Intruder Detector for Webinar with Face Detection

Kumara K.K.K.P. [1]
Department of Information Systems Engineering,
Faculty of Computing Sri Lanka Institute of
Information Technology,
Malabe,Sri lanka
it20228880@my.sliit.lk[1]

Chaminda P.I. [1]
Department of Information Systems Engineering,
Faculty of Computing Sri Lanka Institute of
Information Technology,
Malabe,Sri lanka
it20223144@my.sliit.lk[1]

***Abstract*— Statistics show that employees, people, and organizations carry out their day-to-day operations remotely as a result of the epidemic. Because of the epidemic, all organization meetings, educational sessions, and special events have been transferred to online webinar platforms. There is a risk that confidential information, such as corporate objectives, trade secrets, and future plans, could be disclosed during these webinar meetings. It is imperative that any sensitive information as well as essential data that is discussed during the meetings be kept secure. During the course of the application demonstration, it is important to secure sensitive data and information by releasing it for the goals of an attacker. The levels of confidentiality that are assigned to certain pieces of information are established according to how sensitive they are. Users require an authorization technique in order to identify those individuals who are present at the meetings. The purpose of the Intruder Detector V1.0 is to identify potential trespassers and prevent them from joining the ongoing meeting.**

Keywords—FaceRecognition,Model training,Intruder,Confidentiality

I. INTRODUCTION

Today, in the age of industry 4.0, machine learning technology is quickly evolving. There has been a lot of work done in this area, including examples of its application in several fields, as shown in . Machine learning allows for specialized training in a variety of domains, allowing for applications in areas as diverse as image voice , shortest path , and video pattern recognition . [1]In addition, there have been advancements in IoT technology that allow data to be transferred over the internet without any human intervention. Typically, this method is used for data transmission from microcontroller-based sensors to single-board computers (gateways) and the cloud . One example of an intelligent system that could benefit from the combination of machine learning and IoT is a real-time attendance system. Most current attendance tracking systems still rely on fingerprint scanning, so this approach could help streamline those (including RFID-based machines). Time is a major factor, especially in the initial phases of sign-up and fingerprint verification . A person's or people's presence in a meeting room can be detected rapidly and precisely using machine learning. [2] For this machine to learn and perform well in detection, it needs a large amount of information. Face detection is one area of research that shows a lot of promise for future expansion and development. There have been a significant number of studies on face recognition that are based on machine learning.

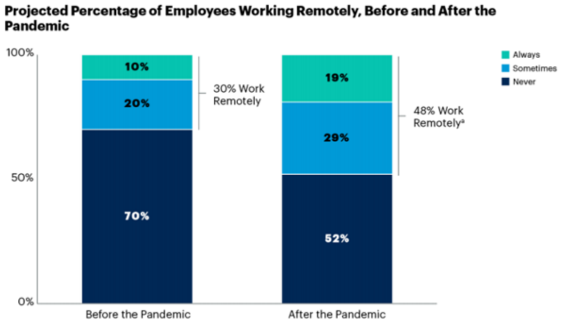


Figure 1 statistics of increasing remote workers after pandemic

According to the statistics [3]given by above images with the pandemic employees and people and organizations do their day to day operations remotely

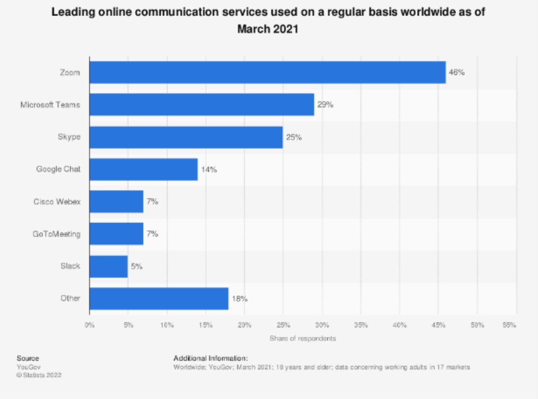


Figure 2statistics about webinar platforms

As well as according to 2nd image of statistics it is about the using of webinar applications in 2021 with the pandemic situation all meetings of organizations [4], education sessions , special events are moved to online webinar platforms .

