Dharmsinh Desai University, Nadiad
Faculty of Technology
Department of Computer Engineering

B. Tech. CE Semester – VI

**Subject: Software Devlopement Practise**

# Project title: Social Distance Monitering

By :-

| sr. | Name | Roll No. | ID |
|---|---|---|---|
| 1 | Devansh P. Maru | CE-077 | 19CEUBG055 |
| 2 | Gaurav K. Mori | CE-086 | 19CEUBG019 |
| 3 | Vedant V. Panchal | CE-097 | 19CEUBS090 |

Guided By :-
Prof. Shaifali P. Malukani
Department Of Computer Engineering

**Faculty of Technology**
**Department of Computer Engineering**
**Dharmsinh Desai University**

# CERTIFICATE

This is to certify that the practical / term work carried out in the subject of

**Software Developement Practise** and recorded in this journal is the

bonafide work of

**Maru Devansh P. (CE-077) (19CEUBG055)**
**Mori Gaurav K. (CE-086) (19CEUBG019)**
**Panchal Vedant V. (CE-097) (19CEUBS090)**

**of** B.Tech semester **VI** in the branch of **Computer Engineering**

during the academic year **2021-22.**

| | |
|---|---|
| **Prof. Shaifali P. Malukani** | **Dr. C. K. Bhensdadia,** |
| **Assistant Professor,** | **Head,** |
| **Dept. of Computer Engg.,** | **Dept. of Computer Engg.,** |
| **Faculty of Technology** | **Faculty of Technology** |
| **Dharmsinh Desai University** | **Dharmsinh Desai University** |
| **Nadiad** | **Nadiad** |

**Contents:-**

## 1. Abstract :-

The ongoing COVID-19 corona virus outbreak has caused a global disaster with its deadly spreading. In the current situation, Social distance is required; therefore, social distancing is thought to be an adequate precaution (norm) against the spread of the pandemic virus. The risks of virus spread can be minimized by avoiding physical contact among people. The purpose of this project is, therefore, to provide a deep learning platform for social distance tracking using an overhead perspective.

In our system a user can a upload a particular video and check the social distance violations. If peoples are less than some distance then their detected box color is red otherwise green. There is also another type violation and that is abnormal violations and that detected color of box is yellow.At the end it display total number of people count,abnormal violations and serious violations.

## 2) Introduction :-

Social distancing associates with the measures that overcome the virus' spread, by minimizing the physical contacts of humans, such as the masses at public places (e.g., shopping malls, parks, schools, universities, airports, workplaces), evading crowd gatherings, and maintaining an adequate distance between people. Social distancing is essential, particularly for those people who are at higher risk of serious illness from COVID-19. So there is too much importance of social distance. As shown in fig(1) if person is not follow social distance then the virus can be spread into more peoples and increase day by day. But after following social distance we can reduce that rate.

The framework uses the YOLOv3 object recognition paradigm to identify humans in video sequences. The transfer learning methodology is also implemented to increase the accuracy of the model. In this way, the detection algorithm uses a pre-trained algorithm that is connected to an extra trained layer using an overhead human data set. The detection model identifies peoples using detected bounding box information. Using the Euclidean distance, the detected bounding box centroid's pairwise distances of people are determined. we used an approximation of physical distance to pixel and set a threshold. Peoples are detected using yolo coco dataset.

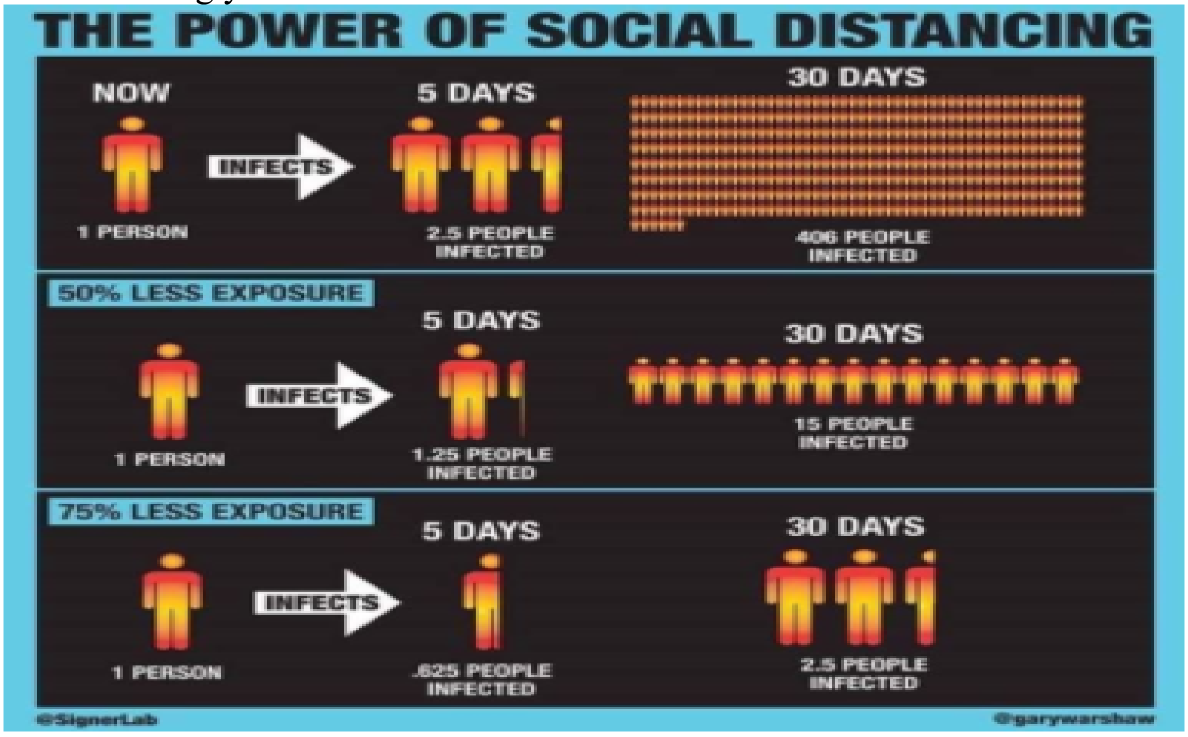


Fig (1) Importance of social distance

## Tools / Technologies Used :

- **Backend : ** Python : In python language we used deep learning for backend purpose.
- **Frontend :** Django framework : For frontend purpose we use django framework to take video input file.
- **Tools : 1)** Github
  **2)** Visual studio code

## How to run the project :

- Our system is implemented using django. So there is need to latest python version installation to your system.
- After that we can run our project in terminal using command **“Python manage.py runserver”.**
- It will give one URL. after clicking that URL you will be redirect to out project in your browser.And final output is in your taskbar.

# 3) Software Requirement Specification:-

## Social Distance Monitoring

### 1. User Management:

#### R.1.1: Video input manage:

| | | |
|---|---|---|
| Description | : | Users are required to upload a video name and video in mp4 format. |
| Input | : | Enter video name and upload a video |
| Output | : | click the submit button and submit video into system |

#### R.1.2: Generated output manage :

| | | |
|---|---|---|
| Description | : | Users can see a generated output of video that total Violations in the output window |
| Output | : | Generated output. |

### 2. System Management:

#### R.2.1: Convert video into Frames:

| | | |
|---|---|---|
| Description | : | System converts video into multiple image frames. |
| Input | : | Video input file. |
| Output | : | Getting multiple image frames. |

#### R.2.2: Detect the person:

| | | |
|---|---|---|
| Description | : | System detect the person from images using yolov3 Model,opencv library and yolo coco dataset. |
| Input | : | Multiple image frames. |
| Output | : | Detect the person into the box with intialise the color Green. |

## R.2.3: Compute distance :

| | | |
|---|---|---|
| Description | : | System compute the distance between detected boxes Using euclidian distance algorithm. |
| Input | : | Detected multiple boxes. |
| Output | : | Get the distance between each boxes in pixels form. |

## R.2.4: Check violations :

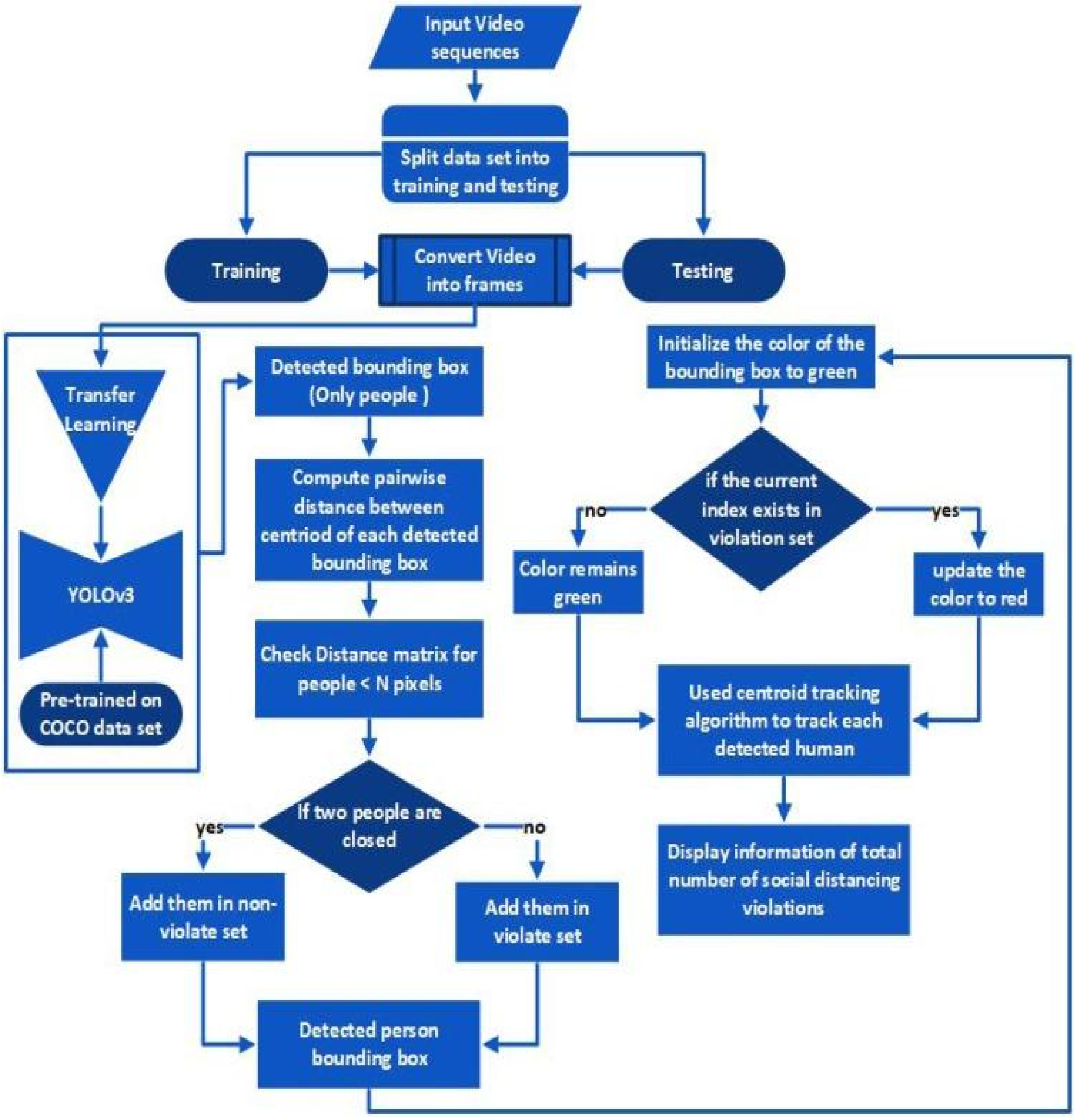
| | | |
|---|---|---|
| Description | : | System checks for violations that if distance between Two boxes is < N pixels then add them into violation Set. and change the color of frames.it also display alert If number of vioaltions are over the limit. |
| Input | : | Detected multiple boxes. |
| Output | : | Display violated frames into another color and also display number of abnormal violation , serious violations , peoples into video , safe distance and Alert message. |

