```



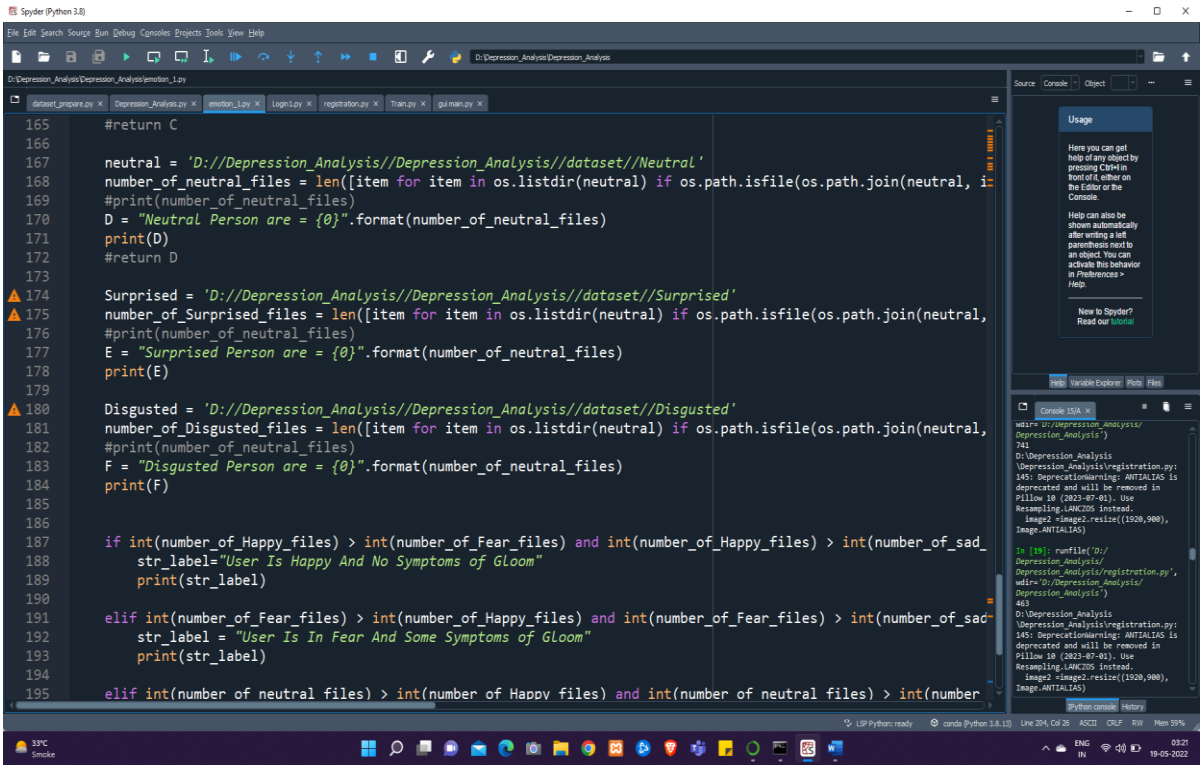
```
                   # cv2.putText(img, 'happy', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
        ##########################################################################################################
         ##########################################################################################################
            elif maxindex == 3:
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/Happy/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                   # cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                   # cv2.putText(img, 'happy', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
        ##########################################################################################################

            elif maxindex == 4:
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/Neutral/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                   # cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                   # cv2.putText(img, 'neutral', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
        ##########################################################################################################
            elif maxindex == 5:
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/Sad/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                   # cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                   # cv2.putText(img, 'sad', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
                    # cv2.waitKey(1);
        ##########################################################################################################
```



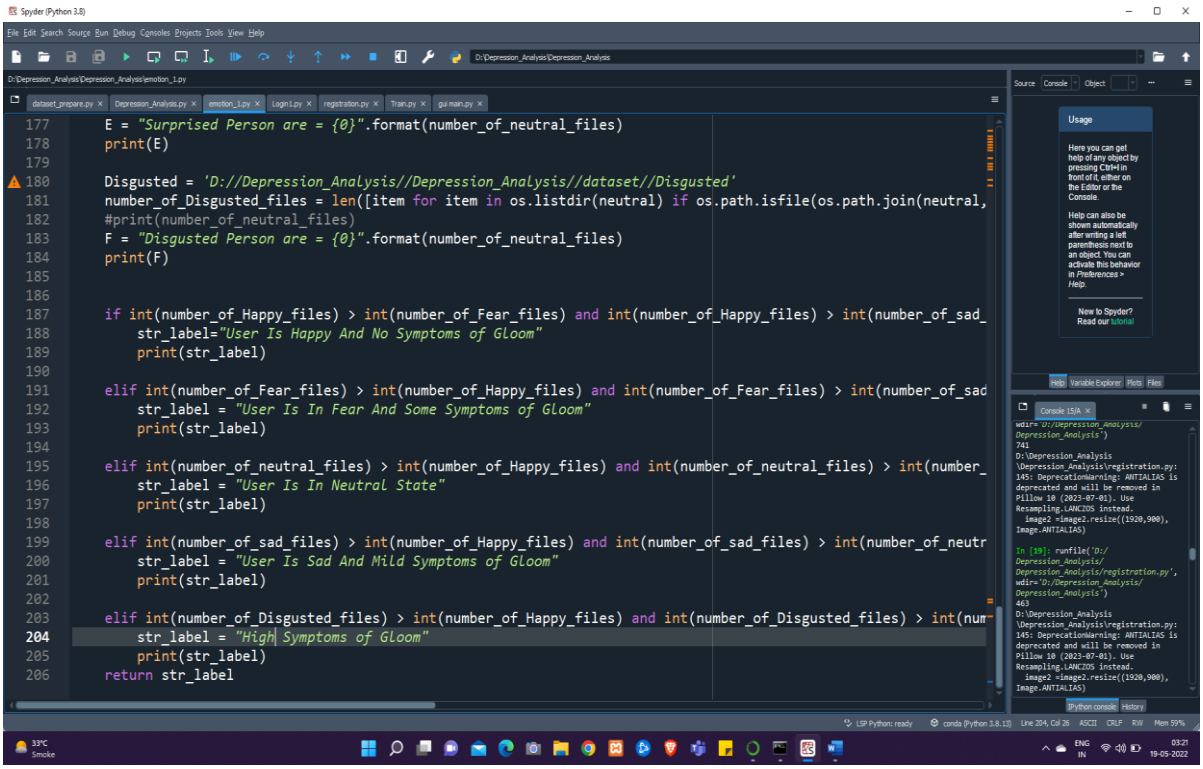
```
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/Sad/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                   # cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                   # cv2.putText(img, 'sad', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
                    # cv2.waitKey(1);
        ##########################################################################################################
      ##########################################################################################################
            elif maxindex == 6:
                    sampleNum = sampleNum + 1
                    cv2.imwrite("dataset/Surprised/" + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                   # cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 1)
                   # cv2.putText(img, 'sad', (x + w, y + h), 1, 1, (255, 0, 0), 1)
                    #cv2.waitKey(100)
                    cv2.imshow('frame', img)
                    # cv2.waitKey(1);

