Abstract

In Modern day, with such advancement in technology, the world is going touchless for security, driving, and many such activities. In this paper, we have tried creating an advanced security system for face and fingerprint recognition by touchless means. Though much efforts can be put to further increase the efficacy and accuracy of the prototype, but this is just a beginning.

First, Face image will be captured and matched with the already present images in the database using a pre-trained model based on Deep Neural Network (DNN) which takes 128 different measurements for face identification. Age and Gender prediction model and prototype is also being used to predict gender and probable age range. Second, if step 1 is successful, the prototype asks for fingerprint image using second camera. The image is then processed to extract fingerprint lines similar to minutiae extraction with the help of Gaussian Adaptive Threshold Algorithm. The lines are then sharpened, blurred and enhanced. The final processed image is then matched with the already saved images in database. The points in the images are matched using SIFT algorithm which enables to detect even if the image is at different angle, size, blurred or any other discrepancy.

Lastly, if both Face and Fingerprint are matched with the same individual data, the prototype grants the access.

Keywords: OpenCV, DNN, Face detection, Face identification, Fingerprint recognition, Gaussian Adaptive threshold, SIFT algorithm.

Introduction

Purpose

The purpose and the reason behind preparing this report is to develop a project based on face identification and verification and fingerprint recognition and give a preview on how we are going to make this software work and how the software will be able to secure any systems using the software that we are building using the technologies. Fingerprints are one of the most popular human features in biometrics safety systems, that's the reason that we included fingerprint recognition in the project. And face verification is also one way of differentiating humans.

Scope

We are going to build a system that will have double system security means, double safety. Both face identification and verification and fingerprint recognition are embedded in the system for the double security. With massive amounts of crime around the world, the system will give high security by using different methods to track and identify people in public places. The system which we are going to build will be used for a prolonged period of time in the concerned place, whenever authorization is required.

1. In order to prevent the frauds of ATM. Whenever user will enter in ATM his photograph will be taken to permit the access after it is being matched with stored photo from the database.
2. Duplicate voters are being reported at the time of voting, the resolution camera and face recognition face 100% and generates the recognition for voting if match is found.
3. Driving license verification can also be exercised face recognition technology as mentioned earlier.
4. In online education, monitoring the attendance of the students is very important as the presence of students is part of a good assessment for teaching and learning.
5. To identify and verify terrorists at airports, railway stations and malls the face recognition technology will be the best choice.

Method

Building such a system from scratch using the Python programming language helped achieve a better understanding of the field as well as its advantages and disadvantages compared to other biometric authentication methods. After some research, the decision to do face detection using an OpenCV library and also pretrained models for Python and we also used face recognition using face recognition library was unavoidable due to not having a system that could reliably do both detection and recognition in the project's circumstances. these were the algorithms used for face detection and identification. It was so important that we arrange the things correctly in order to make all the algorithms perform and give the appropriate and efficient result or the output. Apparently, Gaussian adaptive threshold algorithm is used for the extraction of the fingerprint and we also used Flann based matcher for the matching of the fingerprints, these were the algorithms used for fingerprint recognition.

Objective

Security systems have become an integral part of our environment, having a safe and accurate security system is of the highest importance.

Face detection has magnetized immense attention because it has many applications in computer vision communication and automatic control system. Face detection is a method to detect a face from an image which have several attributes in that image. Research into face detection. By giving a single image, challenge is to detect the face from that image. Face detection is a challenging task because faces are not rigid and it transmutations in size, shape, colour etc.

Human fingerprints are affluent in details called minutiae, which can be utilized as identification marks for dactylogram verification. The goal of this project is to develop a consummate system for dactylogram verification through extracting and matching minutiae. To achieve good minutiae extraction in dactylograms with varying quality, pre-processing in form of image enhancement and binarization is first applied on dactylograms afore they are evaluated. Many methods have been amalgamated to build a minutia extractor and a minutia matcher. Minutiae marking with special consideration of the triple branch counting and erroneous minutiae abstraction methods are utilized in the work. Performance of the developed system is then evaluated on a database with dactylograms from different people.

Problem Statement

In the real world there is a big problem for security. In our daily life we hear about many incidents, which can also be seen in many movies, when there is a security lapse by duplicating the biometric identity of a person. Using both face identification and fingerprint recognition at the same time as a security system will make it almost impossible to bypass the security even after duplicating the biometrics as the system will be comparing many characteristics of the concerned person. Also in the recent year, due to covid 19 people are afraid to touch anything so this is also a problem to touch fingerprint scanner by many people.

In this project we will be making a software that will give more security for the user by recognizing the users face and the users' fingerprint. We are also using a method for fingerprint recognition which is touchless. We can use this software as a login system for any websites or for any apps. We can also use this in cars for unlocking the doors by recognizing face and the fingerprints of the users. This software can also be used in university/schools as the attendance system of students, teacher and staffs.

Functionality

This program will first open the camera using OpenCV to get the face of person. Detection of face is done using a pre-trained model named face proto, Which will detect face and make box around the face. After the face is detected, pressing enter will capture a picture of person to recognise as authorise person. A library named face recognition ia used to match the face with the database. It finds 128 different encodings of face and find the best match with the confidence of 50% or higher.

Along with face detection and recognition we also used a pretrained model to predict the age and gender of that particular person for keeping the record of every authorising person.

After face identification, another camera will take an image of finger then it will be cropped accordingly and converted to grayscale. This image is enhanced and fingerprint is extracted using the “Gaussian adaptive thresholding”. Then the fingerprint obtained is matched to the fingerprint already stored in the database using “Flann based matcher”.

If fingerprint and face both is matched to same person then only person will be authorised and its name with gender and predicted age will be printed.

Literature survey

S.NO	Author	Year of Publication	Title of page	Points to Ponder	Publication/Publisher detail
1.	A. Saxena, P. Singh, S. Narayan Singh	2021	Gender and Age detection using Deep Learning.	• Computer vision involves acquiring, processing, analyzing and extracting information from the images. • For face detection a deep neural network(DNN) is used. • Libraries used: OpenCV,Tensorflow, Keras, Argparse, math. • Deep learning and CNN can be implemented to accurately identify gender and age.	IEEE 11th International Conference on Cloud Computing, Deep Learning & Engineering (Confluence)
2.	N. Boyko, O. Basystiuk, N.Shakhovska	2018	Performance Evaluation and Comparison of Software for Face Recognition, based on Dlib and OpenCV Library	• Face detection based on dlib and OpenCV libraries. • Analysis of performance of OpenCV and dlib libraries based on HOG method implemented in python. • Method: Find faces, Face positoning, Defining unique facial features, identification of a person. • DCNN is used to identify 128 unique numerical facial features. • OpenCV library is more productive and has better	IEEE, Second International Conference on Data Stream Mining & Processing

				performance for face detection and detection.	
3	E. P. Ijjina, G. Kanahasabai, A. Srinivas Joshi	2020	Deep Learning based approach to detect Customer Age, Gender and Expression in Surveillance Video	• Used haar cascade object detection model . • For age and gender estimation it used Wide ResNet 16-8 (WRN-16-8) a Wide Residual Network [16] architecture to estimate age and gender • For Expression recognition it used mini Xception model inspired by well versed Xception architecture • Achieved a 82.9% accuracy rate for gender and 70.8% accuracy for age-range.	IEEE, 11th ICCCNT IIT – Kharagpur
4.	Md. M. H. Ali, Vivek H Mahale, P. Yannawar, A. T. Gaikwad	2016	Overview of Fingerprint Recognition System	• Image Capture: the optical fingerprint reader is used to capture the image of fingerprint and by using ink in the area of finger and then put a sheet of white paper on the fingerprint. • Image Pre-processing Stage: The Gaussian filter, Short Time Fourier Transform analysis are adopted to enhance fingerprint image quality	IEEE, International Conference on Electrical, Electronics, and Optimisation Techniques (ICEEOT)

				• Minutiae extraction algorithm is used. Two types of minutiae points; Ridge ending and Ridge bifurcation.	
5.	A. L. H. Jin, L. Chekima, J. A. Dargaham, L. C. Fan	2002	Fingerprint Identification and Recognition Using Backpropagation Neural I Network	• The back propagation neural network is used in order to train the network • Features extracted from the fingerprint image can be classified into local features and global features, thus increasing the accuracy of further fingerprint identification and recognition process	IEEE, Student Conference on Research and Development Proceedings, Shah Alam, Malaysia
6.	Md. M. H. Ali, Vivek H Mahale, P. Yannawar, A. T. Gaikwad	2016	Fingerprint Recognition for Person Identification and Verification Based on Minutiae Matching	• They used two standard databases which are available online(FVC2000 and FVC2002) • They used minutiae based approach which consists of two approaches ,minutiae detection and minutiae matching . • The accuracy for the two datasets FVC2000 and FVC2002 are 80.03% and 98.55% respectively.	IEEE, 6th International Conference on Advanced Computing

7.	Lin Hong, Anil Jain	1998	Integrating Faces and Fingerprints for Personal Identification	• They integrated face and fingerprint to enhance the accuracy of identification . • Decision fusion which integrates multiple cues has proved beneficial for improving the accuracy of a recognition system. • False reject rates of different methods: Face:-64.1% Fingerprint:- 14.9% After integration:- 9.8% . Accuracy increased.	IEEE, Transactions on Pattern Analysis and Machine Intelligence (Vol. 20)
8.	Asad Mustafa, Kevin Meehan	2020	Gender Classification and Age Prediction using CNN and ResNet in Real-Time	• Classification of the image as male or female are being trained using the neural network. KNN classifier has been used in this architecture for classification. • The aim of their study is to explore the use of saliency, which is equivalent to the human visual system to classify the gender and predict age. • achieved an overall accuracy of approximately 85% for age detection.	IEEE, International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI)
9.	A. Ghildiyal, S. Sharma, U. Verma, U. Marhatta	2020	Age and Gender Predictions using Artificial Intelligence Algorithm	• Face detection with Haar cascades used in OpenCv • Gender and age recognition with CNN algorithm. The CNN's output layer consists of 5 values for 2 gender and 5 age classes ("1–14",	IEEE, Proceedings of the Third International Conference on Intelligent Sustainable Systems (ICISS)

				"14–25", "25–40", "40– 60", "60-").	
10.	G. Aguilar, G. Sanchez, K. Toscano, M. Salinas, Nakano, H. Perez	2007	Fingerprint Recognition	• A combination of Fast Fourier Transform (FFT) and Gabor Filters is used for the enhancement the fingerprint image. • Best algorithms for the enhancement of fingerprints is Gabor Filter. • The performance of a fingerprint feature extraction and matching algorithms heavily depends upon the quality of the input fingerprint image • It has a accuracy of 94.1%,with false acceptance of 2.3% and False rejection of 3.6%.	IEEE, Second International Conference on Internet Monitoring and Protection (ICIMP)
11.	M. Jasmine Pemeena Priyadarsini, G K Rajini, Shaik Naseera, Sardar Bash, Sanjay Kumar, Karunagaran, Kalavagunta Chandra Shekara	2017	Human identification using Face and Fingerprint	• Finger print recognition is the most accurate method of recognition • It can also be surpassed by using fake prints • To overcome this we introduce another facial recognition system which requires both the face and finger print to match to provide authentication.	IEEE - Proceedings of the International Conference on Intelligent Sustainable Systems (ICISS)

12.	Adhiyaman, Ezhilmaran	2015	Fingerprint Matching and Similarity Checking System using Minutiae Based Technique	• Towards the matching, this algorithm is checks each minutiae point for each image and it is tested with query image. • As biometric technology matures, there will be an increasing interaction among the (biometric) market, (biometric) technology, and the (identification) applications	IEEE - International Conference on Engineering and Technology (ICETECH) Coimbatore
13.	Salma Annagrebah, Abderrahim Maizate, Larbi Hassouni	2019	Real-time Face Recognition based on Deep neural network methods to solve occlusion problems	• The major asset of CNN compared to the others algorithms is that in doubt it detects the most imported features without any human assistance. • Not all proposed algorithms used in detection and recognize face are the best for solving occlusion problems.	IEEE

14.	Maciej Szymkowski, Khalid Saeed	2017	A Novel Approach to Fingerprint Identification Using Method of Sectorization	• Biometric methods present a very convenient way to increase the safety of computer system. • Systems secured with passwords can be easily broken because of different hacking strategies. If it comes to Biometrics, spoofing of human traits is not so simple.	IEEE - International Conference on Biometrics and Kansel Engineering
15.	Kai Cao, Anil K. Jain	2018	Automated Latent Fingerprint Recognition	• Latent fingerprints constitute one of the most important and widely used sources of forensic evidence in forensic investigations. • Despite this, efforts to design and build accurate, robust, and fully automated latent fingerprint recognition systems have been limited.	IEEE
16.	Bruno Matarazzo, Jonas Targino, Clodoaldo Lima	2017	Biometric Recognition based on Fingerprint: A Comparative Study	• Biometric recognition refers to the automated identification of individuals based on their physical and behavioral characteristics • Such as fingerprint, face, iris, and voice.	IEEE - Workshop of Computer Vision

