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 | |
| **Saminathan Renganayagi** | | **Rajendra Kissan** | |
| Date issued | Completion date | | Submitted on |
| **1/3/2021** | **13/3/2021** | | **15/3/2021** |
|  | |  | |
| Project title | PAI-0221A Module1 Project Face Recognition | | |

|  |
| --- |
| Learner declaration |
| **I certify that the work submitted for this assignment is my own and research sources are fully acknowledged.**  **Student signature: Date:** |

**Content**

* + - * **Project Overview**
      * **Project Technical Environment**
      * **Forecasting Model**
      * **Setting up the Forecasting Model**
      * **Python Code for Data Filtering and Transformation**
      * **Transformed Data**
      * **Correlation Analysis Process and Output**
      * **Implementing Seasonal Forecasting Model**
      * **Refine Model**
      * **Forecast Output after Model Adjustment:**

**Project Overview: Describe the Project along with Project Outcomes**

**The project Face Recognition is quite interesting as it give us a glimpse of real-life examples of facial recognition using AI, specifically the OpenCV module.**

**The project course has directed us to use the LBPH algorithm – Local Binary Pattern Histogram which is one of the most efficient algorithm in the OpenCV package other than the Fisherfaces & EigenFaces which are older methods.**

**This project involves getting the images and transforming it into gray scale with the classifiers and detect face. But before detecting the faces, we have to gather tr**

**aining data with the subject character for face recognition.**

**Training data involved gathering webcam live feed and also from youtube data for the target character, filtering and cleaning the data accordingly. Once the image data has been sorted, filtered and cleaned the training process takes place. Image data consists of at least 400 data for each character. Total characters I had used for the training data is 4 characters altogether.**

**Initially, I had downloaded the about 20+ pictures for training purposes. After the initial testing of the test images, I felt the accuracy was not ideal. I had to download more pictures for the training purposes and downloaded about 30+ pictures more training. But still the accuracy and hit rate was not ideal.**

**I had tried downloading the Youtube video of the target character and using the video feed to use the python capture individual frames. I had to clean and filter the image data to have a closeup images and higher resolution images.**

**For activity 4, I had to download my own webcam live video into frames. Retraining was executed and used in activity 5 for the live feed identification of my live video image to detect and predict my name as one of the character.**

**1. Project Environment:**

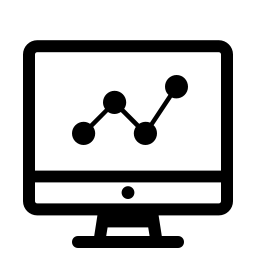
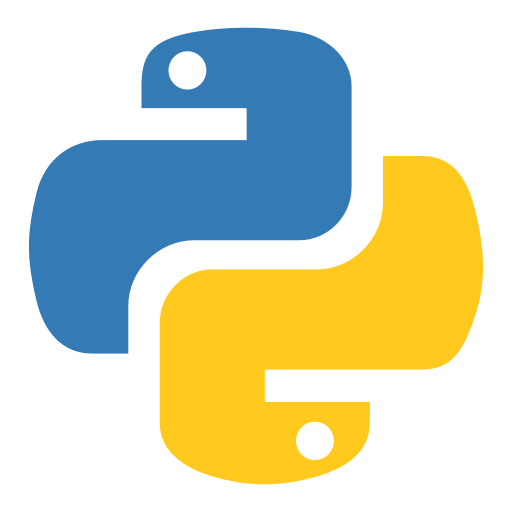
**Jupyter Notebook**

****

**System**

**SS**

**Python 3.8**

**Video source for Data Frame Image Capture**

**Image Source**

**2. Design the Model:**

**We are using the LBPH(Local Binary Pattern Histogram) training model in OpenCV to train the images.**

**It involves the following steps:**

* 1. **Getting the prepared images of the target character**
  2. **Put those images into the subfolders of the directory**
  3. **Parse through the subfolders of images of different target characters**
  4. **Activate the opencv LBPHFaceRecognizer\_create() trainer program for LBPH algorithm**
  5. **Execute the Train function with the parsed through images and subfolder information.**
  6. **Write the train data into “training.yml” for later use**
  7. **Use another program to read and identifiy in activity 3 to identify the test images**

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

1. **Activity 2( .py or ipynb) ( created 5 functions)**
2. **PAI-1021A\_ Saminathan Renganayagi Module1 Project Face Recognition. .py or .ipynb**

**4 functions**

**3. Function 1 = for image**

* **Converting the image to gray Cv2.cvtcolor( cv2.color\_bgr2gray)**
* **Harcascade classifier( frontal face xml )**
* **Call method detect multiscale( grayimage)**
* **Return the faces, grayimage**

**4. Function 2 with a parameter directory**

* + - * **Create empty list Faces = []**
      * **Create empty face id = []**
      * **For path, subdirnames, filenames in os.walk(directory)**
      * **For filename in filenames:**

**If filename.startwith(“.”):**

**print skiping system file**

**Continue**

* **Id = os.path.basename(path) .. reading fetching the subdirectory name ( c:\folder\subfolder**
* **Image path = ( AnithaImg53.jpg)**
* **Full path using join is ( “\folder\subfolder\ AnithaImg53.jpg”)**
* **Print full path image**
* **Full\_path\_image = Cv2.imread ( full path)**
* **If full\_path\_image is none:**