            cv2.putText(img, emotion_dict[maxindex], (x+20, y-60), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255),
            cv2.putText(img, 'Number of Faces : ' + str(len(faces)), (40, 40), font, 1, (255, 0, 0), 2)
        cv2.imshow('Video', cv2.resize(img,(1600,960),interpolation = cv2.INTER_CUBIC))
        if cv2.waitKey(1) & 0xFF == ord('q'):
                break

    cap.release()
    cv2.destroyAllWindows()


def files_count():
```



```
    cap.release()
    cv2.destroyAllWindows()


def files_count():


    happy = 'D://Depression_Analysis//Depression_Analysis//dataset//Happy'

    number_of_Happy_files = len([item for item in os.listdir(happy) if os.path.isfile(os.path.join(happy, item))])
    #print (number_of_Happy_files)
    A = "Happy Person are = {0}".format(number_of_Happy_files)
    print(A)
    #return A

    fear = 'D://Depression_Analysis//Depression_Analysis//dataset//Fearful'
    number_of_Fear_files = len([item for item in os.listdir(fear) if os.path.isfile(os.path.join(fear, item))])
    #print(number_of_Fear_files)
    B = "Fearful Person are = {0}".format(number_of_Fear_files)
    print(B)
    #return B

    sad = 'D://Depression_Analysis//Depression_Analysis//dataset//Sad'
    number_of_sad_files = len([item for item in os.listdir(sad) if os.path.isfile(os.path.join(sad, item))])
    #print(number_of_sad_files)
    C = "Sad Person are = {0}".format(number_of_sad_files)
    print(C)
    #return C

    neutral = 'D://Depression_Analysis//Depression_Analysis//dataset//Neutral'
    number_of_neutral_files = len([item for item in os.listdir(neutral) if os.path.isfile(os.path.join(neutral, i
```



```
#return C

neutral = 'D://Depression_Analysis//Depression_Analysis//dataset//Neutral'
number_of_neutral_files = len([item for item in os.listdir(neutral) if os.path.isfile(os.path.join(neutral, i
#print(number_of_neutral_files)
D = "Neutral Person are = {0}".format(number_of_neutral_files)
print(D)
#return D

Surprised = 'D://Depression_Analysis//Depression_Analysis//dataset//Surprised'
number_of_Surprised_files = len([item for item in os.listdir(neutral) if os.path.isfile(os.path.join(neutral,
#print(number_of_neutral_files)
E = "Surprised Person are = {0}".format(number_of_neutral_files)
print(E)

Disgusted = 'D://Depression_Analysis//Depression_Analysis//dataset//Disgusted'
number_of_Disgusted_files = len([item for item in os.listdir(neutral) if os.path.isfile(os.path.join(neutral,
#print(number_of_neutral_files)
F = "Disgusted Person are = {0}".format(number_of_neutral_files)
print(F)


if int(number_of_Happy_files) > int(number_of_Fear_files) and int(number_of_Happy_files) > int(number_of_sad_
    str_label="User Is Happy And No Symptoms of Gloom"
    print(str_label)

elif int(number_of_Fear_files) > int(number_of_Happy_files) and int(number_of_Fear_files) > int(number_of_sad
    str_label = "User Is In Fear And Some Symptoms of Gloom"
    print(str_label)

elif int(number_of_neutral_files) > int(number_of_Happy_files) and int(number_of_neutral_files) > int(number
```



```
E = "Surprised Person are = {0}".format(number_of_neutral_files)
print(E)

Disgusted = 'D://Depression_Analysis//Depression_Analysis//dataset//Disgusted'
number_of_Disgusted_files = len([item for item in os.listdir(neutral) if os.path.isfile(os.path.join(neutral,
#print(number_of_neutral_files)
F = "Disgusted Person are = {0}".format(number_of_neutral_files)
print(F)


if int(number_of_Happy_files) > int(number_of_Fear_files) and int(number_of_Happy_files) > int(number_of_sad_
    str_label="User Is Happy And No Symptoms of Gloom"
    print(str_label)

elif int(number_of_Fear_files) > int(number_of_Happy_files) and int(number_of_Fear_files) > int(number_of_sad
    str_label = "User Is In Fear And Some Symptoms of Gloom"
    print(str_label)

elif int(number_of_neutral_files) > int(number_of_Happy_files) and int(number_of_neutral_files) > int(number_
    str_label = "User Is In Neutral State"
    print(str_label)

elif int(number_of_sad_files) > int(number_of_Happy_files) and int(number_of_sad_files) > int(number_of_neutr
    str_label = "User Is Sad And Mild Symptoms of Gloom"
    print(str_label)

