**Project Report**

|  |  |
| --- | --- |
| **Course Name (NICF)** | ***PCP Bundle-Artificial Intelligence*** |
| Product Name (Marketing & Sales) | ***PCP Bundle-Artificial Intelligence*** |
| **Module Name (NICF)** | **NICF-Introduction to Python and AI for Data Science(SF)** |
| Product Name (Marketing & Sales) | **NICF-Introduction to Python and AI for Data Science(SF)** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Student name** | | **Assessor name** | |
| Thiha Aung | |  | |
| **Date issued** | **Completion date** | | **Submitted on** |
|  | 10/31/20 | | 11/01/20 |
|  | |  | |
| **Project title** | **Face recognition model** | | |

|  |
| --- |
| **Learner declaration** |
| I certify that the work submitted for this assignment is my own and research sources are fully acknowledged.    No need sign  Student signature: Date: date only |

Project Overview: Describe the Project along with Project Outcomes (Explain the Project in your own words in 15 – 20 lines)

**This project is not only a great opportunity for me to learn and understand about OpenCV with python but also helps me understand and solve the technical challenges faced in order to complete this project.**

**The project outcome was not so satisfying but I have learned different methods to tackle the same problem.**

**While doing this project I came to understand that like a series of waterfalls, the OpenCV cascade breaks the problem of detecting faces into multiple stages. For each block, it does a very rough and quick test. If that passes, it does a slightly more detailed test, and so on. The cascades themselves are just a bunch of XML files that contain OpenCV data used to detect objects. I initialized my code with the cascade I want, and then it does the work for me. Converting images to gray-scale. Many operations in OpenCV are done in gray-scale and altering images in one way or another and I am able to manipulate images with python build-in library. This experience of manipulating images from code is so much fun for me as I am able to do things with python coding without needing other third party software.**

**I have faced a ton of issues while doing this project due to the python version problem and also installation issues of other python libraries. In the end I managed to fix all the issues all by myself and that is very satisfying as I realized that solving a problem and being able to overcome all the pain in the process of fixing it is totally worth it. Able to understand computer vision and some level of machine learning is a great experience for me but most importantly is to be able to see the problem and break it down into small chunks and analyst it allows me to solve the issues quicker. So I am more confident then before that I will be able to overcome any challenges that I may face with his knowledge and experience I had with this project.**