**Print not loaded**

* **Call the function 1 to detect the image based on full path**
* **If face\_rect[0] != 0, continue don’t need to do anything**
* **(x,y,w,h ) = face\_rect[0]**
* **Faces.append()**
* **Faceid.append(int(id)**
* **Return faces, faceid**

**5. Function 3 = training images using faces and faceid of previous function**

* + - **Cv2.face.LBPHFacerecognizer\_create() = algorithm**
    - **Train the haarclassifier using function train(np.array(faceid)**

**6. Function 4 = draw rectangle**

* 1. **Call function cv2.rectangle(image, (x,y),(x+w,y+h),(255,0,0)**

**7. Function 5 = text on the label**

* 1. **Cv2.puttext(image, test)**

**\*\*\*\*\*\*\*Main program to call all the functions \*\*\*\*\*\*\*\*\*\*\*\***

1. **Import packages cv2, numpy, os**
2. **Testimage = Use cv2.imread( “ read test images \TestingImage.jpg“)**
3. **Faces\_detected,grayimg = (fn.function1)function1 of script1(testimage)**

**Point 11-13 to be executed only when running for first time**

1. **Faces, faceid = Call the function 2 of script1( fn.function2)(‘training images’)**
2. **Facerecognizer = Call the function3 of script1(fn.function3)(faces, faceid)**
3. **Facerecognizer.write(‘trainingdata.yml’)**

**Point 14-15 to be executed only when running from the second time onwards**

**14. Cv2.face.LBPHFaceRecogniser\_create()**

**15.  Facerecognizer.read(‘trainingdata.yml’)**

**16. Name = {0:”Sharuk khan”,1:”Anitha”,2:”Dhayanithi”}**

**17. For face in Faces\_detected,:**

* **(x,y,w,h) = face**
* **Grayimage = grayimg [y:y+h,x:x+h]**
* **Label , confidence = face\_recognizer.predict(grayimage)**
* **Print( confidence )**
* **Function 4 of script 1( fn.function4(testimage, face)**
* **Predicted\_name = Name(Label)**
* **If confidence > 40**
* **Continue**
* **Function5 puttext of script1( fn.function5(testimage, predicted\_Name))**

**18. Resizedimage = cv2.resize(testimage,(1000,1000))**

**19. Cv2.imshow()**

**20. Cv2.waitkey(0)**

**21. Cv2.destroyallwindows**

**Activity 4 Script3**

**Create while loop to convert video to image until ‘q’ key is pressed**

**1. Import cv2**

**2. Cap = Cv2.videocapture() --- laptop cam opens**

**3. Count  = 0**

**4. While True:**

1. **Cap.read()—will return return, image**
2. **If image is not return**

**continue**

1. **Cv2.imwrite(<name>%d.jpg) ----save as jpg image format %d number**
2. **Count +=1**
3. **Resizeimg = cv2.resize(image ,1000,700)**
4. **Cv2.imshow(“resized”,resizeimg)**
5. **If cv2.waitkey(0) == ord(“q”)**

**Break**

**5. Cv2.release()**

**6.Cv2.destroyallwindows()**

**Activity 5**

1. **Import packages cv2, numpy, os**
2. **Import script1/py1 as fr**
3. **Facerecognizer = Cv2.face.LBPHFaceRecogniser\_create()**
4. **Facerecognizer.read(‘trainingdata.yml’)**
5. **Name = {0:”Sharuk khan”,1:”Anitha”,2:”Dhayanithi”}**
6. **Cap = cv2.videocapture(0)**
7. **While True:**
   * **Cap.read()—will return return, image**
   * **Facesdetected, grayimg = fn.function1(image)**
   * **For (x,y,w,h) in facesdetected:**
   * **(x,y,w,h) = face**
     + - **Grayimage  = grayimg [y:y+h,x:x+h]**
       - **Label, confidence = face\_recognizer.predict(grayimage)**
       - **Print( confidence , label )**
       - **Function 4 of script 1( fn.function4(testimage, face)**
       - **Predicted\_name  = Name(label)**
       - **If confiden <40**

**Continue**

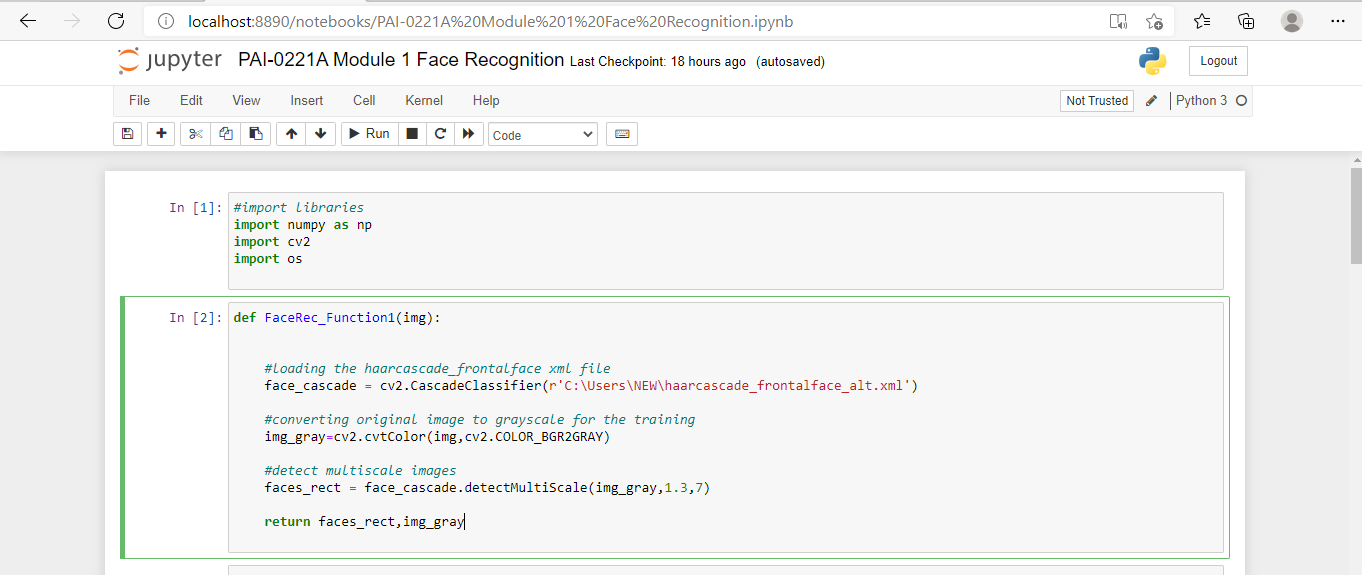
* + - * **Function5puttext of script1( fn.function5(testimage, Predicted\_name**
  + **Resizedimage = cv2.resize(testimage,(1000,1000)**

