# FACE MASK DETECTION USING CNN, OPENCV WITH EMAIL, LOCATION, TIME, AND DATE SUPPORT

**Created By: Hrishikesh Sunil Shinde**          **Date: 10/01/2021**

## OVERVIEW

### 1. Project Background and Description

In today's era of coronavirus, many people don't take the necessary precautions to be safe. Like not maintaining a safe distance from other people, not wearing the mask in public places, wearing the mask below the chin while talking, etc. This could lead to severe problems for all of us.

So, this system aims to detect the people who are not wearing the masks in public places and the people who are wearing the mask below the chin. The system captures a snapshot of the person after a fixed interval of time. If the person is not wearing the mask for a longer time. Then the system will send the snapshot of the person to Surveillance Team with the location, time, date, etc. through email. Further, the surveillance team can take action on the victim can charge a fine for the activity. This will help to lower down the transmission of coronavirus.

### 2. Project Scope

The system will detect the people who are not wearing the mask for a longer time. The system will send the snaps shots of them after a particular time interval through email. If the person is wearing a mask then the system will detect that person as safe. The system can be further extended to recognize the identity (i.e. name, gender, email id) of a person if not wearing the mask. When the identity is detected the system will directly send a mail regarding to him/her details about how much fine the person has to pay. The system will also add snapshots and other details like time, date, location of the activity into the mail.

The system uses "multiprocessing" ability to send the mail over the internet. It is necessary because the main program should be running without any interruption.

### 3. Requirements

Hardware Requirements:

1) Camera.
2) CPU
3) GPU (Optional)

Software Requirements:

1) Python 3.6+
2) Windows Operating System
3) Integrated Development Environment (IDE)

## 4. Affected Parties

This project will be very much useful in public places, also this can be implemented in the company, industrial premises. They can directly identify their employee and warn them.

## 5. Required Python Libraries:

1. Tensorflow: To create and train the CNN model.

2. Computer Vision (cv2): To take input frame from camera.

3. Multiprocessing: For parallel Processing.

4. Numpy: For creating and changing values of arrays.

5. Requests: to get IP address of the system.

6. Urllib: to get current location of system.

7. Ast: to convert binary data to dictionary

8. Datetime: to get system date and time

## 6. Dataset Overview

Dataset is has two classes 1) WithMask Class and 2) Without Mask class. Also, it's 3 parts.
1) Training set: total 10052 images.
2) Validation set: 800 images
3) Testing set: 992 images
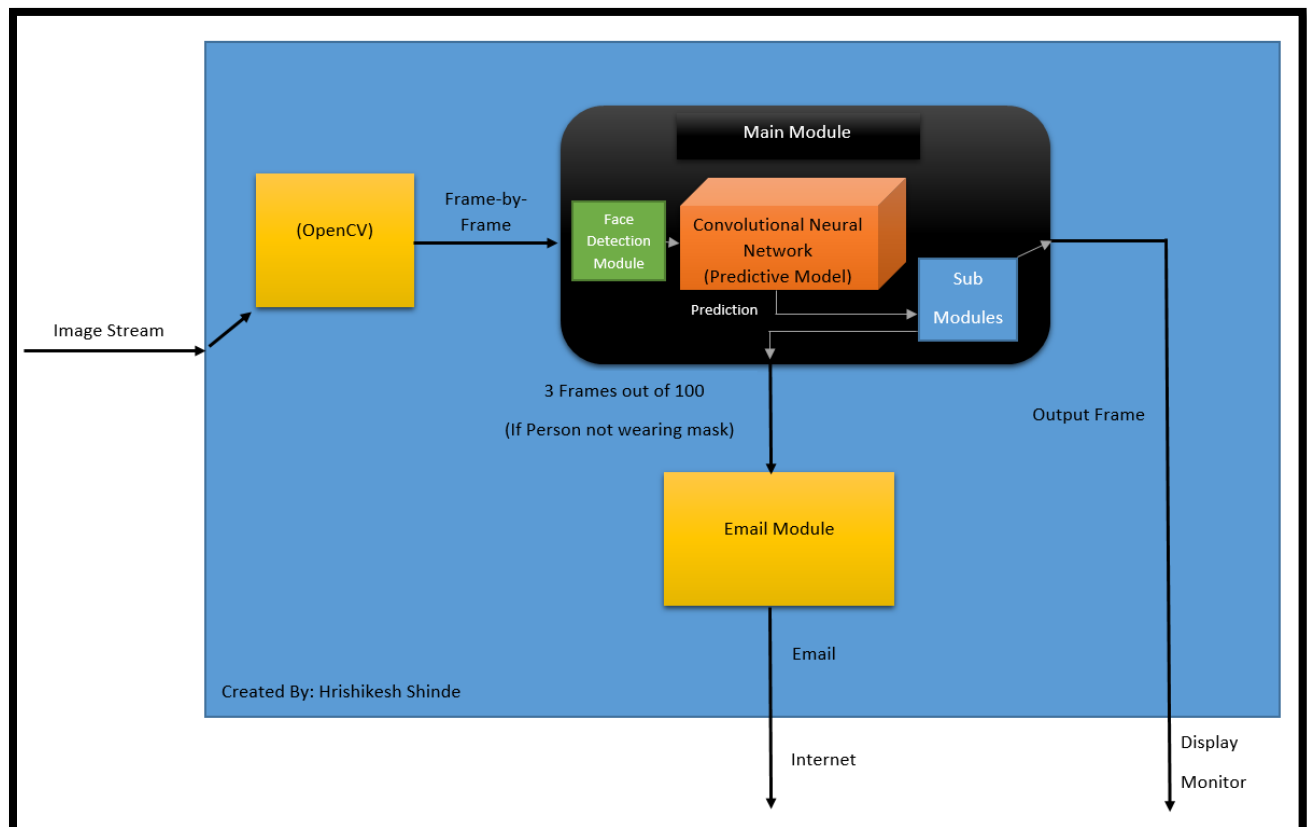The above mages are categorized into 2 class.