17.	Divya Meena, Ravi Sharan	2016	An Approach to Face Detection and Recognition	• Image processing is tool or an algorithm to process an image in order to compress image, enhance image or to extract some useful information from the image • There are two methods for image processing, digital and analog	IEEE - International Conference on Recent Advances and Innovations in Engineering (ICRAIE)
18.	Gurlove Singh, Amit Kumar Goel	2020	Face Detection and Recognition System using Digital Image Processing	• Neural network turns out to be very feasible for the problem of face detection. • The only drawback of this system is that, it requires a very fine-tuned network architecture to get exceptional performance.	IEEE - Proceedings of the Second International Conference on Innovation Mechanism for Industry Applications (ICMIA)
19.	Anagha Dhavalikar, Dr. Kulkarni	2014	Face Detection and Facial Expression Recognition System	• In order to improve detection speed and achieve high robustness, symmetry of the face is checked • All the candidates are removed when symmetry is verified but no facial features are detected.	IEEE - International Conference on Electronics and Communication System (ICECS)

20.	M.Geetha, R.S.Latha, S.K.Nivetha, S.Hariprasath, S.Gowtham, C.S.Deepak	2021	Design of face detection and recognition system to monitor students during online examinations using Machine Learning algorithms	• Eigenface method is used for extracting the facial features through facial vectors • The datasets are trained using an SVM model to improve the detection accuracy.	IEEE - International Conference on Computer Communication and Informatics (ICCCI)
21.	J. S. Shah, M. Sharif, M. Raza, M. Murtaza and S. U. Rehman	2015	Robust Face Recognition Technique under Varying Illumination	• The basic methods for face recognition i.e., PCA and LDA has been used. • OPPM framework, which also use histogram normalization method, is proposed to overcome illumination dilemma. • 5% to 50% recognition rate can be achieved.	Journal of Applied Research and Technology
22.	C. Jiang, Y. Zhao, W. Xu and X. Meng	2009	Research of Fingerprint Recognition	• The scanned image was enhanced, binarized, ridges thinned and then freeman chain code was used to describe fingerprint ridge and matching was done based on fingerprint ridge contour.	2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing
23.	V. Ghenescu, R. E. Mihaescu, S. V. Carata, M. T. Ghenescu, E. Barnoviciu and M. Chindea	2017	Fingerprint Recognition System Based on Big Data and Multi-Feature Fusion	• This method is based on YOLO (You only look once) model that by modifying and testing various parameters proved to be adequate. • The model was based on 17 variations of Darknet-19 neural	IEEE

				network among which best was chosen.	
24.	Mikai Yang	2020	Overview of Fingerprint Recognition System	• This paper is based on deep learning and multi-feature fusion. • It can also handle incomplete fingerprints or small-area fingerprints. • Fingerprint feature extraction stage, FSFD method is also used. • Cloud SaaS services API are provided for fingerprint recognition to other applications.	International Conference on Culture-oriented Science & Technology (ICCST) 978-1- 7281-8138-7/20/$31.00©IEEE
25.	Jagadeesh H. S., Suresh B. K. and K. B. Raja	2016	Rationale of Class and Feature size on Face Recognition	• Basic Local Phase Quantization (BLPQ), Scaled Gray Level Co-occurrence Matrix (SGLCM) and Singular Value Decomposition (SVD) with two mask processing techniques have been proposed to handle multidimensional problems associated with face recognition. • To compute the result Euclidean Distance (ED) measure is used.	AICTC ’16, August 12-13, 2016, Bikaner, India© 2016 ACM

26.	Arya J. L and Safuvan T.	2015	Assessment of Image Quality in Face, Fingerprint, Iris, Palm print and Knuckle point for Detection of Fake Biometrics	• Face recognition, fingerprint, Iris identification, palmprint, knuckle point was assessed for high accuracy and easily detect fake biometrics. • This method has 'multi-attack' and 'multi-biometric' characteristics.	International Journal of Engineering Research & Technology (IJERT)
27.	A. K. Singh, P. Joshi and G. C. Nandi	2013	Development of a Fuzzy Expert System based Liveliness Detection Scheme for Biometric Authentication	• Fuzzy expert system has been introduced which is sufficient enough to detect liveliness. • HAAR cascade to detect eye and mouth, SIFT to detect movement inside, LBP for measuring surface texture. • Various fuzzification method, Linguistic variables and Membership Function are used in order to get best performance.	The 11th International Conference on Signal and Image Processing (ICIP) Published by Elsevier B. V.
28.	Elsevier Research Intelligence	2016	Elsevier Fingerprint Engine	• Back-end software system of state-of-the-art NLP techniques to find out various information. • Identifies relevant technical concepts in a text based on thesaurus.	Elsevier
29.	R. Kute, V. Vyas and A. Anuse	2021	Transfer Learning for Face Recognition using	• Bregman divergence regularization is used. • Regularization helps to reduce the difference between two domains.	Journal of King Saud University – Engineering Sciences

			Fingerprint Biometrics	• Discriminative Locality Alignment (DLA) features are fused. • Face images labelled data are being used for KNN training and fingerprint for testing. • PCA and FLDA have also been used.	
30.	R. Rahim, T. Afriliansyah, H. Winata, D. Nofriansyah, Ratnadewi and S. Aryza	2018	Research of Face Recognition with Fisher Linear Discriminant	• Fisher Linear Discriminant (FLD) method is proposed. • Image changed from RGB shape to grayscale and then black and white by doing threshold. • Lastly FLD calculation is performed and Euclidean distance is measured for face recognition.	4th International Conference on Operational Research (InteriOR) IOP Conference Series: Material Science and Engineering 300

COCOMO Model:

- In 1981 Dr. Berry Boehm proposed Cocomo model (constructive cost model).
- It is the most generally used software estimation model in the world.
- It is used to predict the effort for project and time required for software product development based on the size of software.

Steps for software evaluation in this model are as follows:

1. Get an initial estimate of the development effort from evaluation of thousands of delivered lines of source code (KDLOC).
2. Determine a set of 15 multiplying factors from various attributes of the project.
3. Calculate the effort estimate by multiplying the initial estimate with all the multiplying factors i.e., multiply the values in step1 and step2.

The basic estimation E_i is only determined on the basis of kilo lines of code in unit of person-month by the equation

$$E_i = a \times (kloc)^b$$

Where, a, b are the constants that depends on the project type.
Software projects under COCOMO model are classified into 3 categories:

A. Organic :- All the projects that comes under this if,
   - Project is simple and small.
   - Project team should be small ad must have some prior experience of developing similar type of projects.
   - The problem on which they are working must be well understood and have been solved in the past by the team.
   - Projects should have flexible requirement.

B. Semidetached :- All the projects that comes under this if,
   - Project are complex than organic.
   - Project team should have more experience, more creativity, and better guidance while preparing the software.

- The size of project should neither be so small nor too large, it should have sufficient lines of code.

C. Embedded model :- All the projects that comes under this if,

- Projects should have fixed requirement of resources.
- The development of project should be done with very high constraints.
- Project team should have highest level of complexity, creativity, and experience requirement.
- Requirement of number of person in team is higher in this model.

Types of COCOMO model:

1. Basic COCOMO Model:

It is used for quick and slightly rough calculation of software costs as it only considers based on lines of source code with same constant variables.

$E = a \times (kloc)^b$

$D = c \times (effort)^d$

$P = \frac{effort}{time}$

Where , E = effort in person month

D = development time in months

P = person required

Constants a, b, c, d for the basic cocomo model are

	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

This model does not consider different factors such as reliability, expertise hence estimation is rough.

$E = 2.4 \times (0.365)^{1.05}$

$= 0.83294$

$D = 2.5 (0.83294)^{0.35}$

$= 2.3450$

$$P = \frac{0.83294}{2.3450}$$

$$= 0.3551$$

So the output will be

Effort = 0.83294

Development time = 3 month

Person required = 3

Salary of average junior developer for a month is 40,000

Price of three laptop is 60,000 + 60,000 + 1,30,000 = 2,50,000

Other expenses of camera stand, connecting cables, smartphone = 2,00,000

So, total cost of project is 40,000 x 3 x 3 = 3,60,000

(3.6 + 2.5 + 2) lakh = 8.1 lakh

2. Intermediate COCOMO model:

The Basic COCOMO model is expanded into the Intermediate COCOMO model, which takes a number of cost factors into consideration to improve the cost estimating model's accuracy. In order to determine the price of the programme, the estimating model uses a collection of "cost driver attributes." The connection indicates the anticipated work and time.

$E = a \times (kloc)^b \times EAF$

$D = c \times (effort)^d$

Where , E = effort in person month

D = development time in months

EAF = It is effort Adjustment factor, which is calculated by multiplying parameter values if different cost drivers parameter.

	a	b	c	d
Organic	3.2	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Classification of Cost Drivers and their attributes:

- Product attributes
- Hardware attributes

- Personnel attributes
- Project attributes

Advantages of COCOMO Model

- In contrast to previous models like SLIM, COCOMO is transparent, allowing users to understand how it functions.
- The estimator may better comprehend the effects of many elements that affect project costs by using drivers.
- COCOMO offers suggestions for old projects.
- The COCOMO model makes it simple to calculate the project's overall cost.
- Understanding the effects of the many project crisis-affecting aspects is made much easier with the aid of the drivers.

Limitations of COCOMO Model

- Early in the project, when most effort estimates are needed, KDSI is difficult to estimate.
- The KDSI is a length measurement rather than a size measure.
- very prone to the development mode being misclassified.
- The model must be adjusted to the organization's requirements using past data, which is not always available. This is crucial for success.
- The cost of software is constrained in its precision.
- The abilities, collaboration, and knowledge of the consumer are also disregarded.

Progress Report

MONTH	Aug 10 -Aug 24	Aug 25 -Sep 07	Sep 08 - Sep 21	Sep 22 - Oct 05	Oct 06 - Oct 19	Oct 20 - Oct 26	Oct 27 - Nov 04
Supervisor Meeting							
Submission of proposal							
Data Collection							
Literature Review							
Synopsis Preparation							
Development of Module							
Testing							
Report Preparation							
Report Submission							

Hardware & Software Requirements:

Hardware:

Computer system.

Configuration: intel i5 processor (8 gen or above), 8gb RAM, 20gb free disk space

Cameras- 2

Flash Lights

Camera stand

Software :