Sensitive data can be leaked which discussed on this webinar meetings such as business goals, trade secrets ,future plans. Anyone can join the session as legitimate user of the organization because there is less authorization methods are implemented in default webinar applications. As an example if there is name is right organization approve the entering to the meeting session. So we need authorization method for identify people who attend to that meetings. For that we introduce to you Intruder Detector V1.0

II. EASE OF USE

A. Why intruder detector?

Intruder detector for webinar is a research-based product which using face recognition techniques .
With the provided link any intruder can join the session, but some Online meeting platforms organization can admit only the persons with the names and checking manually in the meeting. It is a time-wasting process. As well as some intruder can identity theft and log to the meetings. So it is a major problem to protect the confidentiality of the company. Because of that problems and as a solution we are introducing our Intruder detector for webinar product.

B. How to use this product?

Any organization which implemented this product they can identify intruders who coming to the very confidential meetings. Organization provide a invite link before the meeting .Intruders also can get that link .But that link is our product link ,not the real meeting link By this product filtering all people who has the invite link that provided by the organization. After by a face recognition system that implemented in product identify all and only authenticated people are redirecting to the real meeting link. Intruder detector run as a web application by using python and flask environment. As a developing team of intruder detector is trying to provide better interfaces for users of this product.

C. Importance of Intruder Detector

Identify intruders who trying to attend confidential meetings ,Attackers who try to compromise a network's security are known as intruders. In this scenario with the face recognition techniques application identifies people who join the sessions so application deny the access request to the meeting to unauthorized people after face recognition process .Monitoring people which attending to the meetings and Get an attendance Reports People should do the security verification with the face recognition after that they can go the meeting after the verification that data goes to the database by that application provides the who were attended to the meeting .This feature can be essential feature for audit team also. When using the application, keep confidential information and important data safe. When demonstrating the application, protect sensitive data and information by disclosing it for attacker purposes. The confidentiality of information is determined by application data and information that are prioritized based on their sensitivity and whether they are intercepted or observed. When using applications on the system, information can be more than secure and can't be broken, making it complex for hackers to access the information and data. even if it is preventing unauthorized access by hiding. Given the importance of data loss prevention, this could be a simple system to implement. According to the software implementation, it aids in the prevention of data and information loss.

III. FACE RECOGNITION

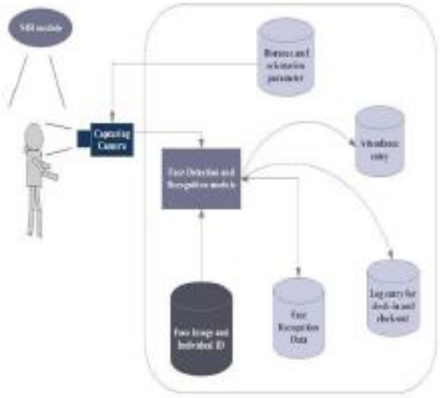

Capturing Camera
Face Detection and Recognition module
Attendance entry
Face Image and Individual ID
Face Recognition Data
Log entry for check-in and checkout

Figure 3 big picture of face recognition system

Ensuring that the system is based on some activities performed for authentication purposes improves its ability to be secured through the use of some identification methods, such as face recognition. Identifying the authentication face and ensuring that that type of face from people is authenticated to the organization's system security [5]When using face recognition, you must use system authentication to implement some security controls and functions. This can be useful for those who have been designated as authorized management and employees who require access to information and data. [6]When someone gains unauthorized access to a system, he or she has access to the database, which should be isolated. As a result, he or she can do anything without gaining access to the system. Face identification must involve the user who must access the system as well as the sensitive data.

A. Detection

[7]Face extraction begins detection. Marking facial features follows. Age and stature don't modify some facial traits. Distance between eyes, eye socket depth, and nose shape. 80 'landmarks' exist. These landmark measurements produce a code. This code is a person's 'faceprint'

B. Matching

Faceprint is matched with system prints. The image undergoes multiple technological stages to assure accuracy. Most of our datasets are 2D photographs, thus the images must be processed. This entails removing face landmarks to make them 3D. Low-resolution images must be encoded and decoded to obtain high-resolution information. Lighting, face expression, and angles must be considered.

C. Identification

Identification depends on whether the program is used for monitoring or authentication. [8]This phase should match the subject 1:1. This can be done in several ways: a rapid pass to reduce alternatives, then complicated layers take over. Some companies use skin texture to improve facial recognition accuracy.

D. Key features of Intruder Detector

• Trained and growing database

Any FRS's accuracy depends on its AI-trained database. The data must increase and be gender and racially diverse. Lighting, angles, and face expressions must vary in training data. A good database includes several image resolutions for the system. The FRS is a fantastic database for machine learning applications to learn from.

