A

Major Project

On

# AN INTELLIGENT SYSTEM FOR IDENTIFICATION CARD DETECTION AND AUTHENTICATION

**Submitted to**

**Jawaharlal Nehru Technological University, Hyderabad**

**(Submitted in partial fulfillment of the requirements for the award of Degree)**

BACHELOR OF TECHNOLOGY
in

COMPUTER SCIENCE AND ENGINEERING

By

| | |
|---|---|
| **M.V.P SAI SRIRAM** | **(187R1A05L6)** |
| **KOTTE SUMA** | **(197R5A0518)** |
| **K. PRATISH REDDY** | **(187R1A05P7)** |
| **S. SANTOSH** | **(197R5A0519)** |

Under the esteemed guidance of

**G.Latha**

(Associate Professor)

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**CMR TECHNICAL CAMPUS**

**UGC AUTONOMOUS**

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

**2018-22**

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

## CERTIFICATE

This is to certify that the project entitled “**AN INTELLIGENT SYSTEM FOR IDENTIFICATION CARD DETECTION AND AUTHENTICATION** being submitted by **M.V.P. Sai Sriram (187R1A05L6), Kotte Suma (197R5A0518), K. Pratish Reddy (187R1A05P7), S. Santhosh (197R5A0519)** in partial fulfillment of the requirements for the award of the degree of B. Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of Bonafide work carried out by him/her under our guidance and supervision during the year 2021-22.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

**INTERNAL GUIDE**
**G.Latha**
Associate Professor

**DIRECTOR**
**Dr. A. Raji Reddy**

**HOD**
**Dr. K. Srujan Raju**

**EXTERNAL EXAMINER**

**Submitted for viva voice Examination held on** ____________________

# ACKNOWLEGDEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We take this opportunity to express my profound gratitude and deep regard to my guide **G. Latha**, Associate Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) **Mr. J. Narasimha Rao, Dr. T. S. Mastan Rao, Mr. A. Uday Kiran, Mr. A. Kiran Kumar, Mrs. G. Latha** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju**, Head, Department of Computer Science and Engineering for providing excellent infrastructure and a nice atmosphere for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy**, Director for being cooperative throughout the course of this project. We also express our sincere gratitude to our Chairman Sri. **Ch. Gopal Reddy** for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed and who contributed to the completion of the project, was vital for the success of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be possible. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project.

| | |
|---|---|
| **M. V. P. SAI SRIRAM** | **(187R1A05L6)** |
| **KOTTE SUMA** | **(197R5A0518)** |
| **K. PRATISH REDDY** | **(187R1A05P7)** |
| **S. SANTHOSH** | **(197R5A0519)** |

# ABSTRACT

This project is titled as “An Intelligent System for Identification card detection and Authentication, Identification Cards help us to make sure that every person in the organization is identified. It is a quick and easy way to determine whether or not a person belongs to particular organization. The main goal of computer vision is to identify and recognize different objects of various size, shape and position. To facilitate object detection in a college environment, the proposed work identifies the presence of a person wearing an ID card using tensor flow object detection API, detects and recognizes using OpenCV. For this, we have to train the student face and then we have to train the id card when the two images are matched then the result will be notified. It also helps the organization to identify if the person belongs to their own organization or not. With the help of this project, we can also evaluate the attendance of the person which would be highly useful for educational institutions.

# LIST OF FIGURES

# LIST OF SCREENSHOTS

## TABLE OF CONTENTS PAGE NO'S

# 1.INTRODUCTION

# 1.INTRODUCTION

## 1.1 PROJECT SCOPE

This project is titled as “An Intelligent System for Identification card detection and authentication”. The main intention of this project is to identify the students in the organization. Uses live face recognition to recognize each individual and identifying the students whether they are belonging to the particular organization on not and also detecting the fake and original ID cards by using face recognition.

## 1.2 PROJECT PURPOSE

The purpose of developing student identification card detection is to computerized way of detecting the id card whether the person is wearing true Identity Card or Not. It helps you to track the person who are wearing fake id card. It is used to keep the track of people in the organization who are wearing invalid Identity card.

## 1.3 PROJECT FEATURES

The core features of this project are an intelligence system can recognize a face ina video and identify students and their identity card thereby recognizing faces in various positions are detected by sliding a window, while faces of varioussizes are detected via scaling. Detecting several faces and their identity cards in live video, and knowing whether the person belongs to particular organization or not. We can store the dataof detected students in the database at any moment. For improving accuracy, we employedthe HAAR Cascade Face Detection approach, which is based on Histograms of OrientedGradients (HOG) and several types of features and classifiers.

# 2. SYSTEM ANALYSIS

# 2. SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system.In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

## 2.1 PROBLEM DEFINITION

This system developed will reduce the manual work and avoid redundant data. It takes a lotof time for manual paper work for both students and faculty, to overcome this situation we proposed a smart attendance system using face recognition technique. After detecting the faces of the students then the attendance information is saved automatically in an excel sheet.

## 2.2 EXISTING SYSTEM

- The existing system interacts with human behavior and issues a manual alert when a student does not wear the id card.
- The second existing model focuses that which is built using some of the deep learning methods. The technique is carried out here using densnet121, which is a transfer learning method, although it does not achieve great accuracy.

### 2.2.1 DISADVANTAGES OF EXISTING SYSTEM

- In these existing methods there is no idea that the person in this organization is true or not.
- Lot of time was consumed by using manual alert when a student doesn’t wear the identity card.

## 2.3 PROPOSED SYSTEM

- We perform detection of ID card photos identification using Convolution Neural Network (CNN) of deep learning and Open cv approaches in the proposed manner.
- As image analysis-based ways for identifying the ID card under a webcam and alerting people who are not wearing it. And also detecting whether the person is his/her own identity card or not by checking the identity validity.

### 2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- From the actual effect the detection speed can be compared with the state-of-the-art methods and achieve a higher accuracy.

- A live video may contain many faces and these faces can be detected and when the face and the registration number of the person matches, then the identity card is true. if not, then notice that the particular person is wearing fake identity card.

- Video surveillance systems are very suitable for the physical security, since the video sequence from many remote areas can be presented to watch them at a time.

## 2.4 FEASIBILITY STUDY

Feasibility analysis begins once the goals are defined. It Starts y generating broad possible solutions, which are possible to give an indication of what the new system should look like. This is where creativity and imagination are used Analysts must think up new ways of doing things generate new ideas. There is no need to go into the detailed system operation yet. Feasibility of a new system means ensuring that the new system, which we are going to implement, is efficient and affordable. There are various types of feasibility to be determined. They are:

- Economic Feasibility
- Technical Feasibility
- Operational Feasibility

### 2.4.1 ECONOMICAL FEASIBILITY

Development of this application is highly economically feasible. The only thing to be done is making an environment with an effective supervision. It is cost effective in the sense that has eliminated in the paperwork completely. The system is also time effective because the calculations are automated as per the user requirement.

### 2.4.2 TECHNICAL FEASIBILITY

The technical requirement for the system is economic and it does not use any other additional hardware and software. Technical evaluation must also assess whether the existing system can be upgraded to use the new technology and whether the organization has the expertise to use it.

### 2.4.3 OPERATIONAL FEASIBILITY

The system working is quite easy to use and learn due to its simple but attractive interface. User requires no special training for operating the system. Technical performance includes issues such as determining whether the system can provide right information for the student details and whether the system can be organized so that it always delivers this information at the right place and on time.

## 2.5 HARDWARE & SOFTWARE REQUIREMENTS

### 2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

| | | |
|---|---|---|
| System | : | Intel i5 processor |
| Hard Disk | : | **256** GB (SSD) |
| RAM | : | 4 GB(min) |

### 2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements.

| | | |
|---|---|---|
| Operating System | : | Windows 10 |
| Language | : | Python 3.9.7 |
| IDE | : | Visual Studio Code 1.67 |
| Libraries used | : | Flask 2.0, Tensorflow 2.8.0, OpenCV 4.5.5 |
| Object Detection | : | YOLO v5 |

# 3.ARCHITECTURE

# 3. ARCHITECTURE

## 3.1 PROJECT ARCITECTURE

This project architecture shows the procedure followed An Intelligent System for Identification card detection using Face Detection Algorithms starting from input to final prediction.

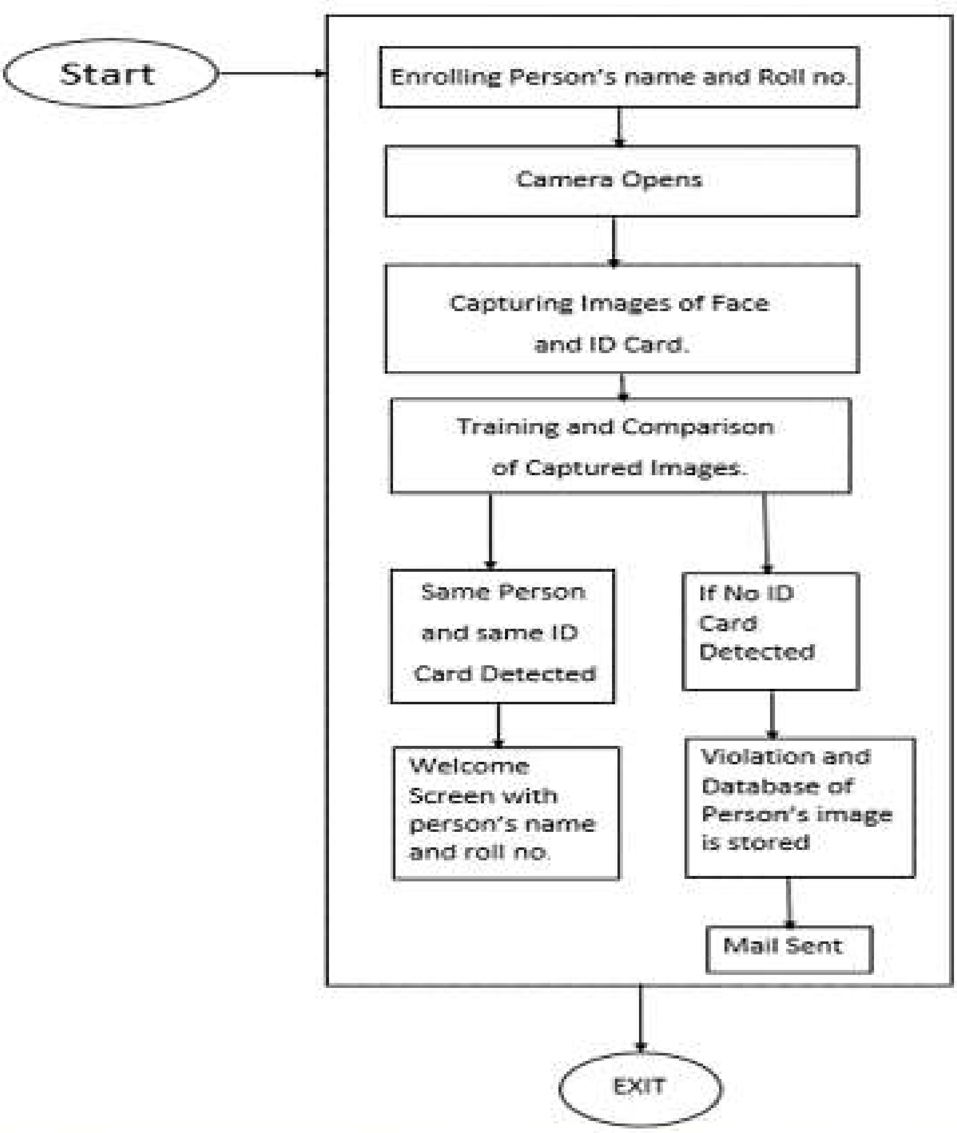

Start
Enrolling Person's name and Roll no.
Camera Opens
Capturing Images of Face and ID Card.
Training and Comparison of Captured Images.
Same Person and same ID Card Detected
If No ID Card Detected
Welcome Screen with person's name and roll no.
Violation and Database of Person's image is stored
Mail Sent
EXIT

Figure: 3.1 Project Architecture

## 3.2 MODULES DESCRIPTION

### 3.2.1 User Module:

In this module user will open camera and track images of every student. For each tracking process it will take student's images and then user should close the process.