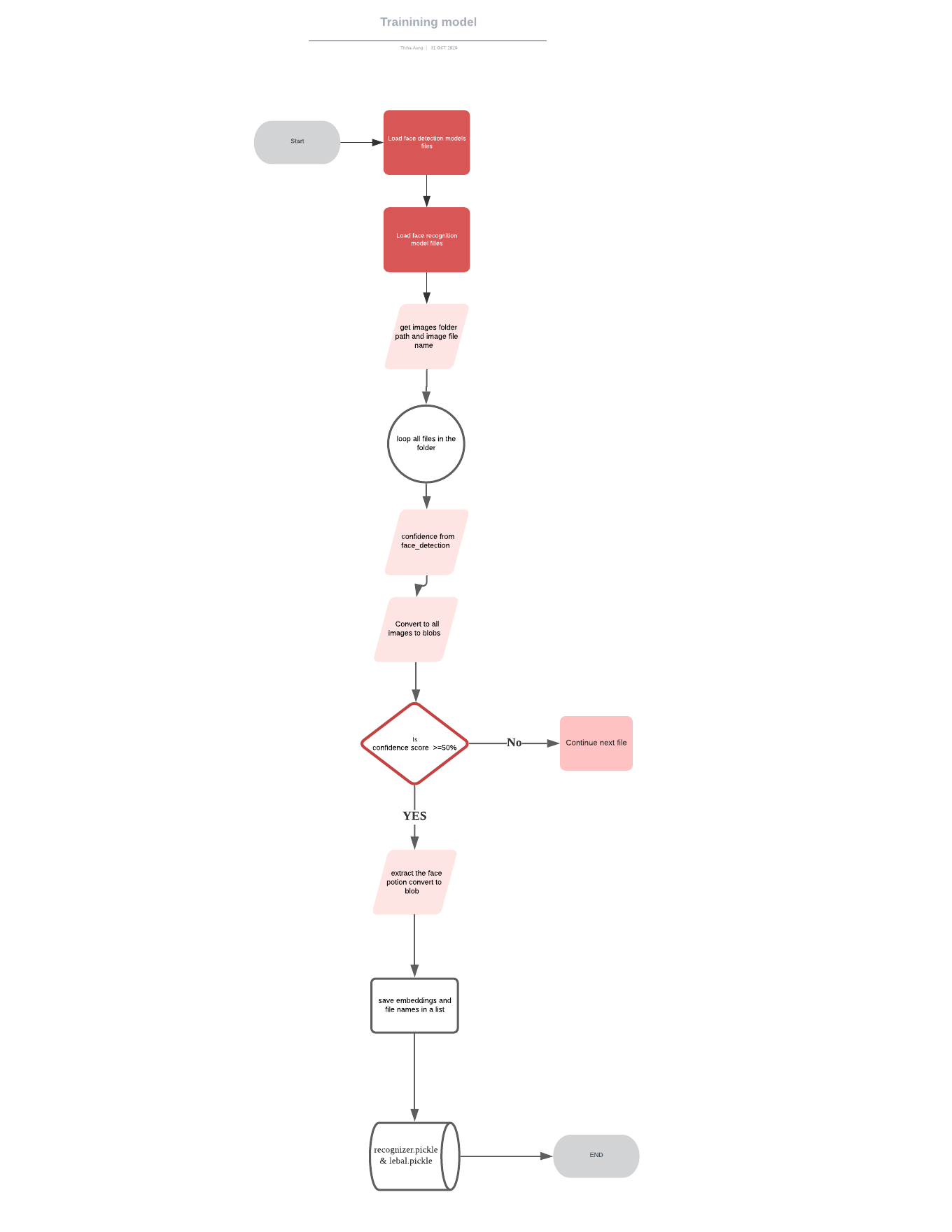
1. **Project Technical Environment:** (Describe the Architecture with Tools used) flowchart -> diagram->programming language