Windows 10/11

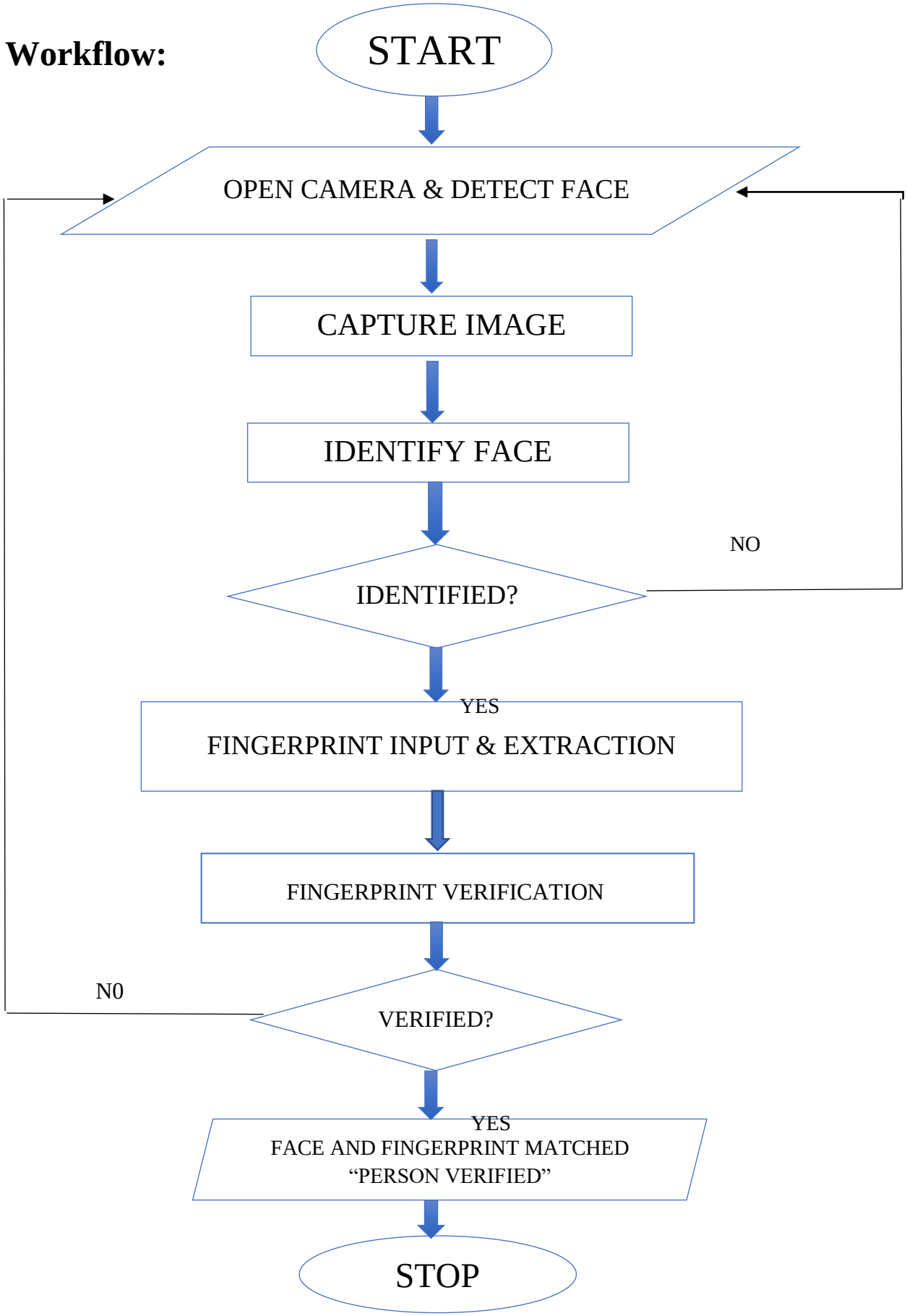
Visual studio community version and visual studio code.

Python and its IDE

Libraries & Modules: dlib, OpenCV, face_recognition, NumPy, math, tensorflow, OS.

Proposed Work

Workflow:



ALGORITHM:-

Step 1: Start

Step 2: Open camera using OpenCV

- VideoCapture() function of Python enables to open the camera.
- Image is read and store in a variable using video.read() function.

Step 3: Detect face using pretrained prototype and model.

- Face Proto "opencv_face_detector.pbtxt" and face model "opencv_face_detector_uint8.pb" file are loaded. These models are read using readnet() function based on deep neural network.
- The model first finds the locations points of face and these points are used by the function blobfromimage() based on deep neural network. This function stores the image as a 4-dimentional blob which is binary form of the imagr and undergoes various factors namely mean subtraction, scaling and some other factors to predict the actual age, gender and also used in the function facebox() to formulate the box in the next step.
- This model then uses the points extracted in the previous step and make box around the detected face.

Step 4: Pre-trained models of age and gender are used to predict the probable age range and gender of the person.

Step 5: Gender and Age range is displayed with the box around the face detected.

Step 6: Press 'Enter' to capture image

- The prototype then waits till user presses any key and this is done by the pre-built function of python that is waitkey().
- It won't save the image till the used presses the enter key.

Step 7: Identify person from the database using deep learning based face recognition algorithm

- Image is read first.
- The image color is converted from bgr to rgb.
- This algorithm then find 128 different types of encodings from face detected.
- These encodings are then matched with the already saved database image using the face recognition library.
- Compare_faces() and face_distance() functions are involved to find the best match.

- Then the algorithm uses the encodings of the already present facial images of the person to be recognized and match it with current face.
- Then if the match accuracy is greater than 55% then person is identified.

Step 6: If identified, Go to next step else Go to step 3

- Once person is identified and image is captured and saved, camera is released using video.release() function.
- All the opened windows during the execution are closed afterwards using the in-built cv2 python function cv2.destroyallwindows()

Step 7: Take input for fingerprint verification

Step 8: Open another camera for taking fingerprint using the earlier used python cv2 function videocapture().

- An already size-defined rectangular box is formed.
- The concerned candidate need to place his finger within the rectangular box for better fingerprint lines extraction.
- The prototype will ignore any object outside the rectangular box for evaluation and matching of points in the further steps.

Step 9: Press enter to take image then this image will be cropped for fingerprint using OpenCV.

- The model with wait till the user presses any key.
- It accepts and saves the input as soon as enter key is pressed.
- The image is read and it's color is converted from BGR to Gray.

Step 10: Fingerprint is extracted from image using "Gaussian Adaptive Threshold Algorithm".

- To apply basic thresholding, a manual threshold value T is supplied.
- Otsu's method can automatically determine the value of T.
- Just having one value of T may not be sufficient. To overcome this problem, adaptive thresholding is used.
- This method considers small neighbors of pixel and find the optimal threshold value.
- Each pixel in neighbourhood contributes equally for calculating the value of T. The Gaussian mean of pixel is used in each region.
- The general formula for calculating the value of T is:

$$T = \text{mean}\ (\ I_L\ ) - C$$

Step 11: Now the keypoints are identified from the fingerprint image using SIFT algorithm and these keypoints are then matched to the database image of the same person whose face is matched using “Flann Based matcher” if fingerprint match user verified.

- In computer vision, a feature detection approach known as SIFT, or Scale Invariant Feature Transform, is used.
- Local features, sometimes referred to as an image's "keypoints," may be found by using SIFT. These keypoints are scale and rotation invariant and may be applied to a number of computer vision tasks, including scene and object recognition and picture matching.
- The keypoints produced by SIFT may be used to train a model by being used as features for the picture. Since SIFT features are unaffected by the size or orientation of the picture, they have a significant advantage over edge features and hog features.
- Fast Library for Approximate Nearest Neighbours is referred to as FLANN. It includes a selection of methods designed for high dimensional features and quick closest neighbour search in huge datasets. It accepts two sets of options for FlannBasedMatcher, specifying the algorithm to be used, its associated parameters, etc. Index comes first. The information that must be given for different algorithms is described in FLANN documentation.

Step 9: If verified, go to next step else Go to step 3

Step 10: User verified now print Name, Face accuracy, Gender, Predicted Probable Age range and fingerprint match accuracy.

- Since our protype requires high-end devices for the better recognition which seems out of reach for us as of now, we have kept the match_points as low as 35%.
- The match accuracy of face is kept to be 0.55 and that of finger is 0.35. So, the overall chances of wrong estimation can be predicted to be as low as 0.3.
- The age and gender prediction is not the very important aspect of this model. Their prediction is also estimated to be around 0.7 which I reckon is good in numbers.
- The pre-trained model can be trained further and with better techniques for more accuracy.

Step 11: Stop

PYTHON

Python is an object-oriented, dynamically-semantic, interpreted high-level programming language. Its high-level built-in data structures, along with dynamic typing and dynamic binding, making it particularly appealing for use in rapid application development as well as as a scripting or glue language to join pre-existing components. The readability of Python code is prioritised, which lowers the cost of programme maintenance. Python's syntax is also straightforward and quick to learn. Python allows packages and modules, which promotes the modularity and reuse of code in programmes. For all popular systems, the Python interpreter and the comprehensive standard library are free to download in source or binary form and can be shared without restriction.

The edit-test-debug cycle is extraordinarily quick because there is no compilation step. Python programmes are simple to debug since a segmentation failure is never caused by a bug or incorrect input. Instead, the interpreter raises an exception when it finds a mistake. The interpreter prints a stack trace if the application does not catch the exception. Setting breakpoints, evaluating arbitrary expressions, inspecting local and global variables, stepping through the code one line at a time, and other features are all possible with a source level debugger. Python's ability to perform introspection is demonstrated by the debugger, which is developed in Python. On the other hand, adding a few print statements to the source code is frequently the easiest way to debug a programme due to the short edit-test-debug cycle.

Libraries used in the implementation:

1. Tkinter: It enables the display of images. Jpeg images are just one of the many formats that may be used for images. It may seem paradoxical, but you can display an image with a label.

   Use the method, Image.open(filename) to open an image. This will search the programmes directory for images; if it finds any, it will append the path to the filename.

2. Cv2: OpenCV is a sizable open-source toolkit for image processing, machine learning, and computer vision. Numerous programming languages, including Python, C++, Java,

etc., are supported by OpenCV. It can analyse photos and movies to recognise items, faces, and even a person's handwriting. The number of weapons in your arsenal rises when it is integrated with other libraries, such as the highly efficient library for numerical operations known as Numpy, as OpenCV is able to do every operation that Numpy can.

3. Numpy: The Python package NumPy is used to manipulate arrays. Additionally, it contains matrices, fourier transform, and functions for working in the area of linear algebra. You can use it for free because it is an open-source project. Numerical Python is referred to as NumPy.

   The goal of NumPy is to offer array objects that are up to 50 times quicker than conventional Python lists. The NumPy array object is referred to as ndarray, and it has a number of supporting methods that make using ndarray relatively simple. In data research, where speed and resources are crucial, arrays are employed a lot.

4. Os: The operating system-dependent functionality can be used portable thanks to this module. The fileinput module can be used to read every line in every command-line file. The open() function can be used to simply read or write a file. The os.path module can be used to alter paths. High-level file and directory handling is covered by the shutil module, while the tempfile module is used to create temporary files and directories. Numerous functions for interacting with the file system are included in the *os* and *os.path* modules.

5. Dlib: DLib is an open-source Python library that implements a number of machine learning techniques, including classification, regression, clustering, data transformation, and structured prediction. DLib comes with a lot of examples that use the library and has excellent unit testing coverage. The library has documentation for each class and function. The library's homepage contains this documentation. DLib is a useful framework for Python developers that want to create machine learning applications.

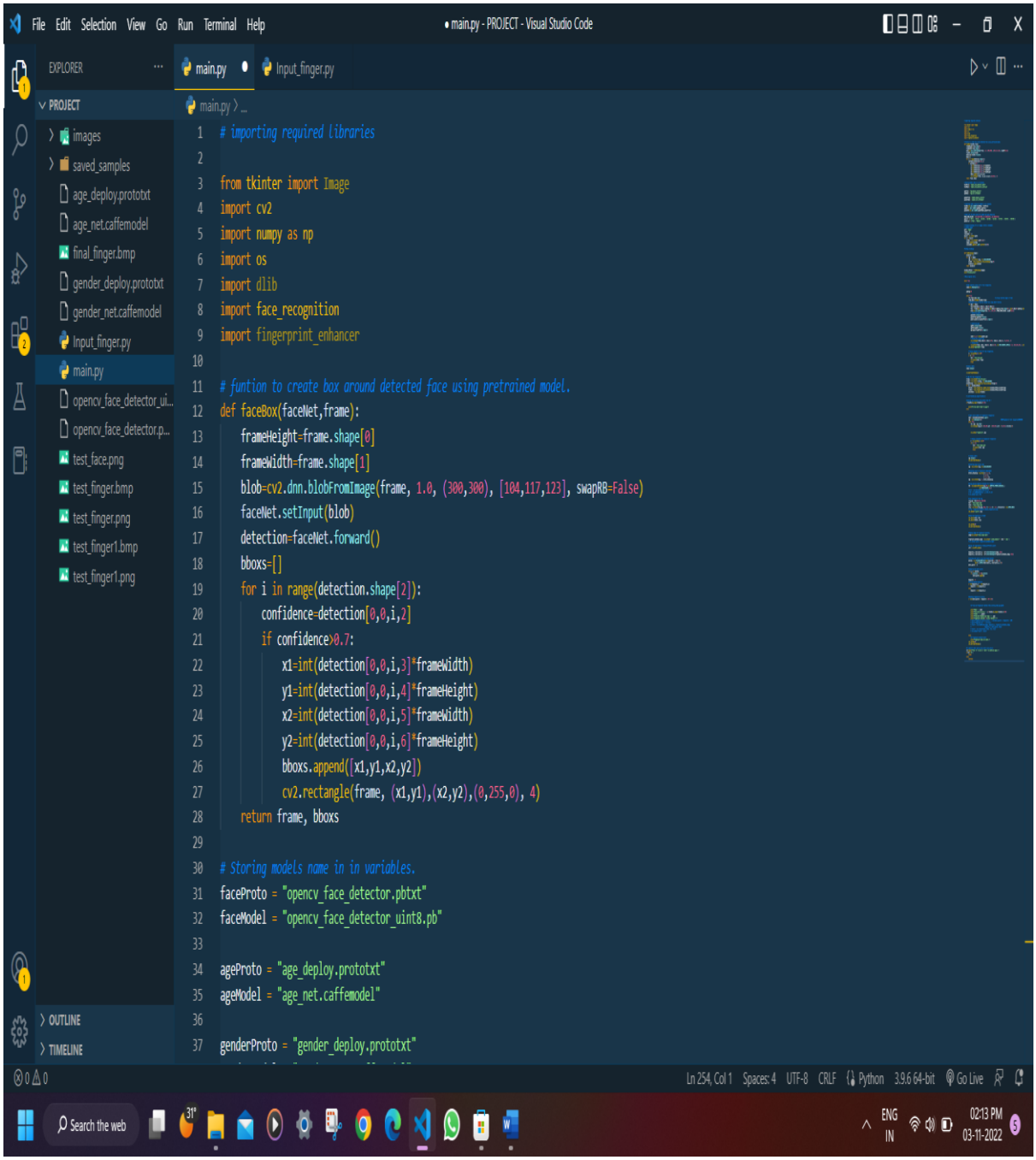
In that it offers a general, high-performance toolbox for machine learning with a variety of methods, DLib is similar to DMTL; however, DLib contains more examples and more current updates. DLib also comes with a tonne additional helper feature.