• Algorithm accuracy:

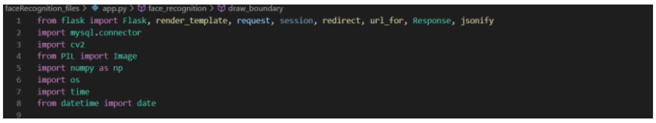
When evaluating a FRS, the two most important metrics to consider are the false acceptance rate (FAR) and the false rejection rate (FRR). False-identity matching (FAR) occurs when two images are incorrectly compared and considered to be the same. If you're trying to use it as a security measure, the wrong person might get in. False positive image matching (FRM) is a technique whereby identical pictures are incorrectly categorized as variants. The appropriate individual might be shut out here. The FAR needs to be small, while the FRR needs to be large, for any realistic security scenario

IV. Methodology

Any company or group that has this product deployed will be able to identify any trespassers that come to highly confidential meetings. Before the meeting, the organization will give a link to the invitation. Intruders can also get that URL if they try. However, that link goes to our product page and not to the actual meeting page. With the use of this solution, the organization can filter out everyone who has the invite URL that was issued to them. After being identified using a facial recognition technology that was included into the product, each individual, and only those who have been validated, are redirected to the actual meeting link. Python and flask are the environments that are used to run the intruder detection as a web application. The intruder detector development team is working hard to improve the user interfaces offered by this product in order to meet customer demand.

A. Setup dependencies

The main objective of this step is to import dependencies. Mainly numpy, open cv environment, pillow and mysql.connector. Open cv environment is used to get camera environment and pillow supports to the face recognition part as well as we have to install mysql.connector to make connections with the database User need to install the libraries by using pip install

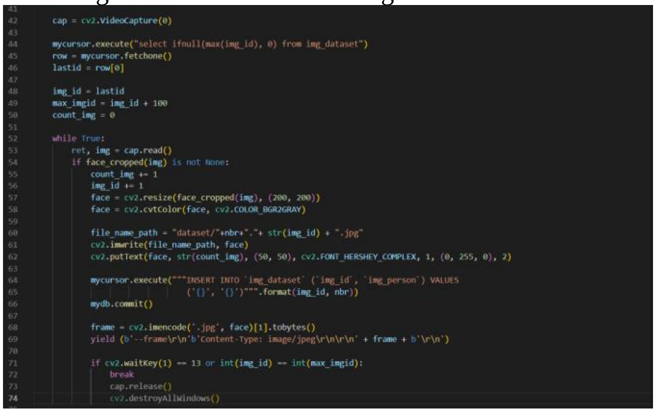


```
faceRecognition_files > app.py > face_recognition > draw_boundary
from flask import Flask, render_template, request, session, redirect, url_for, Response, jsonify
import mysql.connector
import cv2
from PIL import Image
import numpy as np
import os
import time
from datetime import date
```

Figure 4 dependencies that want to run program

B. Data collection for train the model

In line 42 system should get the web camera input by using OpenCV dependencies then it goes to a while loop and by these code segment capture frames of the person who Infront the camera .It takes 100 frames to store in application database and they are used to data training session. After taking 100 frames camera should break according to the last line of code segment



```
42      cap = cv2.VideoCapture(0)
43
44      mycursor.execute("select ifnull(max(img_id), 0) from img_dataset")
45      row = mycursor.fetchone()
46      lastid = row[0]
47
48      img_id = lastid
49      max_imgid = img_id + 100
50      count_img = 0
51
52      while True:
53          ret, img = cap.read()
54          if face_cropped(img) is not None:
55              count_img += 1
56              img_id += 1
57              face = cv2.resize(face_cropped(img), (200, 200))
58              face = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)
59
60              file_name_path = "dataset/"+nbr+"."+ str(img_id) + ".jpg"
61              cv2.imwrite(file_name_path, face)
62              cv2.putText(face, str(count_img), (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2)
63
64              mycursor.execute("""INSERT INTO `img_dataset` (`img_id`, `img_person`) VALUES
65                                  ('{}', '{}')""".format(img_id, nbr))
66              mydb.commit()
67
68              frame = cv2.imencode('.jpg', face)[1].tobytes()
69              yield (b'--frame\r\n'b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')
70
71              if cv2.waitKey(1) == 13 or int(img_id) == int(max_imgid):
72                  break
73                  cap.release()
74                  cv2.destroyAllWindows()
```