elif int(number_of_Disgusted_files) > int(number_of_Happy_files) and int(number_of_Disgusted_files) > int(num
    str_label = "High Symptoms of Gloom"
    print(str_label)
return str_label
```

The above code is used to fetch the frames from the video and detect which kind of face expression is currently on the user face. After the frames is extracted, the user is been given the predicted output whether user is in Gloom or not.

CHAPTER 6
PROJECT PLAN

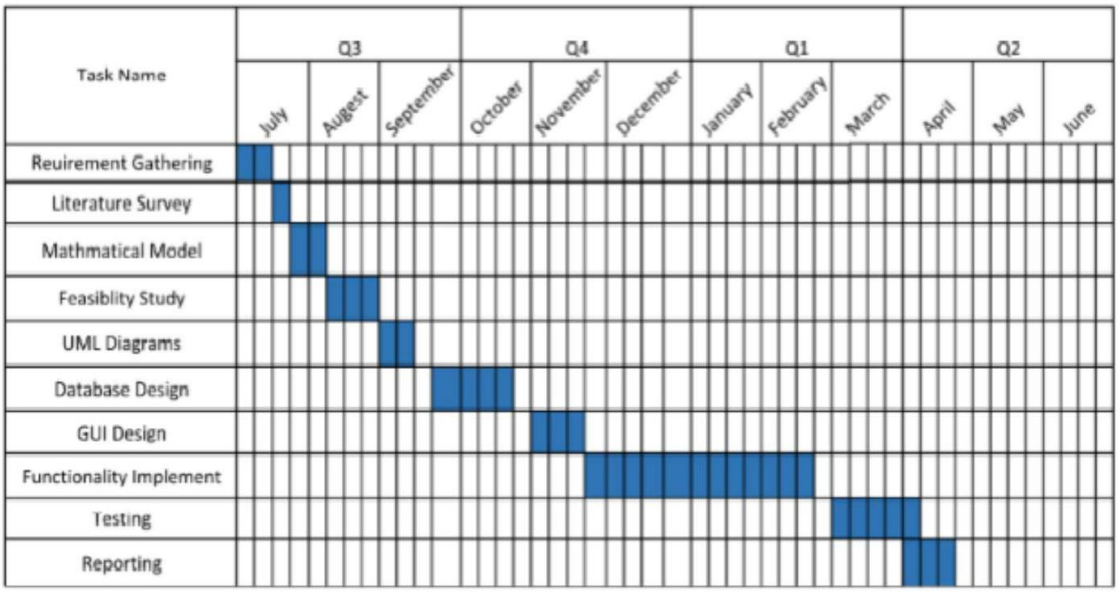
In this chapter we are going to have an overview about how much time does it took to complete each task like Preliminary Survey Introduction and Problem Statement, Literature Survey, Project Statement, Software Requirement and Specification, System Design, Partial Report Submission, Architecture Design, Implementation, Deployment, Testing, Paper Publish, Report Submission and etc. This chapter also gives focus on stakeholder list which gives information about project type, customer of the proposed system, user and project member who developed the system.

6.1 STAKEHOLDER LIST

Sr. No.	Stack holder	User
1.	Customer	Authorized Person
2.	User	User

6.2 SYSTEM IMPLEMENTATION PLAN

The System Implementation plan table, shows the overall schedule of tasks compilation and time duration required for each task.



Chapter 7
Testing

7.1 TEST CASES

Test case	Login Screen- Sign up
Objective	Click on sign up button then check all required/ mandatory fields with leaving all fields blank
Expected Result	All required/ mandatory fields should display with symbol "*". Instruction line "* field(s) are mandatory" should be displayed