## 7. Data Augmentation

Data augmentation like change in brightness, rescale, horizontal flip are performed on training set and validation set to improve the accuracy of the model.
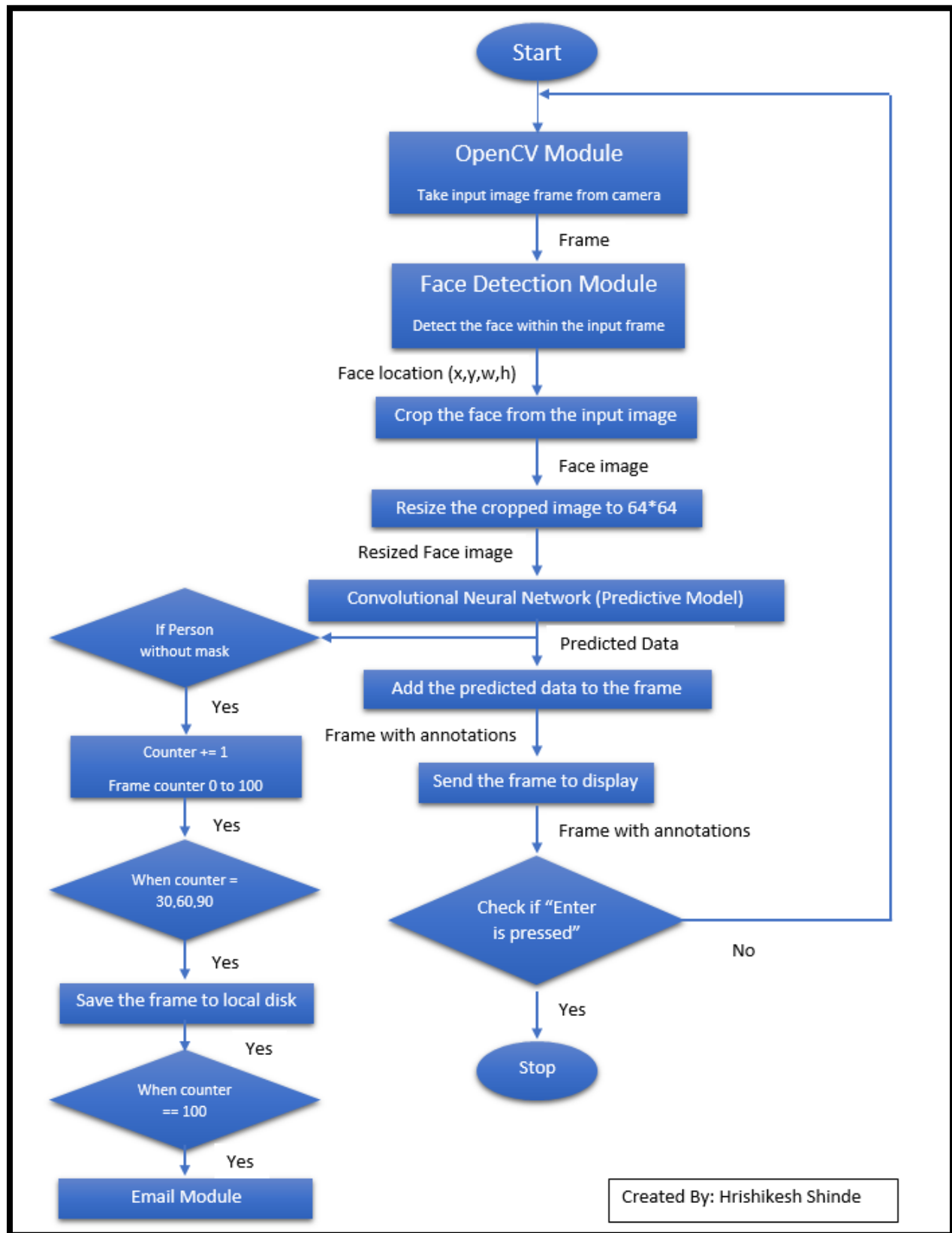
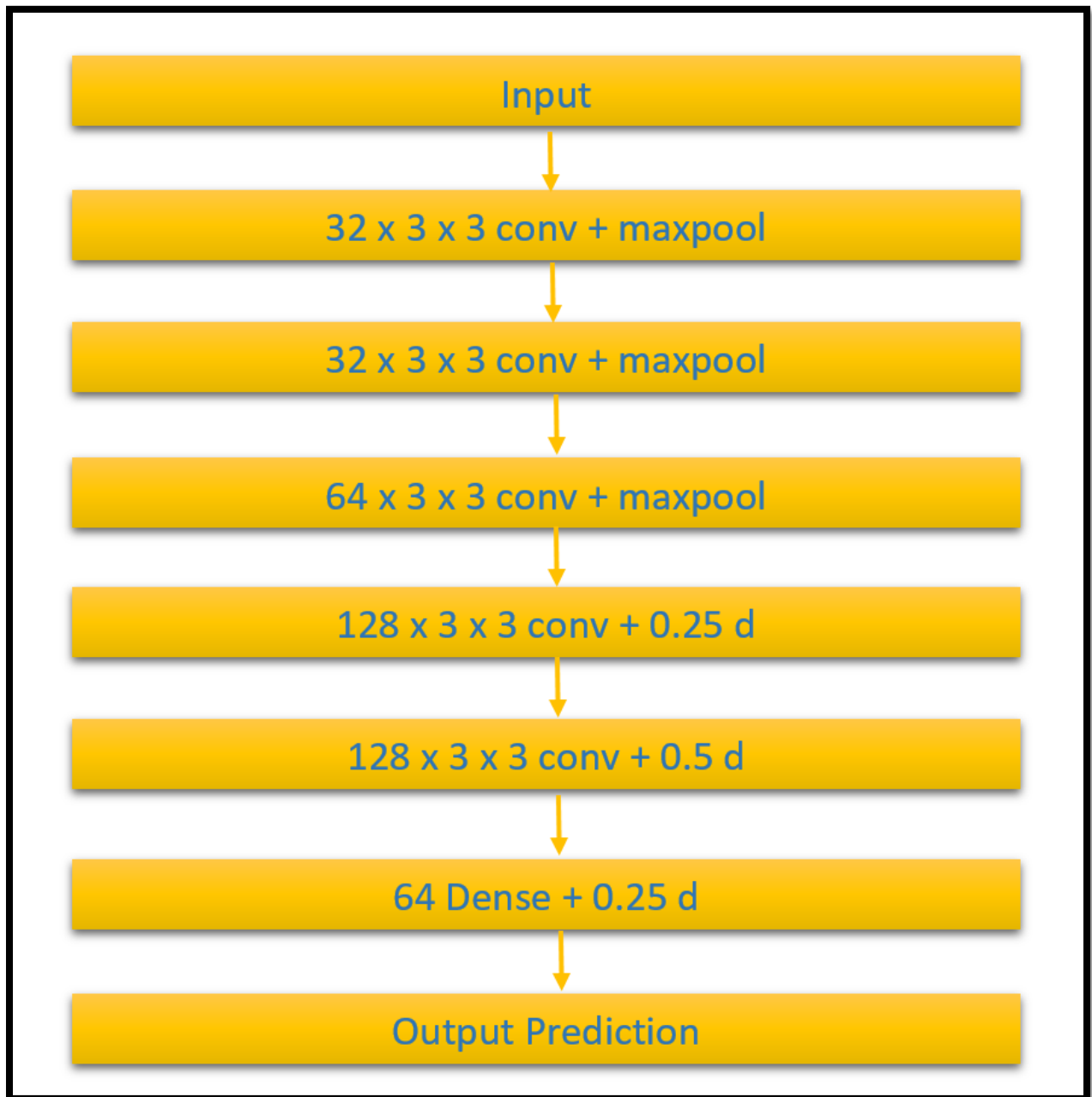# IMPLEMENTATION OVERVIEW