Python programming language

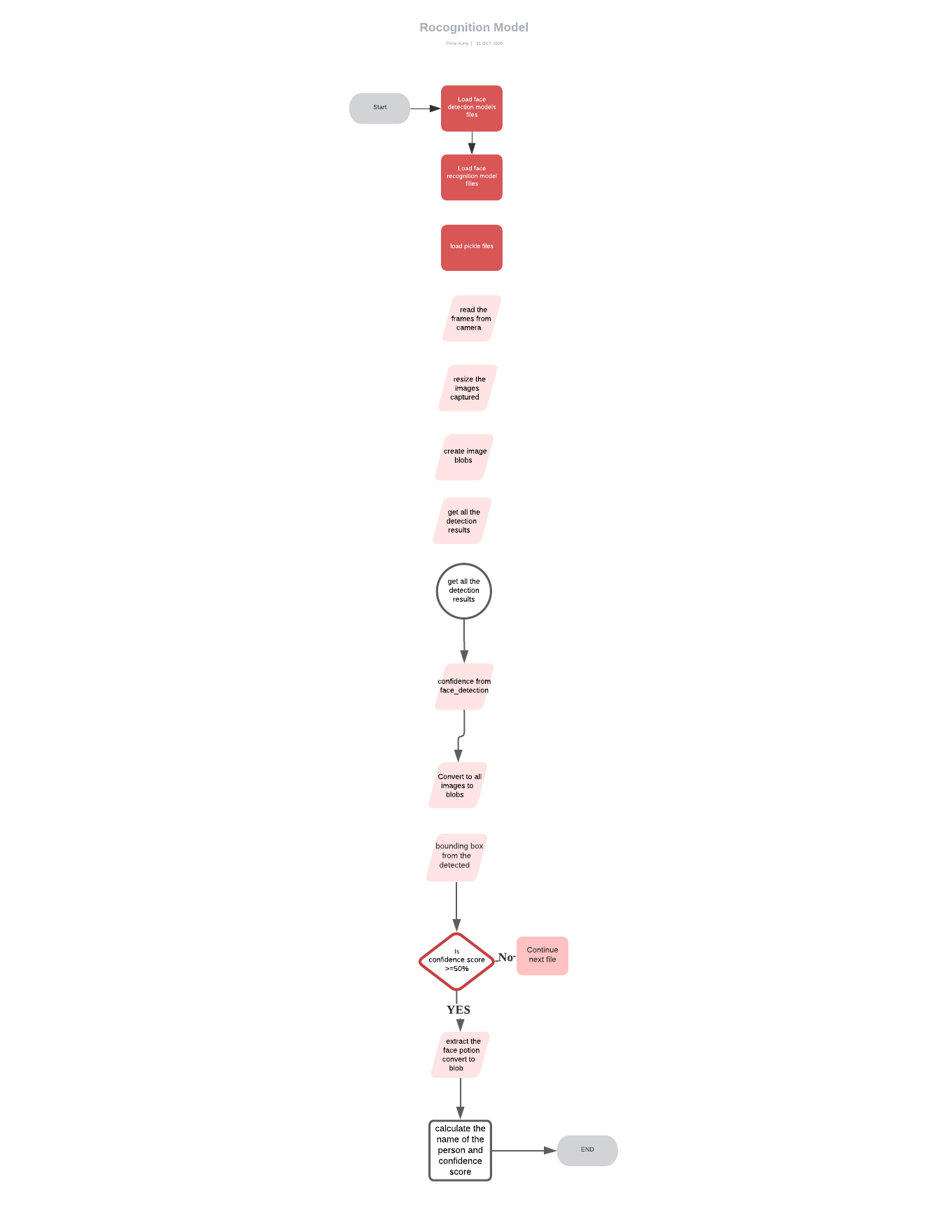
Import tools

* numpy - multi-dimensional array and mathematical operations
* OpenCV2 - facial detection and recognition
* sklearn.svm.SVC - categorizes the data.
* sklearn.preprocessing.LabelEncoder - encoding the levels of categorical features into numeric values
* pickle - save recognized datas as pickle
* imutils - resize image

Training module



1. load face detection model
   1. deploy.protxt file
   2. caffemodel file
2. read both files above and assign as face\_detector
3. load face recognition model
   1. torch openface model
4. read and assign as face\_reconizer
5. get images folder path and image file name
6. assign to data\_base\_path
7. loop all files
   1. in the data\_base\_path and store files names in the filennames list
   2. reading the files one by one
   3. resize the image files
   4. get height and width of image
   5. convert images to blob
   6. passing it face detector
   7. fetch the result and store in face\_detection
   8. get confidence from face\_detection
   9. considers only images if have confidence score >=50% and more
   10. loop and process image one by one and extract the face bounding box then convert to blob and passing it to the recognition model
   11. once detected face in the image, we extract a bounding box of the face potion
   12. create face blob from the face potion
   13. pass the face blob to face\_recognizer model to check with the face embeddings
   14. fetch the result and store to face recognitions
   15. split and get file names
8. save embeddings and file names in a list - face\_embeddings, face\_names
9. create a data dictionary for embeddings and names (data of trained images)
10. create pickle files from data dictionary
    1. recognizer.pickle
    2. label.pickle

Recognizer module

1. load face detection model
2. load face recognition model
3. load pickle files recognizer.pickle and label.pickle that i saved from training script
4. open video camera
5. read the frames from camera
6. resize the images captured from web camera
7. extract height and width of the captured frames
8. create image\_blobs from web camera captured images
9. pass image image\_blog to face\_detector
10. get all the detection results to face\_detections
11. loop all the results in face\_detection
12. get confidence from detection
13. considers only if confidence score >=50%
14. extract the bounding box from the detected images
15. calculate height and width of the images
16. create face blob from images
17. pass it to face recognition model
18. save the result as vec
19. calculate the name of the person and confidence score
20. name get from label.pickle
21. confidence probability from recognizer.pickle
22. **Design the Model:** (Explain the training model which you are designing using ) overview