## R.2.5: Display output :

| | | |
|---|---|---|
| Description | : | Display all details about violations into output window. |
| Output | : | Generated output of video. |

## 4) Diagrams:

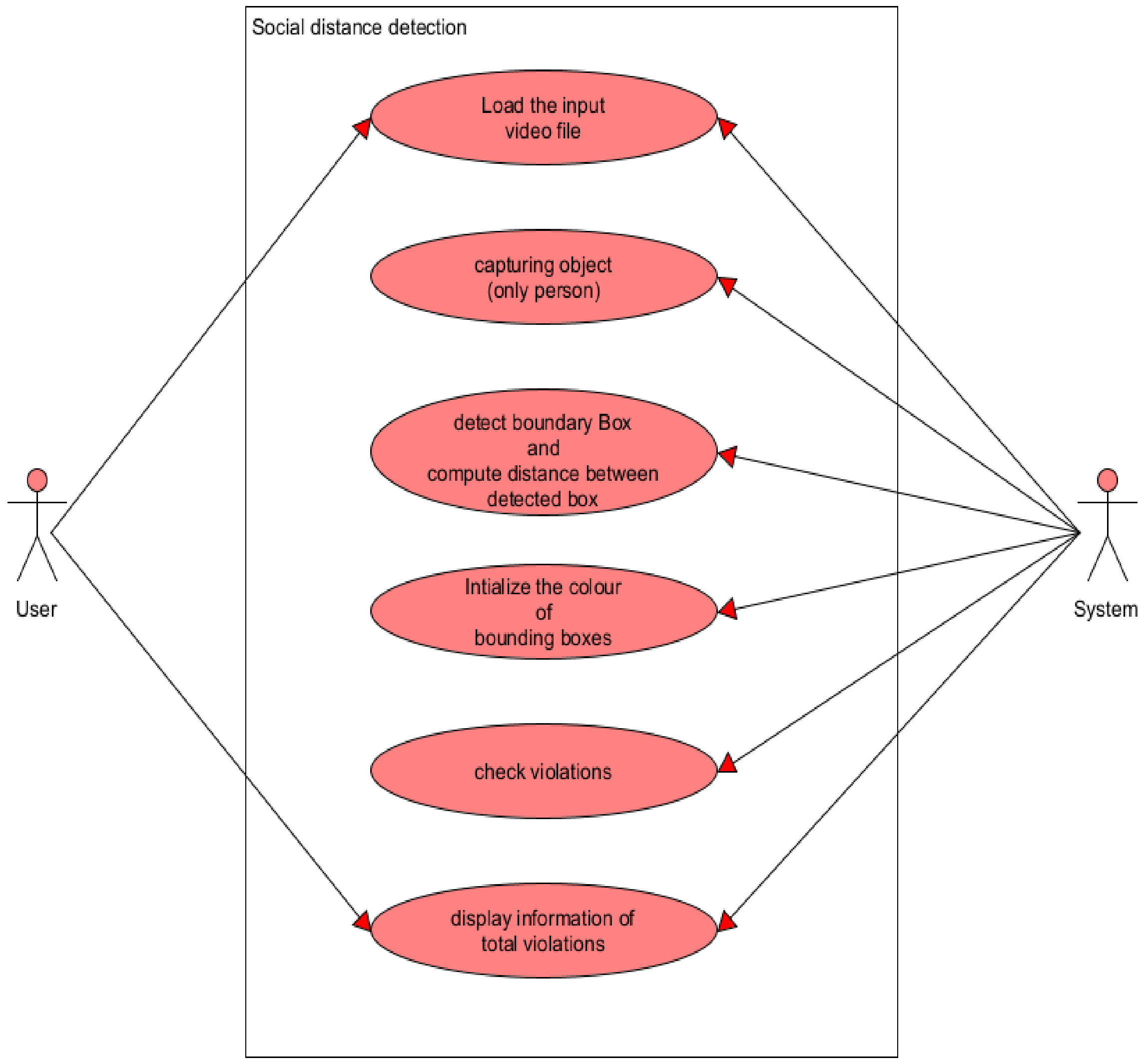
- **Work flow of all-over System:**



Input Video sequences
Split data set into training and testing
Training
Convert Video into frames
Testing
Transfer Learning
YOLOv3
Pre-trained on COCO data set
Detected bounding box (Only people )
Compute pairwise distance between centriod of each detected bounding box
Check Distance matrix for people < N pixels
If two people are closed
yes
no
Add them in non-violate set
Add them in violate set
Detected person bounding box
Initialize the color of the bounding box to green
if the current index exists in violation set
no
yes
Color remains green
update the color to red
Used centroid tracking algorithm to track each detected human
Display information of total number of social distancing violations

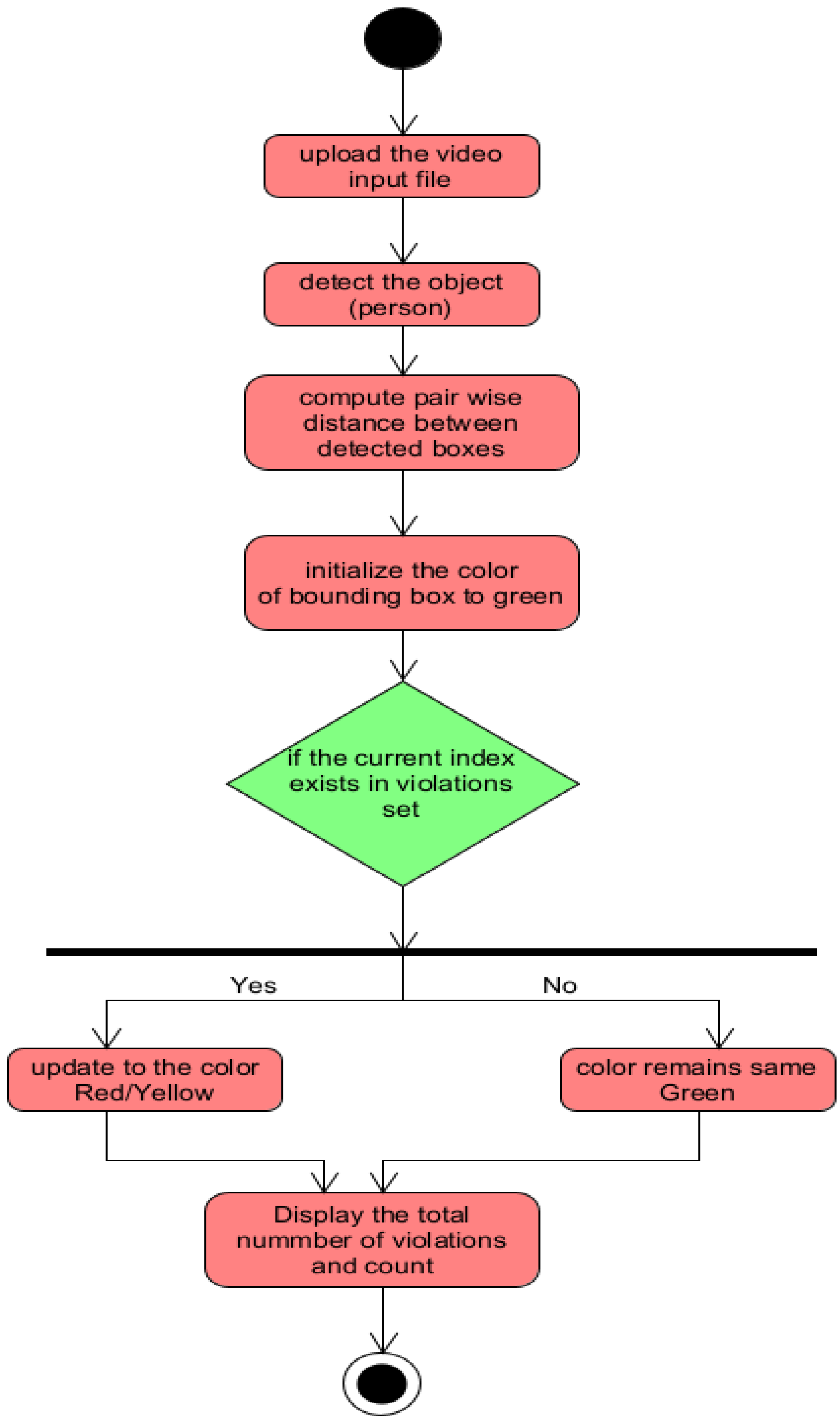
## ● Use case Diagram :

- Usecase diagram describes the highlevel functions and scope of the system.
- There are 6 main functions in our system :
  1) Take the video input file
  2) Capture the object (person).
  3) Detect boundary box
  4) Intialize the color of boundary box.
  5) Checking violations.
  6) Display total information of violations.