## 1. Overview



Created by: Hrishikesh Shinde
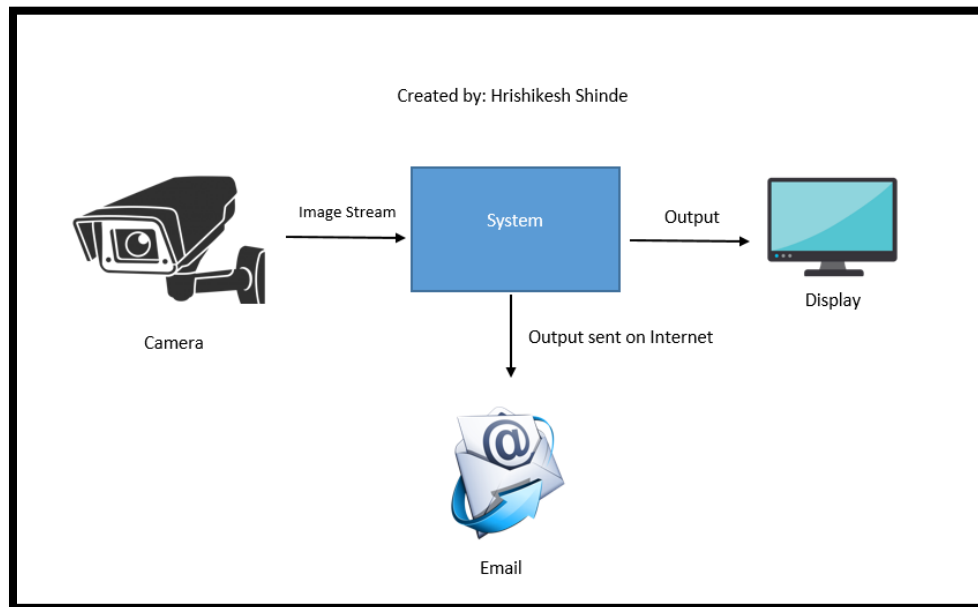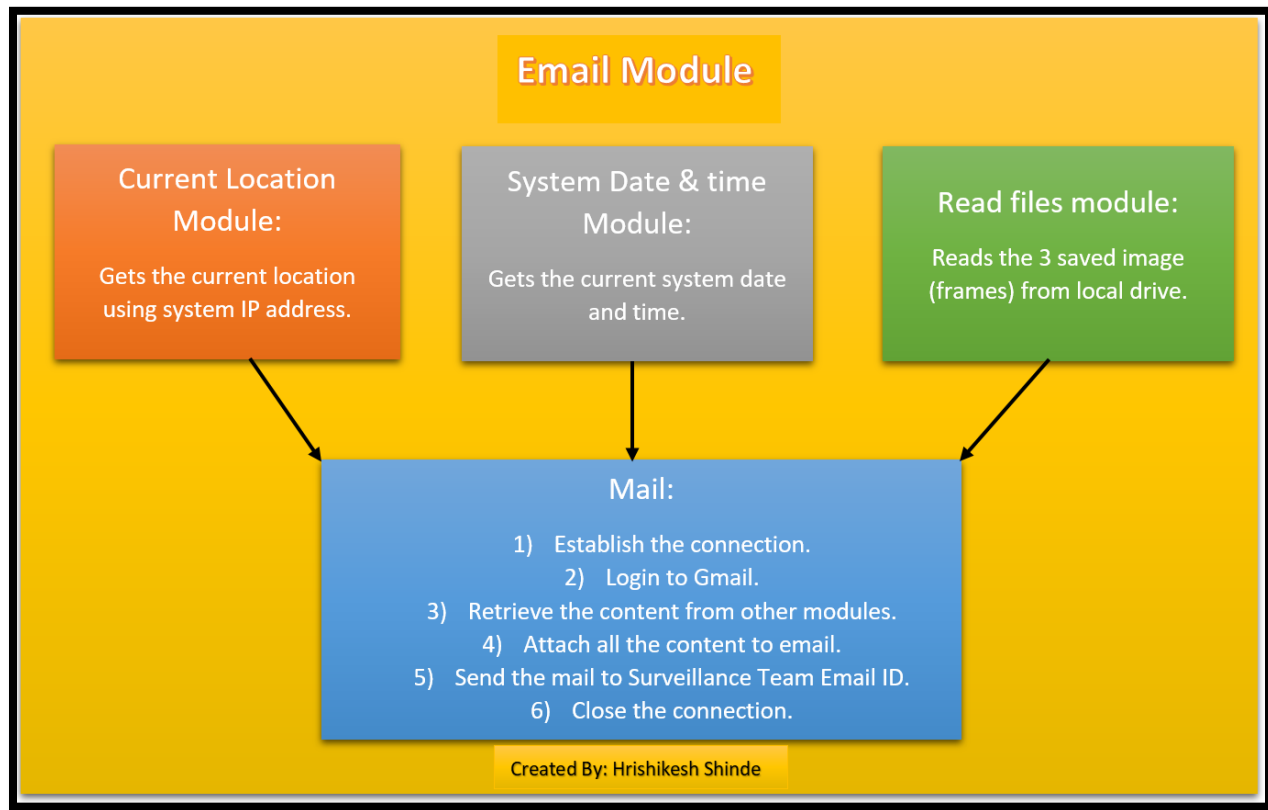
## 2. Testing Model Overview