### 3.2.2 Training Module:

Once taking images process is done from folder and for each image training process is done using open CV. Once the process is done, then the comparison takes place between student image and ID Card image. And when the two images matches then the result will be notified.

### 3.2.3 Detection Module:

In this process when user opens camera it will track live images of user and convert user image and check with face recognition model and then boxes are drawn on each face and features are verified with trained model and output is displayed. And then when the student image and ID card images match then result is notified and the names of students those not wearing ID Card are stored in the database.

## 3.3 USE CASE DIAGRAM

In UML, Use-Case diagrams model the behavior of the system and help to capture the requirements of the system. Use case diagrams describe the High-Level functions and scope of the system. These diagrams also identify the interactions between the System and its actors.

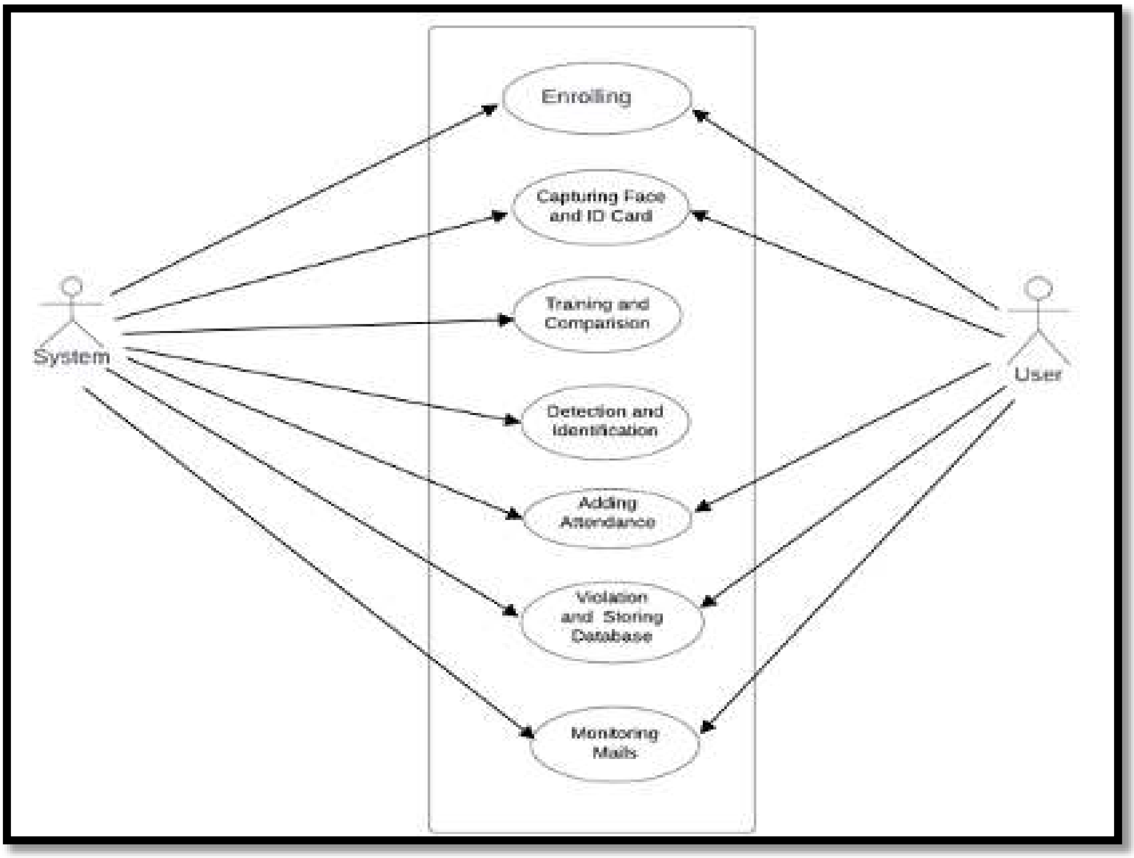

Enrolling
Capturing Face and ID Card
Training and Comparision
Detection and Identification
Adding Attendance
Violation and Storing Database
Monitoring Mails
System
User

Figure: 3.3 Use Case Diagram

## 3.4 ACTIVITY DIAGRAM

In UML, an activity provides a view of the behaviour of a system by describing the sequence of actions in a process, in activity diagrams, you use activity nodes and activity edges to model the flow of control and data between actions.

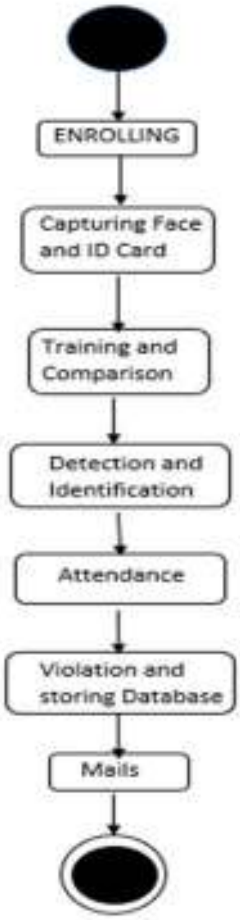

ENROLLING
Capturing Face and ID Card
Training and Comparison
Detection and Identification
Attendance
Violation and storing Database
Mails

Figure: 3.4 Activity Diagram

## 3.5 CLASS DIAGRAM

A class diagram resembles a flowchart in which classes are portrayed as boxes, each box having three rectangles inside. The top rectangle contains the name of the class; the middle rectangle contains the attributes of the class; the lower rectangle contains the methods, also called operations, of the class.

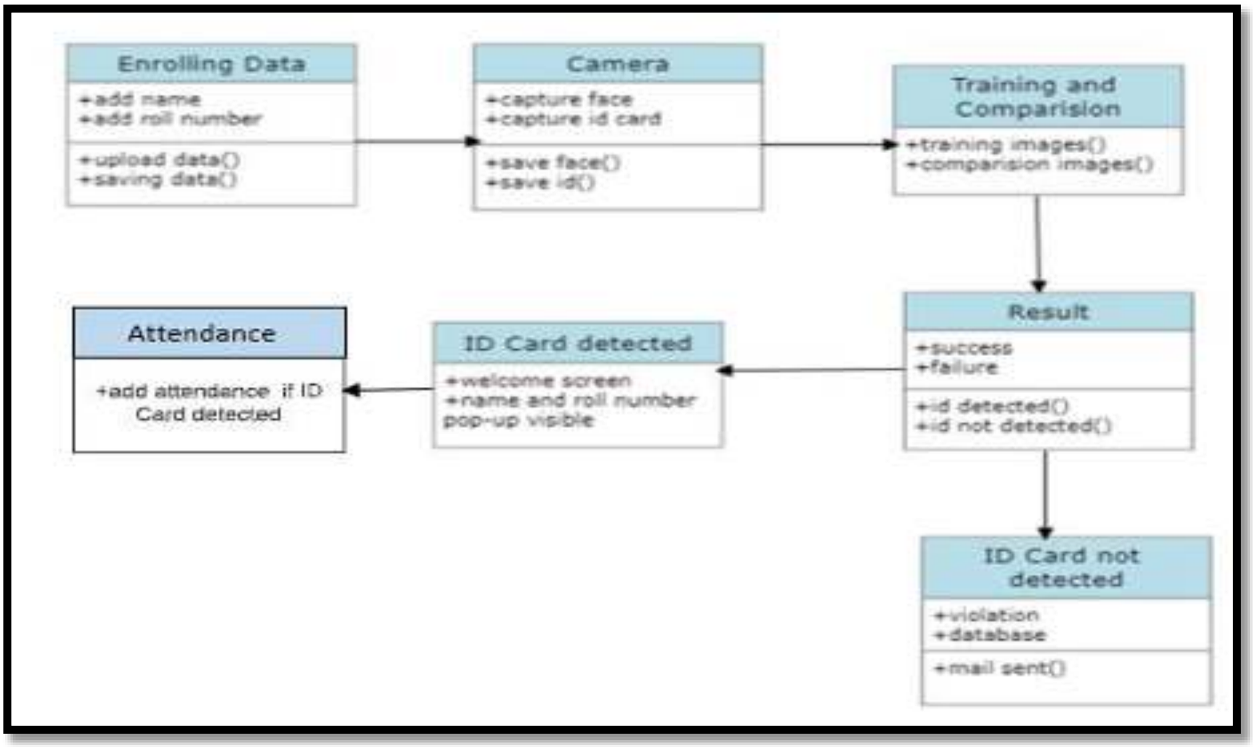


Figure: 3.5 Class Diagram

## 3.6 SEQUENCE DIAGRAM

A Sequence diagram is a UML diagram that illustrates the sequence of messages between objects in an interaction.

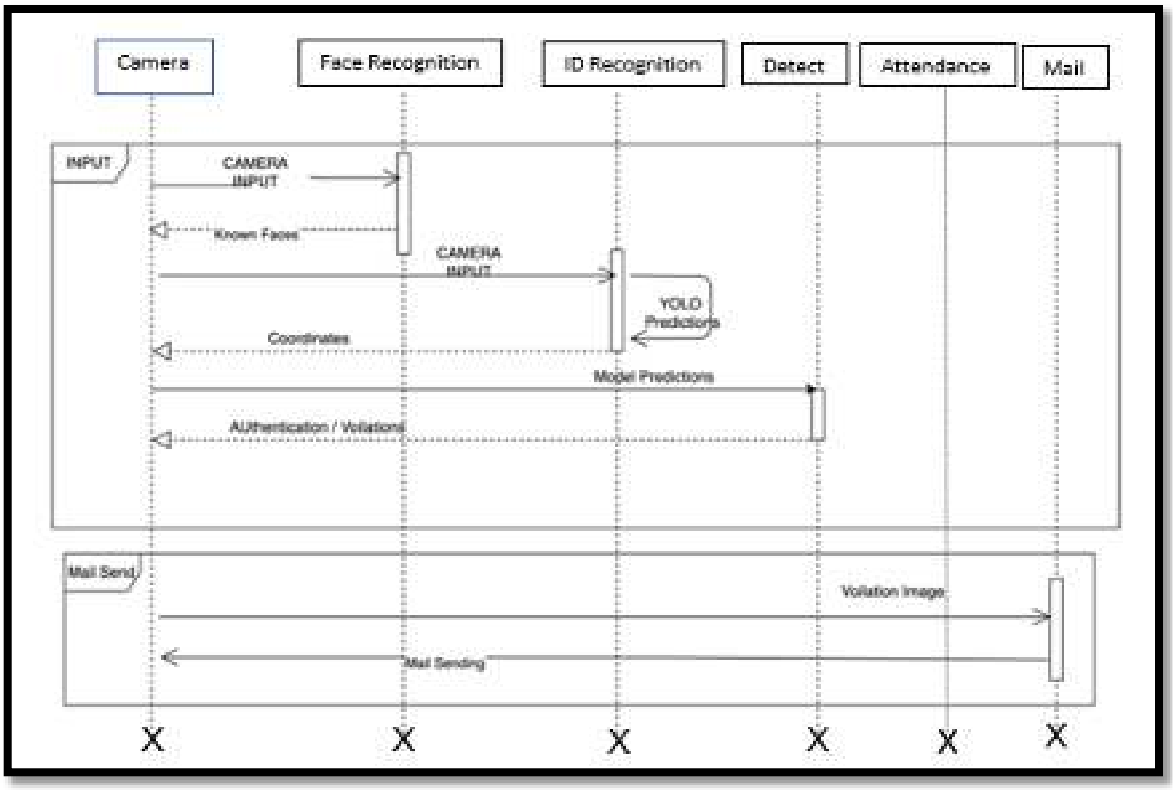


Figure:3.6 Sequence Diagram

## 3.7 DATAFLOW DIAGRAM

A data-flow diagram is a way of representing a flow of data through a process or a system.