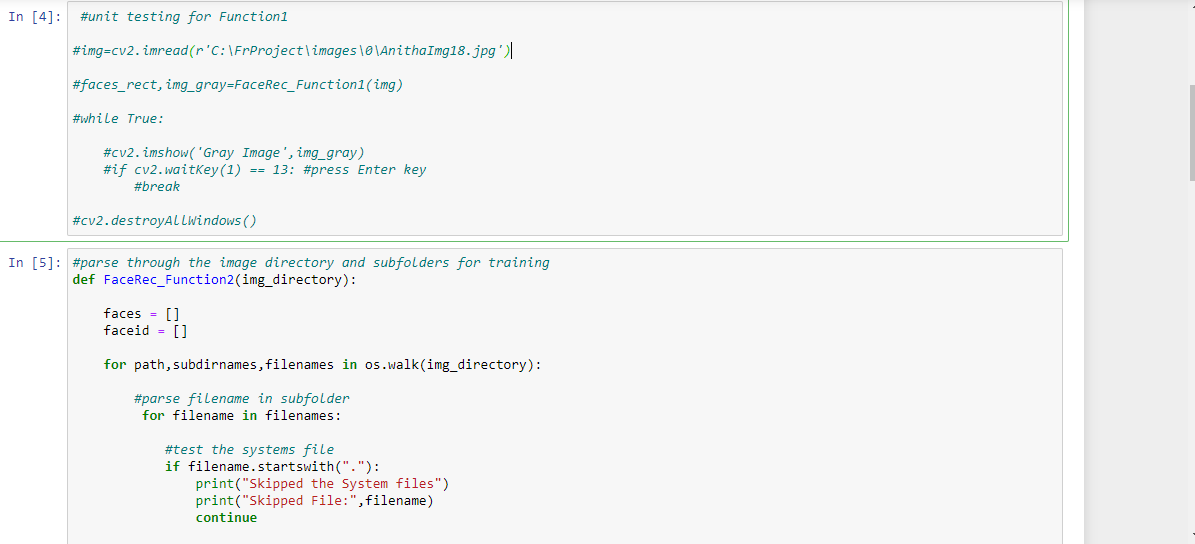
8. **Cv2.imshow(resizedimage)**

**9. Cv2.waitkey(0).**

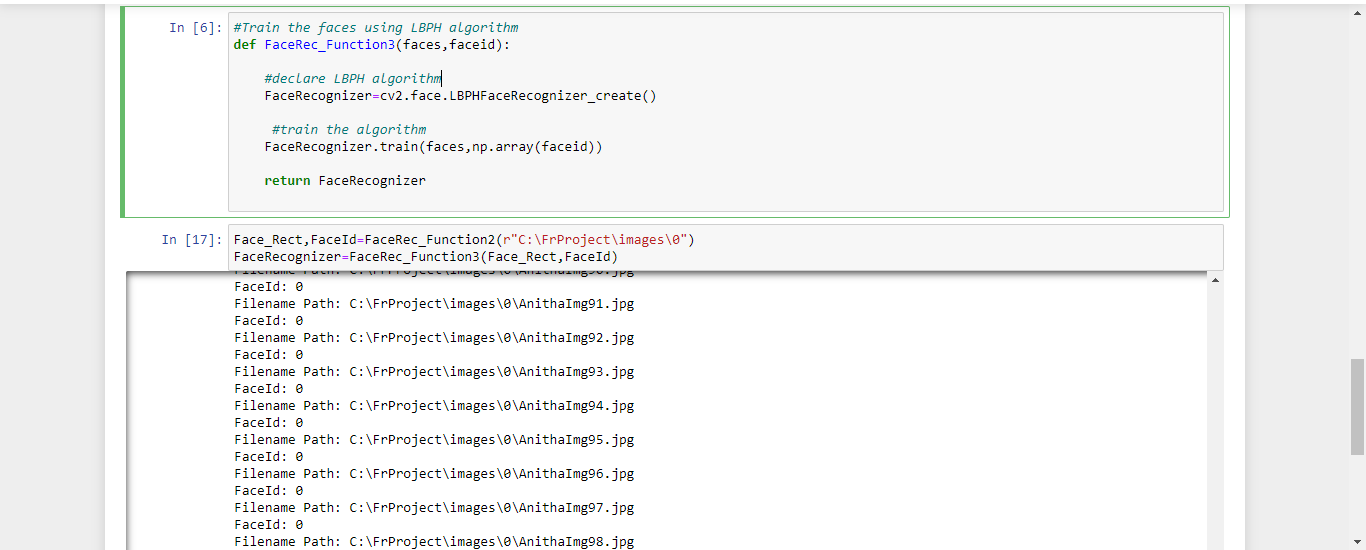
**10. Cv2.destroyallwindows**

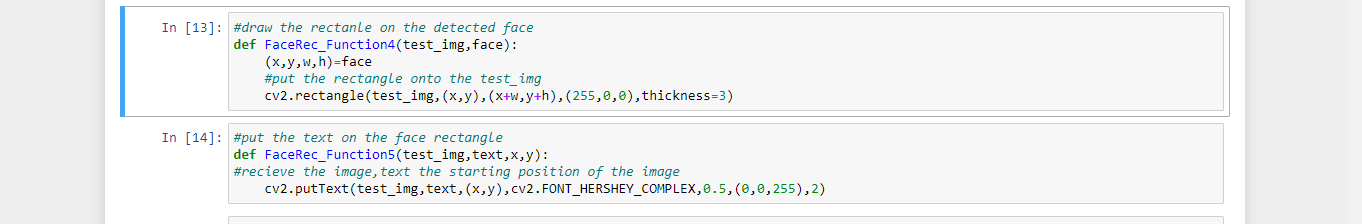
4. Python Code For Data Filtering And Transfermation:



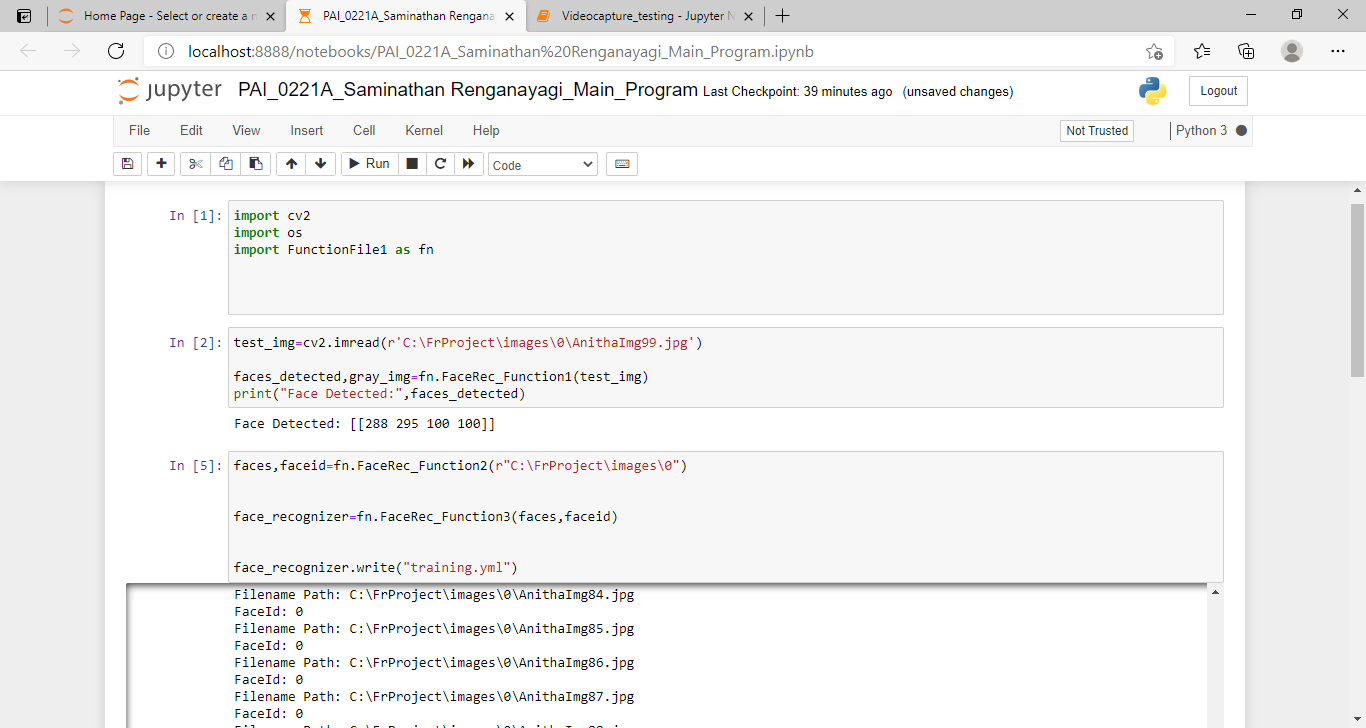
****

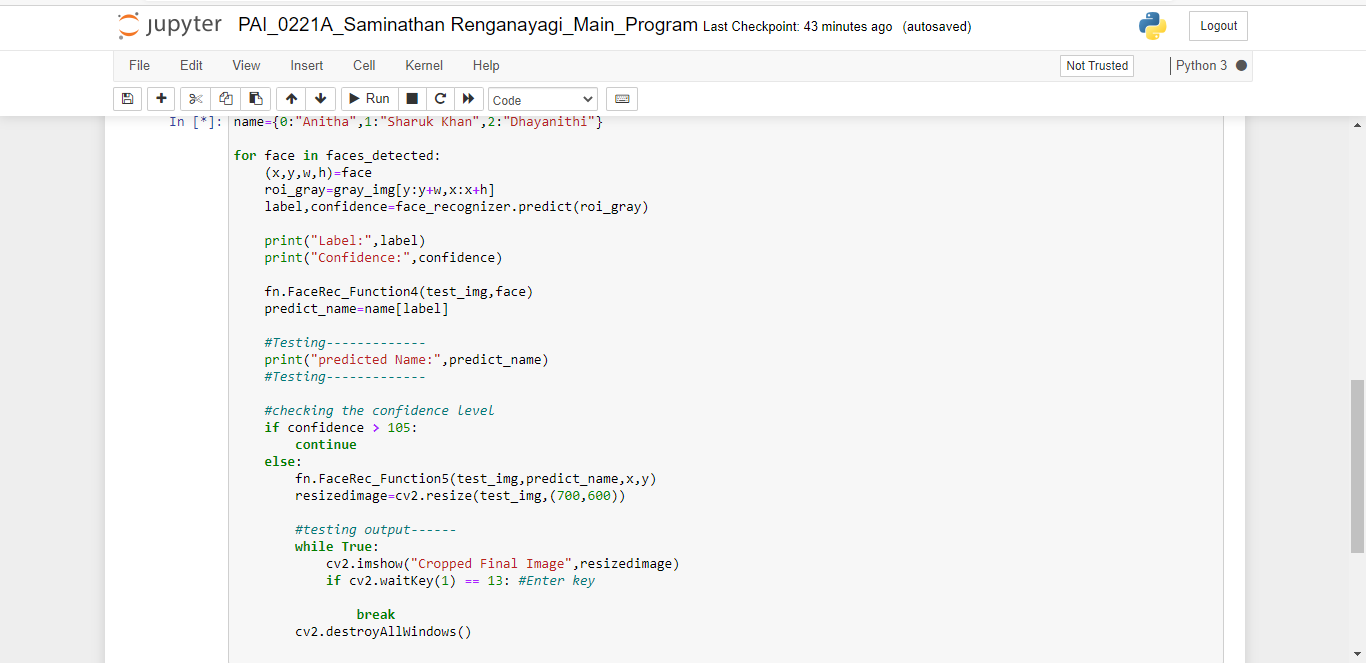
Project Technical Environment: 