## 3. Flow Chart

```
                              ┌─────────┐
                              │  Start  │
                              └─────────┘
                                   │
                    ┌──────────────────────────────┐
                    │        OpenCV Module          │
                    │ Take input image frame from    │
                    │           camera               │
                    └──────────────────────────────┘
                                   │  Frame
                    ┌──────────────────────────────┐
                    │     Face Detection Module     │
                    │ Detect the face within the     │
                    │          input frame           │
                    └──────────────────────────────┘
                                   │  Face location (x,y,w,h)
                    ┌──────────────────────────────┐
                    │ Crop the face from the input   │
                    │           image                │
                    └──────────────────────────────┘
                                   │  Face image
                    ┌──────────────────────────────┐
                    │  Resize the cropped image to   │
                    │            64*64               │
                    └──────────────────────────────┘
                                   │  Resized Face image
                    ┌──────────────────────────────┐
                    │ Convolutional Neural Network   │
                    │      (Predictive Model)        │
                    └──────────────────────────────┘
```

If Person without mask

Counter += 1
Frame counter 0 to 100

When counter = 30,60,90

Save the frame to local disk

When counter == 100

Email Module

Predicted Data

Add the predicted data to the frame

Frame with annotations

Send the frame to display

Frame with annotations

Check if "Enter is pressed"

No

Yes

Stop

Yes (multiple)

Created By: Hrishikesh Shinde

4

## 4. Neural Network Architecture

Input

↓

32 x 3 x 3 conv + maxpool

↓

32 x 3 x 3 conv + maxpool

↓

64 x 3 x 3 conv + maxpool

↓

128 x 3 x 3 conv + 0.25 d

↓

128 x 3 x 3 conv + 0.5 d

↓

64 Dense + 0.25 d

↓

Output Prediction

## 5. Email Module



Email Module

**Current Location Module:**

Gets the current location using system IP address.

**System Date & time Module:**

Gets the current system date and time.

**Read files module:**

Reads the 3 saved image (frames) from local drive.

**Mail:**

1) Establish the connection.
2) Login to Gmail.
3) Retrieve the content from other modules.
4) Attach all the content to email.
5) Send the mail to Surveillance Team Email ID.
6) Close the connection.