Social distance detection
Load the input video file
capturing object (only person)
detect boundary Box and compute distance between detected box
Intialize the colour of bounding boxes
check violations
display information of total violations
User
System

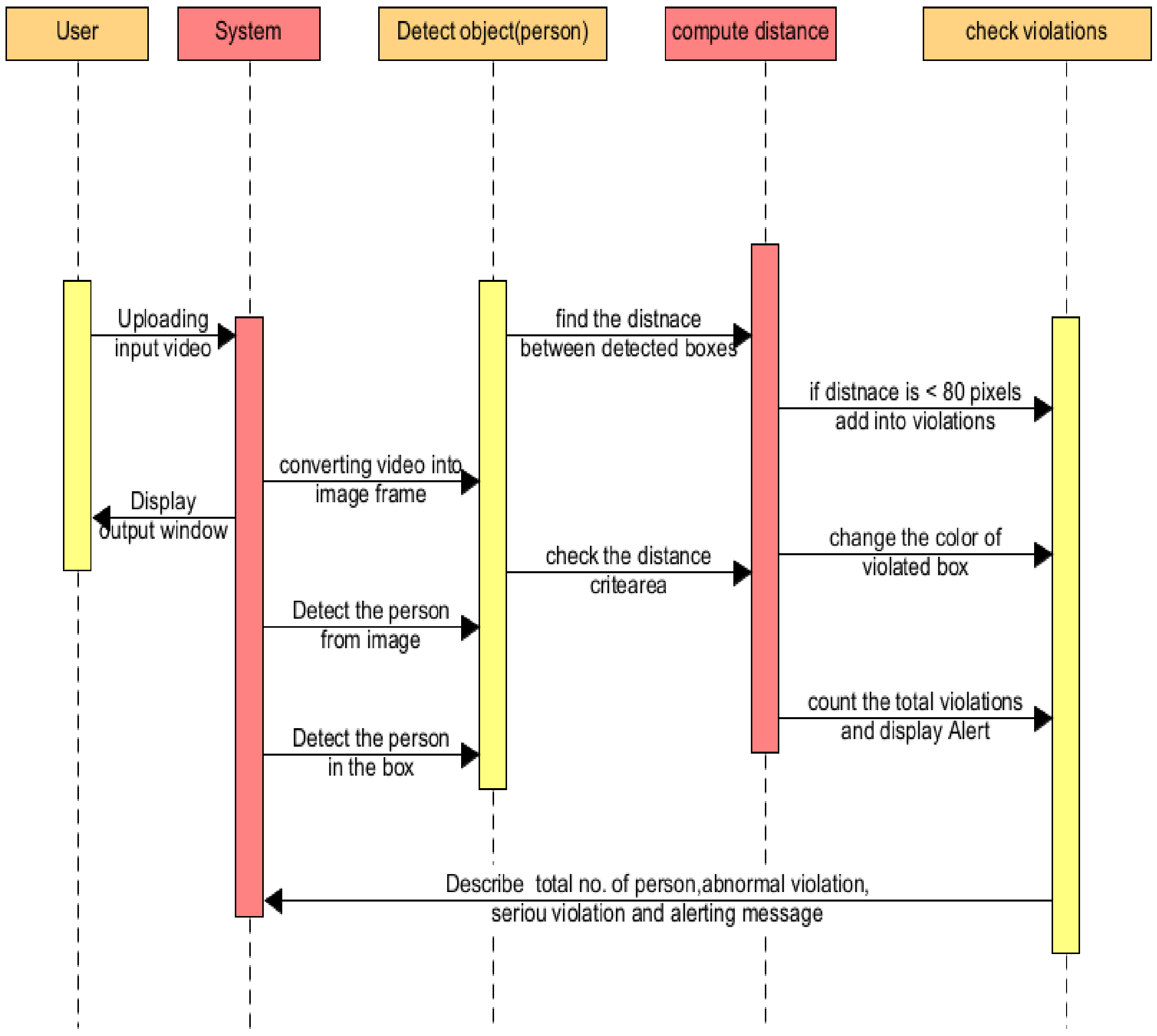
- **Activity Diagram -**

- Activity diagram shows the description of processes and particular events.


upload the video input file
detect the object (person)
compute pair wise distance between detected boxes
initialize the color of bounding box to green
if the current index exists in violations set
Yes
No
update to the color Red/Yellow
color remains same Green
Display the total nummber of violations and count

## ● Sequence Diagram -

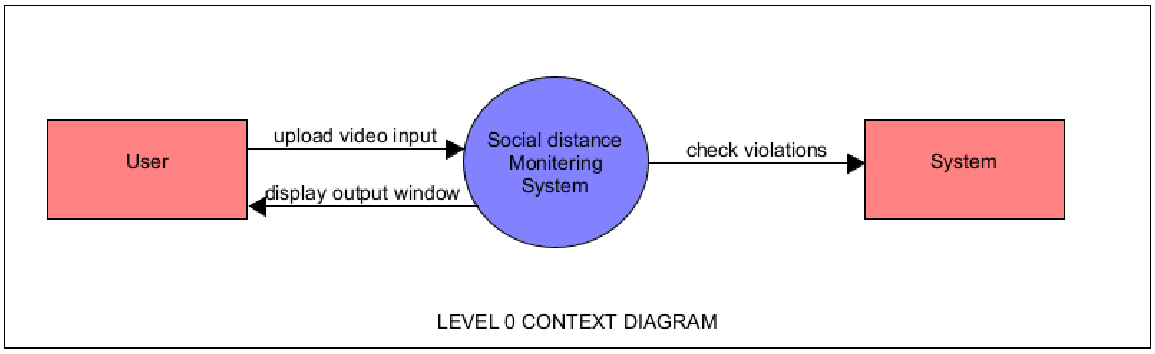
- Sequence diagram shows the logic between the objects in the system In the time order that interaction take place
- It display how and in what order the group of objects work together
- In this diagram shown all the objects and how they works in their time sequence one by one.

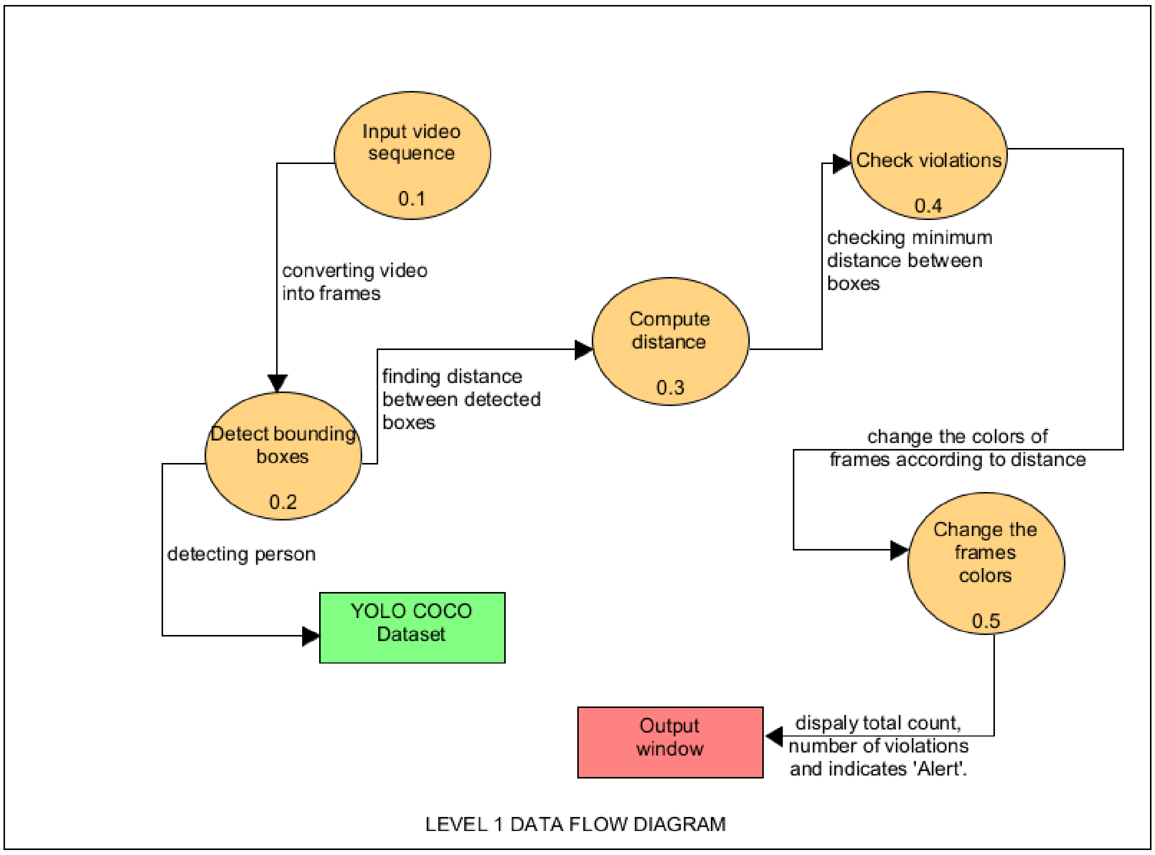


User
System
Detect object(person)
compute distance
check violations
Uploading input video
find the distnace between detected boxes
if distnace is < 80 pixels add into violations
converting video into image frame
Display output window
check the distance critearea
change the color of violated box
Detect the person from image
count the total violations and display Alert
Detect the person in the box
Describe total no. of person,abnormal violation, seriou violation and alerting message

- **Data Flow Diagram -**

- Data flow diagram shows the way information flow through a process or system
- It helps to visualize the major steps and data involved in the software System processes.




Input video sequence 0.1
converting video into frames
Detect bounding boxes 0.2
detecting person
YOLO COCO Dataset
finding distance between detected boxes
Compute distance 0.3
checking minimum distance between boxes
Check violations 0.4
change the colors of frames according to distance
Change the frames colors 0.5
dispaly total count, number of violations and indicates 'Alert'.
Output window
LEVEL 1 DATA FLOW DIAGRAM

## 5) Methods and Dataset :

The most important thing in our system is detect the object(person). For that purpose we used YOLO v3 (transfer learning) so here is the brief introduction about YOLO v3.

- **YOLOv3:**
  - YOLO (You Only Look Once) is a method / way to do object detection. It is the algorithm /strategy behind how the code is going to detect objects in the image/video.
  - It looks at the entire image only once and goes through the network once and detects objects. Hence the name. It is very fast. That's the reason it has got so popular.
  - This is very powerful algorithm because of its speed and accuracy.
  - YOLO is implemented using the OpenCV deep learning library.
  - YOLOv3 is a one of the version of YOLO.

- **How does YOLOv3 works:**
  - YOLO is a Convolutional Neural Network (CNN) for performing object detection in real-time.it allows the model to look at the whole image at test time, so its predictions are informed by the global context in the image
  - The first step to using YOLOv3 would be to decide on a specific object detection project. YOLOv3 performs real-time detections.
  - We used YOLO's COCO pretrained weights by initializing the model with model = YOLOv3().
  - Using COCO's pre-trained weights means that you can only use YOLO for object detection with any of the 80 pretrained classes that come with the COCO dataset.(in our system we used only Person).