Two design models

* Training Model
* Recognition Model

Training Model

This model will train the system using images from the database

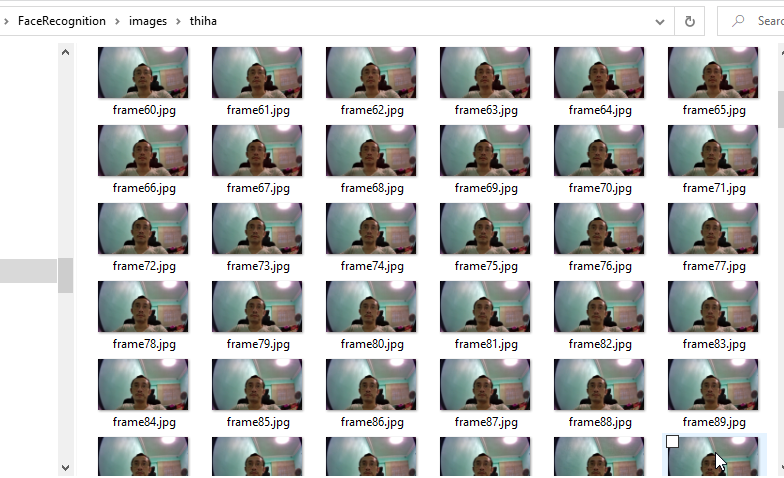
Recognition Model

This will detect the user face and predict the name using the result from training data model

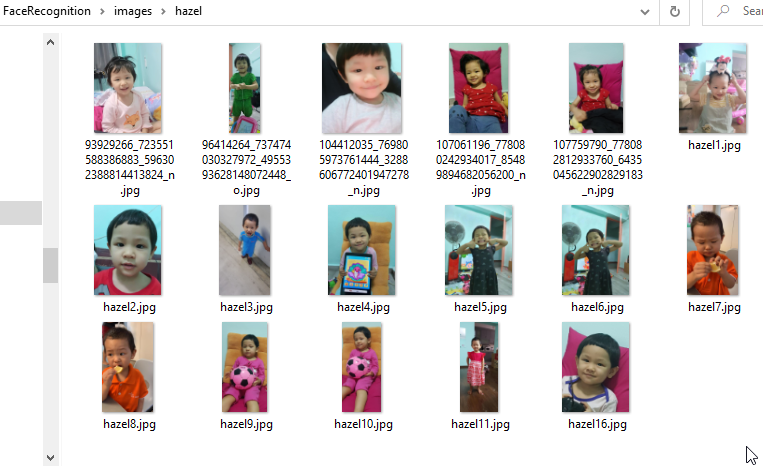
1. **Setting up the Facial Recognition Model:** (Explain the Process for setting up facial recognition Model using Python ) screenshot of all the above steps

**Break down into step by step**

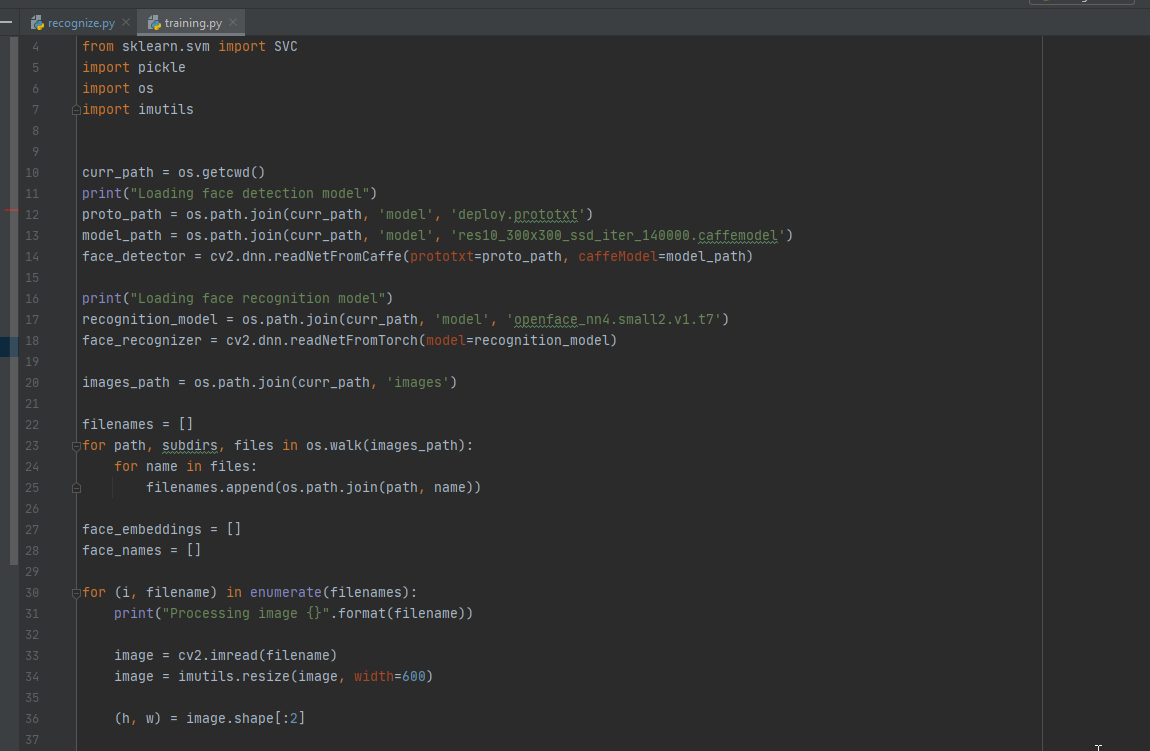
**Using webcam to capture thiha images and save it in thiha folder**