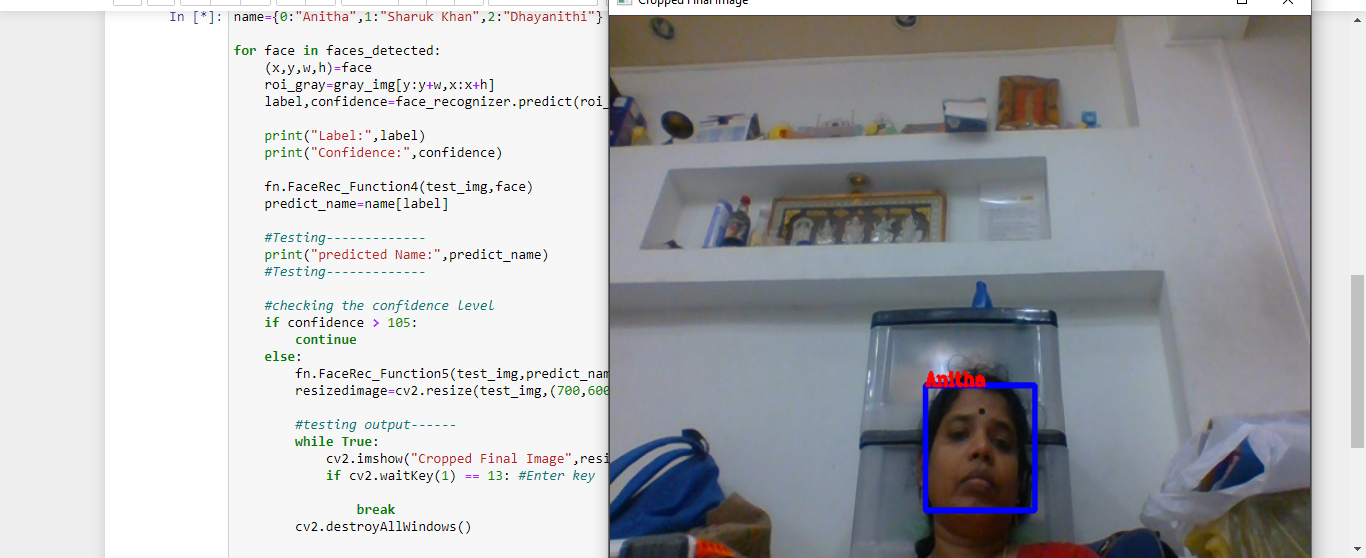


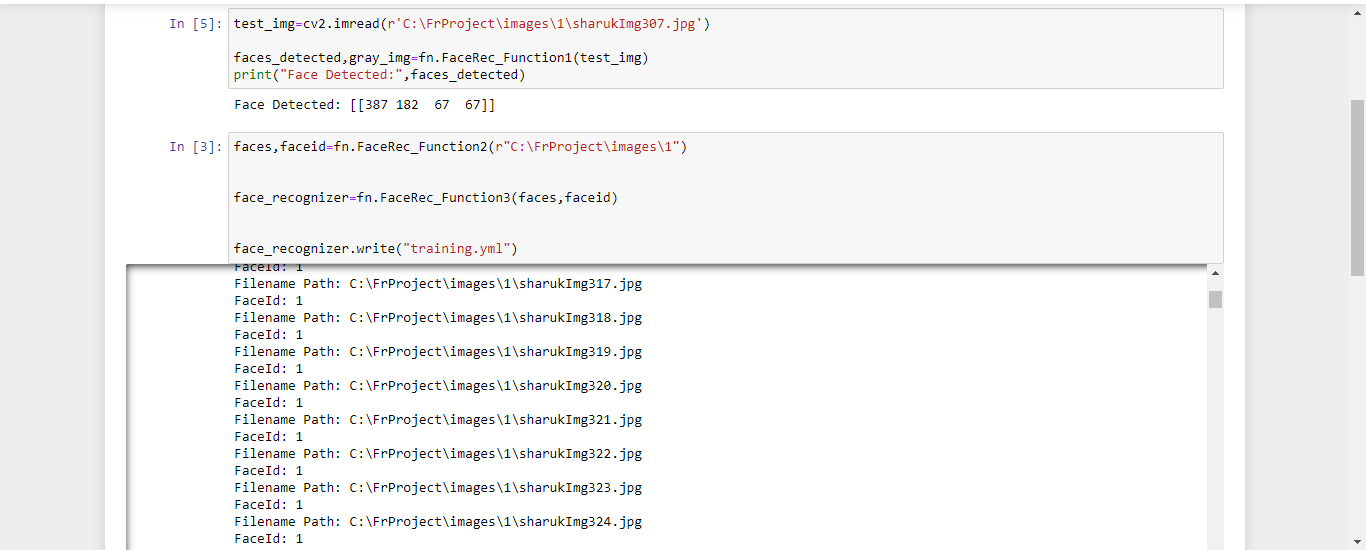


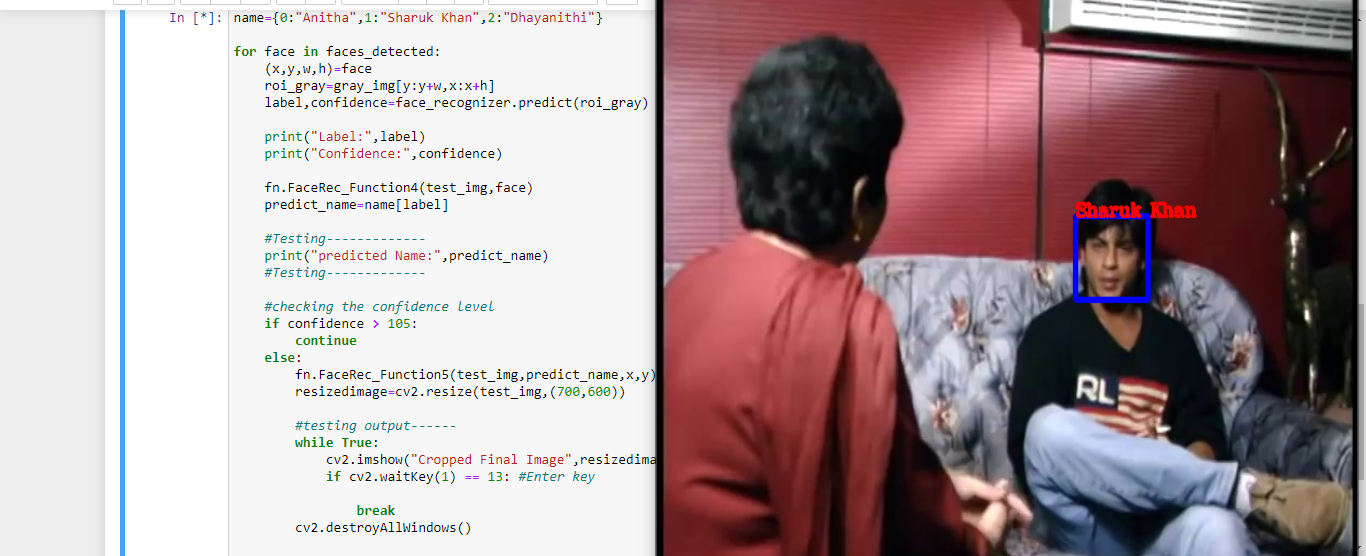
5.Traing Data:( Calling the FaceRec\_Function 3,FaceRec\_Function4,faceRec\_Function for Training the Model)

1. 