Created By: Hrishikesh Shinde

## 6. Training Module Overview



Input Image Set → Data Generator → Augment → ? → Yes → Data Augmented

No

Batch Data

Training Module

Training Set | Validation Set

Convolution Neural Network

Camera → Video → Video preprocessing → Processed Data → Predictive Model

Output Prediction
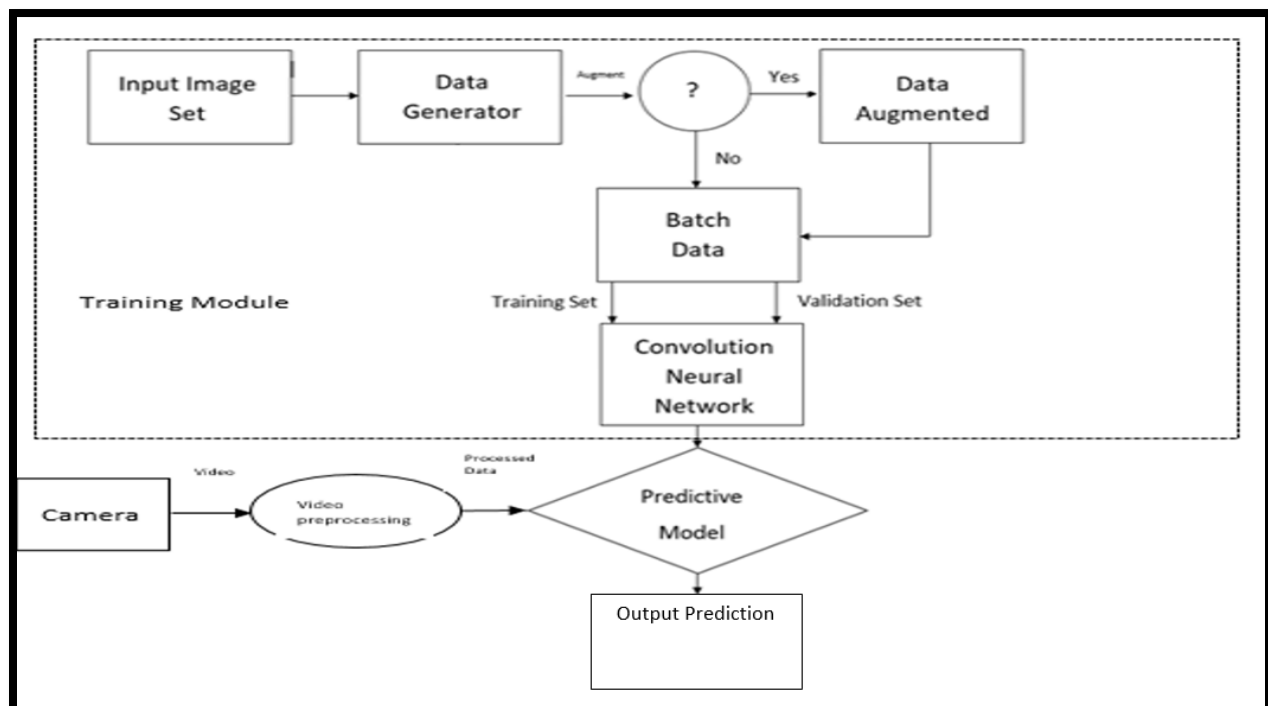
## 7. Training History

```
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
Train for 625 steps, validate for 50 steps
Epoch 1/50
624/625 [============================>.] - ETA: 0s - loss: 0.2213 - accuracy: 0.9215   ETA - ETA - ETA: 40s - loss: 0.2506 - ac
cura
Epoch 00001: val_loss improved from inf to 0.10199, saving model to Mask_best_fit
WARNING:tensorflow:From C:\Users\Acer\Anaconda3\envs\cv\lib\site-packages\tensorflow_core\python\ops\resource_variable_ops.py:1
786: calling BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint is deprecated and
will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
INFO:tensorflow:Assets written to: Mask_best_fit\assets
625/625 [==============================] - 170s 272ms/step - loss: 0.2210 - accuracy: 0.9216 - val_loss: 0.1020 - val_accuracy:
0.9750
Epoch 2/50
623/625 [============================>.] - ETA: 0s - loss: 0.1120 - accuracy: 0.9672
Epoch 00002: val_loss improved from 0.10199 to 0.04126, saving model to Mask_best_fit
INFO:tensorflow:Assets written to: Mask_best_fit\assets
625/625 [==============================] - 21s 34ms/step - loss: 0.1117 - accuracy: 0.9673 - val_loss: 0.0413 - val_accuracy:
0.9912
Epoch 3/50
624/625 [============================>.] - ETA: 0s - loss: 0.0849 - accuracy: 0.9763
Epoch 00003: val_loss improved from 0.04126 to 0.03469, saving model to Mask_best_fit
INFO:tensorflow:Assets written to: Mask_best_fit\assets
625/625 [==============================] - 20s 33ms/step - loss: 0.0849 - accuracy: 0.9764 - val_loss: 0.0347 - val_accuracy:
0.9887
Epoch 4/50
624/625 [============================>.] - ETA: 0s - loss: 0.0769 - accuracy: 0.9785
Epoch 00004: val_loss did not improve from 0.03469
625/625 [==============================] - 17s 27ms/step - loss: 0.0770 - accuracy: 0.9785 - val_loss: 0.0930 - val_accuracy:
0.9750
Epoch 5/50
624/625 [============================>.] - ETA: 0s - loss: 0.0791 - accuracy: 0.9814
Epoch 00005: val_loss did not improve from 0.03469
625/625 [==============================] - 18s 28ms/step - loss: 0.0790 - accuracy: 0.9815 - val_loss: 0.0706 - val_accuracy:
0.9800
Epoch 6/50
623/625 [============================>.] - ETA: 0s - loss: 0.0797 - accuracy: 0.9835 ETA: 0s - loss: 0.0774 - accuracy:
Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.

Epoch 00006: val_loss did not improve from 0.03469
625/625 [==============================] - 19s 31ms/step - loss: 0.0800 - accuracy: 0.9835 - val_loss: 0.0469 - val_accuracy:
0.9825
Epoch 7/50
624/625 [============================>.] - ETA: 0s - loss: 0.0346 - accuracy: 0.9909
Epoch 00007: val_loss improved from 0.03469 to 0.01385, saving model to Mask_best_fit
INFO:tensorflow:Assets written to: Mask_best_fit\assets
625/625 [==============================] - 21s 34ms/step - loss: 0.0348 - accuracy: 0.9908 - val_loss: 0.0138 - val_accuracy:
0.9962
Epoch 8/50
624/625 [============================>.] - ETA: 0s - loss: 0.0295 - accuracy: 0.9920
Epoch 00008: val_loss did not improve from 0.01385
625/625 [==============================] - 18s 29ms/step - loss: 0.0296 - accuracy: 0.9919 - val_loss: 0.0385 - val_accuracy:
0.9962
Epoch 9/50
623/625 [============================>.] - ETA: 0s - loss: 0.0291 - accuracy: 0.9934
Epoch 00009: val_loss did not improve from 0.01385
625/625 [==============================] - 21s 34ms/step - loss: 0.0290 - accuracy: 0.9934 - val_loss: 0.0361 - val_accuracy:
0.9937
Epoch 10/50
623/625 [============================>.] - ETA: 0s - loss: 0.0276 - accuracy: 0.9931
Epoch 00010: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.

Epoch 00010: val_loss did not improve from 0.01385
625/625 [==============================] - 19s 31ms/step - loss: 0.0275 - accuracy: 0.9931 - val_loss: 0.0484 - val_accuracy:
0.9987
Epoch 11/50
624/625 [============================>.] - ETA: 0s - loss: 0.0218 - accuracy: 0.9951
Epoch 00011: val_loss did not improve from 0.01385
625/625 [==============================] - 19s 31ms/step - loss: 0.0218 - accuracy: 0.9951 - val_loss: 0.0292 - val_accuracy:
0.9975
Epoch 12/50
623/625 [============================>.] - ETA: 0s - loss: 0.0150 - accuracy: 0.9960Restoring model weights from the end of the
best epoch.

Epoch 00012: val_loss did not improve from 0.01385
625/625 [==============================] - 20s 32ms/step - loss: 0.0150 - accuracy: 0.9960 - val_loss: 0.0539 - val_accuracy:
0.9975
Epoch 00012: early stopping
```
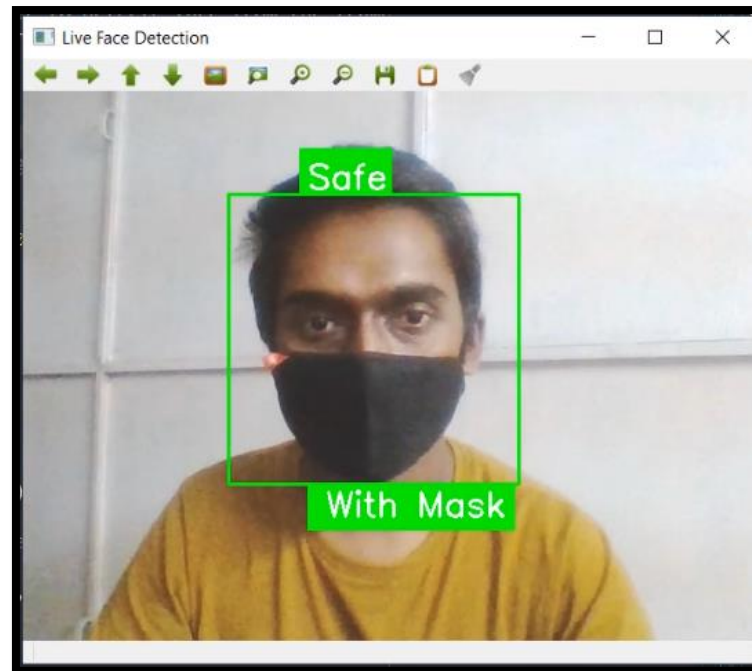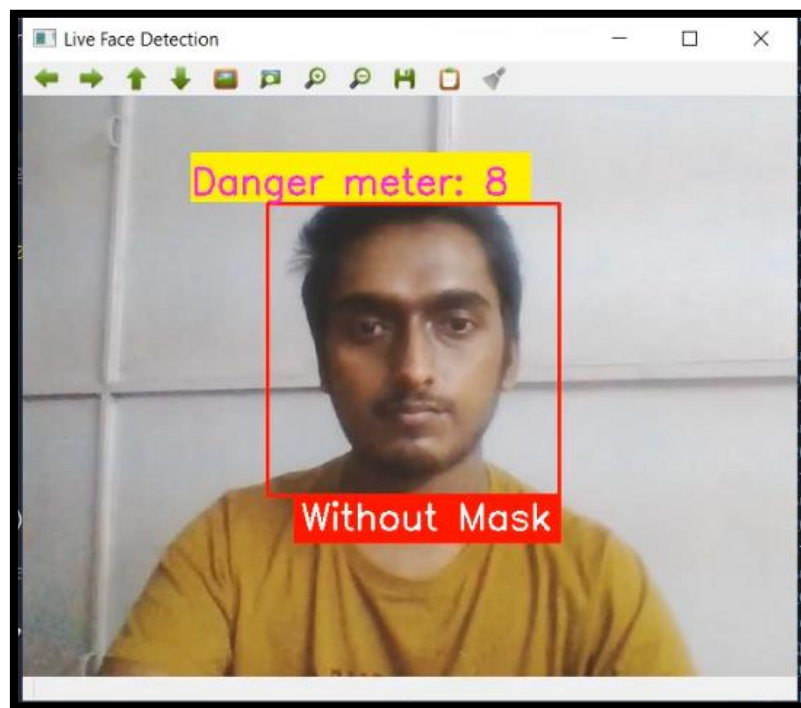
## 8. ROC Curve