****

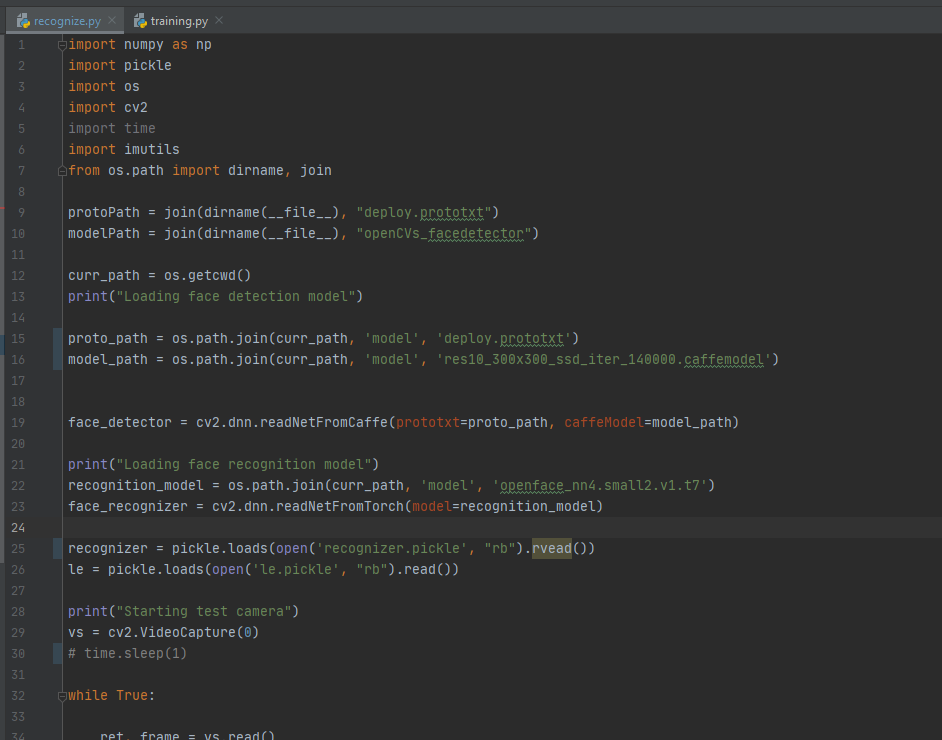
**2. Save hazel images files in images hazel folder**