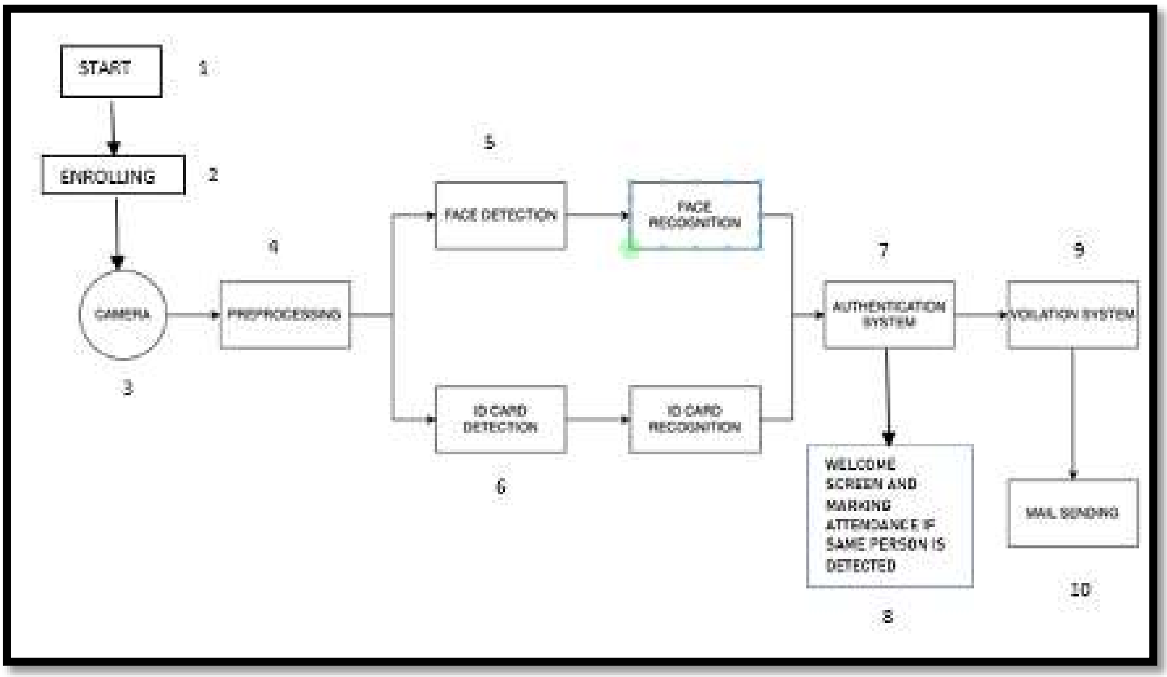


Figure 3.7 Dataflow Diagram

# 4. IMPLEMENTATION

# 4.IMPLEMENTATION

## 4.1 SAMPLE CODE

```
import numpy as np
import os
import systemcheck
import cv2
import torch

from models.common import DetectMultiBackend
from utils.datasets import LoadImages
from utils.general import check_img_size, non_max_suppression, scale_coords
from utils.plots import Annotator, colors
from utils.torch_utils import select_device

import torchvision.ops.boxes as bops
from statistics import mean
import time
from datetime import datetime

import face_recognition
from mailsend import sendmail

MODEL = "HOG"
TOLERANCE = 0.5
ID_TOLERANCE = 0.3
ID_FACE_TOLERANCE = 0.55

inp = input("Press Enter to run System or 'add' to Add new person:")
if 'add' in inp.lower():
   add_face = True
   print("New Person Enrolling Mode...\n\n")
   time.sleep(3)
else:
   add_face = False
   print("Normal Mode...\n\n")
   time.sleep(3)

def get_model(weights,  # model.pt path(s)
       imgsz=640,  # inference size (pixels)
       device='',  # cuda device, i.e. 0 or 0,1,2,3 or cpu
       half=False,  # use FP16 half-precision inference
       dnn=False,  # use OpenCV DNN for ONNX inference
       ):
```

```
    device = select_device(device)
    model = DetectMultiBackend(weights, device=device, dnn=dnn)
    stride, names, pt, jit = model.stride, model.names, model.pt, model.jit
    imgsz = check_img_size(imgsz, s=stride)  # check image size

    # print("Names:", names)

    # Half
    half &= pt and device.type != 'cpu'  # half precision only supported by PyTorch on CUDA
    if pt:
        model.model.half() if half else model.model.float()

    if pt and device.type != 'cpu':
        model(torch.zeros(1, 3, *imgsz).to(device).type_as(next(model.model.parameters())))  #
warmup

    ###################### Model ready for Detection
#########################################
    model_yolo = (device,model,stride, names, pt, jit, imgsz, half )
    print("Model ready for Detection")
    return model_yolo


def detect(model_yolo,
        source,
        classes=None,  # filter by class: --class 0, or --class 0 2 3
        visualize = False,
        line_thickness=3,  # bounding box thickness (pixels)
        hide_labels=False,  # hide labels
        hide_conf=False,  # hide confidences
        conf_thres=0.20,  # confidence threshold
        iou_thres=0.45,  # NMS IOU threshold
        max_det=100,  # maximum detections per image
        ):

    device,model,stride, names, pt, jit, imgsz, half = model_yolo
     # Dataloader
    dataset = LoadImages(source, img_size=imgsz, stride=stride, auto=pt and not jit)

    for path, im, raw, vid_cap, s in dataset:

        im = torch.from_numpy(im).to(device)
        im = im.half() if half else im.float()  # uint8 to fp16/32
        im /= 255  # 0 - 255 to 0.0 - 1.0
        if len(im.shape) == 3:
            im = im[None]  # expand for batch dim

        # Inference
        pred = model(im)

        pred = non_max_suppression(pred, conf_thres, iou_thres, classes, max_det=max_det)
        coords = list()
        classes = list()
        # Process predictions


        for i, det in enumerate(pred):  # per image

            annotator = Annotator(raw, line_width=line_thickness, example=str(names))

                if len(det):
                # Rescale boxes from img_size to raw size
                det[:, :4] = scale_coords(im.shape[2:], det[:, :4], raw.shape).round()
                # Write results
                for *xyxy, conf, cls in reversed(det):
                    c = int(cls)  # integer class
                    label = None if hide_labels else (names[c] if hide_conf else f'{names[c]}
{conf:.2f}')
                    # if names[c] in ['id card', 'person']:
                    #     annotator.box_label(xyxy, label, color=colors(c, True))
                    xyxy.append(conf)
                    coords.append([int(xyxy[0]),int(xyxy[1]),int(xyxy[2]),int(xyxy[3]),float(xyxy[4])])
                    classes.append(names[c])

            annotated_image = annotator.result()

    return annotated_image, coords, classes


def train_recogniser():

    print('Loading known faces for Retraining...')
    known_faces,known_names = [], []
    known_ids, known_id_names = [], []
    known_id_faces, known_id_faces_names = [], []

    KNOWN_FACES_DIR = 'data'

    #Training For FACE RECOGNITION
    for name in os.listdir(KNOWN_FACES_DIR):
        if "DS_Store" in name:
            continue
        for filename in os.listdir(f"{KNOWN_FACES_DIR}/{name}/face"):
            if "DS_Store" in filename:
                continue

            print(f"{KNOWN_FACES_DIR}/{name}/face/{filename}", end = "")
            image =
face_recognition.load_image_file(f"{KNOWN_FACES_DIR}/{name}/face/{filename}")


            try:
                encoding = face_recognition.face_encodings(image, model = "hog")[0]
                known_faces.append(encoding)
                known_names.append(name)
                print(" --> Trained")
            except Exception as e:
                    print(e, 'Face not detected')
                    os.remove(f"{KNOWN_FACES_DIR}/{name}/face/{filename}")




#Training For ID RECOGNITION
    for name in os.listdir(KNOWN_FACES_DIR):
        if "DS_Store" in name:
            continue
        for filename in os.listdir(f"{KNOWN_FACES_DIR}/{name}/id"):
            if "DS_Store" in filename:
                continue

            print(f"{KNOWN_FACES_DIR}/{name}/id/{filename}", end = "")
            image =
face_recognition.load_image_file(f"{KNOWN_FACES_DIR}/{name}/id/{filename}")
            try:
                id_encoding = face_recognition.face_encodings(image,[(0,image.shape[0],
image.shape[1],0)])[0]
                known_ids.append(id_encoding)
                known_id_names.append(name)
                print(" --> Trained")
            except Exception as e:
                    print('ID not detected: ', e)
                    # os.remove(f"{KNOWN_FACES_DIR}/{name}/id/{filename}")


    #Training For FACE IN ID RECOGNITION
    for name in os.listdir(KNOWN_FACES_DIR):
        if "DS_Store" in name:
            continue
        for filename in os.listdir(f"{KNOWN_FACES_DIR}/{name}/id"):
            if "DS_Store" in filename:
                continue

            print(f"{KNOWN_FACES_DIR}/{name}/id/{filename}", end = "")
            image =
face_recognition.load_image_file(f"{KNOWN_FACES_DIR}/{name}/id/{filename}")

            try:
                encoding = face_recognition.face_encodings(image, model = "hog")[0]
                known_id_faces.append(encoding)
                known_id_faces_names.append(name)
                print(" --> Trained")
            except Exception as e:
                    print(e, ': Face not detected in ID')
    # print("All Known Names:", set(known_names))
    np.save('known_faces', np.array(known_faces))
    np.save('known_names', np.array(known_names))

    np.save('known_ids', np.array(known_ids))
    np.save('known_id_names', np.array(known_id_names))




   np.save('known_id_faces', np.array(known_id_faces))
    np.save('known_id_faces_names', np.array(known_id_faces_names))


def get_main_face_coord(img, min_area = 1500):
    locations = face_recognition.face_locations(img, model=MODEL)
    main_face_coord = []
    main_face = []
    area = 0

    if len(locations) == 0:
        print("No Face Found")
        return main_face_coord, main_face, None

    for location in locations:

        # cv2.rectangle(image,(location[3],location[0]), (location[1], location[2]), (0,0,255), 2 )
        width = int(location[3] - location[1])
        height = int(location[0] - location[2])
        calc_area = width*height
        print("Face Area:", calc_area)
        if calc_area > area and calc_area > min_area:
            main_face_coord = list(location)
            area = calc_area
            print("Got Main Face", main_face_coord)

    try:
        if len(main_face_coord) > 0:
            if add_face:
                main_face_coord[3] -= 50 #start x
                main_face_coord[0] -= 100 #start Y

                main_face_coord[1] += 50 #end x
                main_face_coord[2] += 50 #end y


                if main_face_coord[3] < 0:
                    main_face_coord[3] = 0
                if main_face_coord[0] < 0:
                    main_face_coord[0] = 0
```

```
      if main_face_coord[2] > img.shape[0] :
             main_face_coord[2] = img.shape[0] #height

           if main_face_coord[1] > img.shape[1] :
             main_face_coord[1] = img.shape[1] #width


         print("Detected Main Face Coord:", main_face_coord)
         print(img.shape)

         main_face= img[main_face_coord[0]:main_face_coord[2],
main_face_coord[3]:main_face_coord[1]]
         # cv2.imshow("Main Face", main_face)
   except Exception as e:
      print("Face Crop image Failed:", e)

   return main_face_coord, main_face, locations


if add_face:
   name = input("Enter Name:")
   roll = input("Enter Hall Ticket Number:")

   final_name = name+"["+roll+"]"
   try:
      os.mkdir(f"data")
   except:
      pass
   try:
      os.mkdir(f"data/{final_name}")
      os.mkdir(f"data/{final_name}/face")
      os.mkdir(f"data/{final_name}/id")
   except:
      print("Person Already Exists, Images will be Appended.")

else:

   print("Loading Available Faces...  ", end ="")
   train_recogniser()

   known_faces = np.load('known_faces.npy', allow_pickle=True).tolist()
   known_names = np.load('known_names.npy',allow_pickle=True).tolist()

   known_ids = np.load('known_ids.npy', allow_pickle=True).tolist()
   known_id_names = np.load('known_id_names.npy',allow_pickle=True).tolist()

   known_id_faces = np.load('known_id_faces.npy', allow_pickle=True).tolist()
   known_id_faces_names = np.load('known_id_faces_names.npy',allow_pickle=True).tolist()

   print("Done")
   print("Known Names:", set(known_names))
```