Figure 5 code segment for open the camera for data collection

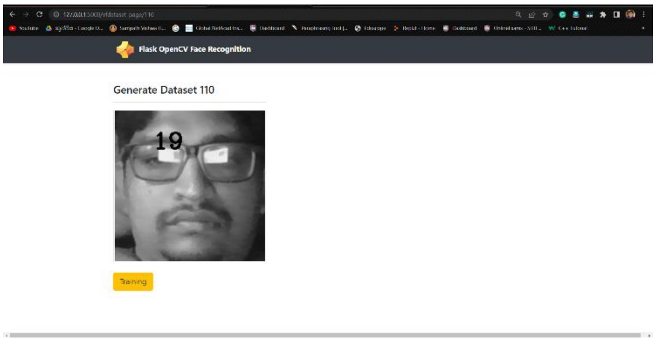
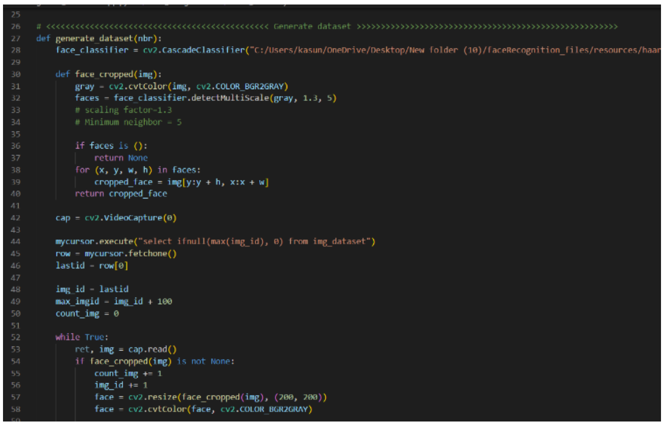


Figure 6 output of code, collecting 100 frames for train the model and generate the dataset

Then need to store by generating the dataset for each user or employee in the organization



```
25
26  # <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< Generate dataset >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
27  def generate_dataset(nbr):
28      face_classifier = cv2.CascadeClassifier("C:/Users/kasun/OneDrive/Desktop/New folder (10)/faceRecognition_files/resources/haar
29
30      def face_cropped(img):
31          gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
32          faces = face_classifier.detectMultiScale(gray, 1.3, 5)
33          # scaling factor=1.3
34          # Minimum neighbor = 5
35
36          if faces is ():
37              return None
38          for (x, y, w, h) in faces:
39              cropped_face = img[y:y + h, x:x + w]
40          return cropped_face
41
42      cap = cv2.VideoCapture(0)
43
44      mycursor.execute("select ifnull(max(img_id), 0) from img_dataset")
45      row = mycursor.fetchone()
46      lastid = row[0]
47
48      img_id = lastid
49      max_imgid = img_id + 100
50      count_img = 0
51
52      while True:
53          ret, img = cap.read()
54          if face_cropped(img) is not None:
55              count_img += 1
56              img_id += 1
57              face = cv2.resize(face_cropped(img), (200, 200))
58              face = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)
```

Figure 7 code segment for generate dataset and store them to database

Application takes 100 frames of face and according to the code it assigns a number to each photo with regards of dataset number like following.

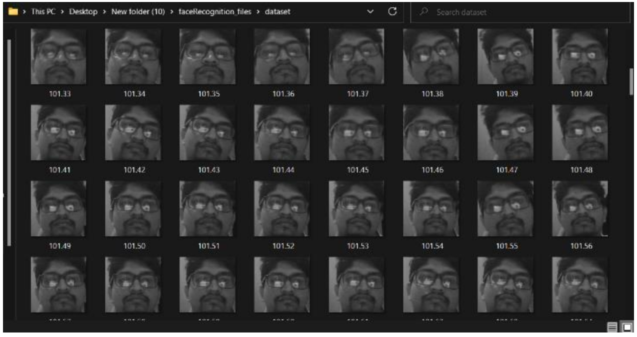
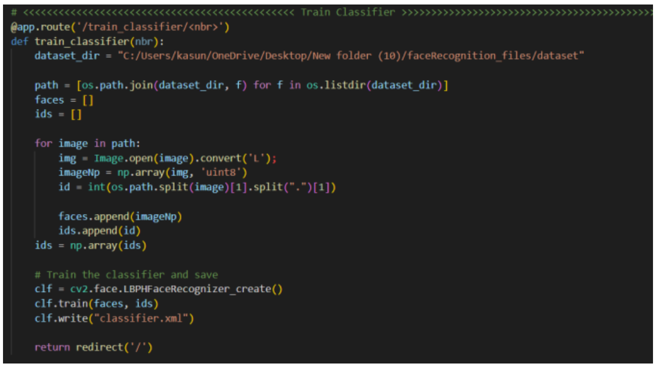