**3. Run the training script to train the model**



**4. Run the recognition script**

****

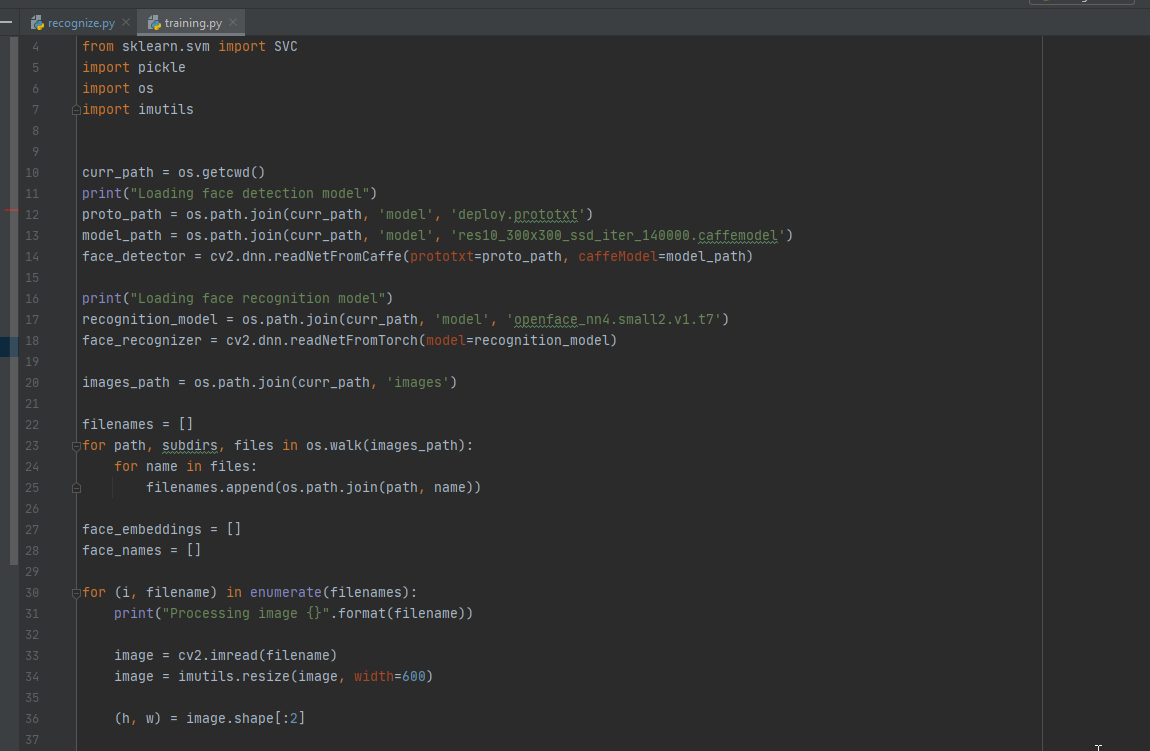
1. **Python code for Data Filtering and Transformation:** (Attach the Python Code used in the Project for Data Transformation)
   1. **Script1 - activity 2** 
      1. **Function 1 – transformation**
      2. **Function2 – 3 filters and one transformation**
      3. **Function 4,5 - filtering, rectangle , put text in desired length , font**
      4. **Text

         Description automatically generated**
   2. **Script2 – activity 3** 
      1. **Filtering based on the confidence**
      2. **Transforming the image**

**Text

Description automatically generated**

1. **Training Data:** (Attach the transformed data as Annexure)
   1. **Trainging script: training will train the system and save the label and recognized data in pickle file format**