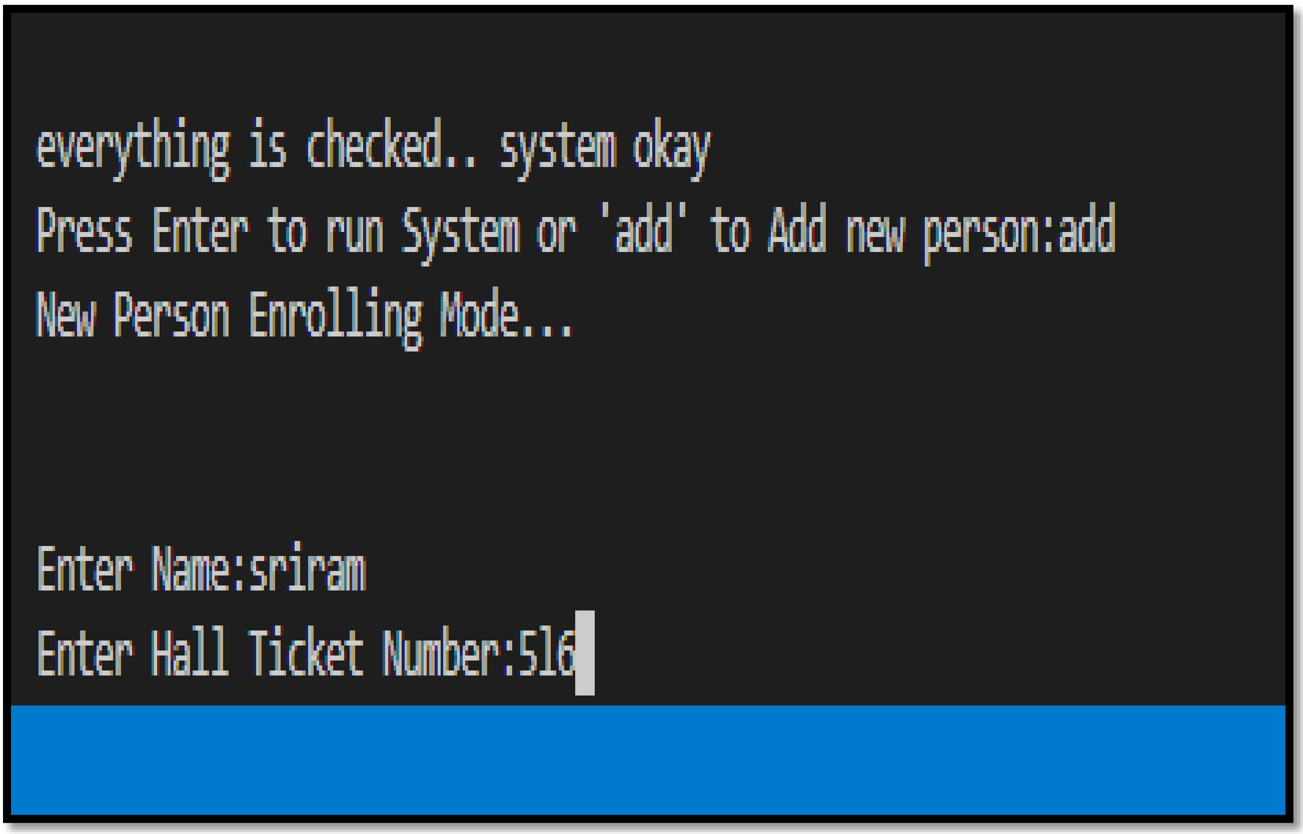
```
model_custom = get_model(weights= "best.pt")
model_person = get_model(weights= "yolov5m.pt")


cam = cv2.VideoCapture(0
) #Make 0 as 1 for External Camera
for i in range(30):
    _, img = cam.read()
    cv2.waitKey(20)

count =0
noidcount = 0
while True:

    id_name, id_face_name, id_face_face_name, face_name = "","","",""
    _, img = cam.read()
    if _:
        count += 1
        if add_face and count > 200: #Give limited time to add face and ID
            break
        cv2.imwrite("live.png", img)
        image, custom_coords, custom_classes = detect(model_custom, source = f"live.png")
        if not add_face:
            image2, person_coords, person_classes = detect(model_person, source = f"live.png")

        final_class = []
        final_coords = []

        for cc in custom_classes:
            final_class.append(cc)
        if not add_face:
            for pc in person_classes:
                final_class.append(pc)

        for ctc in custom_coords:
            final_coords.append(ctc)
        if not add_face:
            for psc in person_coords:
                final_coords.append(psc)


        #DTECT ID CARD
        if 'id card' not in final_class:
            print("ID Card Not Found")
        for i in range(len(final_class)):
            if final_class[i] == 'id card':
                # print("GOT:", final_class[i])
                y_offset = 50
                id_image = image[final_coords[i][1]+y_offset:final_coords[i][3],
final_coords[i][0]:final_coords[i][2]]
```

```
        if add_face: #Save ID
            id_face_coord, id_face, id_face_locations = get_main_face_coord(id_image)
            if id_face_locations is not None :
                for face_location in id_face_locations:
                    if not add_face:

cv2.rectangle(image,(face_location[3]+final_coords[i][0],face_location[0]+final_coords[i][1]+y
_offset),
                    (face_location[1]+final_coords[i][0],
face_location[2]+final_coords[i][1]+y_offset),
                    (0,0,255), 3 )

                print("Saved ID")
                cv2.putText(image, "ID Saved",(100,50),cv2.FONT_HERSHEY_PLAIN, 2,
(255,100,100), 2)
                cv2.imwrite(f"data/{final_name}/id/{str(time.time())[4:-
7]}.png".replace("..","."),id_image)
            else:
                print("Make Sure Face is visible in ID")
                cv2.putText(image, "FACE NOT VISIBLE IN
ID",(100,50),cv2.FONT_HERSHEY_PLAIN, 2, (100,0,250), 2)

        else:
            #Compare ID Cards
            check_id_encodings = []

check_id_encodings.append(face_recognition.face_encodings(id_image,[(0,id_image.shape[0],
id_image.shape[1],0)]))
            check_id_encodings = np.array(check_id_encodings)


            for id_encoding in check_id_encodings:
                id_results = face_recognition.compare_faces(known_ids, id_encoding,
ID_TOLERANCE)

            if True in id_results:
                id_match = known_id_names[id_results.index(True)]
                print("Found ID Match: ", id_match)
                id_name = id_match


            #Compare Face in ID Cards
            id_face_coord, id_face, id_face_locations = get_main_face_coord(id_image)
            id_face_encodings = face_recognition.face_encodings(id_image, id_face_locations)
            if len(id_face_encodings) == 0:
                print("No Face Visible in ID.")

            if id_face_locations is not None and not add_face:
                for face_location in id_face_locations:

cv2.rectangle(image,(face_location[3]+final_coords[i][0],face_location[0]+final_coords[i][1]+y
_offset),
                    (face_location[1]+final_coords[i][0],
face_location[2]+final_coords[i][1]+y_offset),
                              (0,0,255), 3 )

                          for id_face_encoding, id_face_location in zip(id_face_encodings,
id_face_locations):
                              id_face_results = face_recognition.compare_faces(known_faces,
id_face_encoding, ID_FACE_TOLERANCE)

                              if True in id_face_results:
                                  id_face_match = known_names[id_face_results.index(True)]
                                  print("Found ID Face Match1: ", id_face_match)
                                  id_face_name = id_face_match



if id_face_locations is not None and not add_face:
                          for id_face_encoding, id_face_location in zip(id_face_encodings,
id_face_locations):
                              id_face_results = face_recognition.compare_faces(known_id_faces,
id_face_encoding, ID_FACE_TOLERANCE)

                              if True in id_face_results:
                                  id_face_match = known_id_faces_names[id_face_results.index(True)]
                                  print("Found ID Face Match2: ", id_face_match)
                                  id_face_face_name = id_face_match

                 # cv2.imshow("ID Card", id_image)



 #Detect Main Face for Face Detection

main_face_coord, main_face, all_face_locations = get_main_face_coord(img, min_area=20000)

        if len(main_face_coord)>0:
            if add_face:
                print("Saved Face")
                cv2.putText(image, "Face Saved",(100,100),cv2.FONT_HERSHEY_PLAIN, 2,
(200,100,0), 2)
                cv2.imwrite(f"data/{final_name}/face/{str(time.time())[4:-7]}.png",main_face)

            #RECOGNISE ALL FACES
            encodings = face_recognition.face_encodings(img, all_face_locations)
            if all_face_locations is not None and not add_face:
                for face_encoding, face_location in zip(encodings, all_face_locations):
                    face_results = face_recognition.compare_faces(known_faces, face_encoding,
TOLERANCE)

                    if True in face_results:
                        face_match = known_names[face_results.index(True)]
                        print("Found Face Match: ", face_match)
            # RECOGNISE MAIN FACE
            main_face_encodings = []

main_face_encodings.append(face_recognition.face_encodings(img,[(main_face_coord)]))
            main_face_encodings = np.array(main_face_encodings)
            if all_face_locations is not None and not add_face:
                for face_encoding, face_location in zip(main_face_encodings, [(main_face_coord)]):
                    face_results = face_recognition.compare_faces(known_faces, face_encoding,
TOLERANCE)
                    if True in face_results:
                        face_match = known_names[face_results.index(True)]
                        print("Main Face Match: ", face_match)
                        face_name = face_match




for face_location in all_face_locations:
                cv2.rectangle(image,(face_location[3],face_location[0]), (face_location[1],
face_location[2]), (0,0,255), 3 )
            cv2.rectangle(image,(main_face_coord[3],main_face_coord[0]), (main_face_coord[1],
main_face_coord[2]), (0,255,0), 3 )

        #PLOT ALL YOLO DETECTIONS
        for p in range(len(final_class)):
            if final_class[p] == "person" and not add_face:
                print("Plotting Person")
                color = (200, 0, 0) #BLUE
                start_point = (int(final_coords[p][0]), int(final_coords[p][1]))
                end_point = (int(final_coords[p][2]), int(final_coords[p][3]))
                image = cv2.rectangle(image, start_point, end_point, color, 2)

        for id in range(len(final_class)):
            if final_class[id] == "id card":
                print("Plotting ID Card")
                color = (150, 150, 255) #Pink
                start_point = (int(final_coords[id][0]), int(final_coords[id][1]))
                end_point = (int(final_coords[id][2]), int(final_coords[id][3]))
                image = cv2.rectangle(image, start_point, end_point, color, 2)

        for id in range(len(final_class)):
            if final_class[id] == "id card tag" and not add_face:
                print("Plotting ID Card Tags")
                color = (1, 100, 1) #Dark Green
                start_point = (int(final_coords[id][0]), int(final_coords[id][1]))
                end_point = (int(final_coords[id][2]), int(final_coords[id][3]))
                image = cv2.rectangle(image, start_point, end_point, color, 2)

        for id in range(len(final_class)):
            if final_class[id] == "tag" and not add_face:
                print("Plotting Tags")
                color = (1, 100, 100) #Dark Yello2
            start_point = (int(final_coords[id][0]), int(final_coords[id][1]))
            end_point = (int(final_coords[id][2]), int(final_coords[id][3]))
            image = cv2.rectangle(image, start_point, end_point, color, 2)


        if (len(face_name) > 0) and ((face_name == id_face_face_name) or (face_name ==
id_face_name) or (face_name == id_name)):
            print("Welcome:", face_name)
            cv2.putText(image, f"WELCOME
{face_name}",(200,80),cv2.FONT_HERSHEY_PLAIN, 2, (100,255,100), 3)
            try:
                f1 = open("attendance.csv", 'x')
                f1.write("Date Time,Name RollNo\n")
            except:
                f1 = open("attendance.csv", 'a')




            f1.write(f"{str(datetime.now())[:-7]},{face_name}\n")
            f1.close()



        if 'person' in final_class and not add_face:
            if 'id card' not in final_class and 'id card tag' not in final_class:
                noidcount += 1
                if noidcount > 3:
                    print("Person without ID Card Found")
                    cv2.putText(image, f"Person without ID
Card",(200,80),cv2.FONT_HERSHEY_PLAIN, 2, (0,5,250), 3)

                    imgname = "voilation/"+"live"+str(int(time.time()))+".png"
                    cv2.imwrite(imgname, image)
                    sendmail(imgname)
            else:
                noidcount = 0
        else:
                noidcount = 0


        cv2.imshow("Output", image)
        cv2.waitKey(100)
        print("#"*20)
```