From below image Fig(2) we can understand how it works:

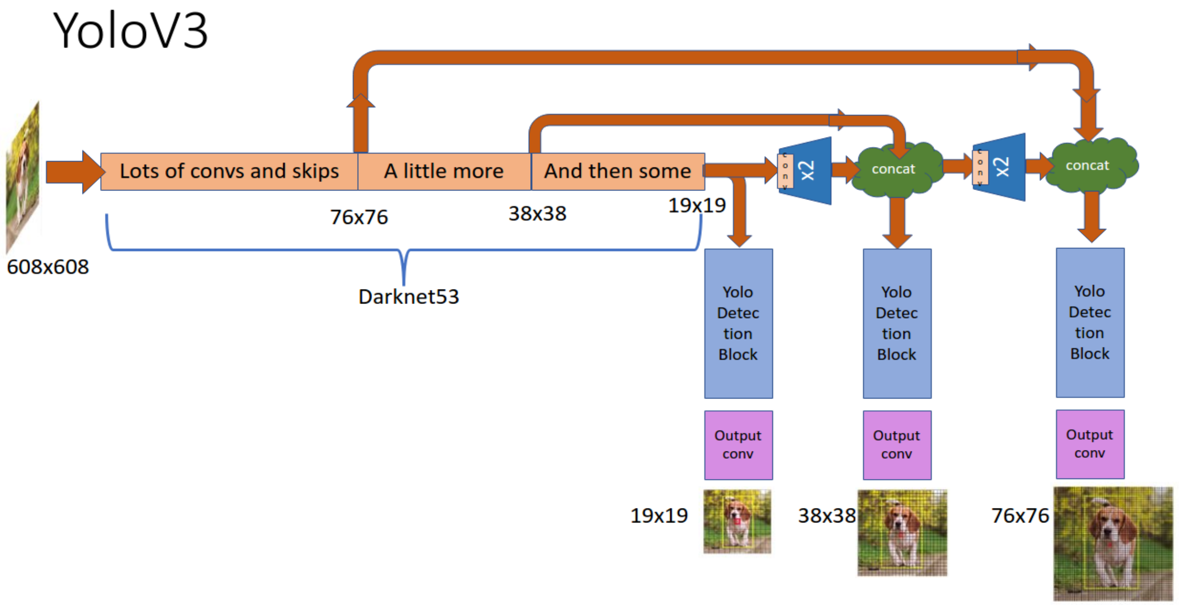


Fig (2) Working Process of YOLO V3 model

Other most import thing is converting video input to the image frame and for that purpose we used OpenCV library.

- **OpenCV Libray:**
  - OpenCV is a great tool for **image processing and performing computer vision tasks**. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more.
  - There are so many functions in this library that is used for image processing.
  - OpenCV has a bunch of pre-trained classifiers that can be used to **identify objects such as trees, number plates, faces, eyes, etc**. We can use any of these classifiers to detect the object as per our need.
  - The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms.

- **Dataset :**

  - Dataset is used for our system is YOLO-COCO pretrained dataset.
  - COCO stands for Common Objects in Context. COCO is often used to benchmark algorithms to compare the performance of real-time object detection.
  - 80 object categories, the “COCO classes”, which include “things” for which individual instances may be easily labeled (person, car, chair, etc.)
  - 91 stuff categories, where “COCO stuff” includes materials and objects with no clear boundaries (sky, street, grass, etc.) that provide significant contextual information.
  - Our system is used only “person” from 80 object categories.

## 6) Algorithms:

### Measurement of Distance:

- The Other important thing is measurement the distance between detected boxes for that we used euclidean distance between centroids.
- The set of individuals whose interval is lower than the preset minimum threshold value is considered as violation. The people who violate the condition are marked using a red box, and the remaining people are marked using a green box.
- The code for computing the centers of the boxes of are given below:
- First of all there is one detected box shown in the fig(3).

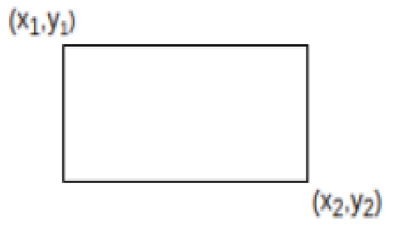


Fig (3) detected box using bounding box method

1) **Algorithm for detect only person class :**

The goal is to identify "Only Person" class map bounding boxes related to only the people.

```
#To identity "Person Only" class
x = np.where(classes==0)[0]
p=box[x]
count= len(p)
x1,y1,x2,y2 = p[0]
print(x1,y1,x2,y2)
```

## 2) Algorithm for find the center:

```
x_c = int ((x1+x2)/2)
Y_c = int (y2)
C = (x_c , y_c) = cv2.circle(image , C , 5, (255,0,0) , -1)
[#where image = image, center coordinates = C, radius = 5 , color = (255,0,0) ,
 Thickness = -1 ]

plt.figure(figsize=(20,10))
plt.imshow(image)
def mid(image,p,id):
x1,y1,x2,y2 = p[id] _ = cv2.rectangle(image, (x1, y1), (x2, y2), (0,0,255),
2)
[#where  image = image , start point = (x1,y1) , end point = (x2,y2) , color =
(0,0,255) , thickness = 2 ]
```

## 3) Algorithm for compute pair wise distance between detected people :

```
from scipy.spatial import distance
def dist(midpt,n):
d = np.zeros((n,n))
for i in range(n):
for j in range(i+1,n):
if i!=j:
dst = distance.euclidean(midpt[i], midpt[j])
d[i][j]=dist
return d
```

If the result obtained in the previous method is less than the minimum acceptable threshold value, then the box around the set of people is represented using red color. The code that defines a function to change the color of the closest people to red is given below:

```
def red(image,p,p1,p2):
unsafe = np.unique(p1+p2)
for i in unsafe:
x1,y1,x2,y2 = p[i] _ = cv2.rectangle(image, (x1, y1), (x2, y2),  (2)
return image
```

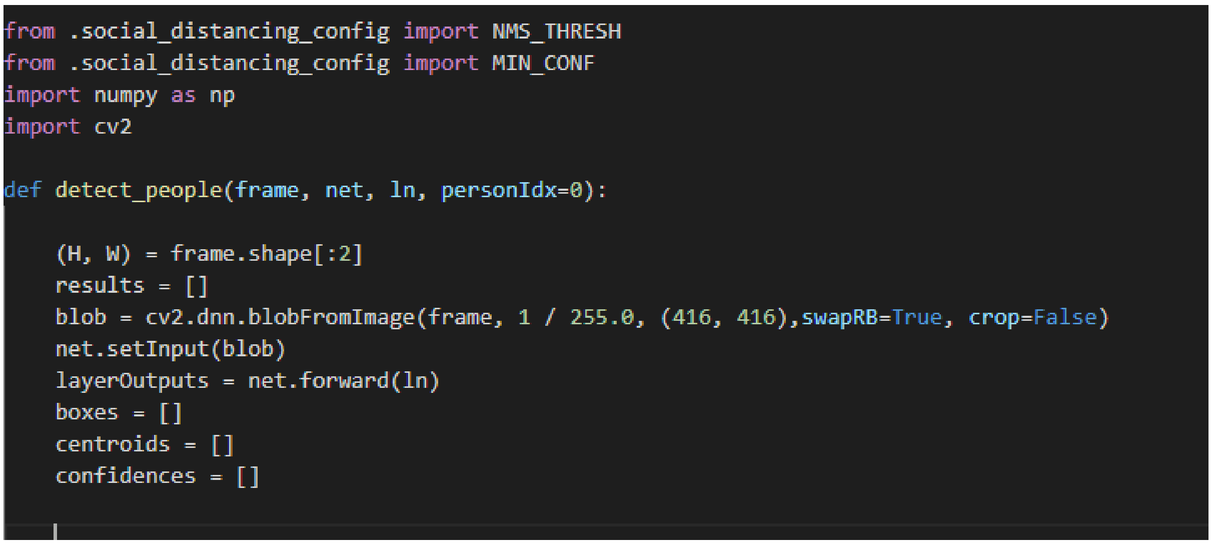
## 7) Testing :

- Manual Testing was performed in order to find and fix the bugs in the development process.
- **Testing method :-** Manual Testing

| Sr.no | Test Scenario | Expected Result | Actual Result | Status |
|---|---|---|---|---|
| TC_01 | For one image detect the person | Person detected into the rectangle box | Person detected into the rectangle box | Success |
| TC_02 | Convert video into multiple image frames | Video is converted into multiple images | Video is converted into multiple images | Success |
| TC_03 | Compute pairwise distance between Centroid of each detected boxes | Get the distance between all detected boxes | Get the distance between all detected boxes | Success |
| TC_04 | If distance is < N pixels add them into violate set | Color is changed if box is in violated set | Color is changed if box is in violated set | Success |
| TC_05 | Count the Total number of people in video and if distance is > thresold value display alert | Display total number of people and display Alert (if distance is > thresold otherwise not) | Display total number of people and display Alert (if distance is > thresold otherwise not) | Success |
| TC_06 | Display total violations , count, And alert in the output window | Show all outputs in output window | Show all outputs in output window | Success |

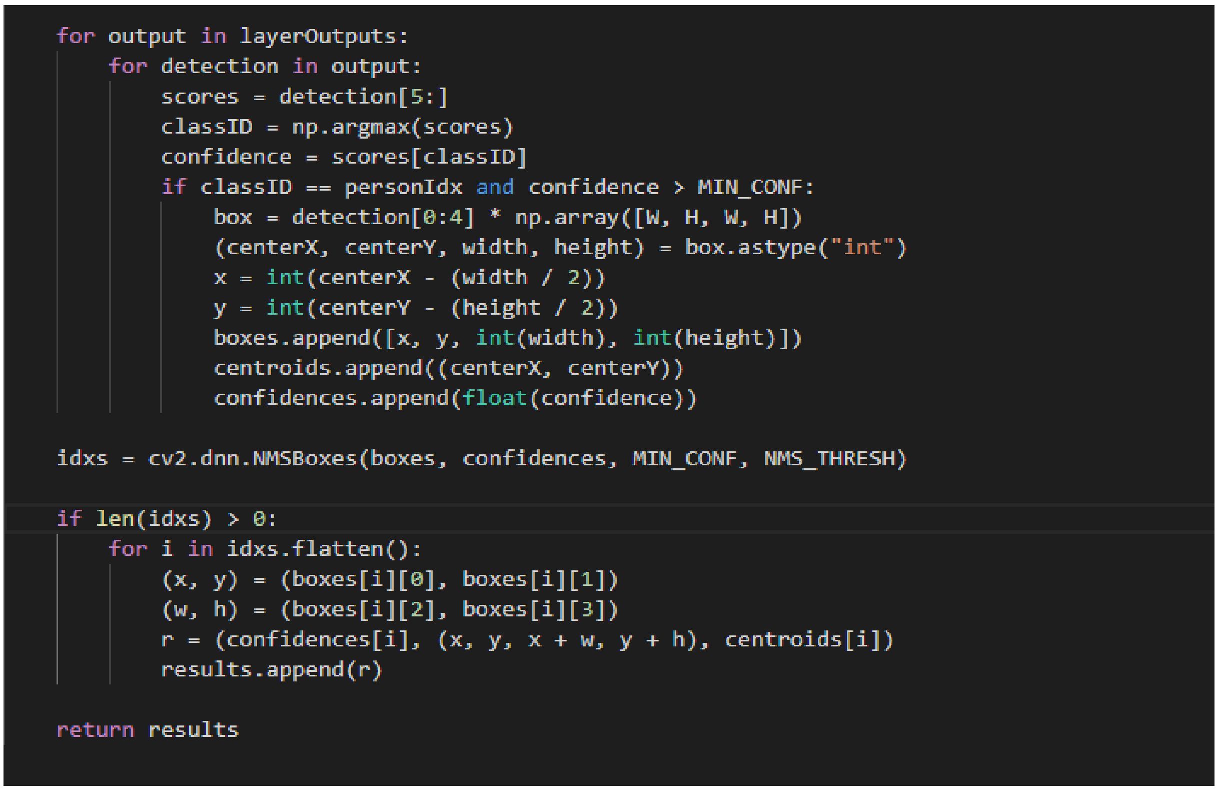
## 8) Implementation Details :

- Dimension of frames and pass the yolo object detector & intialize the detected bounding box,centroids and confidences.