Figure 8 generated dataset of user

C. Train the model

With the dataset that application have collected .Following coding part do the training the model for face recognition and it writes data on classifier.xml file

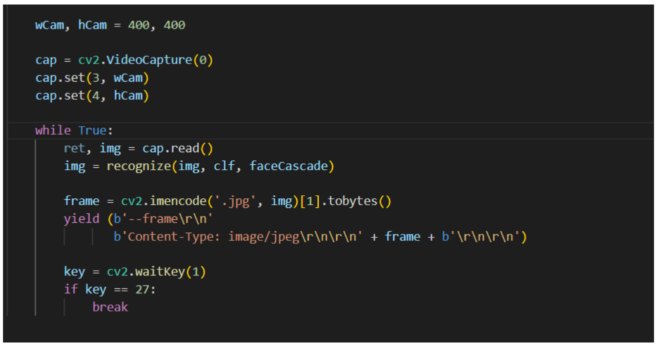


```
# <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< Train Classifier >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
@app.route('/train_classifier/<nbr>')
def train_classifier(nbr):
    dataset_dir = "C:/Users/kasun/OneDrive/Desktop/New folder (10)/faceRecognition_files/dataset"

    path = [os.path.join(dataset_dir, f) for f in os.listdir(dataset_dir)]
    faces = []
    ids = []

    for image in path:
        img = Image.open(image).convert('L');
        imageNp = np.array(img, 'uint8')
        id = int(os.path.split(image)[1].split(".")[1])

        faces.append(imageNp)
        ids.append(id)
    ids = np.array(ids)

    # Train the classifier and save
    clf = cv2.face.LBPHFaceRecognizer_create()
    clf.train(faces, ids)
    clf.write("classifier.xml")

    return redirect('/')
```

Figure 9 code segment for train the model and saves them in classifier.xml

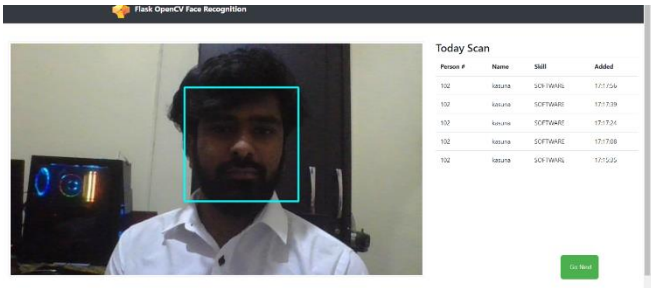
D. Face Recognition

By using the dataset application uses that data to verify people who comes to the application .When looking at the code first we had to define sizes of camera input .



```
    wCam, hCam = 400, 400

    cap = cv2.VideoCapture(0)
    cap.set(3, wCam)
    cap.set(4, hCam)

    while True:
        ret, img = cap.read()
        img = recognize(img, clf, faceCascade)

        frame = cv2.imencode('.jpg', img)[1].tobytes()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')

        key = cv2.waitKey(1)
        if key == 27:
            break
```

Figure 10 code segment for face detection



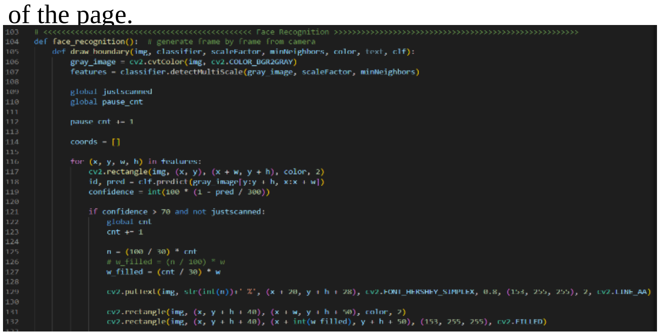
Today Scan

Person #	Name	Skill	Added
102	kasuna	SOFTWARE	17:17:56
102	kasuna	SOFTWARE	17:17:39
102	kasuna	SOFTWARE	17:17:24
102	kasuna	SOFTWARE	17:17:08
102	kasuna	SOFTWARE	17:15:35

Figure 11 web application output for face recognition

By using web camera generate frame by frame to identify the person that in front of the camera and it uses the dataset that we trained in previous part.