6. Face recognition: Face recognition is a technique for identifying people's faces from images and videos. These systems are often utilised, particularly in law enforcement and capitals.
   This sort of system can now be created using only a few libraries, such as those for facial recognition in Python, thanks to the development of deep learning, and that is what this lesson is all about.

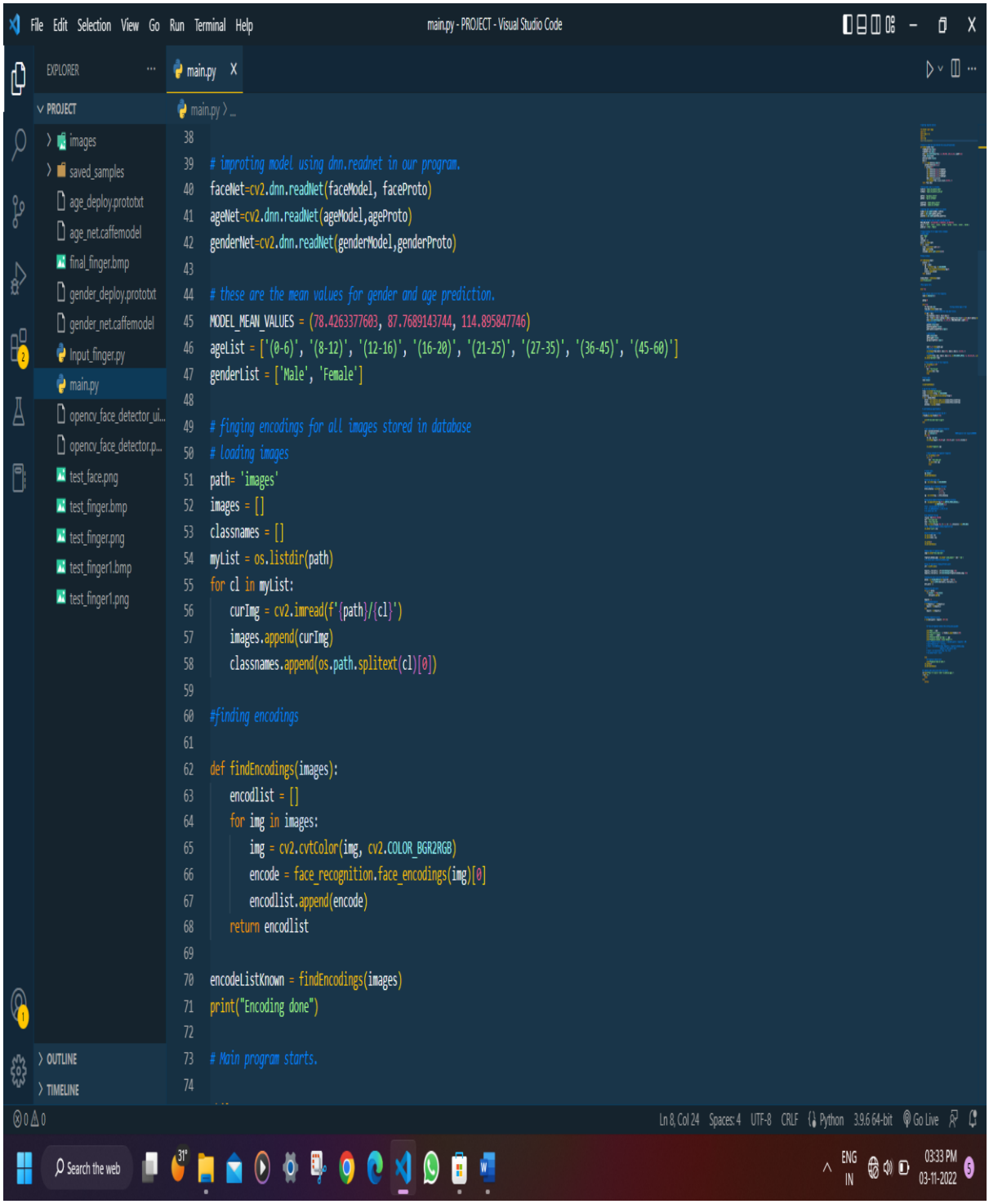
   Only the BGR picture format is supported by the face recognition library. We should use OpenCV to transform the generated picture into RGB before printing it.
   We don't need to utilise haar cascade and other methods because the library face recognition can detect faces rapidly on its own.

   The deep learning-based library face recognition offers single-shot learning, which means it only requires one image to teach itself to recognise a person.

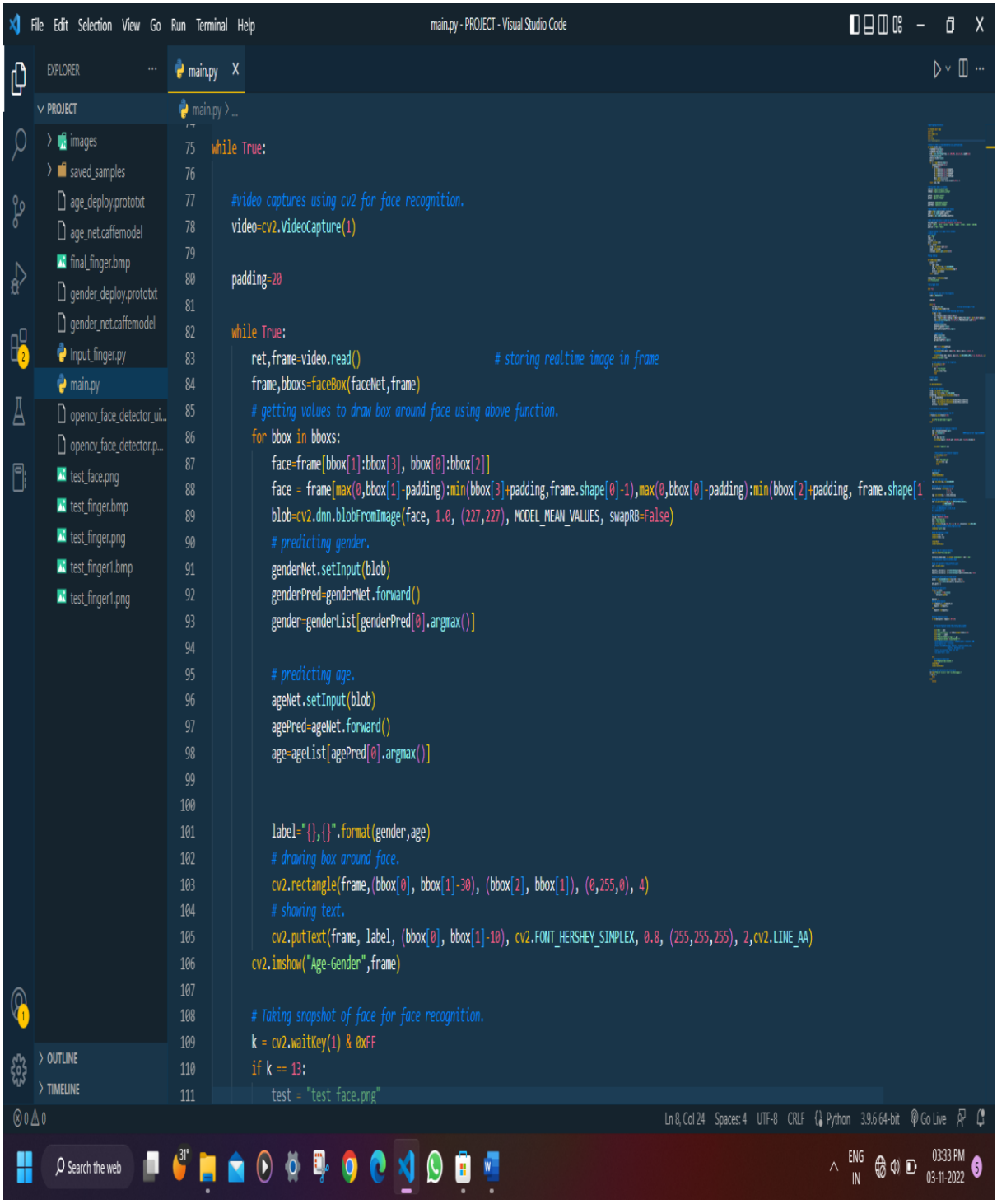
Implementation:-



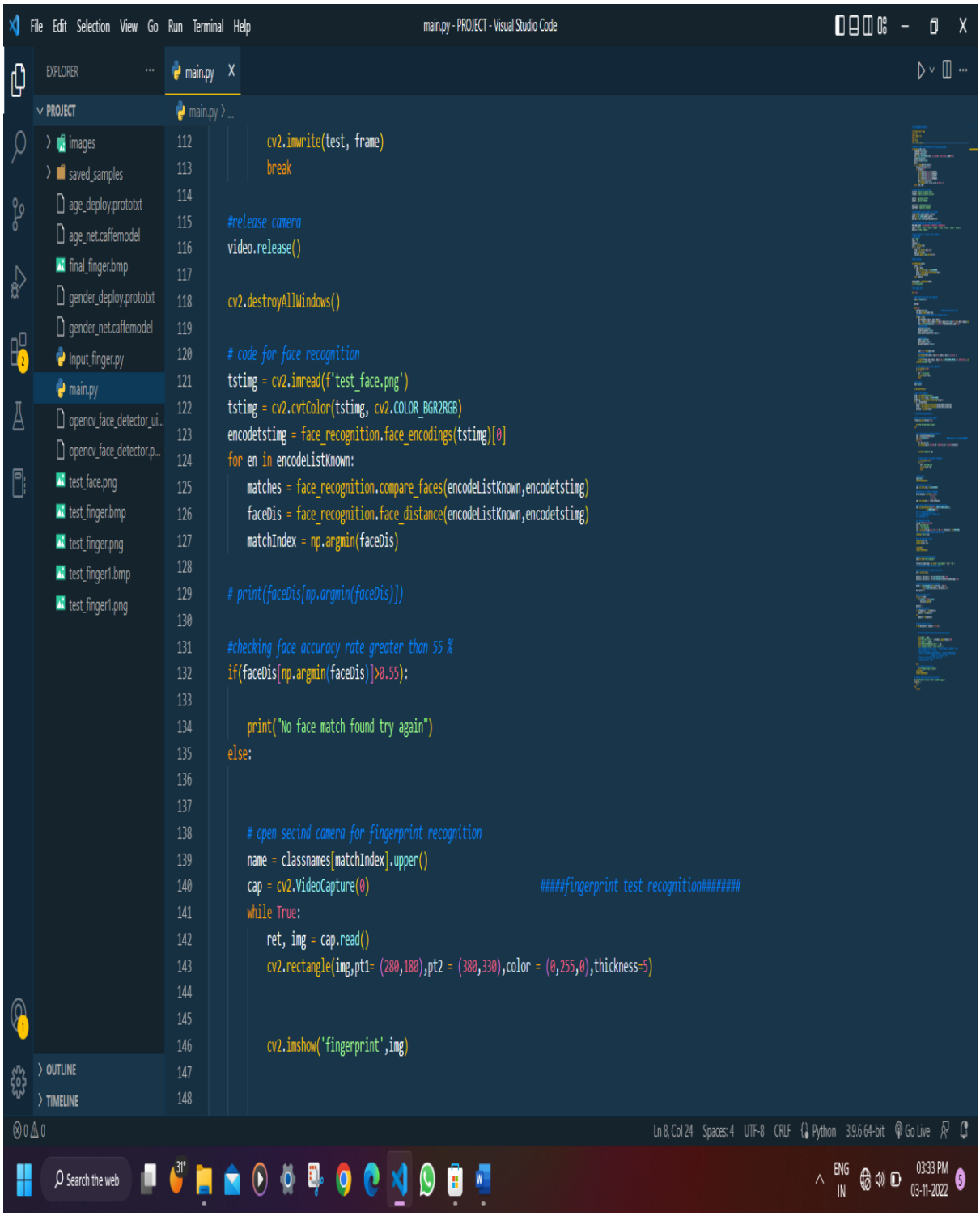
Code (1)