# 5.RESULTS

# 5.RESULTS

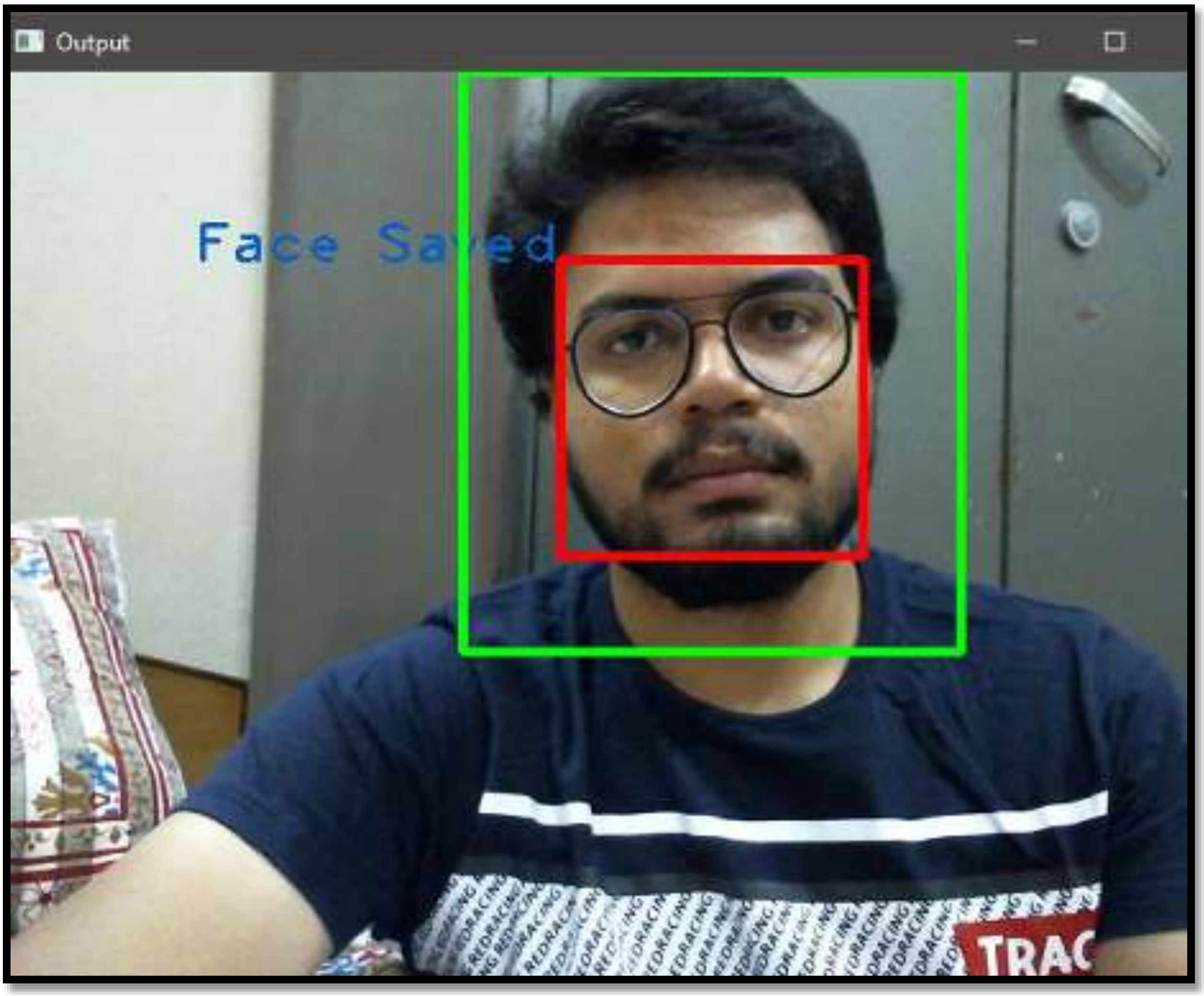
## 5.1 ENROLLING THE PERSON



```
everything is checked.. system okay
Press Enter to run System or 'add' to Add new person:add
New Person Enrolling Mode...


Enter Name:sriram
Enter Hall Ticket Number:516
```