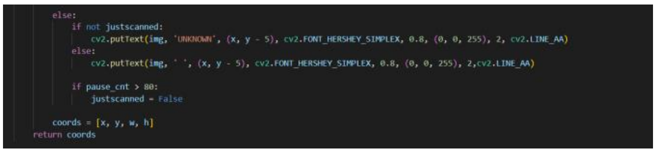
If there is data about the person in the database by following command application send their attendance to the database .After that result is shown to the user also from right corner of the page.



```
# <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< Face Recognition >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
def face_recognition():  # generate frame by frame from camera
    def draw_boundary(img, classifier, scaleFactor, minNeighbors, color, text, clf):
        gray_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        features = classifier.detectMultiScale(gray_image, scaleFactor, minNeighbors)

        global justscanned
        global pause_cnt

        pause_cnt += 1

        coords = []

        for (x, y, w, h) in features:
            cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)
            id, pred = clf.predict(gray_image[y:y + h, x:x + w])
            confidence = int(100 * (1 - pred / 300))

            if confidence > 70 and not justscanned:
                global cnt
                cnt += 1

                n = (100 / 30) * cnt
                # w_filled = (n / 100) * w
                w_filled = (cnt / 30) * w

                cv2.putText(img, str(int(n))+' %', (x + 20, y + h + 28), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (153, 255, 255), 2, cv2.LINE_AA)

                cv2.rectangle(img, (x, y + h + 40), (x + w, y + h + 50), color, 2)
                cv2.rectangle(img, (x, y + h + 40), (x + int(w_filled), y + h + 50), (153, 255, 255), cv2.FILLED)
```

Figure 12 full code segment for face detection and how to match with trained dataset

When face recognition is started user can see the progress of their scanning with a percentage that part is done by this code segmentation, if there are no data about person who in front of the camera it shows the unknown.



```
            else:
                if not justscanned:
                    cv2.putText(img, 'UNKNOWN', (x, y - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 0, 255), 2, cv2.LINE_AA)
                else:
                    cv2.putText(img, ' ', (x, y - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 0, 255), 2,cv2.LINE_AA)

                if pause_cnt > 80:
                    justscanned = False

            coords = [x, y, w, h]
        return coords
```

Figure 13 code segment for show the progress on video feed

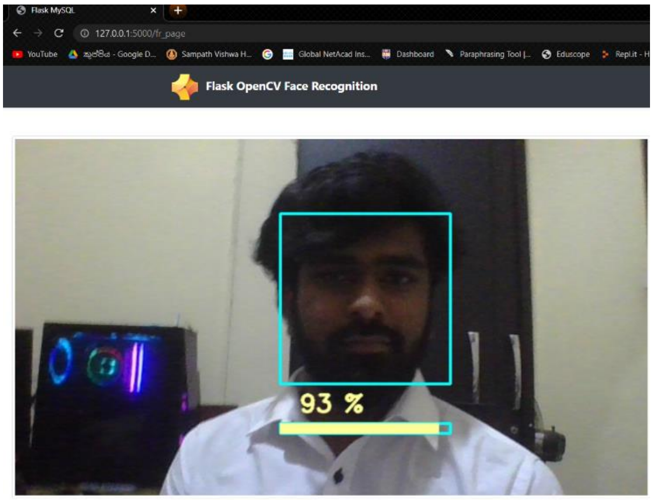


Figure 14 output of web application with showing the progress of scanning the face of user

After 100% completed scanned data stores in the database and that data can use for create attendance reports also. This application is using the database for get dataset and some inserting processes should be done by the application . so we used following codes for that .

V. DESIGN AND IMPLEMETATION CONSTRAINTS

Intruder detector can run on windows operating systems System (server) requirements : Minimum 8GB RAM(Random Access Memory)Average CPU usage Intruder detector v1.0 is powered by python language with flask environment .It need python 3.9 + to run the program. Database should be maintained .According to the customer requirements can be modified some functions.