****

* 1. **Screenshot of training image folders**

**Text

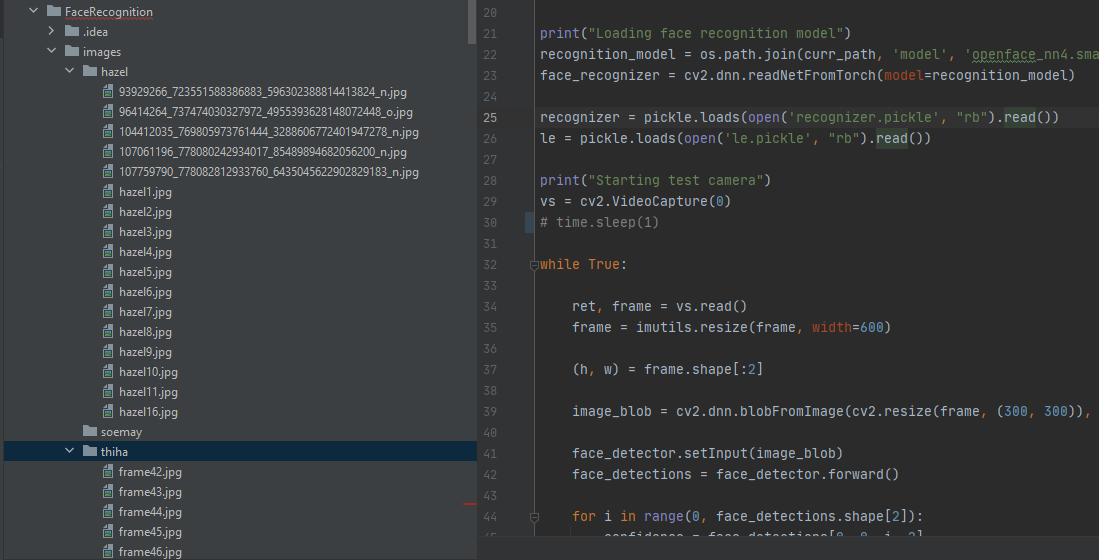
Description automatically generated**

**Graphical user interface, text

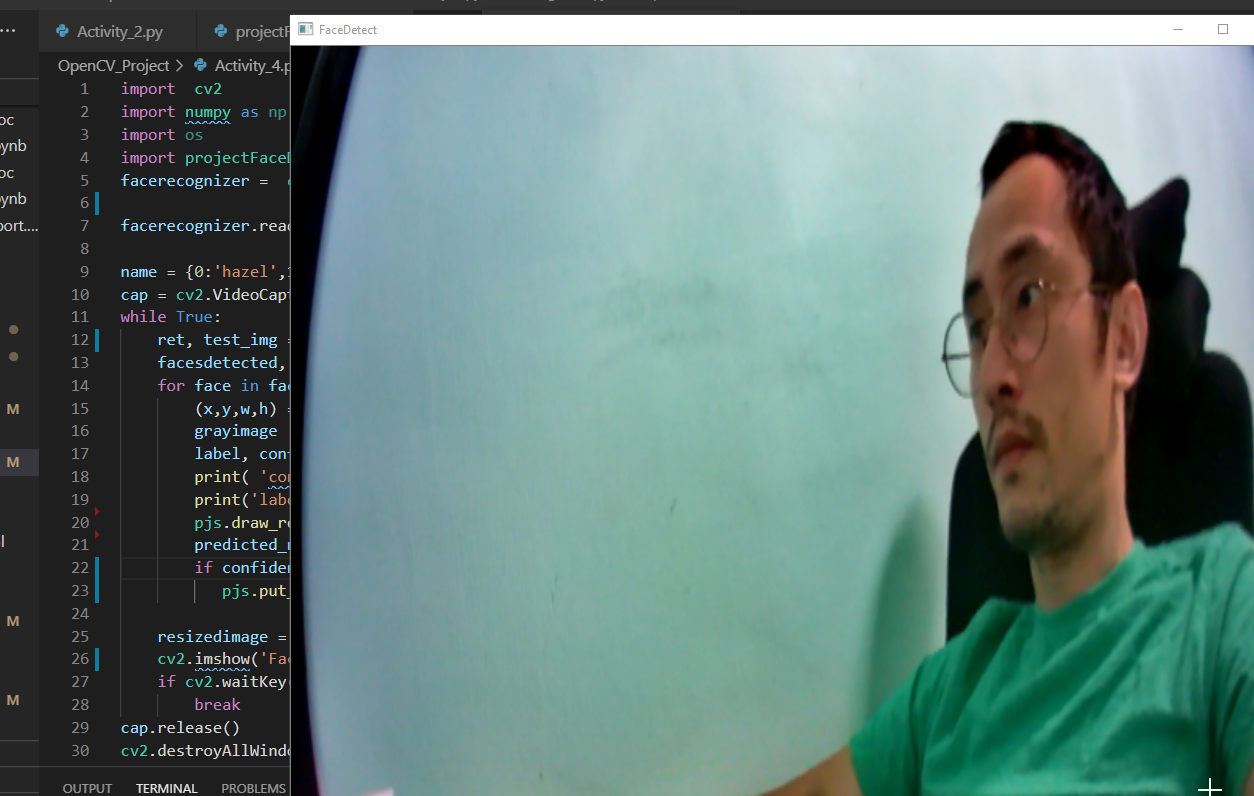
Description automatically generated**

1. **Testing dataset:** (Explain how you performed the activity 3 along with the output as Annexure) (script 2 output)