Test case	Create a Password >>Text Box Confirm Password >>Text Box
Objective	Check the validation message for Password and Confirm Password field
Expected Result	Correct validation message should be displayed accordingly or "Password and confirm password should be same" in place of "Password mismatch".

Table 7.1: GUI TESTING

Registration test cases:

Test Case ID	Test Case	Test Case I/P	Actual Result	Expected Result	Test case criteria(P/F)
001	Enter the number in username, middle name, last name field	Number	Error Comes	Error Should Comes	P
001	Enter the character in username, middle name, last name field	Character	Accept	Accept	p
002	Enter the invalid email id format in email id field	Kkgmail,com	Error comes	Error Should Comes	P
002	Enter the valid email id format in email id field	kk@gmail.com	Accept	Accept	P
003	Enter the invalid digit no in phone no field	99999	Error comes	Error Should Comes	P
003	Enter the 10 digit no in phone no field	9999999999	Accept	Accept	P

Table 7.2: Registration test case

Login test cases:

Test Case ID	Test Case	Test Case I/P	Actual Result	Expected Result	Test case criteria(P/F)
001	Enter The Wrong username or password click on submit button	Username or password	Error comes	Error Should come	P
002	Enter the correct username and password click on submit button	Username and password	Accept	Accept	P

Table 7.3: Login test case

System Test Cases:

Test Case ID	Test Case	Test Case I/P	Actual Result	Expected Result	Test case criteria(P/F)
001	Store Xml File	Xml file	Xml file store	Error Should come	P
002	Parse the xml file for conversion	parsing	File get parse	Accept	P