**Receiver operating characteristic example**
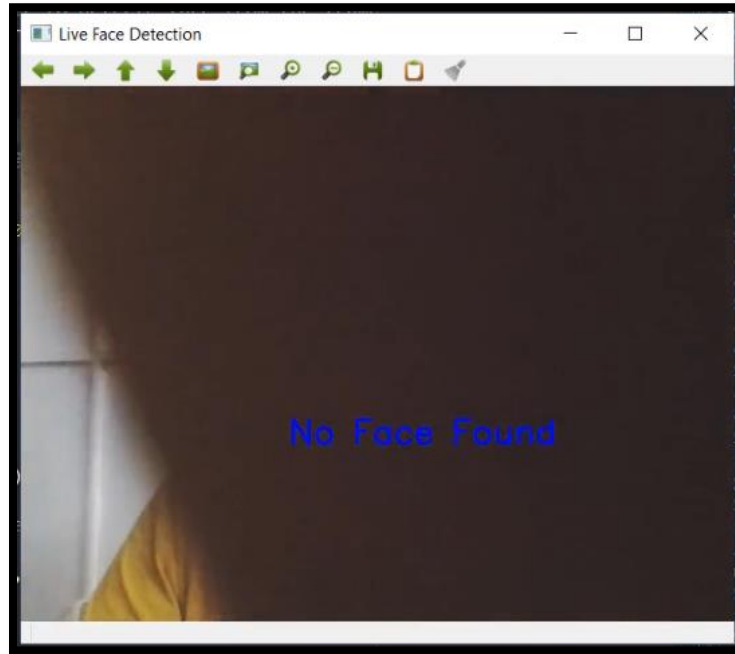
ROC curve (area = 1.00)
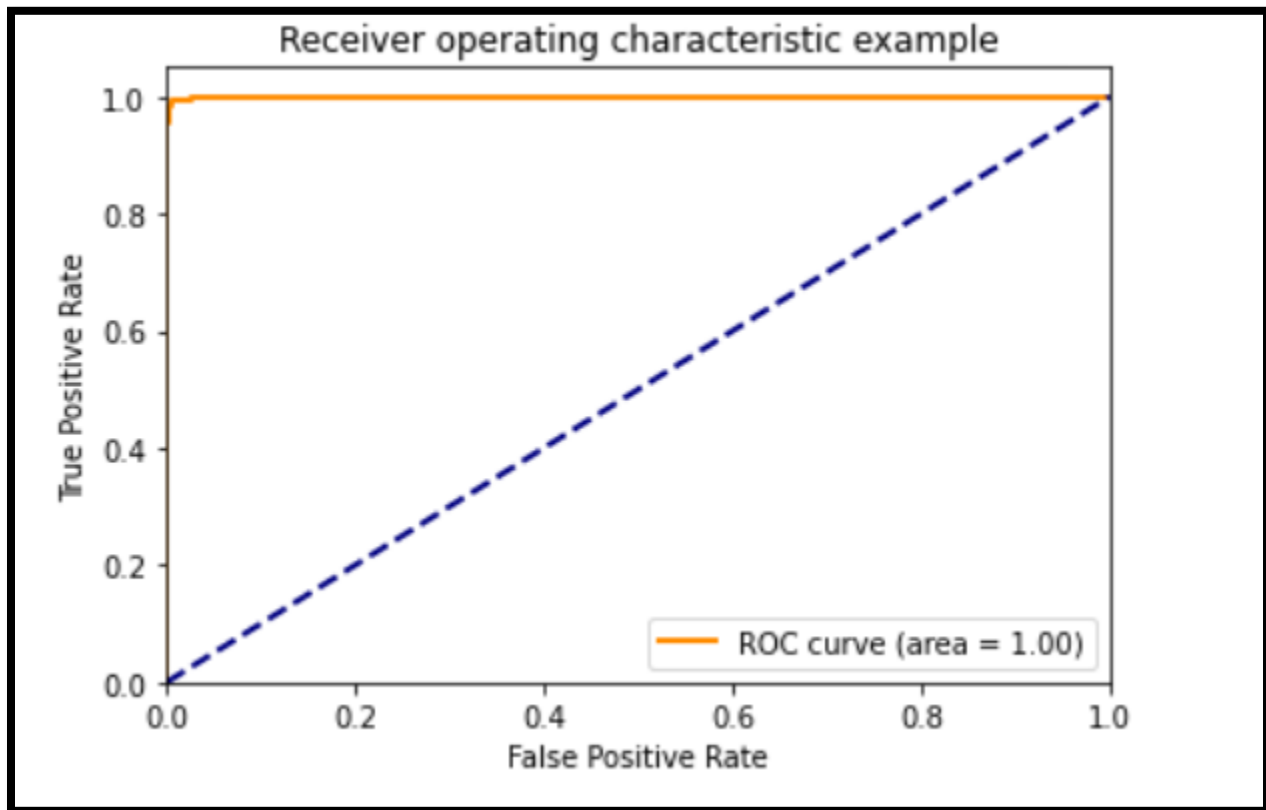
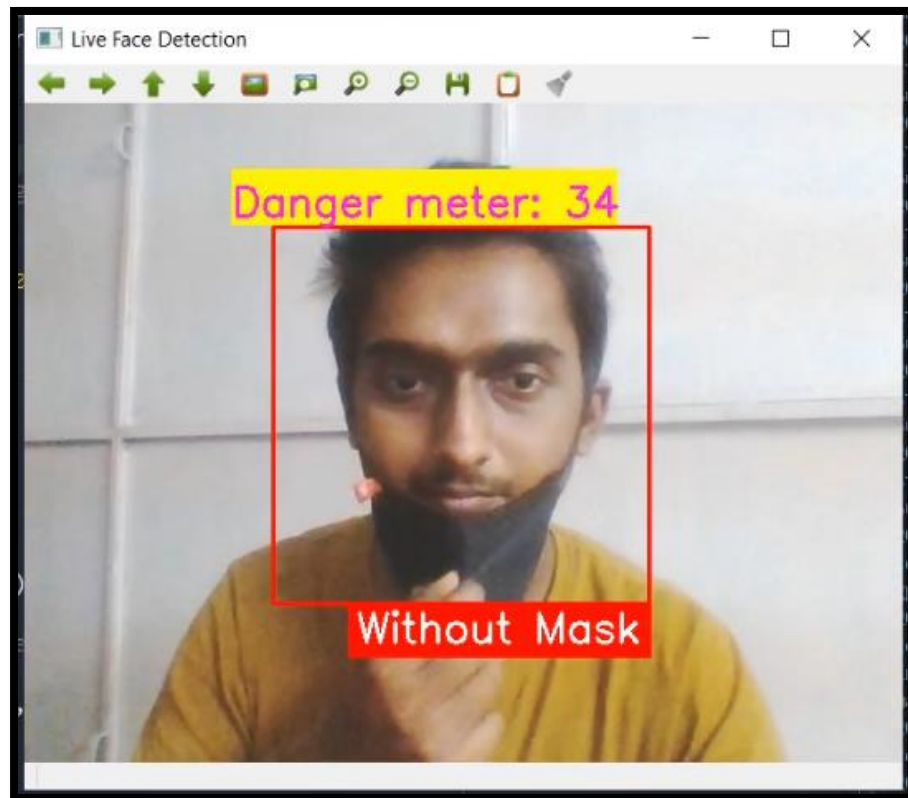## 9. Output Predictions