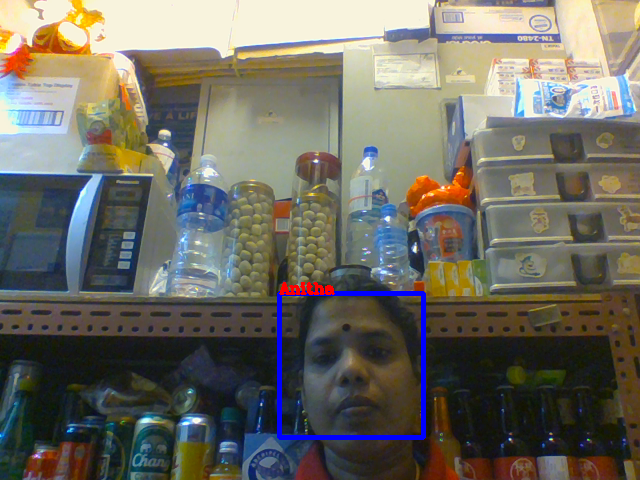




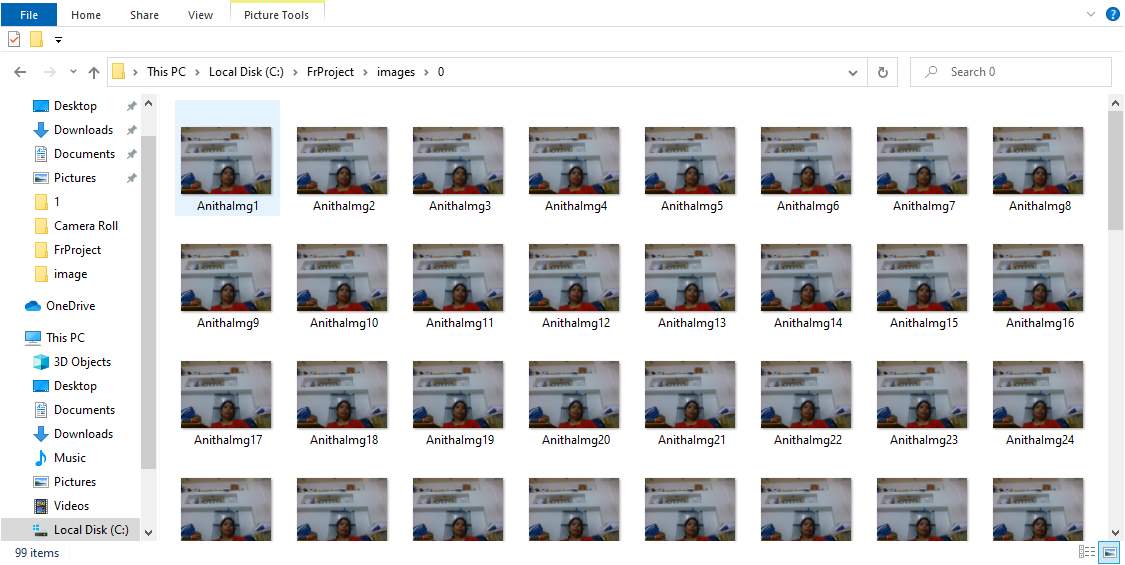


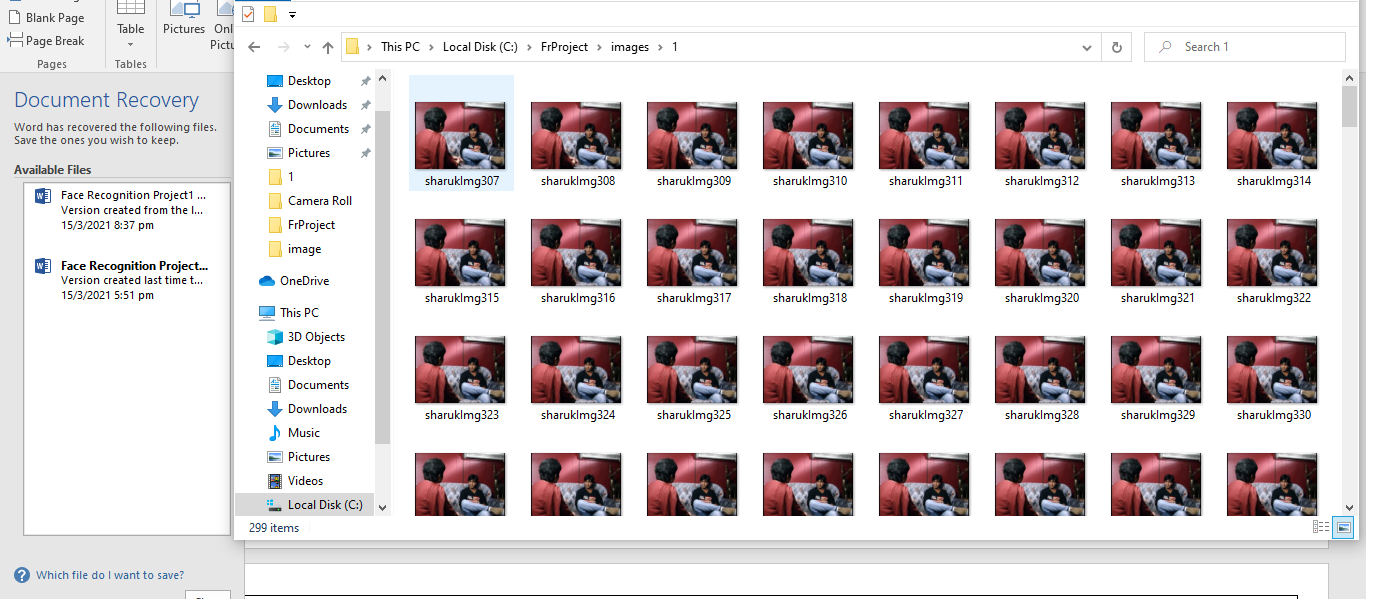
Video Capture Testing



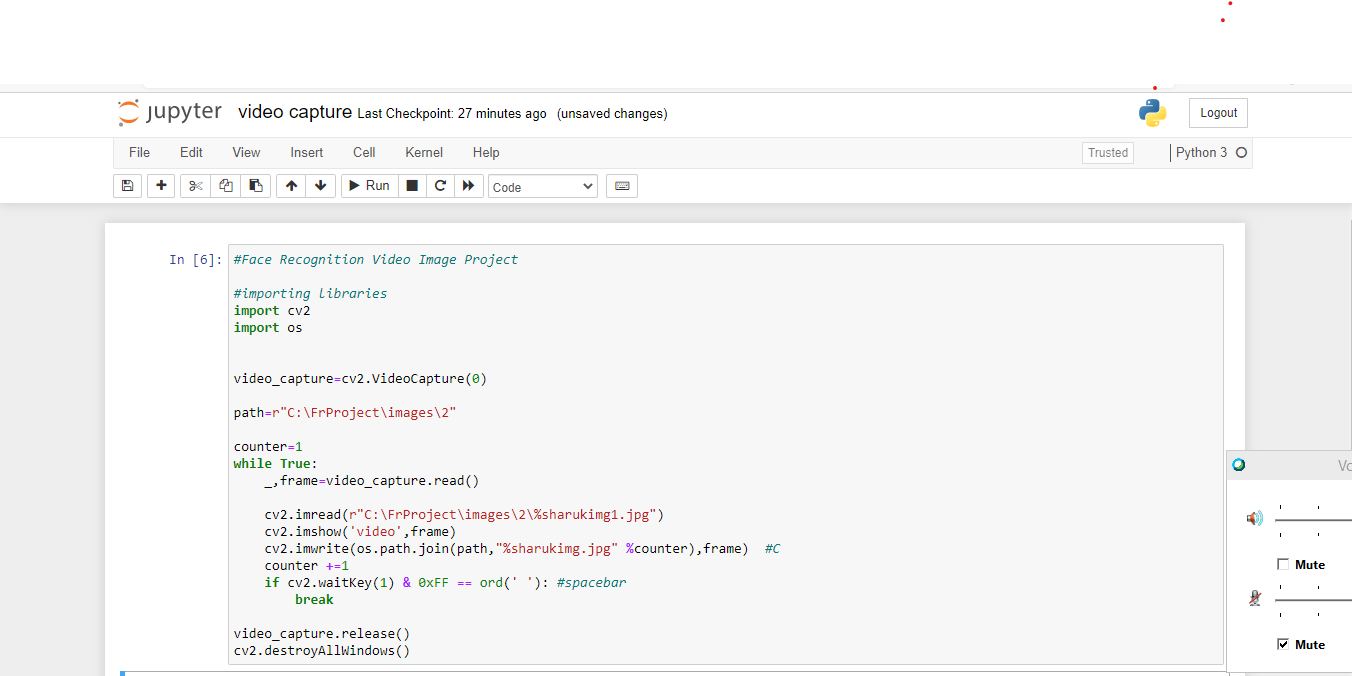


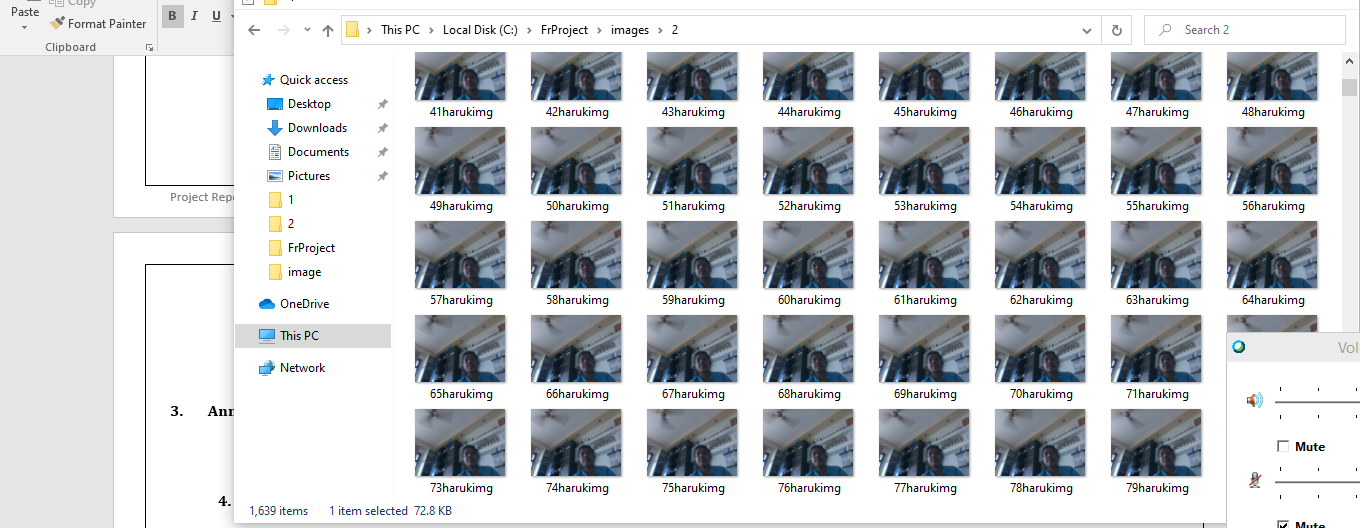
6. Testing Data Set





Activity 4 is Capture The Frame image Frame the video live feed And II have implemented as below the python code





1. Annexure 1
   1. Script1
2. Annexure 2
   1. Script 2
3. Annexure 3
   1. Script 3
4. Annexure 4
   1. Script 4