003	Attribute identification	Check individual Attribute	Identify Attributes	Accepted	P
004	Weight Analysis	Check Weight	Analyze Weight of individual Attribute	Accepted	P
005	Tree formation	Form them-Tree	Formation	Accepted	P
006	Cluster Evaluation	Check Evaluation	Should check Cluster	Accepted	P
007	Algorithm Performance	Check Evaluation	Should work Algorithm Properly	Accepted	P
008	Query Formation	Check Query Correction	Should check Query	Accepted	P

Table 7.4: System Test Cases:

Chapter 8

Result and Discussion

8.1 RESULT

Main GUI :

- This is the first home page where user will be able to register or login

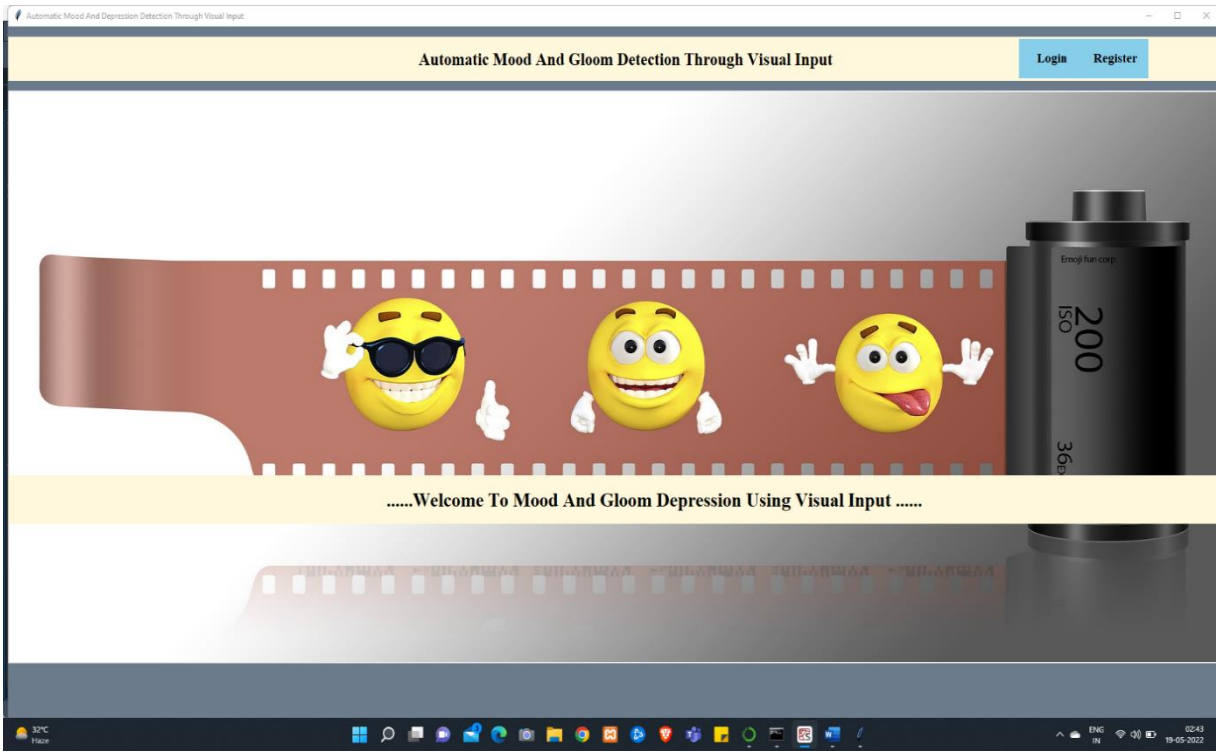


Figure 8.1: Main File

Registration snapshot:

- When the new user arrives, user have to register himself through registration form
- Entering all the required and Validate details user will be able to register himself Like Full name, Last name, etc.

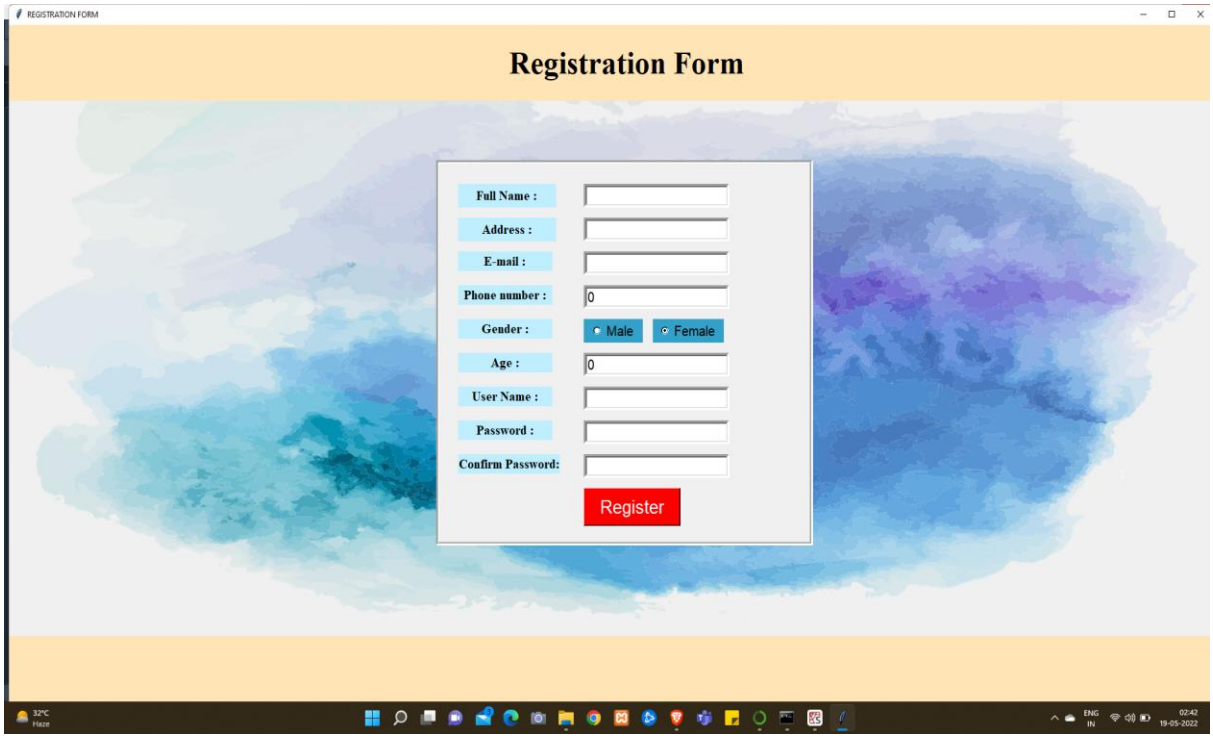


Figure 8.2: Registration File

Login Snapshot:

- After Successful Registration User have to enter valid credential to successfully login to their account

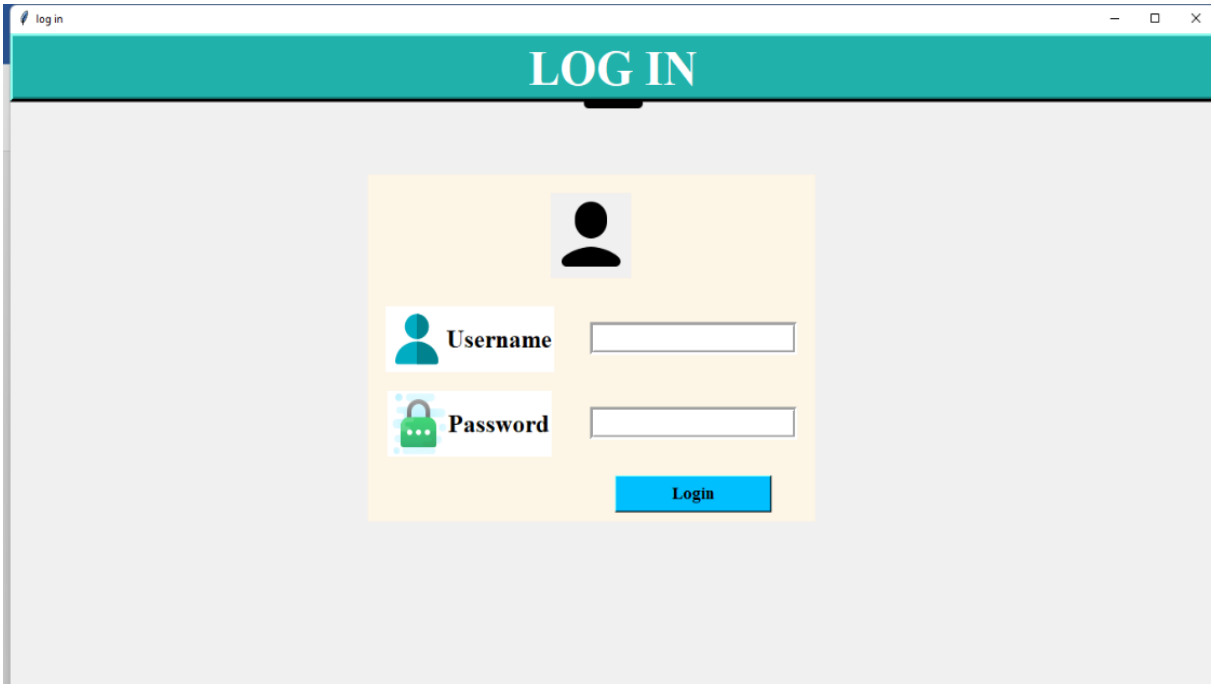


Figure 8.3: Login File

Services:

- After successfully login of user the user will see the master view through which he will be able to access the services

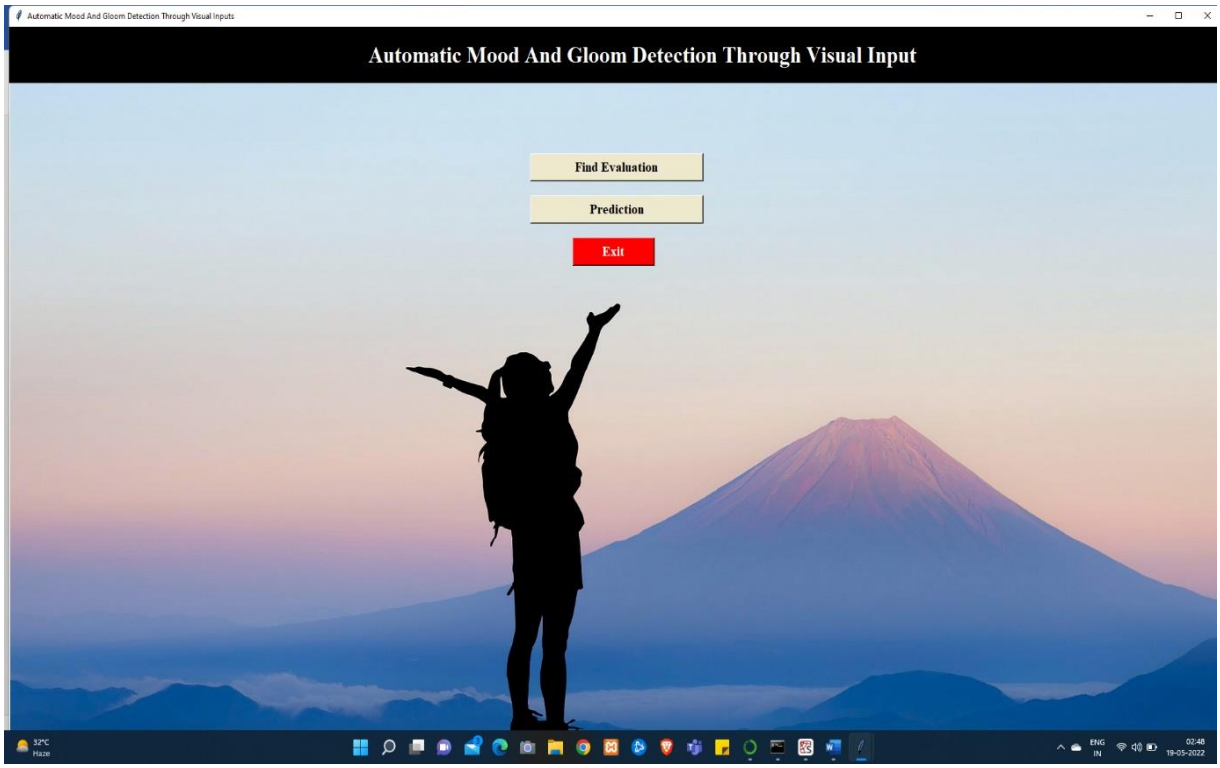


Figure 8.4: Master

Output Frame:

- This is the live output detected by the software predicted using facial emotions

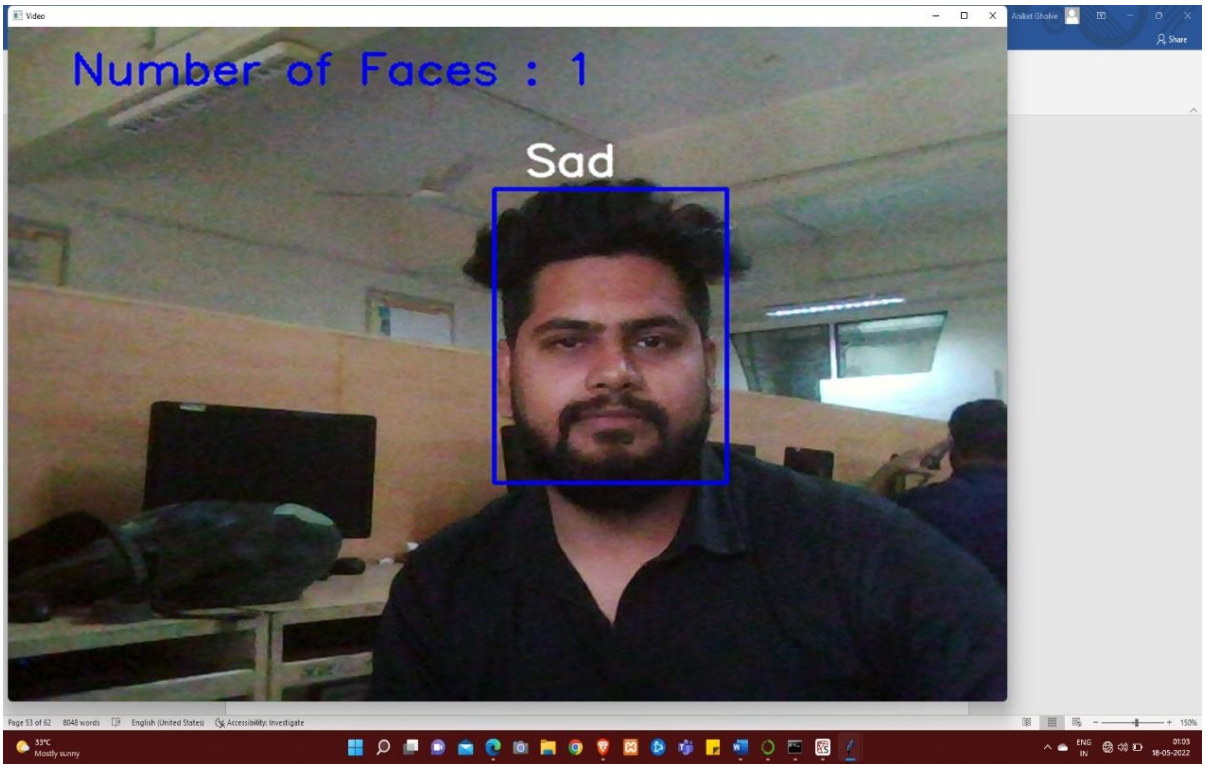


Figure 8.5: Output Frame Sad

Output Frame:

- This is the live output detected by the software predicted using facial emotions

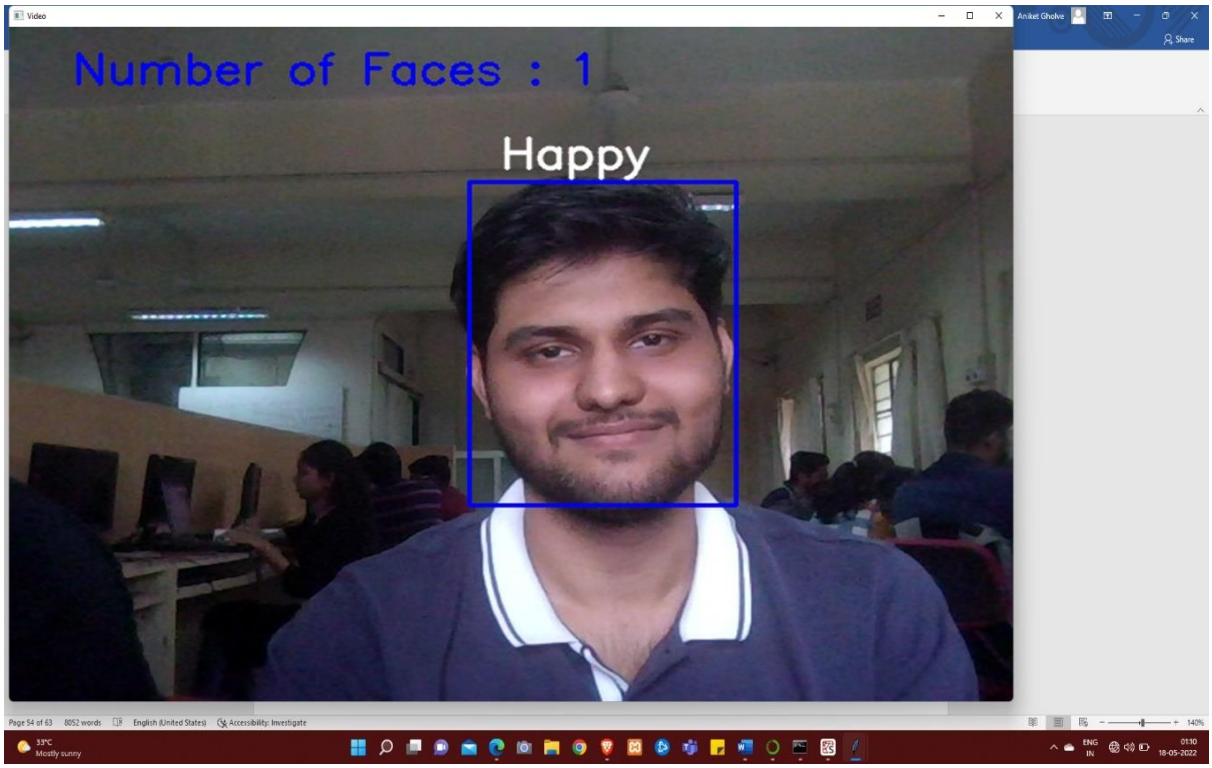