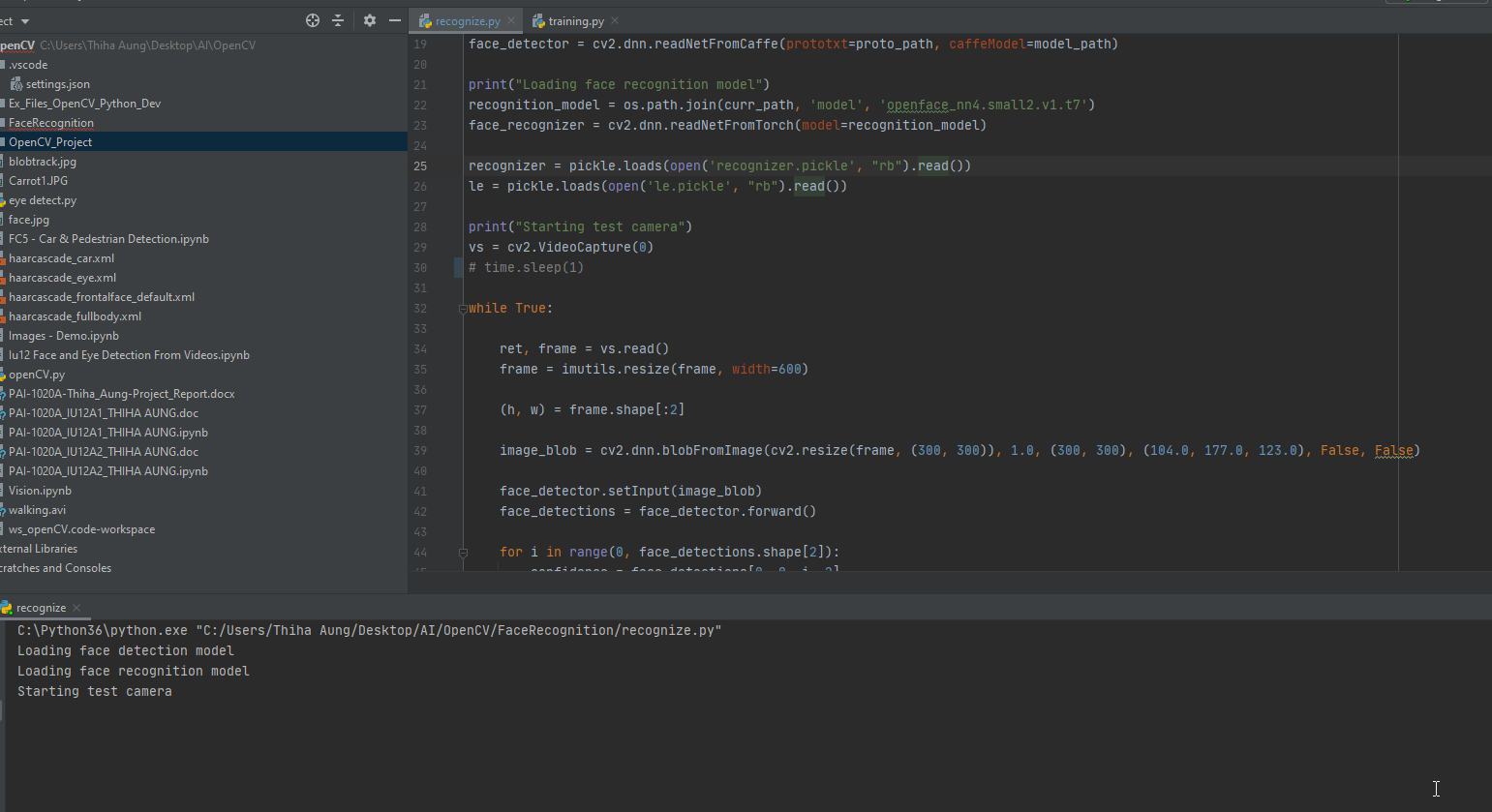
**Testing image Folder**

****

1. **Implementing script 4:** (Explain how you have evaluated the Model) script 4
   1. **Run activity 4 script - video to image conversion**

****

* 1. **Run training script:**
     1. **To train the system for recognition script to recognize face and name of the person**
  2. **Run recognition Script**
     1. **To test the training model**

****

1. **Annexure 1**
   1. **Training model**
2. **Annexure 2**
   1. **Recognition model**
3. **Annexure 3**