File Edit Selection View Go Run Terminal Help
main.py - PROJECT - Visual Studio Code
EXPLORER
main.py
PROJECT
images
saved_samples
age_deploy.prototxt
age_net.caffemodel
final_finger.bmp
gender_deploy.prototxt
gender_net.caffemodel
Input_finger.py
main.py
opencv_face_detector_ui...
opencv_face_detector.p...
test_face.png
test_finger.bmp
test_finger.png
test_finger1.bmp
test_finger1.png
OUTLINE
TIMELINE
38
39 # improting model using dnn.readnet in our program.
40 faceNet=cv2.dnn.readNet(faceModel, faceProto)
41 ageNet=cv2.dnn.readNet(ageModel,ageProto)
42 genderNet=cv2.dnn.readNet(genderModel,genderProto)
43
44 # these are the mean values for gender and age prediction.
45 MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
46 ageList = ['(0-6)', '(8-12)', '(12-16)', '(16-20)', '(21-25)', '(27-35)', '(36-45)', '(45-60)']
47 genderList = ['Male', 'Female']
48
49 # finging encodings for all images stored in database
50 # loading images
51 path= 'images'
52 images = []
53 classnames = []
54 myList = os.listdir(path)
55 for cl in myList:
56 curImg = cv2.imread(f'{path}/{cl}')
57 images.append(curImg)
58 classnames.append(os.path.splitext(cl)[0])
59
60 #finding encodings
61
62 def findEncodings(images):
63 encodlist = []
64 for img in images:
65 img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
66 encode = face_recognition.face_encodings(img)[0]
67 encodlist.append(encode)
68 return encodlist
69
70 encodeListKnown = findEncodings(images)
71 print("Encoding done")
72
73 # Main program starts.
74
Ln 8, Col 24 Spaces: 4 UTF-8 CRLF Python 3.9.6 64-bit Go Live
Search the web
ENG
IN
03:33 PM
03-11-2022

Code (2)

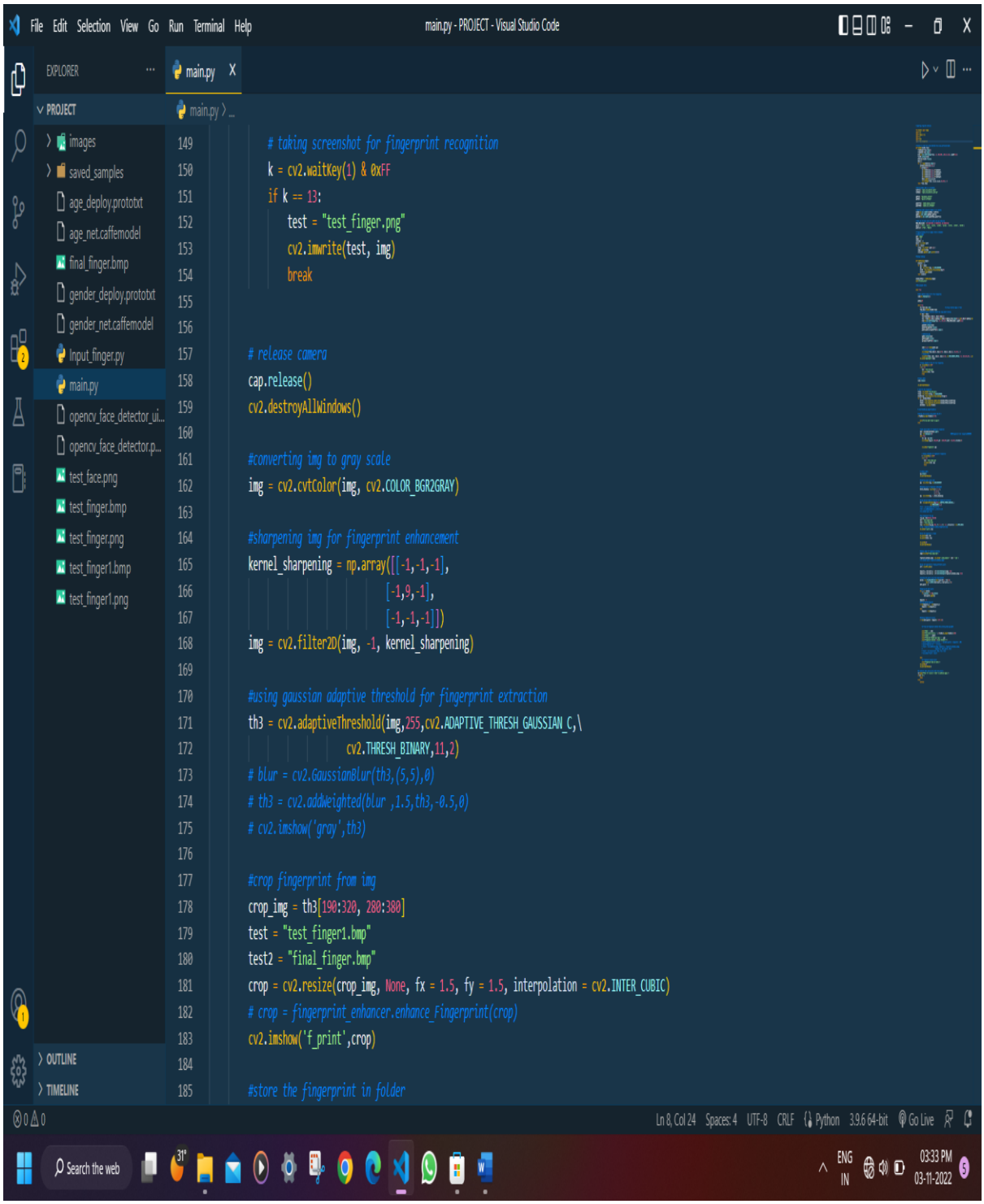

File Edit Selection View Go Run Terminal Help
main.py - PROJECT - Visual Studio Code
EXPLORER
main.py
PROJECT
images
saved_samples
age_deploy.prototxt
age_net.caffemodel
final_finger.bmp
gender_deploy.prototxt
gender_net.caffemodel
Input_finger.py
main.py
opencv_face_detector_ui...
opencv_face_detector.p...
test_face.png
test_finger.bmp
test_finger.png
test_finger1.bmp
test_finger1.png
OUTLINE
TIMELINE
while True:
#video captures using cv2 for face recognition.
video=cv2.VideoCapture(1)
padding=20
while True:
ret,frame=video.read() # storing realtime image in frame
frame,bboxs=faceBox(faceNet,frame)
# getting values to draw box around face using above function.
for bbox in bboxs:
face=frame[bbox[1]:bbox[3], bbox[0]:bbox[2]]
face = frame[max(0,bbox[1]-padding):min(bbox[3]+padding,frame.shape[0]-1),max(0,bbox[0]-padding):min(bbox[2]+padding, frame.shape[1
blob=cv2.dnn.blobFromImage(face, 1.0, (227,227), MODEL_MEAN_VALUES, swapRB=False)
# predicting gender.
genderNet.setInput(blob)
genderPred=genderNet.forward()
gender=genderList[genderPred[0].argmax()]
# predicting age.
ageNet.setInput(blob)
agePred=ageNet.forward()
age=ageList[agePred[0].argmax()]
label="{},{}".format(gender,age)
# drawing box around face.
cv2.rectangle(frame,(bbox[0], bbox[1]-30), (bbox[2], bbox[1]), (0,255,0), 4)
# showing text.
cv2.putText(frame, label, (bbox[0], bbox[1]-10), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255,255,255), 2,cv2.LINE_AA)
cv2.imshow("Age-Gender",frame)
# Taking snapshot of face for face recognition.
k = cv2.waitKey(1) & 0xFF
if k == 13:
test = "test face.png"
Ln 8, Col 24 Spaces: 4 UTF-8 CRLF Python 3.9.6 64-bit Go Live
Search the web
ENG
IN
03:33 PM
03-11-2022

Code (3)

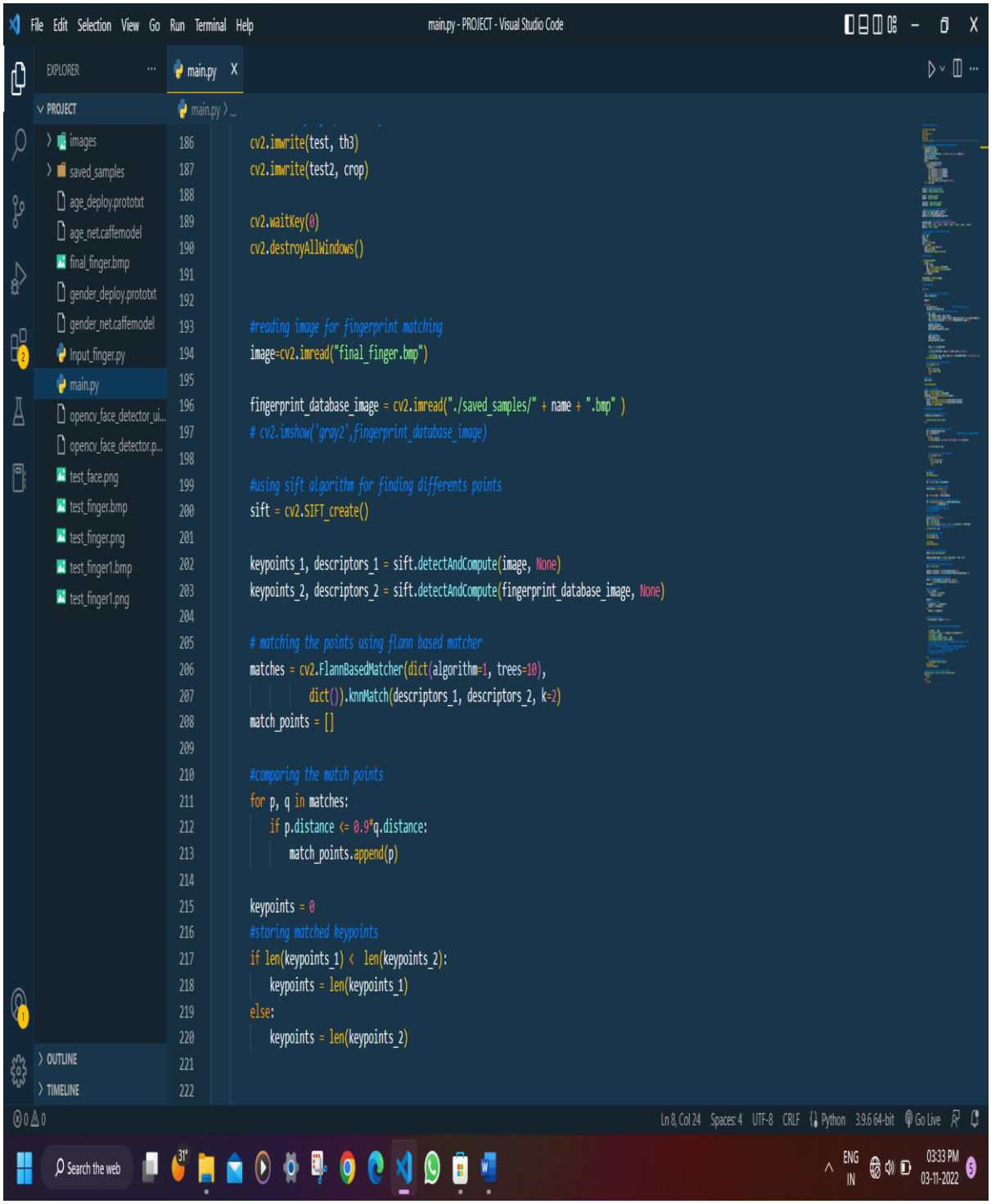


```
            cv2.imwrite(test, frame)
            break

    #release camera
    video.release()

    cv2.destroyAllWindows()

    # code for face recognition
    tstimg = cv2.imread(f'test_face.png')
    tstimg = cv2.cvtColor(tstimg, cv2.COLOR_BGR2RGB)
    encodetstimg = face_recognition.face_encodings(tstimg)[0]
    for en in encodeListKnown:
        matches = face_recognition.compare_faces(encodeListKnown,encodetstimg)
        faceDis = face_recognition.face_distance(encodeListKnown,encodetstimg)
        matchIndex = np.argmin(faceDis)

    # print(faceDis[np.argmin(faceDis)])

    #checking face accuracy rate greater than 55 %
    if(faceDis[np.argmin(faceDis)]>0.55):

        print("No face match found try again")
    else:


        # open secind camera for fingerprint recognition
        name = classnames[matchIndex].upper()
        cap = cv2.VideoCapture(0)                    ######fingerprint test recognition########
        while True:
            ret, img = cap.read()
            cv2.rectangle(img,pt1= (280,180),pt2 = (380,330),color = (0,255,0),thickness=5)


            cv2.imshow('fingerprint',img)
```