Figure 8.6 : Output Frame Happy

Output Frame:

- This is the live output detected by the software predicted using facial emotions

Fig 8.7 Output Frame Fearful

Output Frame:

- This is the live output detected by the software predicted using facial emotions

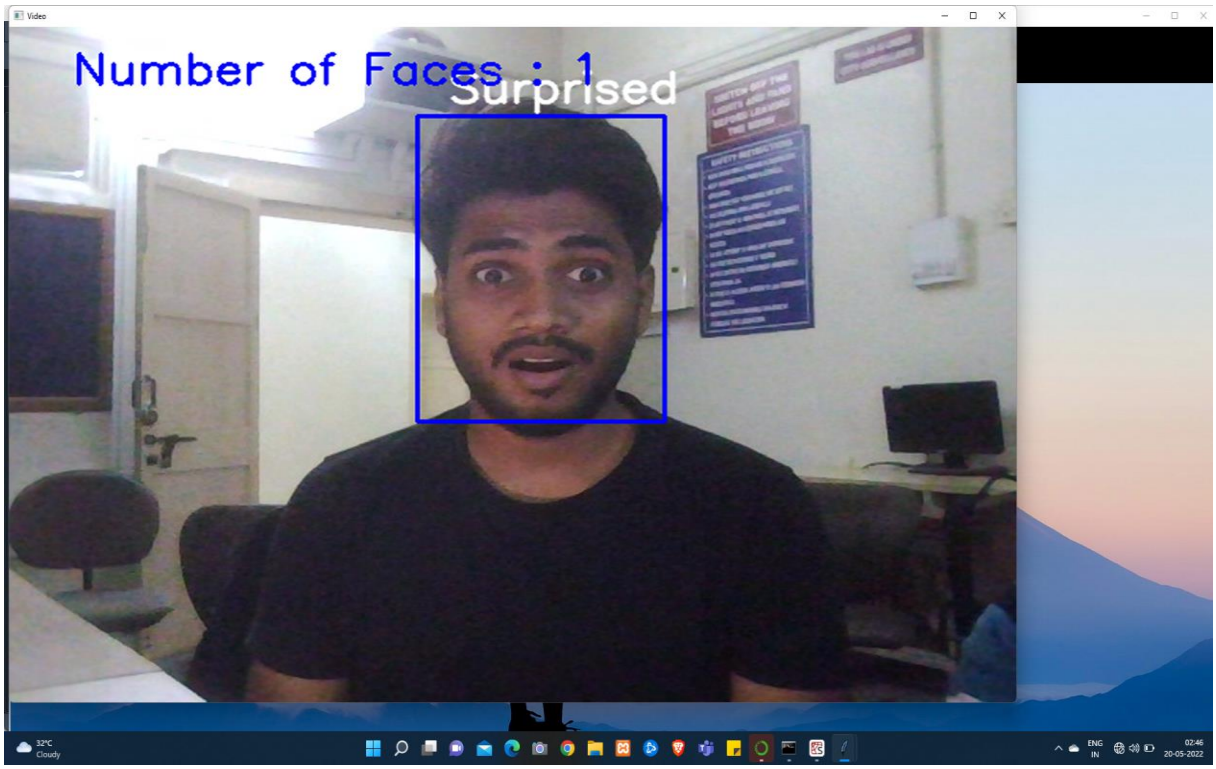


Fig 8.8 Output Frame Surprised

Predicted Output:

- Snapshot shows the predicted output of the code wither the user is Gloomed or not

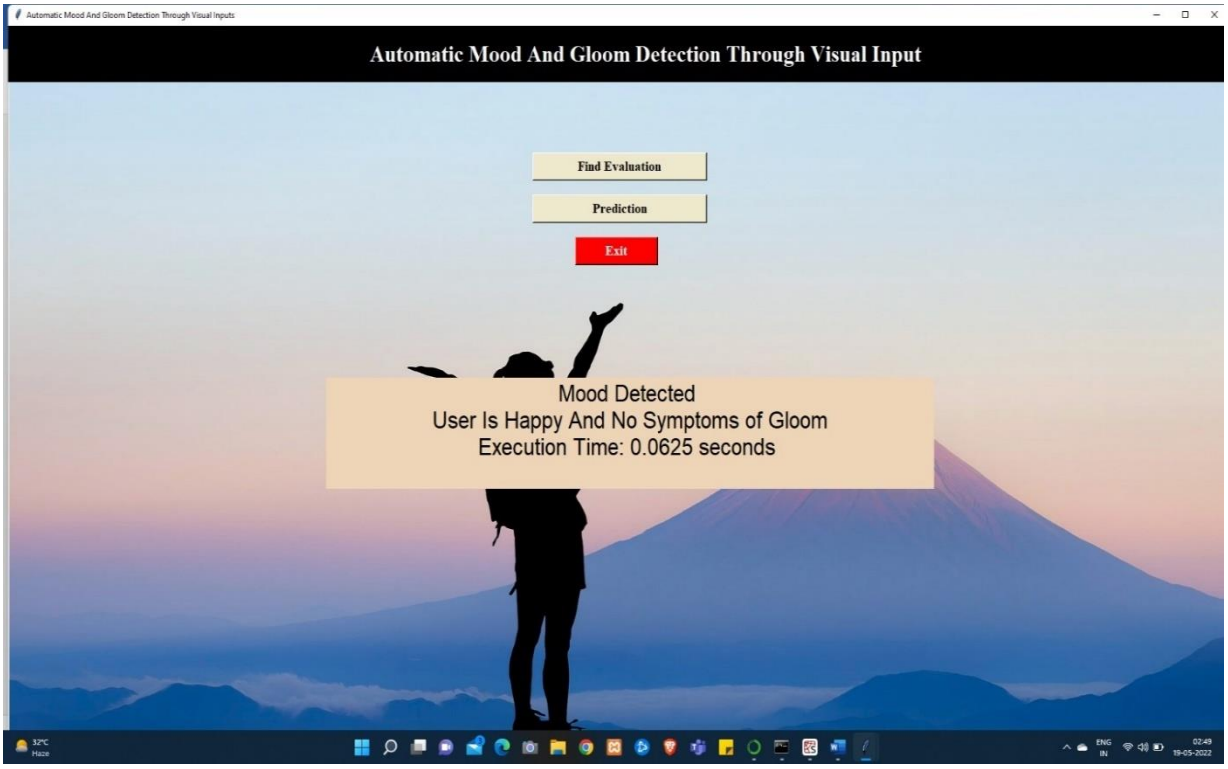


Fig 8.9: Check File

Predicted Output:

- Snapshot shows the predicted output of the code with user current Mood And Gloom or not

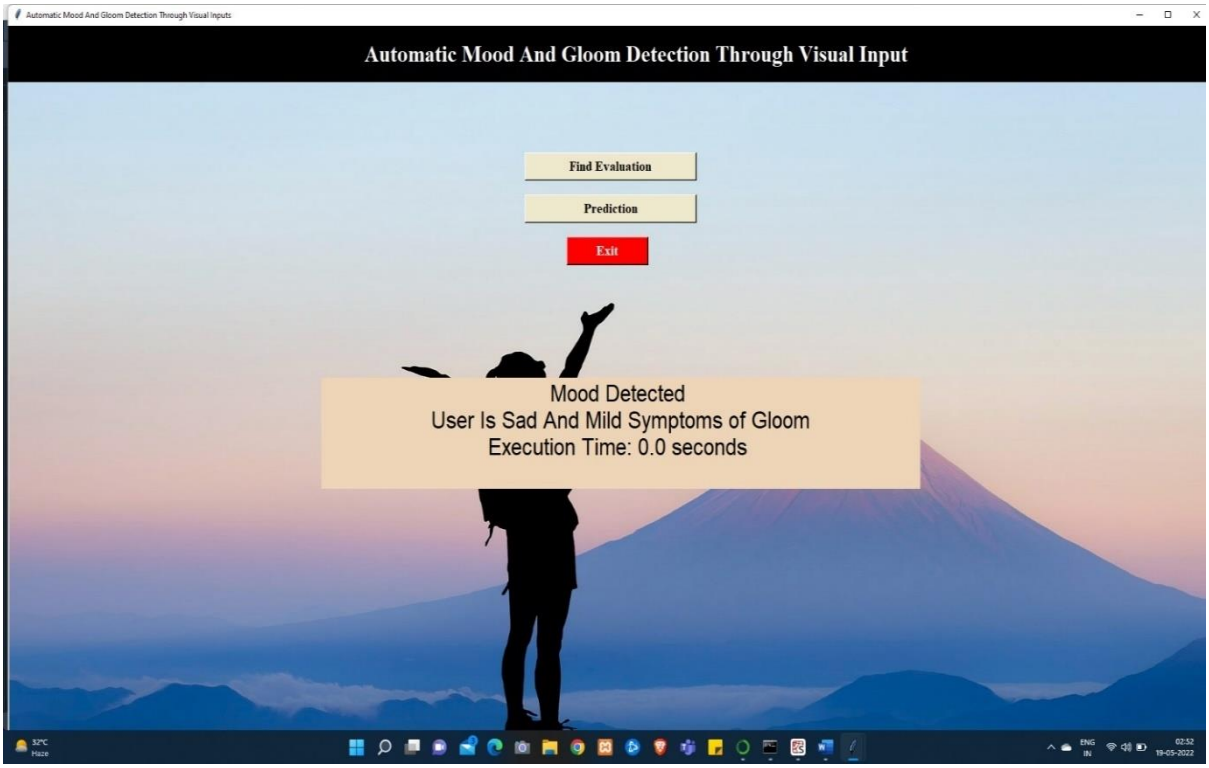


Fig 8.10: Check File

Chapter 9
Conclusion and Future Work

Conclusion:

Different Mood and Gloom detector models were equipped with multiple ECGs in this project. Features such as interval of QT, RR. This method of detecting ECG signal tension will assist to determine one's psychological state and also physical health, aperson from user will be capable of taking the steps expected. It was concluded, too, that the more properties user use, the more detailed the model becomes.

This study was undertaken for finding out the level of moods in five different videos of user. The presence of 'Happy', 'Neutral', - (positive emotion) and 'Contempt' and 'Degust'- (Negative emotion) facial features, which are found prominent in videos were found out and analyzed. The dataset for training and testing was captured separately and the facial features of the same were classified using a CNN classifier. The amount of the positive and negative emotions in each image was analyzed and the image were predicted as image with 'Happy And Not Gloom', 'Sad and Mild Gloom' or 'High Symptoms of Gloom'.