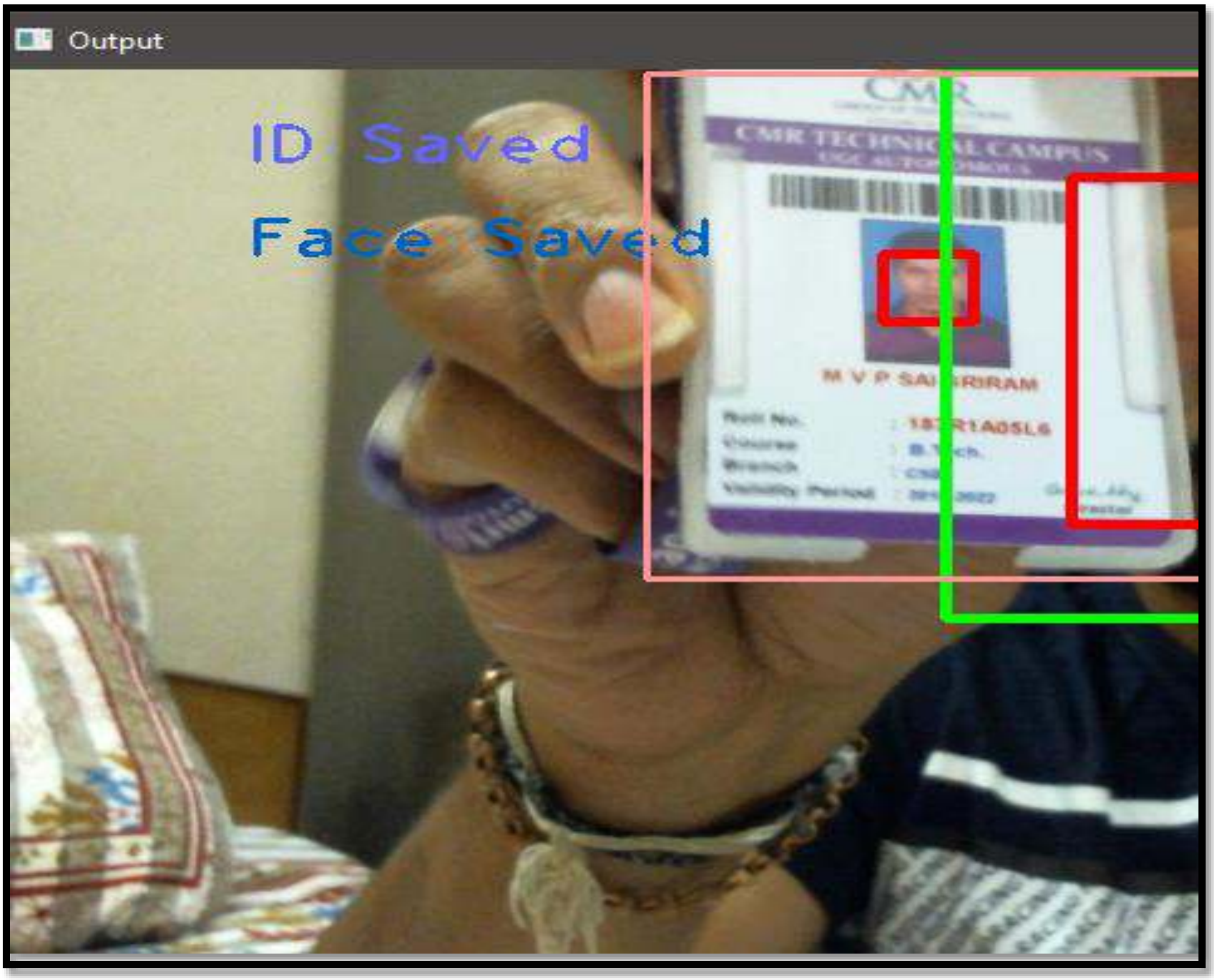
Screenshot 5.1 Home Page

## 5.2 SAVING THE FACE



Screenshot 5.2 Saving the person

## 5.3 SAVING THE ID CARD

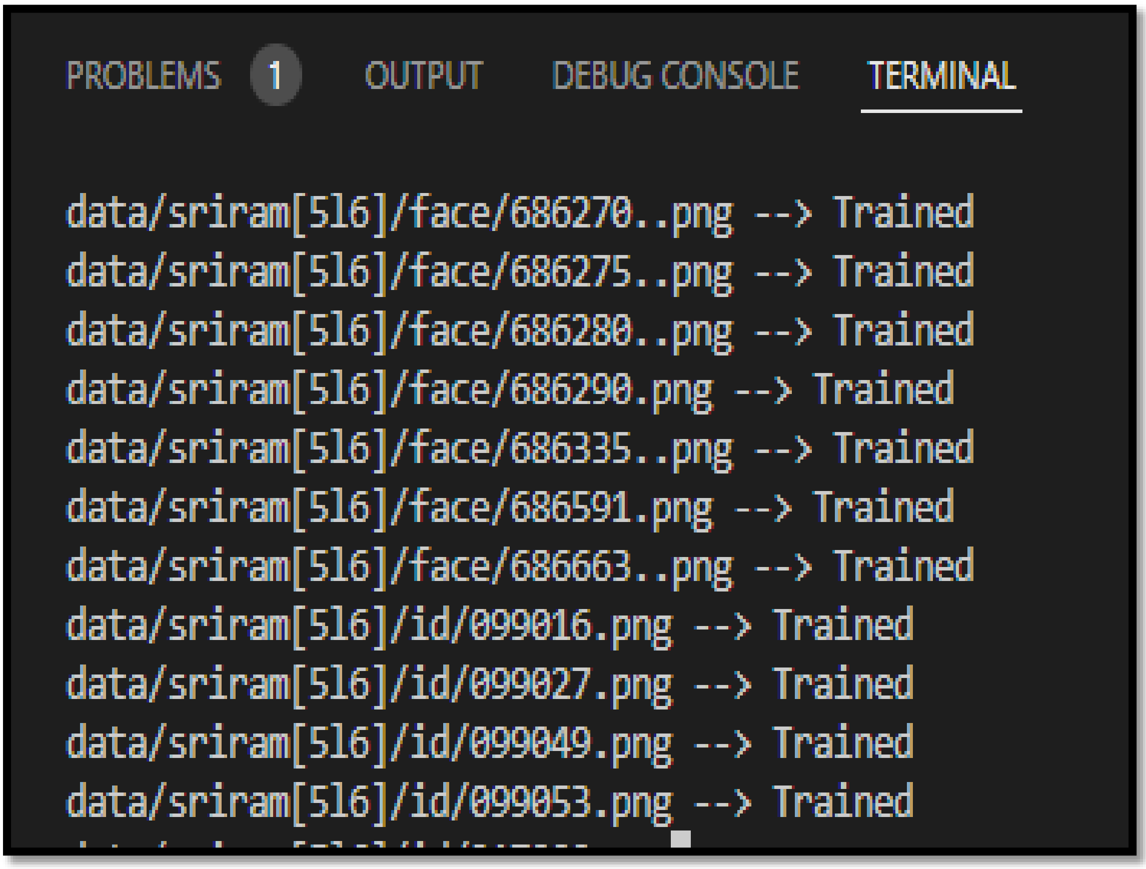


ID Saved

Face Saved

Screenshot 5.3 Saving the ID Card

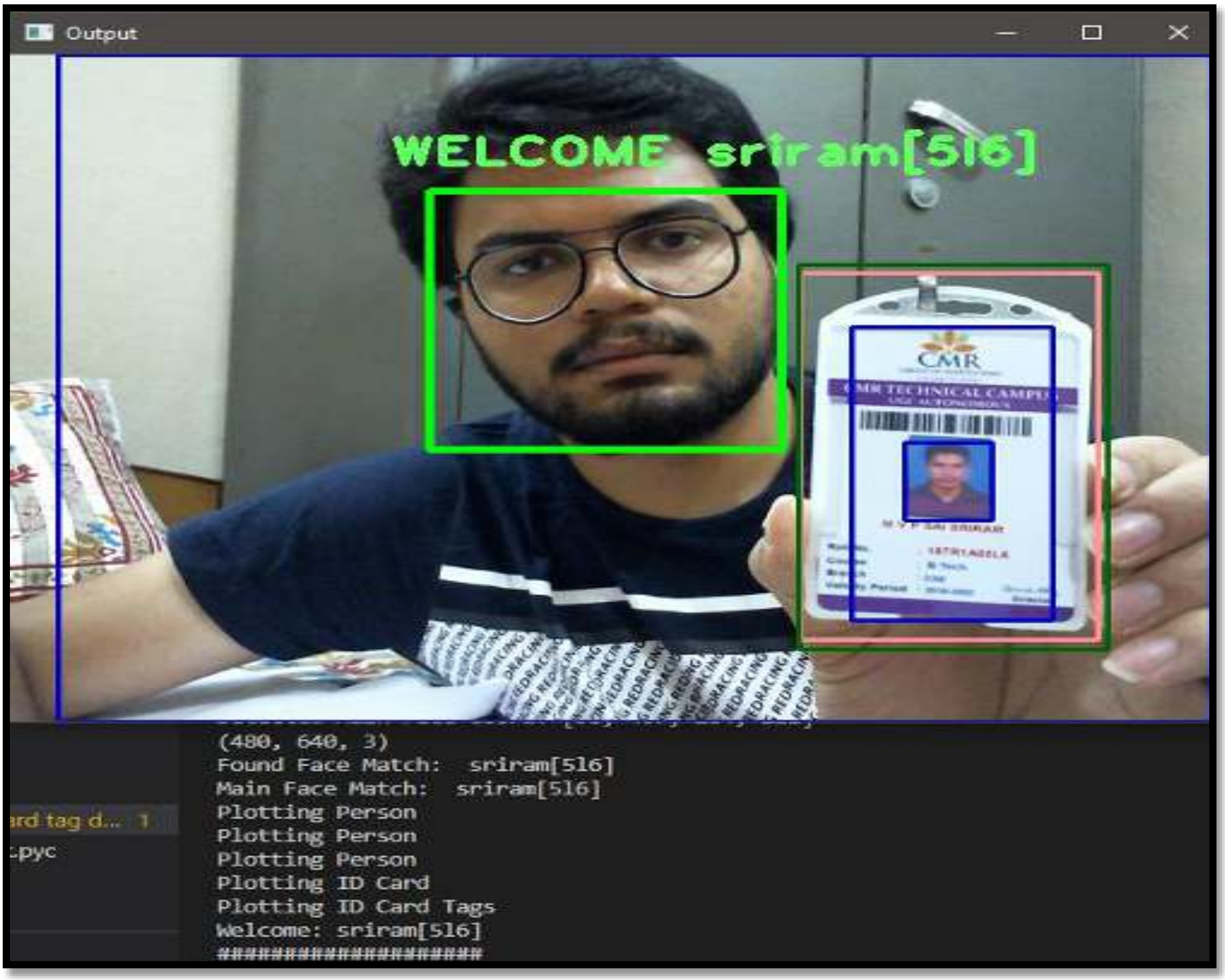
## 5.4 TRAINING THE CAPTURED IMAGES



```
data/sriram[516]/face/686270..png --> Trained
data/sriram[516]/face/686275..png --> Trained
data/sriram[516]/face/686280..png --> Trained
data/sriram[516]/face/686290.png --> Trained
data/sriram[516]/face/686335..png --> Trained
data/sriram[516]/face/686591.png --> Trained
data/sriram[516]/face/686663..png --> Trained
data/sriram[516]/id/099016.png --> Trained
data/sriram[516]/id/099027.png --> Trained
data/sriram[516]/id/099049.png --> Trained
data/sriram[516]/id/099053.png --> Trained
```

Screenshot 5.4 Training the Captured Images

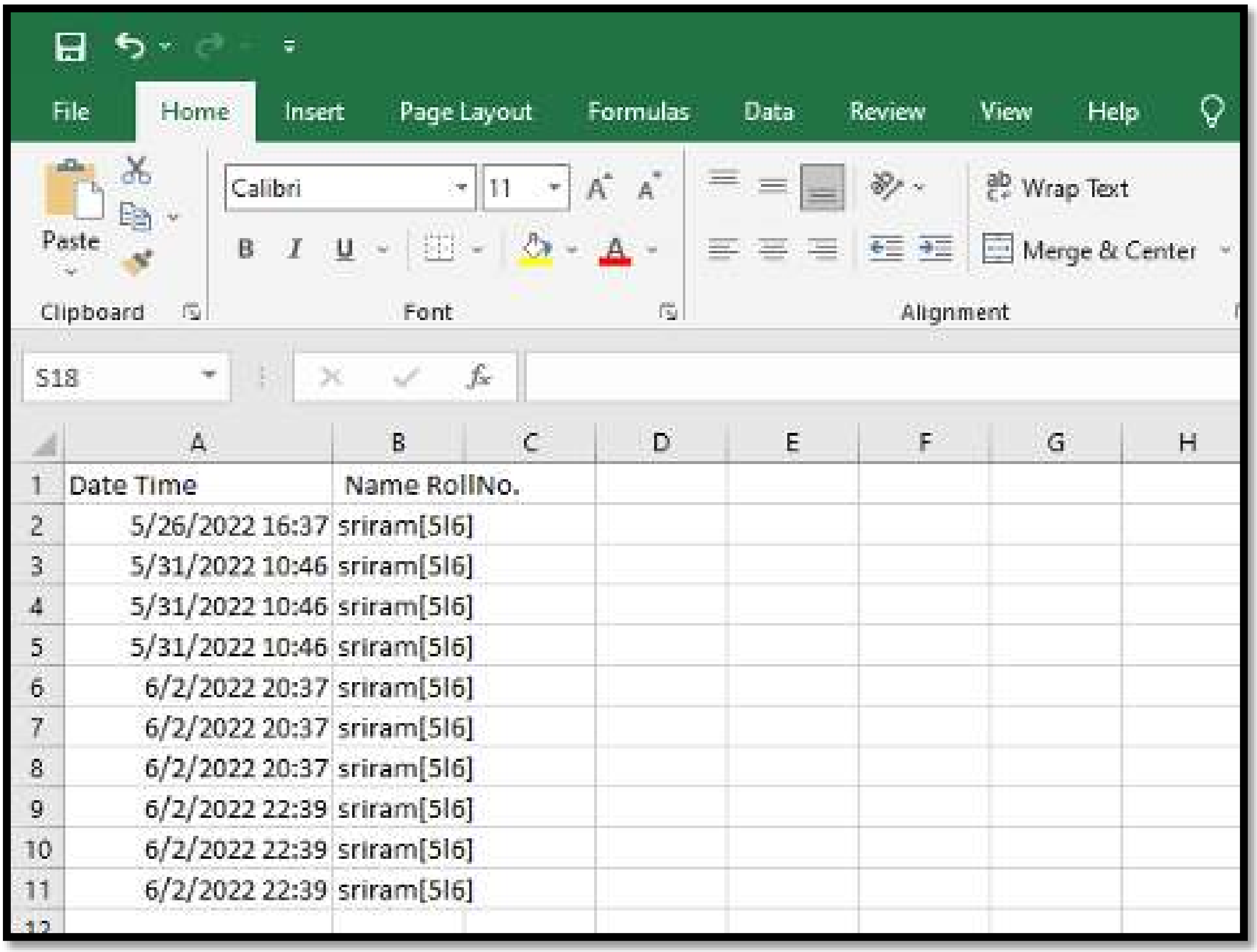
## 5.5 PERSON DETECTED WITH THEIR OWN ID CARD



```
(480, 640, 3)
Found Face Match:  sriram[516]
Main Face Match:  sriram[516]
Plotting Person
Plotting Person
Plotting Person
Plotting ID Card
Plotting ID Card Tags
Welcome: sriram[516]
####################
```