1) If person is wearing Mask
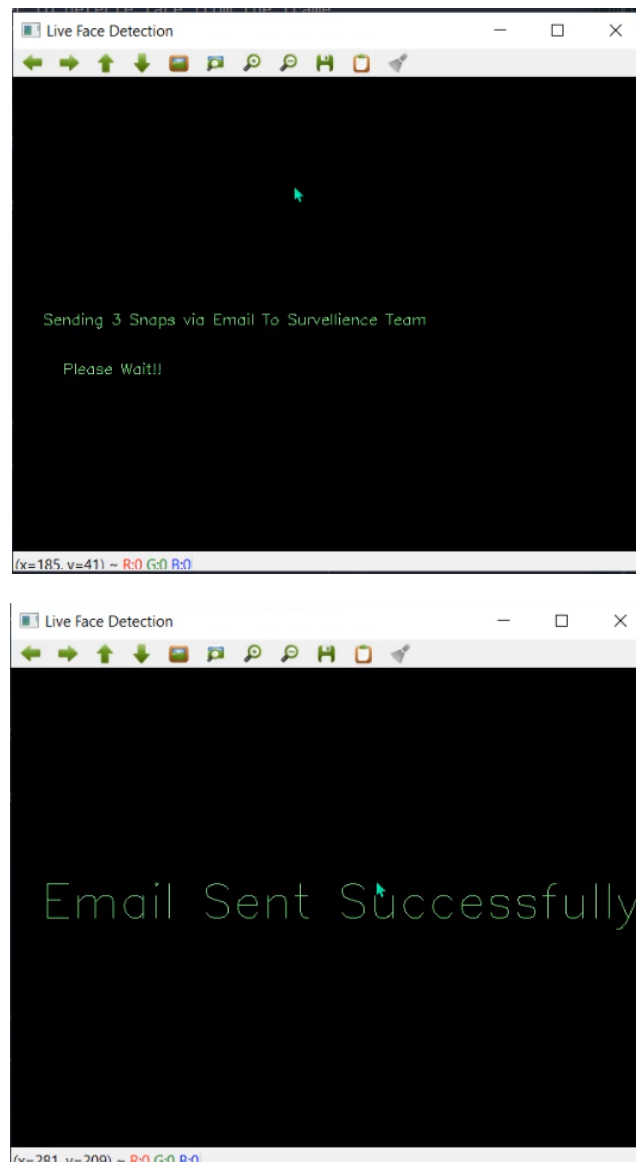


2) If the person not wearing mask

3) When no person detected



4) If person wearing mask below chin

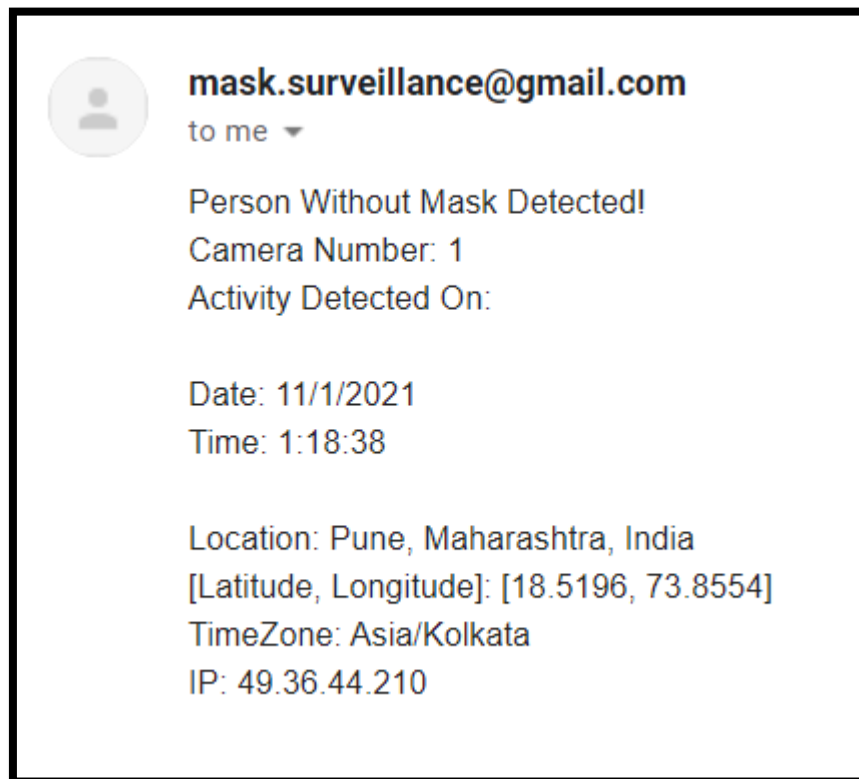5) When damage meter reaches 30,60,90 these 3 frames are captured and saved onto local drive



image 0.png        image 1.png        image 2.png

6) When danger meter reaches 90 email is sent with the attachments and location, time details to surveillance team.

**mask.surveillance@gmail.com**
to me ▾

01:18 (2 hours ago)  ☆  ↩  ⋮

Person Without Mask Detected!
Camera Number: 1
Activity Detected On:

Date: 11/1/2021
Time: 1:18:38

Location: Pune, Maharashtra, India
[Latitude, Longitude]: [18.5196, 73.8554]
TimeZone: Asia/Kolkata
IP: 49.36.44.210

**3 Attachments**  ↓  ⧉



**mask.surveillance@gmail.com**
to me ▾

Person Without Mask Detected!
Camera Number: 1
Activity Detected On:

Date: 11/1/2021
Time: 1:18:38

Location: Pune, Maharashtra, India
[Latitude, Longitude]: [18.5196, 73.8554]
TimeZone: Asia/Kolkata
IP: 49.36.44.210

# THANK YOU

Created By: Hrishikesh Sunil Shinde