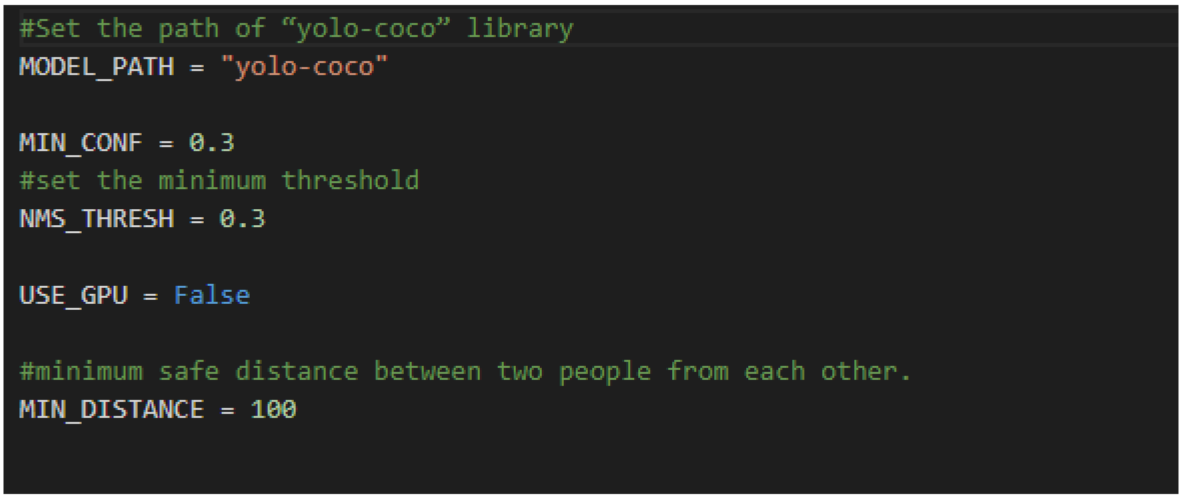
```
from .social_distancing_config import NMS_THRESH
from .social_distancing_config import MIN_CONF
import numpy as np
import cv2

def detect_people(frame, net, ln, personIdx=0):

    (H, W) = frame.shape[:2]
    results = []
    blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416),swapRB=True, crop=False)
    net.setInput(blob)
    layerOutputs = net.forward(ln)
    boxes = []
    centroids = []
    confidences = []
```

- Scale the bounding box coordinates back relieve to the size of images and returns the center(x,y) coordinates and after updating the Bounding box coordinates,centroids.
- Ensure the detection conditions. Extract the bounding box coordinates. And update the results with person detection.



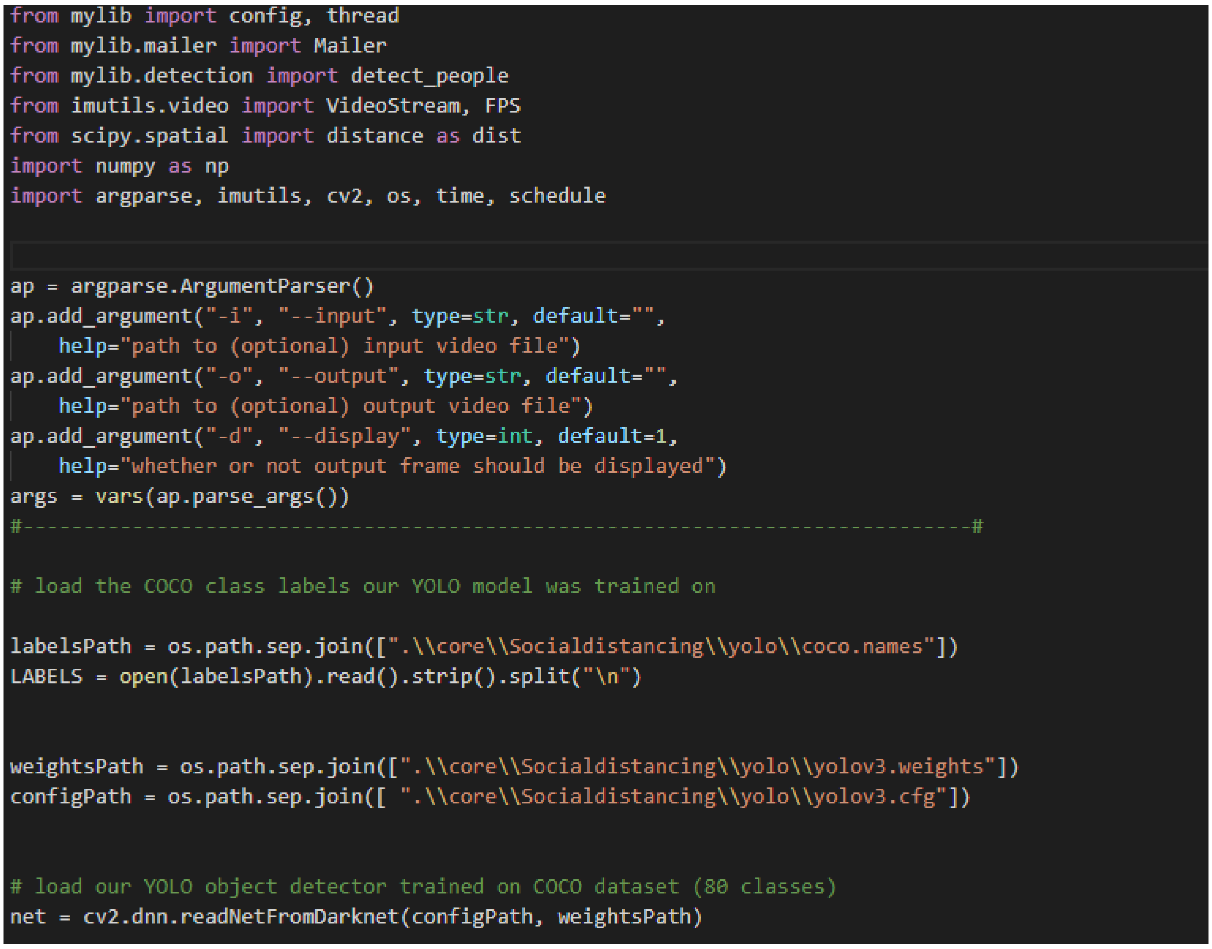
```
    for output in layerOutputs:
        for detection in output:
            scores = detection[5:]
            classID = np.argmax(scores)
            confidence = scores[classID]
            if classID == personIdx and confidence > MIN_CONF:
                box = detection[0:4] * np.array([W, H, W, H])
                (centerX, centerY, width, height) = box.astype("int")
                x = int(centerX - (width / 2))
                y = int(centerY - (height / 2))
                boxes.append([x, y, int(width), int(height)])
                centroids.append((centerX, centerY))
                confidences.append(float(confidence))

    idxs = cv2.dnn.NMSBoxes(boxes, confidences, MIN_CONF, NMS_THRESH)

    if len(idxs) > 0:
        for i in idxs.flatten():
            (x, y) = (boxes[i][0], boxes[i][1])
            (w, h) = (boxes[i][2], boxes[i][3])
            r = (confidences[i], (x, y, x + w, y + h), centroids[i])
            results.append(r)

    return results
```

- Set the path of “yolo-coco” library and set the minimum threshold & define the minimum safe distance between two people from each other.



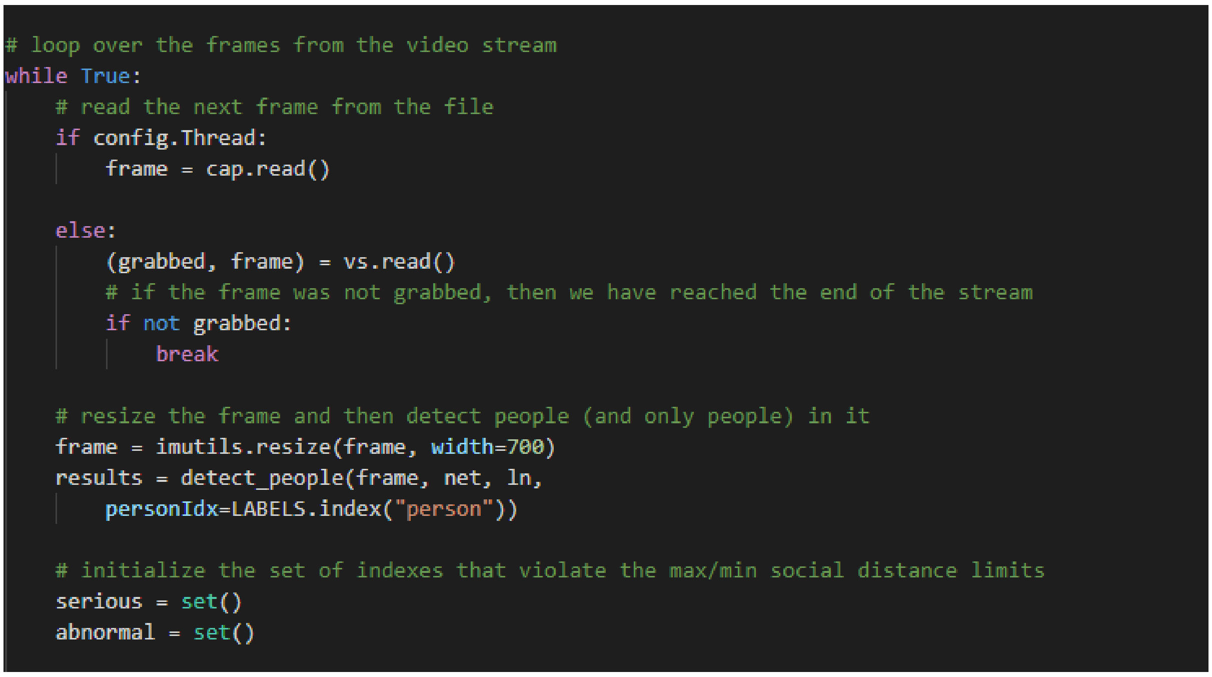
```
#Set the path of “yolo-coco” library
MODEL_PATH = "yolo-coco"

MIN_CONF = 0.3
#set the minimum threshold
NMS_THRESH = 0.3

USE_GPU = False

#minimum safe distance between two people from each other.
MIN_DISTANCE = 100
```

- Load the yolo coco models, yolo weights & loading the yolo detector and train the dataset.