Screenshot 5.5 Person detected with their own id card

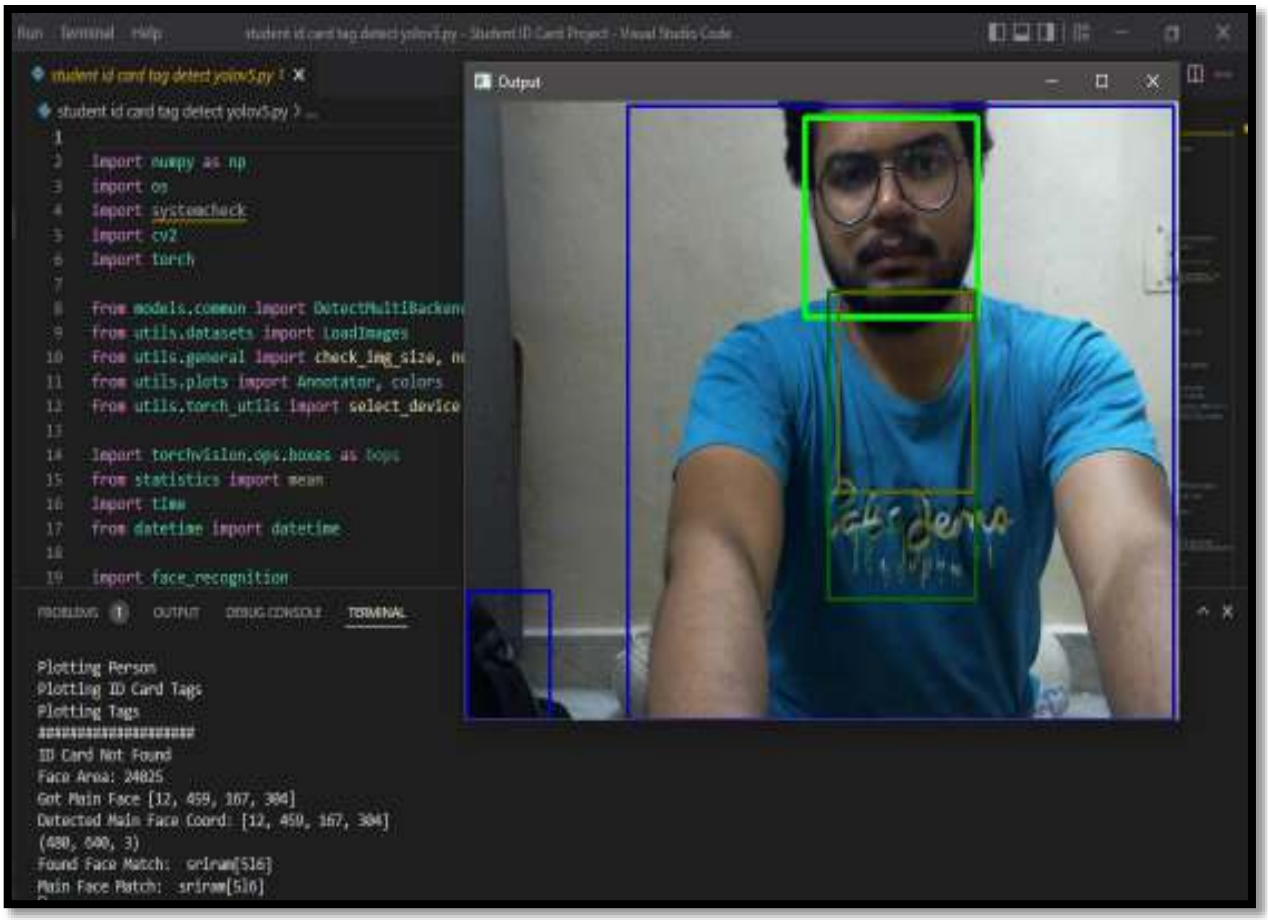
## 5.6 MARKING ATTENDANCE



|  | A | B |
|---|---|---|
| 1 | Date Time | Name RollNo. |
| 2 | 5/26/2022 16:37 | sriram[5l6] |
| 3 | 5/31/2022 10:46 | sriram[5l6] |
| 4 | 5/31/2022 10:46 | sriram[5l6] |
| 5 | 5/31/2022 10:46 | sriram[5l6] |
| 6 | 6/2/2022 20:37 | sriram[5l6] |
| 7 | 6/2/2022 20:37 | sriram[5l6] |
| 8 | 6/2/2022 20:37 | sriram[5l6] |
| 9 | 6/2/2022 22:39 | sriram[5l6] |
| 10 | 6/2/2022 22:39 | sriram[5l6] |
| 11 | 6/2/2022 22:39 | sriram[5l6] |

Screenshot 5.6 Marking Attendance

## 5.7 PERSON IS IDENTIFIED BY SYSTEM BUT ID CARD IS NOT DETECTED

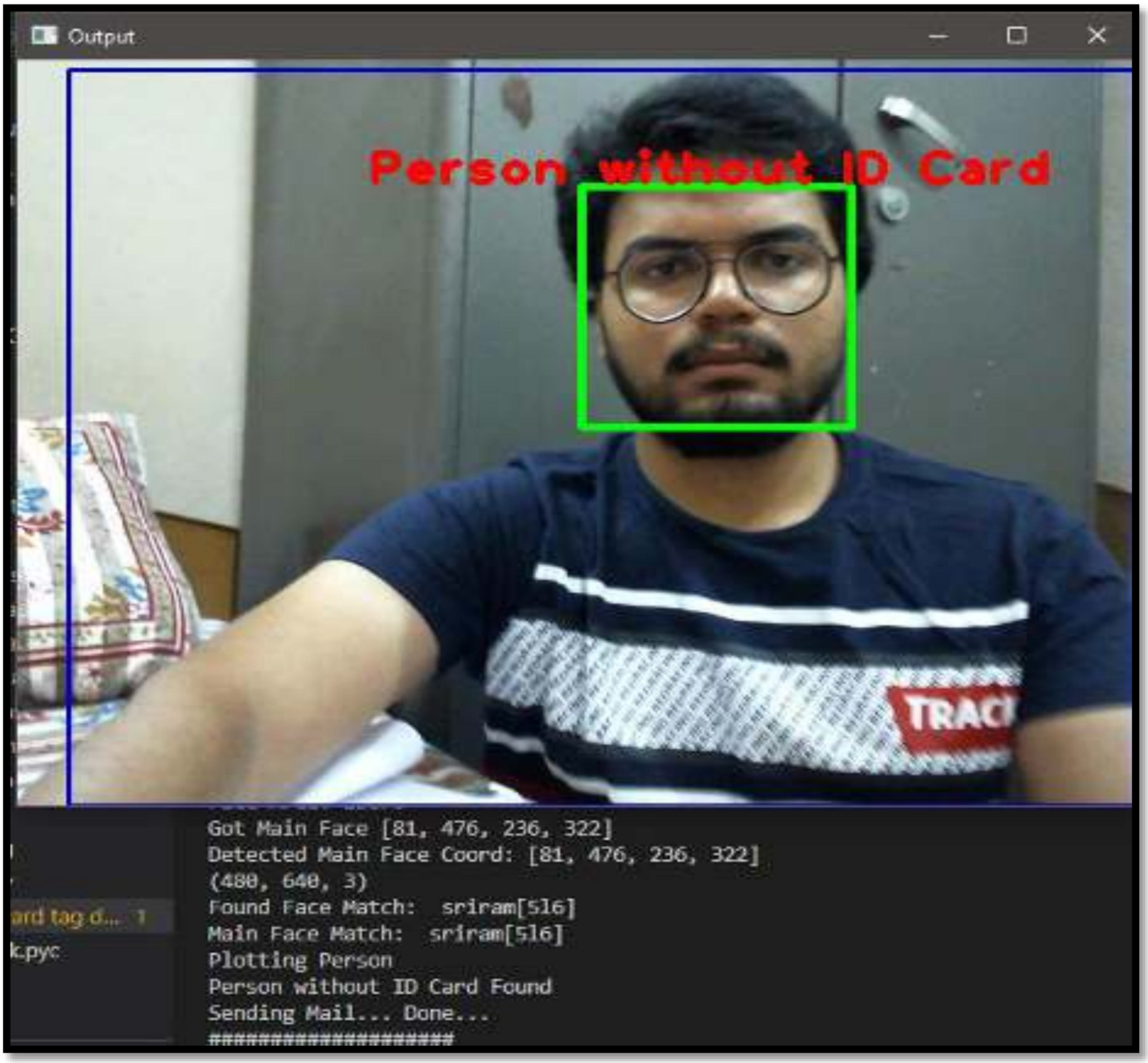

Run
Terminal
Help
student id card tag detect yolov5.py - Student ID Card Project - Visual Studio Code
student id card tag detect yolov5.py
student id card tag detect yolov5.py > ...
Output
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL

```

import numpy as np
import os
import systemcheck
import cv2
import torch

from models.common import DetectMultiBacken
from utils.datasets import LoadImages
from utils.general import check_img_size, n
from utils.plots import Annotator, colors
from utils.torch_utils import select_device

import torchvision.ops.boxes as bops
from statistics import mean
import time
from datetime import datetime

import face_recognition
```

```
Plotting Person
Plotting ID Card Tags
Plotting Tags
####################
ID Card Not Found
Face Area: 24025
Got Main Face [12, 459, 167, 304]
Detected Main Face Coord: [12, 459, 167, 304]
(480, 640, 3)
Found Face Match:  sriram[516]
Main Face Match:  sriram[516]
```

Screenshot 5.7 Person Is Identified by System but ID Card Is Not Detected

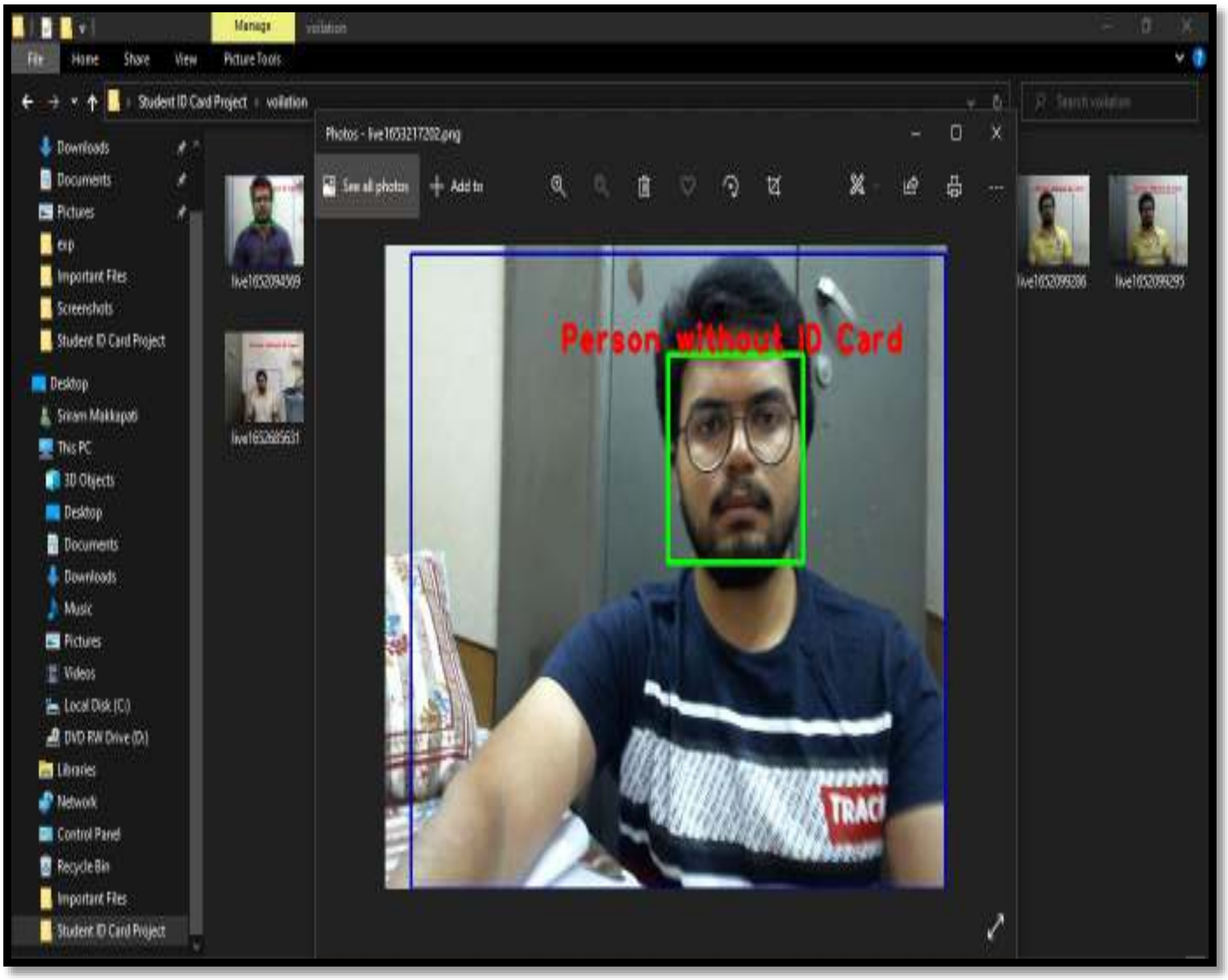
## 5.8 VIOLATION DETECTED



```
Got Main Face [81, 476, 236, 322]
Detected Main Face Coord: [81, 476, 236, 322]
(480, 640, 3)
Found Face Match:  sriram[516]
Main Face Match:  sriram[516]
Plotting Person
Person without ID Card Found
Sending Mail... Done...
####################
```