VI. IMPLEMETATION

Step one is identify the business scope In the step one team should identify what are the critical information in the organization and what are the content of their meetings ,team should consider content of meetings, Are they discuss sensitive information in their online meetings?. Who involved into for discuss important facts of organization on meeting and.Audit the attendance of the meetings previous

Second step is Implementation of the product and enter the necessary data to database In this step , team implement the intruder detector system and enter the data that identified in step 01 and check whether entered data is correct .Then, Training models for face recognition. In this phase , application need the 100 photos of each of employees to train the model to identify in face recognition part .So all employees need to train their data and enter the database as registering to the application. If there is no data in database that employee can not attend any meeting in the organization. After entering the all necessary data to databases , using some employees as test cases should be tested application properly working or not.
At the last step , all employees should awareness about the policies .As well as they should training how to work with this application.

VII. Secure testing

By locating security flaws and vulnerabilities in source code, application security testing (AST) strengthens applications' resistance to security threats.
AST was initially a manual procedure. AST needs to be automated in the modern day due to the increasing modularity of corporate software, the enormous amount of open source components, and the great number of known vulnerabilities and threat vectors. The majority of businesses combine various application security technologies [9]

A. White box testing

We can use tool such as SonarQube to identify vulnerabilities of the code reviewing

B. Black box testing

run code, inspect it at runtime, and look for problems that might be security flaws. [10]This can involve problems with query strings, requests and answers, script usage, memory leaks, management of cookies and sessions, authentication, running third-party components, data injection, and DOM injection, among other things. Development team ran the code and it works perfectly according to the purpose so intruder detector could achieve the black box testing easily.

VIII. Future improvements of application

Monitor user activities and webinar platform for improvement is our next target, According to the implementation of the application, there must be some improvement in the application's effectiveness and efficiency. To do so, we must implement a system for monitoring user logs and user activities. Ensure that when an authenticated user logs in to the system via our application, a record of the user's activities during the specified time period is kept. For the user to use the application, it must have disclosure activations. When a user needs access to a user log or other types of tracking , he should send an email to management. An authenticated user can view the activities performed in the application during their session. as well as to applications running on the system, maintain a webinar template for getting user logs and a database for the identification of user activities.

IX. Conclusion

When it comes to sensitive information, confidentiality refers to the idea and practice of keeping it secret until the owner or data custodian explicitly consents to sharing it with another party. Another definition of confidentiality is the request to uphold the rule and custom.
The primary objective of the application is the preserve the confidentiality .With the face recognition system only authorized people can access the meeting and they can get the confidential facts of organization.

Acknowledgment

This research was made possible by Dr. Lakmal Rupasinghe and Mrs. Laneesha Ruggahakotuwa, an assistant lecturer at SLIIT. We would also want to extend our gratitude to all of the instructors at the institution who shared their knowledge and insights, which were of tremendous assistance to the research.

References

[1] N. A. A. R. Z. M. Z. A. Behzad Shoarian Satari, "Face Recognition for Security Efficiency".

[2] W. L. W. a. S. S. D. M. A. Dabbah, "Secure Authentication for Face Recognition".

[3] "fitsmallbusiness.com," [Online]. Available: https://fitsmallbusiness.com/remote-work-statistics/?__cf_chl_tk=LG8fgdyi90h0KbM52KkvpnzaPFdzoxCe7S4mqSIV1Ys-1670530413-0-gaNycGzNB30.

[4] "Webinar Statistics, Trends And Facts 2023," [Online]. Available: https://abdalslam.com/webinar-statistics.

[5] "www.techtarget.com," [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/facial-recognition.

[6] M. K. D. A. S. a. D. R. P. Nirmalya Kar, "Study of Implementing Automated Attendance System".

[7] "Top 11 Facial Recognition Software in 2021," [Online].

[8] C. Mingtsung, "Research on the Application of Face Recognition System".

[9] "www.sonarqube.org," [Online]. Available: https://www.sonarqube.org/developer-edition/?gads_campaign=Class-3-Brand-SQ&gads_ad_group=SonarQube&gads_keyword=sonarqube&gclid=CjwKCAiAs8acBhA1EiwAgRFdwwp1Ao4fQsofPGU_6CeXDPIkcnaCvRw0NNvKGxoj8BJluy5GlUvLDBoCuksQAvD_BwE.

[10] "www.guru99.com," [Online]. Available: https://www.guru99.com/black-box-testing.html.