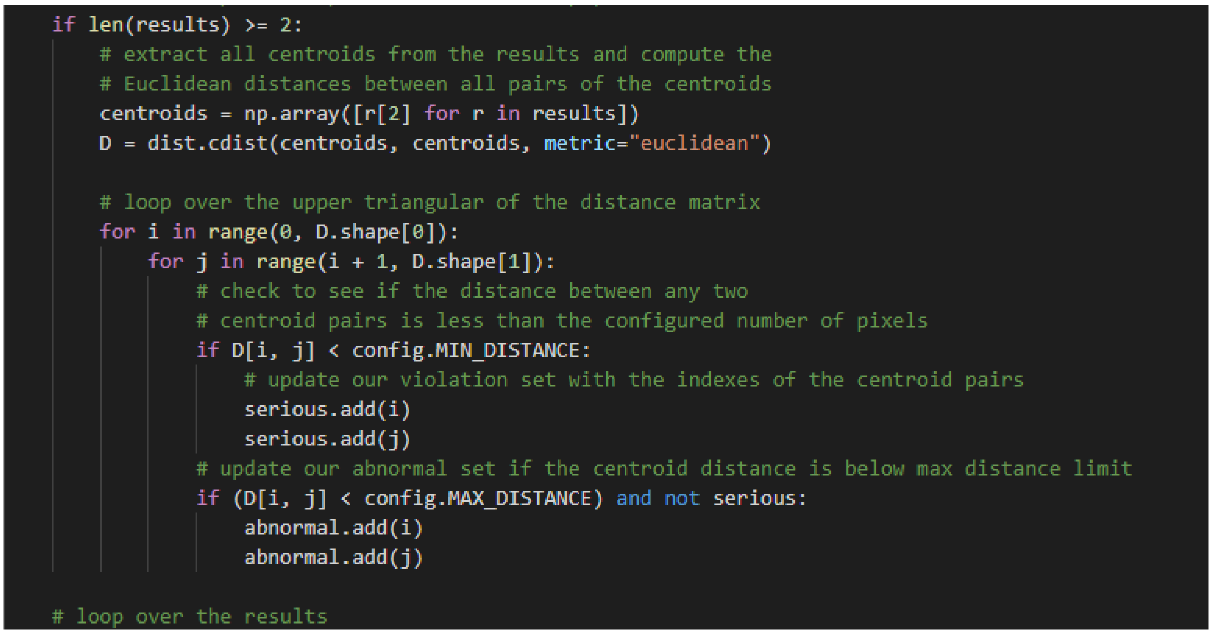
```
from mylib import config, thread
from mylib.mailer import Mailer
from mylib.detection import detect_people
from imutils.video import VideoStream, FPS
from scipy.spatial import distance as dist
import numpy as np
import argparse, imutils, cv2, os, time, schedule


ap = argparse.ArgumentParser()
ap.add_argument("-i", "--input", type=str, default="",
    help="path to (optional) input video file")
ap.add_argument("-o", "--output", type=str, default="",
    help="path to (optional) output video file")
ap.add_argument("-d", "--display", type=int, default=1,
    help="whether or not output frame should be displayed")
args = vars(ap.parse_args())
#-------------------------------------------------------------------------------#

# load the COCO class labels our YOLO model was trained on

labelsPath = os.path.sep.join([".\\core\\Socialdistancing\\yolo\\coco.names"])
LABELS = open(labelsPath).read().strip().split("\n")


weightsPath = os.path.sep.join([".\\core\\Socialdistancing\\yolo\\yolov3.weights"])
configPath = os.path.sep.join([ ".\\core\\Socialdistancing\\yolo\\yolov3.cfg"])


# load our YOLO object detector trained on COCO dataset (80 classes)
net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)
```

- Initialize the video stream and read a continuous another frames for video & intialize the set of people who violates minimum social distance.



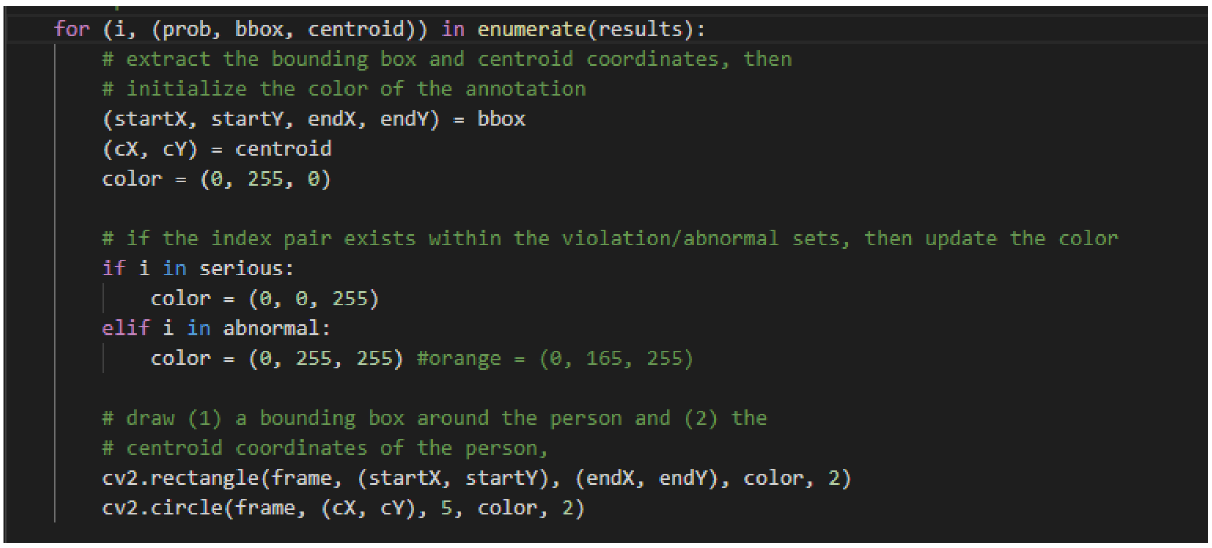
```
# loop over the frames from the video stream
while True:
    # read the next frame from the file
    if config.Thread:
        frame = cap.read()

    else:
        (grabbed, frame) = vs.read()
        # if the frame was not grabbed, then we have reached the end of the stream
        if not grabbed:
            break

    # resize the frame and then detect people (and only people) in it
    frame = imutils.resize(frame, width=700)
    results = detect_people(frame, net, ln,
        personIdx=LABELS.index("person"))

    # initialize the set of indexes that violate the max/min social distance limits
    serious = set()
    abnormal = set()
```

- Ensuring atleast two people detect and find euclidiean distance between all pairs of the centroids & checked if the distance between any two pairs is less than the configure number then add it into violation set.



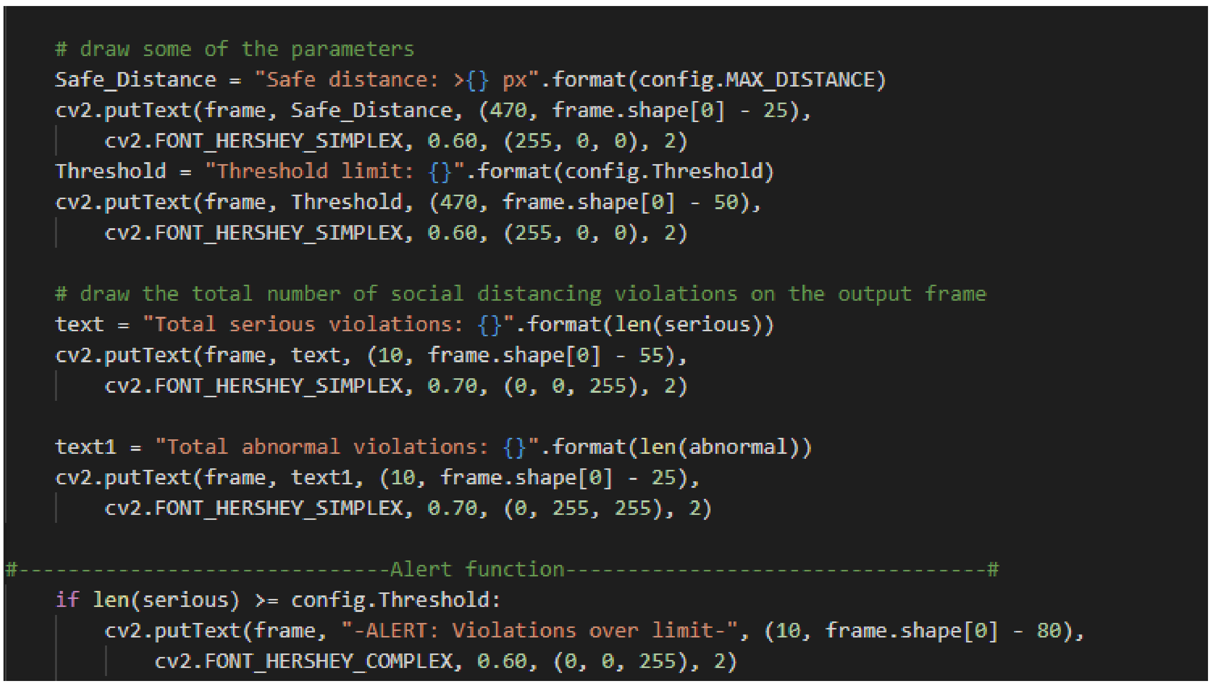
```
    if len(results) >= 2:
        # extract all centroids from the results and compute the
        # Euclidean distances between all pairs of the centroids
        centroids = np.array([r[2] for r in results])
        D = dist.cdist(centroids, centroids, metric="euclidean")

        # loop over the upper triangular of the distance matrix
        for i in range(0, D.shape[0]):
            for j in range(i + 1, D.shape[1]):
                # check to see if the distance between any two
                # centroid pairs is less than the configured number of pixels
                if D[i, j] < config.MIN_DISTANCE:
                    # update our violation set with the indexes of the centroid pairs
                    serious.add(i)
                    serious.add(j)
                # update our abnormal set if the centroid distance is below max distance limit
                if (D[i, j] < config.MAX_DISTANCE) and not serious:
                    abnormal.add(i)
                    abnormal.add(j)

    # loop over the results
```

- Intalize the color of extract frames & updating a color of frames who are violated and display the total number of violations also checked some output cardinality.



```
    for (i, (prob, bbox, centroid)) in enumerate(results):
        # extract the bounding box and centroid coordinates, then
        # initialize the color of the annotation
        (startX, startY, endX, endY) = bbox
        (cX, cY) = centroid
        color = (0, 255, 0)

        # if the index pair exists within the violation/abnormal sets, then update the color
        if i in serious:
            color = (0, 0, 255)
        elif i in abnormal:
            color = (0, 255, 255) #orange = (0, 165, 255)

        # draw (1) a bounding box around the person and (2) the
        # centroid coordinates of the person,
        cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)
        cv2.circle(frame, (cX, cY), 5, color, 2)
```

- Display safe distance,total serious and abnormal violations and Alert to the output window.