Future Work:

In the current system we have been successful in integrating the Gloom Detection. The current system can easily show whether user is Gloomed or not

However, we would like to work further to achieve the following goals in the near future:

1. When the user is quits the application, it should automatically delete the backend images
2. When user uses the application the inputs data when they quit application should be given to training dataset so that it will train the model automatically and this will increase the accuracy of the model.
3. Predict detection of user on based on their chats
4. Predict detection of user asking them questions based on standards question

Chapter 10
References

1 Nur Alia Syahirah Badrulhisham and Nur Nabilah Abu Mangshor, “Emotion Recognition Using Convolutional Neural Network (CNN)” . The 1st International Conference on Engineering and Technology (ICoEngTech) , Journal of Physics: Conference Series 2021 doi:10.1088/1742-6596/1962/1/012040.

2 D Y Liliana, “Emotion recognition from facial expression using deep convolutional neural network” 2018 International Conference of Computer and Informatics Engineering (IC2IE), IOP Conf. Series: Journal of Physics: Conf. Series 1193 (2019) 012004 I .doi:10.1088/1742-6596/1193/1/012004.

3 Rahul Chauhan, Kamal Kumar Ghanshala and R.C Joshi, “Convolutional Neural Network (CNN) for Image Detection and Recognition” 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)

4 Akash Saravanan, Gurudutt Perichetla, Dr. K.S.Gayathri, “Facial Emotion Recognition using Convolutional Neural Networks” doi : arXiv:1910.05602v1 [cs.CV] 12 Oct 2019

5 Lang Hea,∗ , Cui Caob, “Automated depression analysis using convolutional neural networks from speech,” Journal of Biomedical Informatics Volume 83, July 2018, Pages 103-111.

6 Ninad Mehendale “Facial emotion recognition using convolutional neural networks (FERC)”. Received: 16 July 2019 / Accepted: 12 February 2020 / Published online: 18February 2020 © Springer Nature Switzerland AG 2020

7 Erhan Tiryaki Akshay Sonawane, Lakshman Tamil. “Real-Time CNN Based ST Depression Episode Detection Using Single-Lead ECG”. 021 22nd International Symposium on Quality Electronic Design (ISQED) | 978-1-7281-7641-3/20/$31.00 ©2021 IEEE | DOI: 10.1109/ISQED51717.2021.9424275;

8 Lang Hea,b,, Jonathan Cheung-Wai Chanc , Zhongmin Wang,
"Automatic Depression Recognition using CNN with Attention Mechanism from Videos" ESVEM Investigators. Electrophysiologic

Study Versus Electrocardiographic Monitoring Trial. J Am Coll Cardiol.; 32:1891-9.73, 1998.

9 Milan Tripathi. "FACIAL EMOTION RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK" ISSN: 0976-9102 (ONLINE) ICTACT JOURNAL ON IMAGE AND VIDEO PROCESSING, AUGUST 2021, VOLUME: 12, ISSUE: 01 DOI: 10.21917/ijivp.2021.0359

10 Arselan Ashraf1 , Teddy Surya Gunawan2 , Bob Subhan Riza3 , Edy Victor Haryanto4 , Zuriati Janin5 "On the review of image and video-based depression detection using machine learning." Indonesian Journal of Electrical Engineering and Computer Science Vol. 19, No. 3, September 2020, pp. 1677~1684 ISSN: 2502-4752, DOI: 10.11591/ijeecs.v19.i3.pp1677-1684;