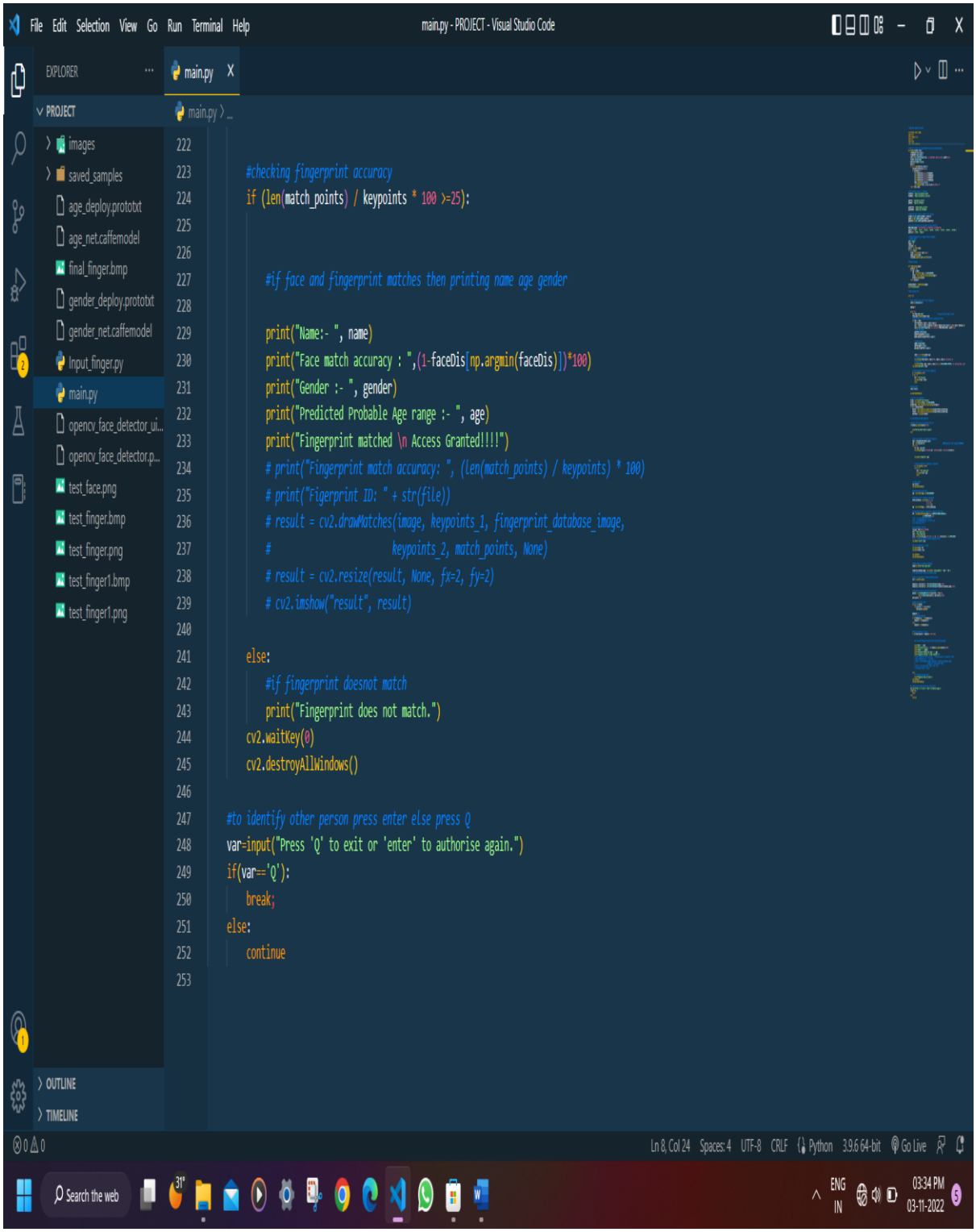
Code (4)


File Edit Selection View Go Run Terminal Help
main.py - PROJECT - Visual Studio Code
EXPLORER
main.py
PROJECT
images
saved_samples
age_deploy.prototxt
age_net.caffemodel
final_finger.bmp
gender_deploy.prototxt
gender_net.caffemodel
input_finger.py
main.py
opencv_face_detector_ui...
opencv_face_detector.p...
test_face.png
test_finger.bmp
test_finger.png
test_finger1.bmp
test_finger1.png
OUTLINE
TIMELINE
# taking screenshot for fingerprint recognition
k = cv2.waitKey(1) & 0xFF
if k == 13:
test = "test_finger.png"
cv2.imwrite(test, img)
break
# release camera
cap.release()
cv2.destroyAllWindows()
#converting img to gray scale
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
#sharpening img for fingerprint enhancement
kernel_sharpening = np.array([[-1,-1,-1],
[-1,9,-1],
[-1,-1,-1]])
img = cv2.filter2D(img, -1, kernel_sharpening)
#using gaussian adaptive threshold for fingerprint extraction
th3 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,\
cv2.THRESH_BINARY,11,2)
blur = cv2.GaussianBlur(th3,(5,5),0)
th3 = cv2.addWeighted(blur ,1.5,th3,-0.5,0)
cv2.imshow('gray',th3)
#crop fingerprint from img
crop_img = th3[190:320, 280:380]
test = "test_finger1.bmp"
test2 = "final_finger.bmp"
crop = cv2.resize(crop_img, None, fx = 1.5, fy = 1.5, interpolation = cv2.INTER_CUBIC)
# crop = fingerprint_enhancer.enhance_Fingerprint(crop)
cv2.imshow('f_print',crop)
#store the fingerprint in folder
Ln 8, Col 24 Spaces: 4 UTF-8 CRLF Python 3.9.6 64-bit Go Live
Search the web
ENG
IN
03:33 PM
03-11-2022

Code (5)


File Edit Selection View Go Run Terminal Help
main.py - PROJECT - Visual Studio Code
EXPLORER
main.py
PROJECT
images
saved_samples
age_deploy.prototxt
age_net.caffemodel
final_finger.bmp
gender_deploy.prototxt
gender_net.caffemodel
Input_finger.py
main.py
opencv_face_detector_ui...
opencv_face_detector.p...
test_face.png
test_finger.bmp
test_finger.png
test_finger1.bmp
test_finger1.png
OUTLINE
TIMELINE
cv2.imwrite(test, th3)
cv2.imwrite(test2, crop)
cv2.waitKey(0)
cv2.destroyAllWindows()
#reading image for fingerprint matching
image=cv2.imread("final_finger.bmp")
fingerprint_database_image = cv2.imread("./saved_samples/" + name + ".bmp")
# cv2.imshow('gray2',fingerprint_database_image)
#using sift algorithm for finding differents points
sift = cv2.SIFT_create()
keypoints_1, descriptors_1 = sift.detectAndCompute(image, None)
keypoints_2, descriptors_2 = sift.detectAndCompute(fingerprint_database_image, None)
# matching the points using flann based matcher
matches = cv2.FlannBasedMatcher(dict(algorithm=1, trees=10),
dict()).knnMatch(descriptors_1, descriptors_2, k=2)
match_points = []
#comparing the match points
for p, q in matches:
if p.distance <= 0.9*q.distance:
match_points.append(p)
keypoints = 0
#storing matched keypoints
if len(keypoints_1) < len(keypoints_2):
keypoints = len(keypoints_1)
else:
keypoints = len(keypoints_2)
Ln 8, Col 24 Spaces: 4 UTF-8 CRLF Python 3.9.6 64-bit Go Live
Search the web
ENG
IN
03:33 PM
03-11-2022

Code (6)

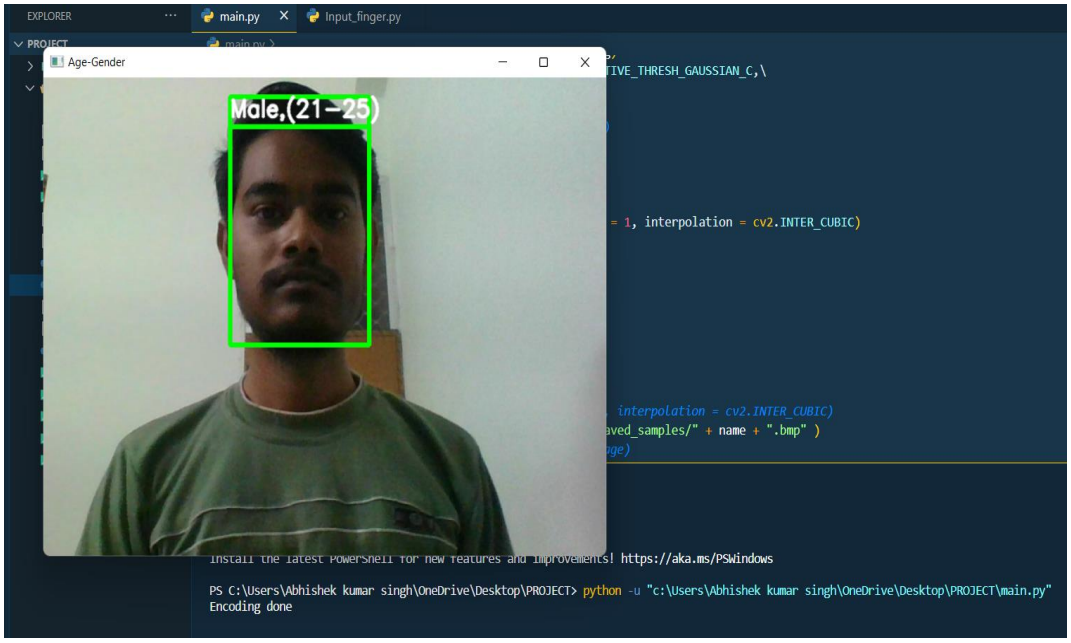

File Edit Selection View Go Run Terminal Help
main.py - PROJECT - Visual Studio Code
EXPLORER
main.py
PROJECT
images
saved_samples
age_deploy.prototxt
age_net.caffemodel
final_finger.bmp
gender_deploy.prototxt
gender_net.caffemodel
Input_finger.py
main.py
opencv_face_detector_ui...
opencv_face_detector.p...
test_face.png
test_finger.bmp
test_finger.png
test_finger1.bmp
test_finger1.png
OUTLINE
TIMELINE
#checking fingerprint accuracy
if (len(match_points) / keypoints * 100 >=25):
#if face and fingerprint matches then printing name age gender
print("Name:- ", name)
print("Face match accuracy : ",(1-faceDis[np.argmin(faceDis)])*100)
print("Gender :- ", gender)
print("Predicted Probable Age range :- ", age)
print("Fingerprint matched \n Access Granted!!!!")
# print("Fingerprint match accuracy: ", (len(match_points) / keypoints) * 100)
# print("Figerprint ID: " + str(file))
# result = cv2.drawMatches(image, keypoints_1, fingerprint_database_image,
# keypoints_2, match_points, None)
# result = cv2.resize(result, None, fx=2, fy=2)
# cv2.imshow("result", result)
else:
#if fingerprint doesnot match
print("Fingerprint does not match.")
cv2.waitKey(0)
cv2.destroyAllWindows()
#to identify other person press enter else press Q
var=input("Press 'Q' to exit or 'enter' to authorise again.")
if(var=='Q'):
break;
else:
continue
Ln 8, Col 24 Spaces: 4 UTF-8 CRLF Python 3.9.6 64-bit Go Live
Search the web
ENG
IN
03:34 PM
03-11-2022

Code (7)

Steps to Execute the Code:

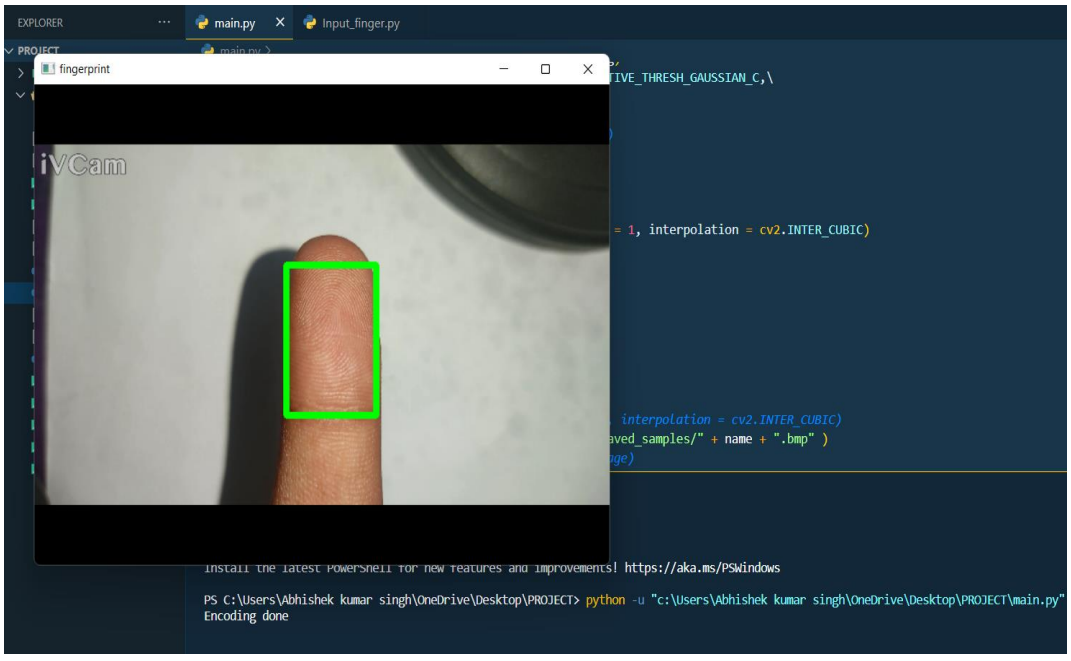
- Download and install VS code in your system
- Download and install python compiler, install and its path to the environment variable.
- Download and install Visual Studio Community version
  Select and install Desktop development with C++
- Open Windows Powershell
  Install cmake module
  Type: pip install cmake
  Install dlib library
  Type: pip install dlib
  Install cv2 library
  Type: pip install cv2
  Install face recognition module
  Type: pip install face_recognition
  Install numpy library:
  Type: pip install numpy
- Download and install iVCam app in both smartphone and PC
- Enable Developer Mode in your smartphone and connect to PC
- Run the code

Result:



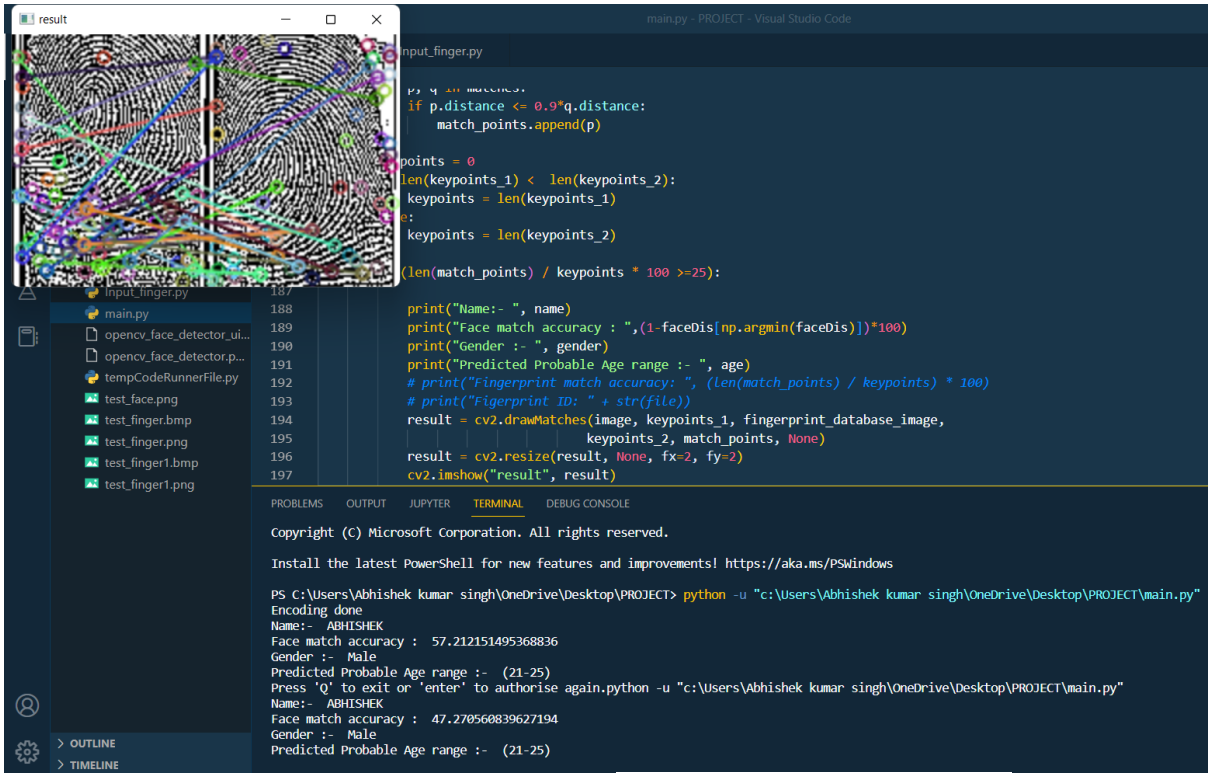
Face (1)

Here we are using camera to capture video for face detection. Our program is detecting face and making box around it. It is also predicting about the gender and age range of person. After pressing enter it will match face with database image for identification of person.



Finger image

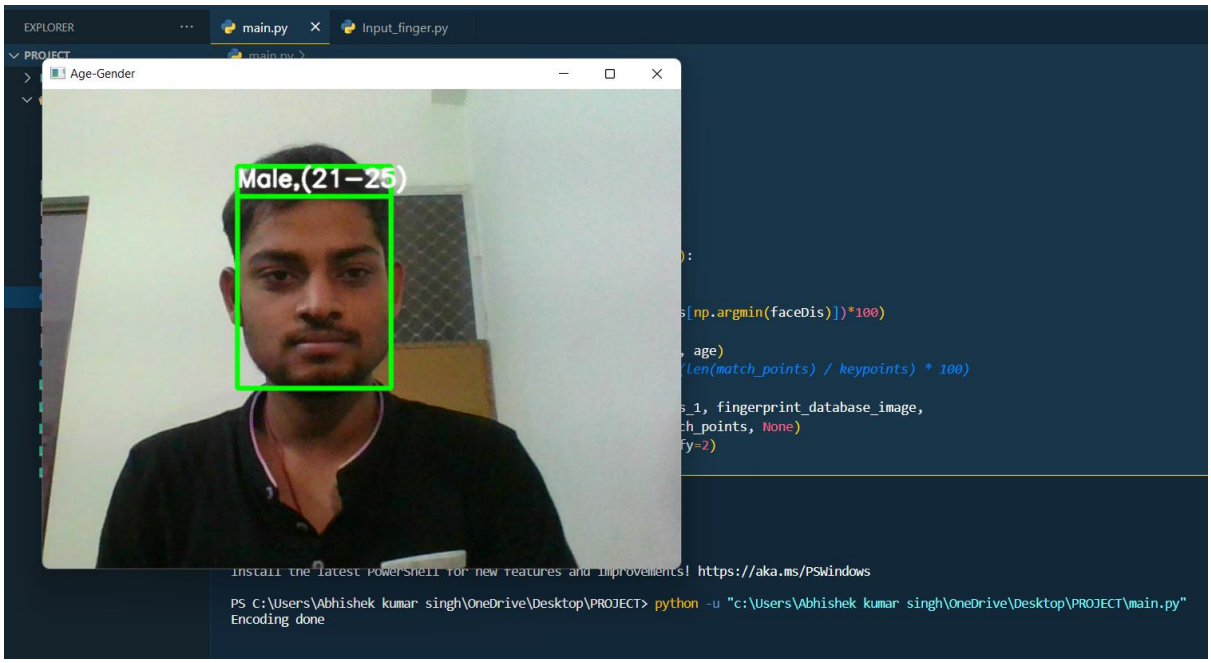
After face detection and identification another camera will open to take image of finger, process this image and extract fingerprint from the image. Then it will match it with database image.



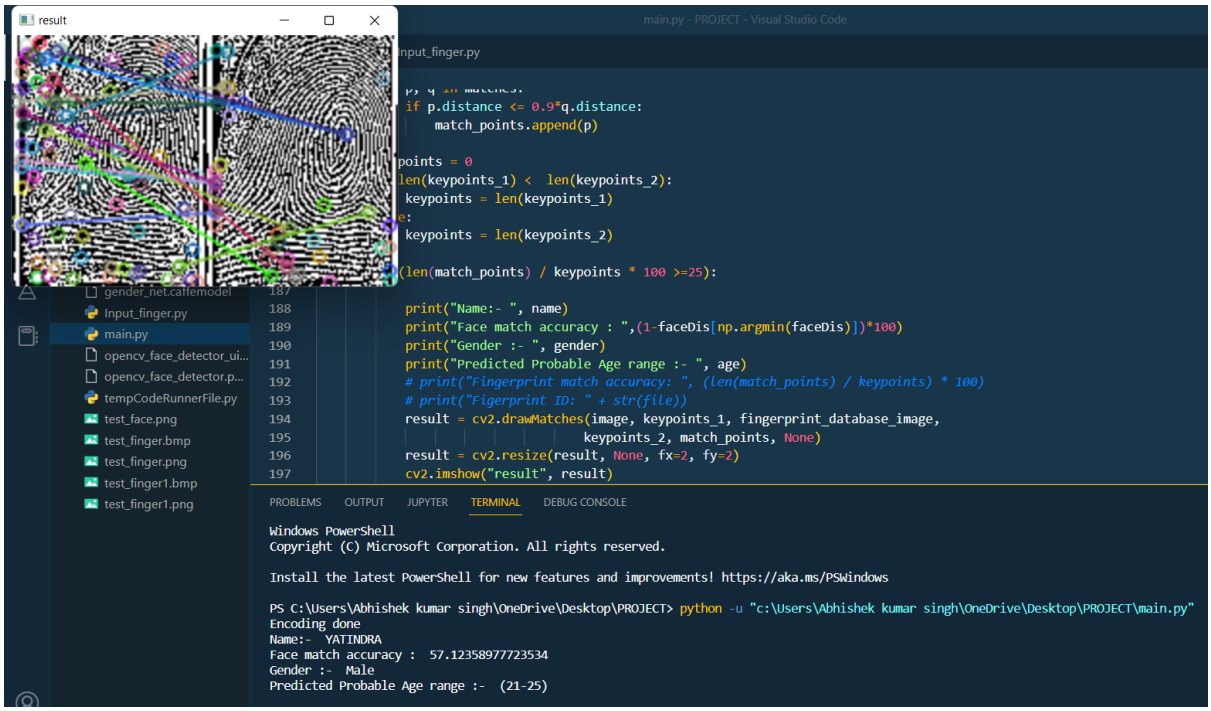
Fingerprint image (1)

After successful fingerprint verification, it had print result as Name of person, face match accuracy, its predicted age range and gender. And if fingerprint doesn't match it will print fingerprint does not match.