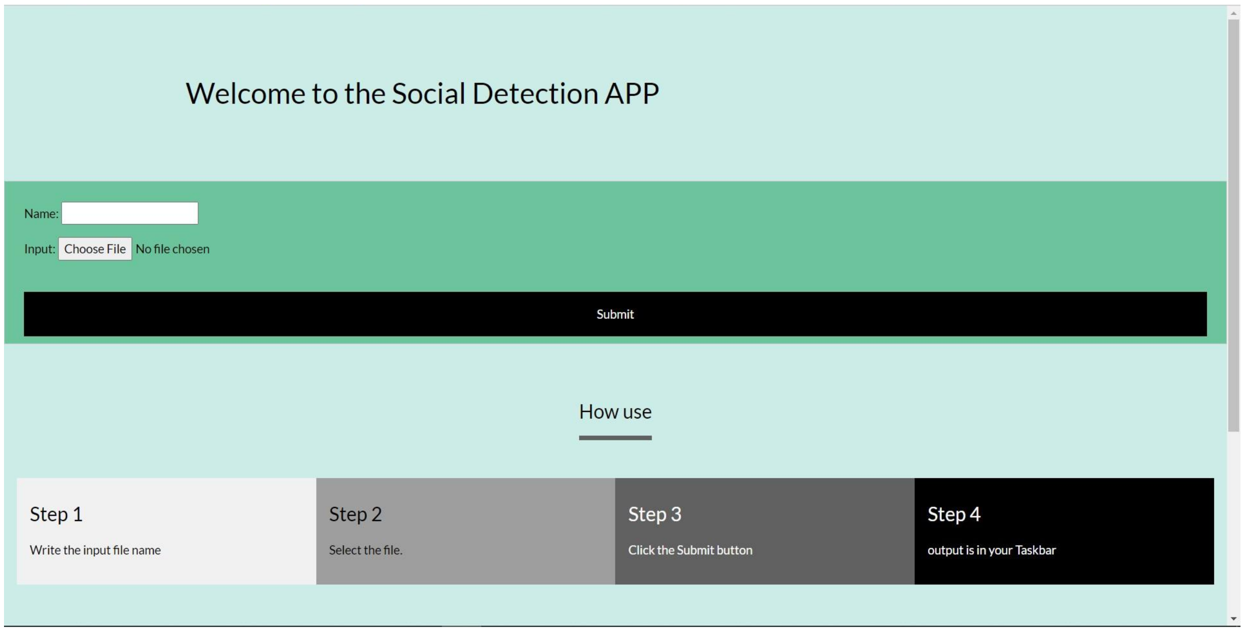
```
    # draw some of the parameters
    Safe_Distance = "Safe distance: >{} px".format(config.MAX_DISTANCE)
    cv2.putText(frame, Safe_Distance, (470, frame.shape[0] - 25),
        cv2.FONT_HERSHEY_SIMPLEX, 0.60, (255, 0, 0), 2)
    Threshold = "Threshold limit: {}".format(config.Threshold)
    cv2.putText(frame, Threshold, (470, frame.shape[0] - 50),
        cv2.FONT_HERSHEY_SIMPLEX, 0.60, (255, 0, 0), 2)

    # draw the total number of social distancing violations on the output frame
    text = "Total serious violations: {}".format(len(serious))
    cv2.putText(frame, text, (10, frame.shape[0] - 55),
        cv2.FONT_HERSHEY_SIMPLEX, 0.70, (0, 0, 255), 2)

    text1 = "Total abnormal violations: {}".format(len(abnormal))
    cv2.putText(frame, text1, (10, frame.shape[0] - 25),
        cv2.FONT_HERSHEY_SIMPLEX, 0.70, (0, 255, 255), 2)

#------------------------------Alert function----------------------------------#
    if len(serious) >= config.Threshold:
        cv2.putText(frame, "-ALERT: Violations over limit-", (10, frame.shape[0] - 80),
            cv2.FONT_HERSHEY_COMPLEX, 0.60, (0, 0, 255), 2)
```

## 9) Screenshots :

Github link :

- https://github.com/marudevansh25111/Social_Distance_Detection_App

Main page -


Welcome to the Social Detection APP

Name:

Input: Choose File No file chosen

Submit

How use

Step 1

Write the input file name

Step 2

Select the file.

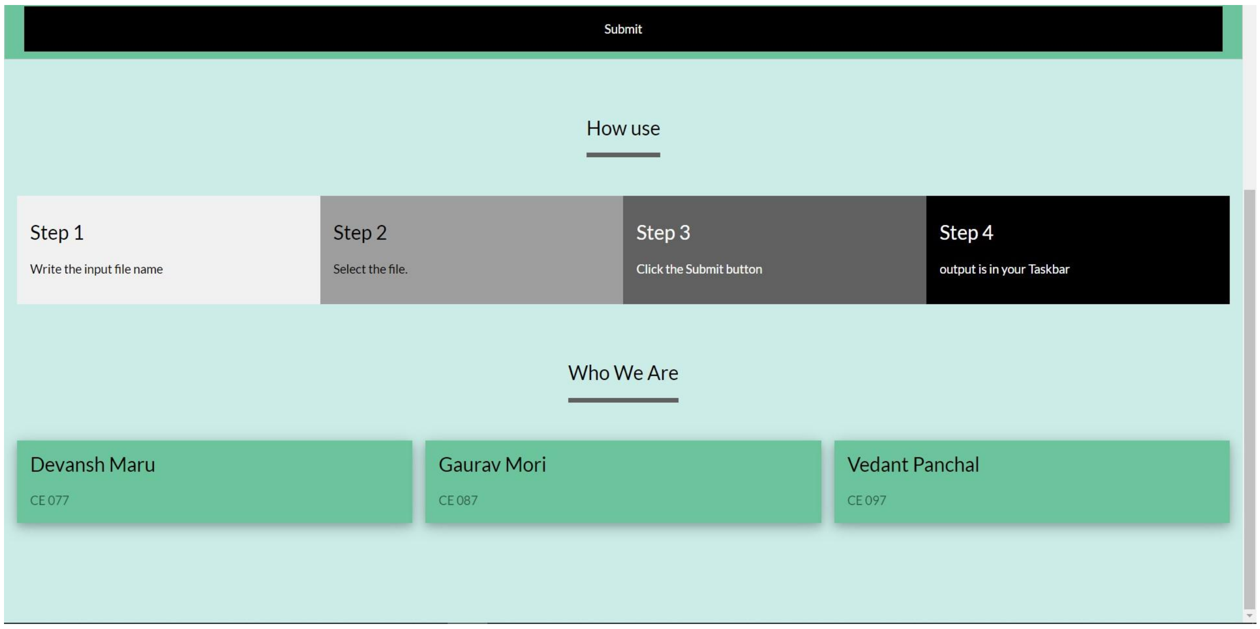
Step 3

Click the Submit button

Step 4

output is in your Taskbar

Steps of selecting file-


Submit

How use

Step 1

Write the input file name

Step 2

Select the file.