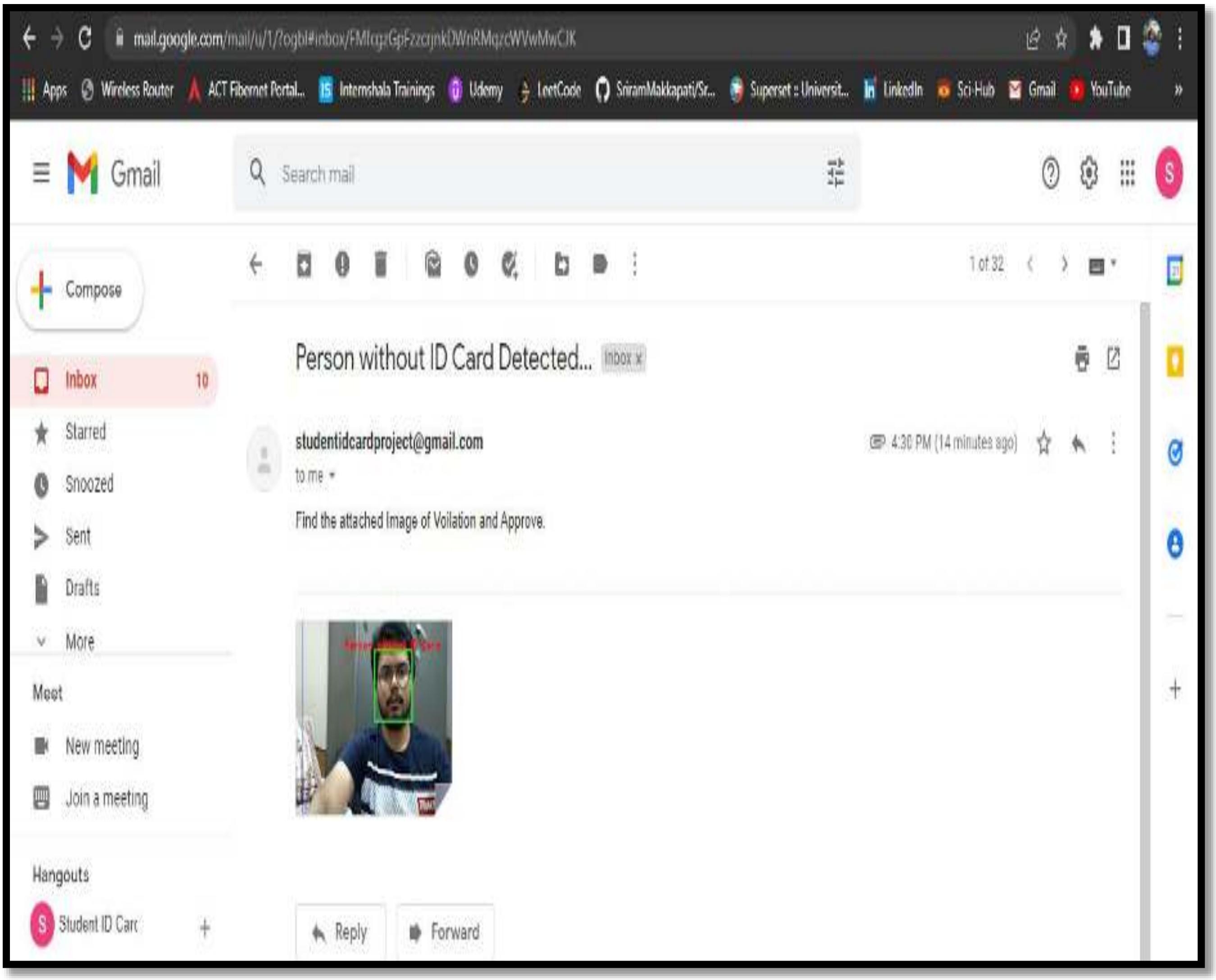
Screenshot 5.8 Violation Detected

## 5.9 SAVING THE DATABASE OF A PERSON NOT WEARING ID CARD



Screenshot 5.9 Saving The Database Of A Person Not Wearing Id Card

## 5.10 SENDING MAIL



mail.google.com/mail/u/1/?ogbl#inbox/FMfcgzGpFzzcrjnkDWnRMqzcWVwMwCJK

Person without ID Card Detected... Inbox

studentidcardproject@gmail.com

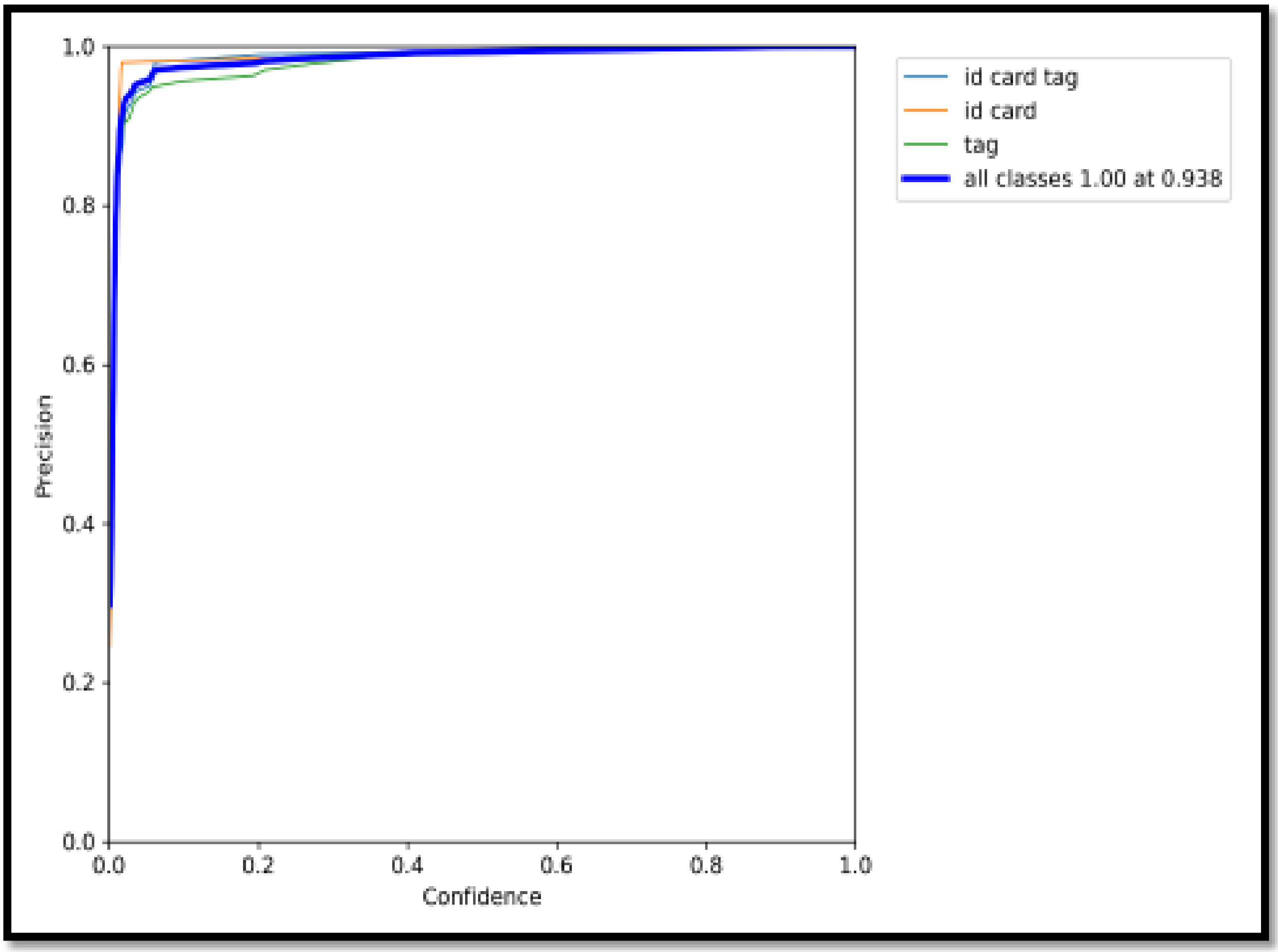
to me

4:30 PM (14 minutes ago)

Find the attached Image of Voilation and Approve.

Screenshot 5.10 Sending the Mail

## 5.11 ACCURACY OF THE MODEL



Screenshot 5.11 Accuracy of The Model

# 6.TESTING

# 6. TESTING

## 6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

## 6.2 TYPES OF TESTING

### 6.2.1 UNIT TESTING

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

### 6.2.2 INTEGRATION TESTING

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

### 6.2.3 FUNCTIONAL TESTING

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

## 6.3 TEST CASES

| Test Case ID | Test Case Name | Purpose | Input | Output |
|---|---|---|---|---|
| 01 | Enrolling person details | To enroll the details like name & roll no. | Name & roll no. of person are enrolled | Enrolled successfully |
| 02 | Images of capturing face & ID of person | To save the data of face& ID of that person | Face and id are shown to the camera | Images are captured successfully |
| 03 | Detection of the person with his/her own ID card | To detect if the person is wearing their own ID card | Showing the face & ID towards the camera simultaneously | Person is detected successfully & attendance is added |
| 04 | Person without ID card | To detect of the person is using ID card | Showing the face without Id towards the camera | Person without ID card is detected and the mail is sent successfully |

# 7.CONCLUSION

# 7. CONCLUSION AND FUTURE ENHANCEMENT

## 7.1 PROJECT CONCLUSION

Identification cards have become an important component of the corporate sector, allowing organizations to simplify students and faculty identification while also improving their integrity and security. The proposed work identifies the presence of a person wearing an ID card using tensor flow object detection API, detects and recognizes. For this, we have to train the student face and then we have to train the id card when the two images are matched then the result will be notified.

## 7.2 FUTURE ENHANCEMENT

- Can be used in Schools, Colleges, Offices etc. wherever ID Cards are issued.
- Can be used in Exam Centres.
- This model can be directly placed at the entrance of the organization so the people only with their own ID card can enter.

# 8.BIBLIOGRAPHY

# 8.BIBLIOGRAPHY

## 8.1 REFERENCES

- https://www.kaggle.com/dataset/a6d0819aeaa4c13e76eeb0c3776e1acdb81e6e5243c570bb80def4ce8040c2af
- https://www.drssica.com/
- https://nanonets.com/blog/id-card-digitization-deep-learning/
- https://www.google.com/url?sa=t&source=web&rct=j&url=https://aip.scitation.org/doi/pdf/10.1063/1.5033788&ved=2ahUKEwjWxJC0lKP4AhWEumMGHSTJAcEQFnoECB4QAQ&usg=AOvVaw1c9ua_3v6JvtGcs6H04eY1
- https://ieeexplore.ieee.org/document/9183342

## 8.2 WEBSITES

- https://www.google.com/url?sa=t&source=web&rct=j&url=https://arxiv.org/pdf/2003.12103&ved=2ahUKEwjWxJC0lKP4AhWEumMGHSTJAcEQFnoECB8QAQ&usg=AOvVaw0-_NYGvMmtbMXHzw23rad7

## 8.3 GITHUB LINK

https://github.com/SriramMakkapati/Student-ID-CARD-Project