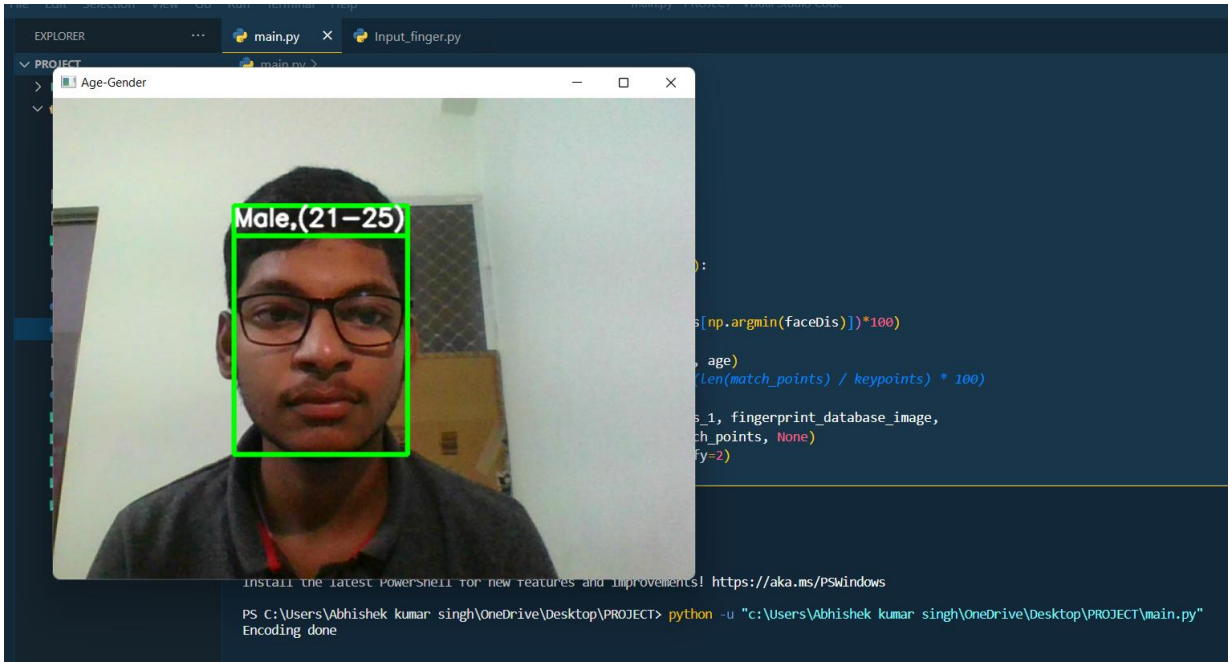
Testing our program on different person:



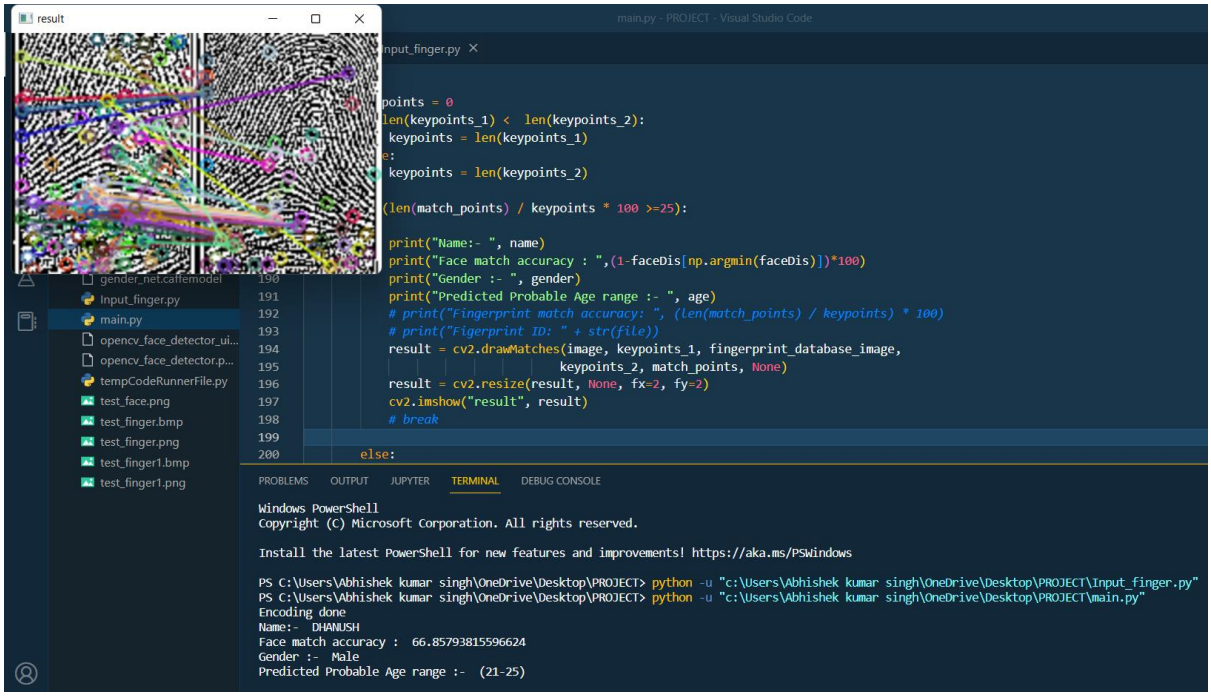
Face (2)



Fingerprint image (2)



Face (3)



Fingerprint image (3)

Conclusion and Future work:

Currently, the efficiency of this prototype is expected to be around 60%. In future it can be further improved using more fingerprint image enhancement techniques. The accuracy can also be increased by using advanced camera and proper lighting. This will increase the accuracy specially of age and gender prediction. Some extra security layers can also be added to the prototype such as voice matching, retina scanning, etc as per the requirement of the concerned institution. Now this prototype is predicting the age range (0-6), (21-25), etc. It can be further trained to predict the single integer age.

In future the database can be updated or new data can be added from within the prototype itself which now is being done manually.
More points to catch up with in future:

- Where to implement?
- Improve processing time.
- Extent
- Adding more layers of security.

References

[1] A. Saxena, P. Singh and S. N. Singh (2021) "Gender and age detection using deep learning" 11th International Conference on Cloud Computing, Deep Learning & Engineering (Confluence) 978-1-6654-1451-7/20/$31.00 Ⓒ IEEE DOI: 10.1109/Confluence51648.2021.9377041

[2] N. Boyko, O. Basystiuk and N. Shakhovska (2018) "Performance Evaluation and Comparison of Software for Face Recognition, based on Dlib and Opencv Library" IEEE Second International Conference on Data Stream Mining & Processing 978-1-5386-2784-4/18/$31.00

[3] Dr. E. P. Ijjina, G. Kanahasabai and A. S. Joshi (2020) "Deep Learning based approach to detect Customer Age, Gender and Expression in Surveillance Video" 11th ICCCNT IIT – Kharagpur Ⓒ IEEE

[4] M. M. H. Ali, V. H. Mahale, P. Yannawar and A. T. Gaikwad (2016) "Overview of Fingerprint Recognition System" International Conference on Electrical, Electronics, and Optimisation Techniques (ICEEOT) 978-1-4673-9939-5/16/$31.00ⒸIEEE

[5] A. L. H. Jin, A. Chekima, J. A. Dargham and L. C. Fan (2002) "Fingerprint Identification and Recognition Using Backpropagation Neural Network" Student Conference on Research and Development Proceedings, Shah Alam, Malaysia 0-7803-7565-3/02/$1700 ⒸIEEE

[6] M. M. H. Ali, V. H. Mahale, P. Yannawar and A. T. Gaikwad (2016) "Fingerprint Recognition for Person Identification and Verification Based on Minutiae Matching" IEEE 6th International Conference on Advanced Computing 978-1-4673-8286-1/16 $31.00 Ⓒ IEEE DOI: 10.1109/IACC.2016.69

[7] L. Hong and A. Jain (1998) "Integrating Faces and Fingerprints for Personal Identification" IEEE Transactions on Pattern Analysis and Machine Intelligence (Vol. 20) 0162-8828/98/$10.00 Ⓒ IEEE

[8] A. Mustafa and K. Meehan (2020) "Gender Classification and Age Prediction using CNN ResNet in Real-Time" International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI) 978-1-7281-9675-6/20/$31.00ⒸIEEE

[9]A. Ghildiyal, S. Sharma, I. Verma and U. Marhata (2020) "Age and Gender Predictions using Artificial Intelligence Algorithm" Proceedings of the Third International Conference on Intelligent Sustainable Systems (ICISS) 978-7281-7089-3/20/$31.00 Ⓒ IEEE

[10]G. Aguilar, G. Sanchez. K. T. Scano, M. Salinas, M. Nakano and H. Perez (2007) "Fingerprint Recognition" Second International Conference on Internet Monitoring and Protection (ICIMP) 0-7695-2911-9/07/$25.00ⒸIEEE

[11]M. J. P. Priyadarshini, G. K. Rajini, S. Naseera, N. S. Bash, S. S.Kumar, M. Karunagaran and K. C. Shekara (2017) "Human Identification using Face and Fingerprint" Proceedings of the International Conference on Intelligent Sustainable Systems (ICISS) 978-1-5386-1959-9/17/$31.00ⒸIEEE

[12]Adhiyaman M. and Ezhilmaran D (2015) "Fingerprint Matching and Similarity Checking System using Minutiae Based Technique" International Conference on Engineering and Technology (ICETECH) Coimbatore, TN, India ⒸIEEE

[13]Salma A., P. A. Maizate and P. L. Hassouni (2019) "Real-time Recognition based on Deep neural network methods to solve occlusion problems" 978-1-7281-0003-6/19/$31.00ⒸIEEE

[14]M. Szymkowski and K. Saeed (2017) "A Novel Approach to Fingerprint Identification Using Method of Sectorization" International Conference on Biometrics and Kansel Engineering 978-1-5386-3401-1/17/$31.00ⒸIEEE

[15]K. Cao and A. K. Jain (2018) "Automated Latent Fingerprint Recognition" 0162-8828 ⒸIEEE DOI:10.1109/TPAMI.2018.2818162

[16]B. M. Durti, J. M. Targino and C. A. D. M. Lima (2017) "Biometric Recognition based on Fingerprint: A Comparative Study" Workshop of Computer Vision" 0-7695-6357-0/17/$31.00 ⒸIEEE DOI: 10.1109/WVC.2017.00022

[17]D. Meena and R. Sharan (2016) "An Approach to Face Detection and Recognition" International Conference on Recent Advances and Innovations in Engineering (ICRAIE) Jaipur, India 978-1-5090-2807-8/16/$31.00 ⒸIEEE

[18] G. Singh and A. K. Goel (2020) "Face Detection and Recognition System using Digital Image Processing" Proceedings of the Second International Conference on Innovation Mechanism for Industry Applications (ICMIA) 978-1-7281-4167-1/20/$31.00ⒸIEEE

[19] A. S. Dhavalikar and Dr. R. K. Kulkarni (2014) “Face Detection and Facial Expression Recognition System” International Conference on Electronics and Communication System (ICECS) ©IEEE

[20] M. Geetha, R. S. Latha, S. K. Nivetha, S. Hariprasath, S. Gowtham and C.S. Deepak (2021) “Design of face detection and recognition system to monitor students during online examinations using Machine Learning algorithms” International Conference on Computer Communication and Informatics (ICCCI) 978-1-7281-5875-4/21/$31.00 © IEEE

[21] J. S. Shah, M. Sharif, M. Raza, M. Murtaza and S. U. Rehman (2015) “Robust Face Recognition Technique under Varying Illumination” Journal of Applied Research and Technology Vol. 13 February, 2015

[22] C. Jiang, Y. Zhao, W. Xu and X. Meng (2009) “Research of Fingerprint Recognition” Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing 978-0-7695-3929-4/09/$26.00 ©IEEE DOI: 10.1109/DASC.2009.102

[23] V. Ghenescu, R. E. Mihaescu, S. V. Carata, M. T. Ghenescu, E. Barnoviciu and M. Chindea “Face Detection and Recognition Based on General Purpose DNN Object Detector”

[24] Mikai Yang (2020) “Fingerprint Recognition System Based on Big Data and Multi-Feature Fusion” International Conference on Culture-oriented Science & Technology (ICCST) 978-1-7281-8138-7/20/$31.00©IEEE DOI: 10.1109/ICCST50977.2020.00097

[25] Jagadeesh H. S., Suresh B. K. and K. B. Raja (2016) “Rationale of Class and Feature size on Face Recognition” AICTC ’16, August 12-13, 2016, Bikaner, India©2016 ACM ISBN: 978-1-4503-4213-1/16/08..$15.00 DOI: 10.1145/2979779.2979810

[26] Arya J. L and Safuvan T. (2015) “Assessment of Image Quality in Face, Fingerprint, Iris, Palm print and Knuckle point for Detection of Fake Biometrics” International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 NCETET-2015 Conference Proceedings Volume 3, Issue 05

[27] A. K. Singh, P. Joshi and G. C. Nandi (2013) “Development of a Fuzzy Expert System based Liveliness Detection Scheme for Biometric Authentication” The 11th International Conference on Signal and Image Processing (ICIP) Published by Elsevier B. V.

[28] Elsevier Research Intelligence (May 2016) “Elsevier Fingerprint Engine”

[29] R. Kute, V. Vyas and A. Anuse (2021) “Transfer Learning for Face Recognition using Fingerprint Biometrics” in Journal of King Saud University – Engineering Sciences DOI: 101016/j.jksues.2021.07.011

[30] R. Rahim, T. Afriliansyah, H. Winata, D. Nofriansyah, Ratnadewi and S. Aryza (2018) “Research of Face Recognition with Fisher Linear Discriminant” in 4th International Conference on Operational Reasearch(InteriOR) IOP Conference Series: Material Science and Engineering 300 (2018) 012037 DOI: 10.1088/1757-899X/300/1/012037