Step 3

Click the Submit button

Step 4

output is in your Taskbar

Who We Are

Devansh Maru

CE 077

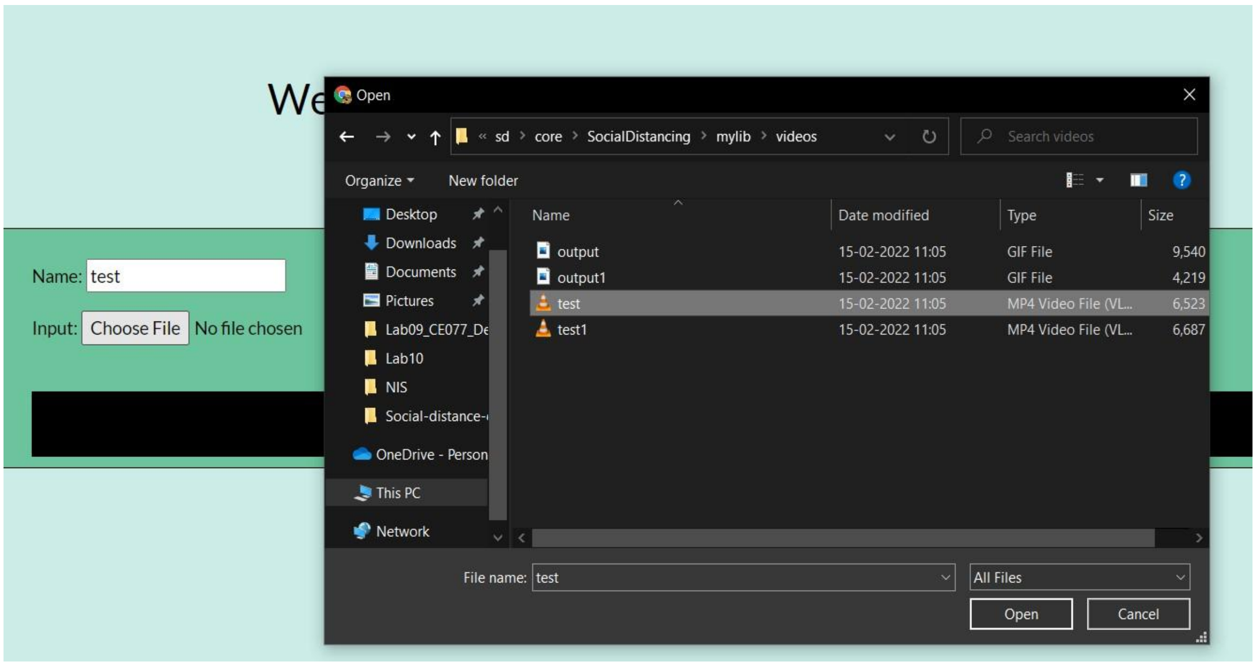
Gaurav Mori

CE 087

Vedant Panchal

CE 097

## Input file selection -



Name: test

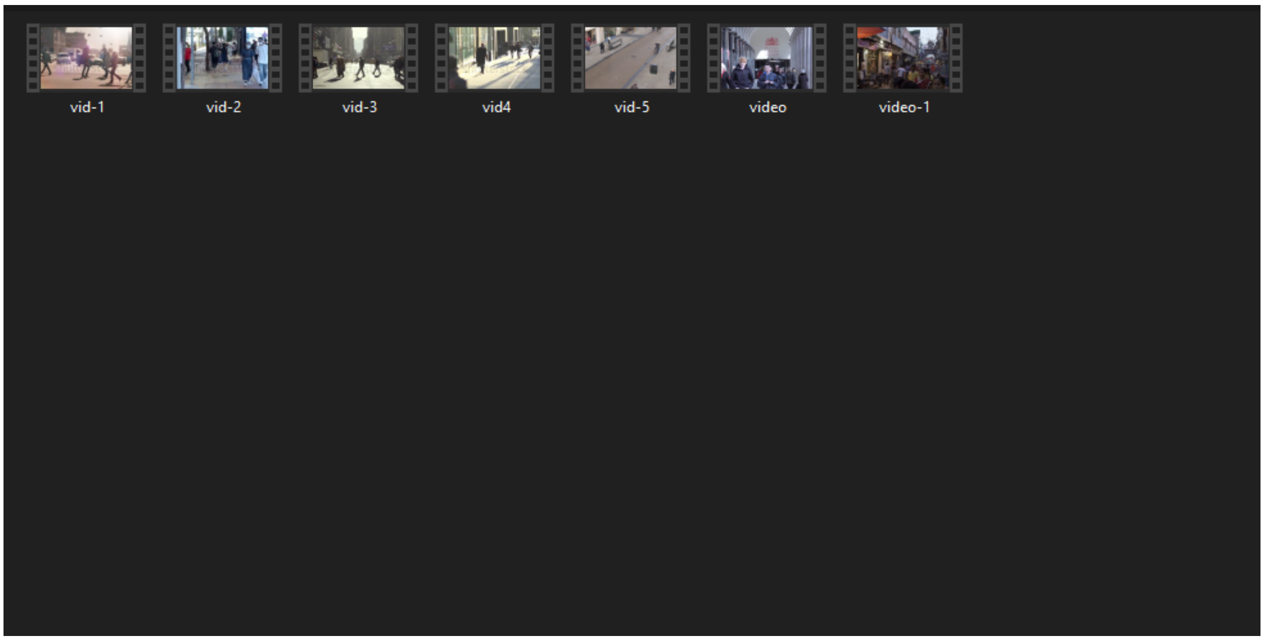
Input: Choose File No file chosen

| Name | Date modified | Type | Size |
|---|---|---|---|
| output | 15-02-2022 11:05 | GIF File | 9,540 |
| output1 | 15-02-2022 11:05 | GIF File | 4,219 |
| test | 15-02-2022 11:05 | MP4 Video File (VL... | 6,523 |
| test1 | 15-02-2022 11:05 | MP4 Video File (VL... | 6,687 |

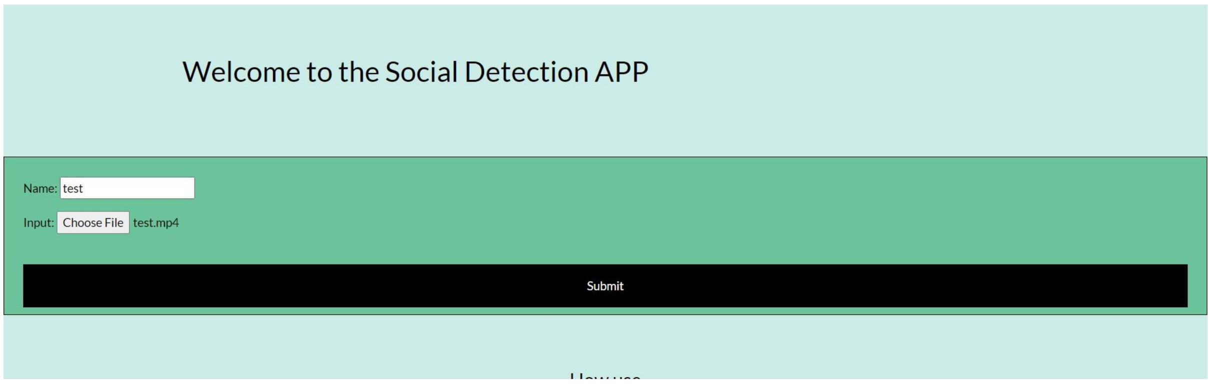
File name: test

All Files

## Input file location -



## Submit the file -

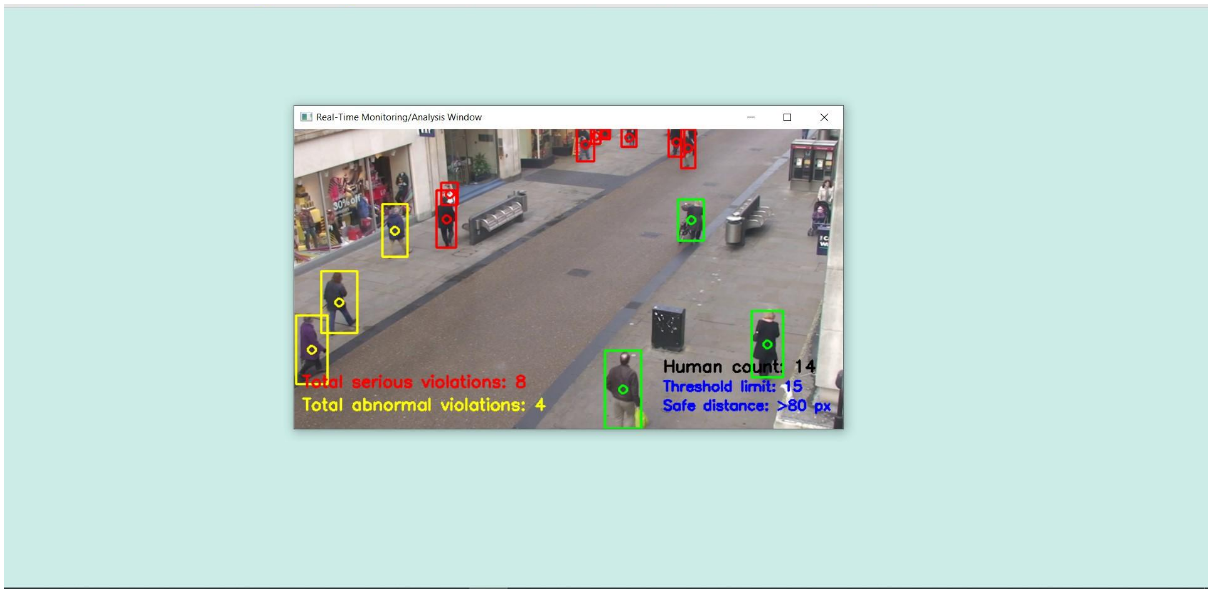

Welcome to the Social Detection APP

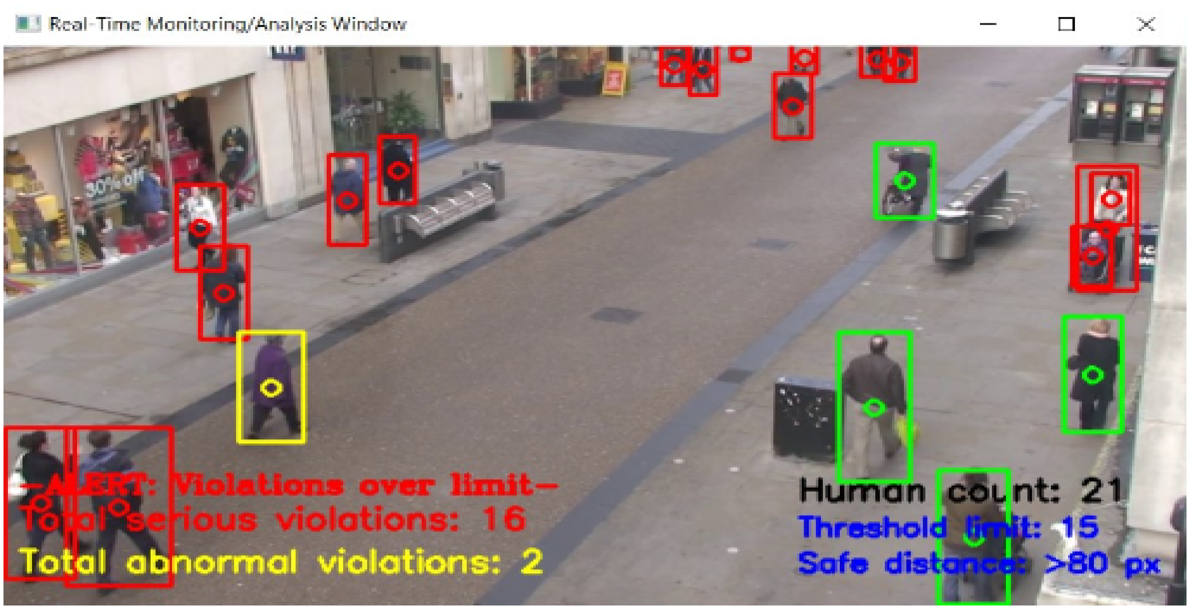
Name: test

Input: Choose File test.mp4

Submit

## Output of video file -


Real-Time Monitoring/Analysis Window
Total serious violations: 8
Total abnormal violations: 4
Human count: 14
Threshold limit: 15
Safe distance: >80 px


Real-Time Monitoring/Analysis Window
-ALERT: Violations over limit-
Total serious violations: 16
Total abnormal violations: 2
Human count: 21
Threshold limit: 15
Safe distance: >80 px

## 10) Conclusion :

- The studies concluded that the early and immediate practice of social distancing could gradually reduce the peak of the virus attack.
- In this work, a deep learning-based social distance monitoring framework is presented using an overhead perspective. The pre-trained YOLOv3 paradigm is used for human detection.
- The detection model gives bounding box information, containing centroid coordinates information. Using the Euclidean distance, the pairwise centroid distances between detected bounding boxes are measured. To check social distance violations between people, an approximation of physical distance to the pixel is used, and a threshold is defined.
- A violation threshold is used to check if the distance value violates the minimum social distance set or not.And display all the violations of the video in output window.

## 11) Limitations and FutureEnhancements:

- **Limitations :**

  - System is not working for direct live cctv camera.
  - The tracking accuracy of peoples is quite lesser.

- **FutureEnhancements :**

  - The work may be improved in the future for different indoor and outdoor environments.
  - The system  also works for live cctv camera footage.
  - Different detection and tracking algorithms might be used to help track the person or people who are violating or breaches the social distancing threshold.
  - More accuracy for detecting peoples.
  - More accuracy for checking violations.

## 12) Reference / Bibliography :

- “COCO - Common Objects in Context,” *COCO - Common Objects in Context*. https://cocodataset.org/#home (accessed Mar. 08, 2022).

- U. Almog, “YOLO V3 Explained. In this post we’ll discuss the YOLO… | by Uri Almog | Towards Data Science,” *Medium*, Oct. 09, 2020. https://towardsdatascience.com/yolo-v3-explained-ff5b850390f (accessed Mar. 08, 2022).

- J. Solawetz, “YOLOv3 Versus EfficientDet for State-of-the-Art Object Detection,” *Roboflow Blog*, May 01, 2020. https://blog.roboflow.com/yolov3-versus-efficientdet-for-state-of-the-art-object-detection/ (accessed Mar. 08, 2022).

- “Electronics | Free Full-Text | Social Distance Monitoring Approach Using Wearable Smart Tags | HTML,” *MDPI*. https://www.mdpi.com/2079-9292/10/19/2435/htm (accessed Mar. 08, 2022).