USER GUIDE

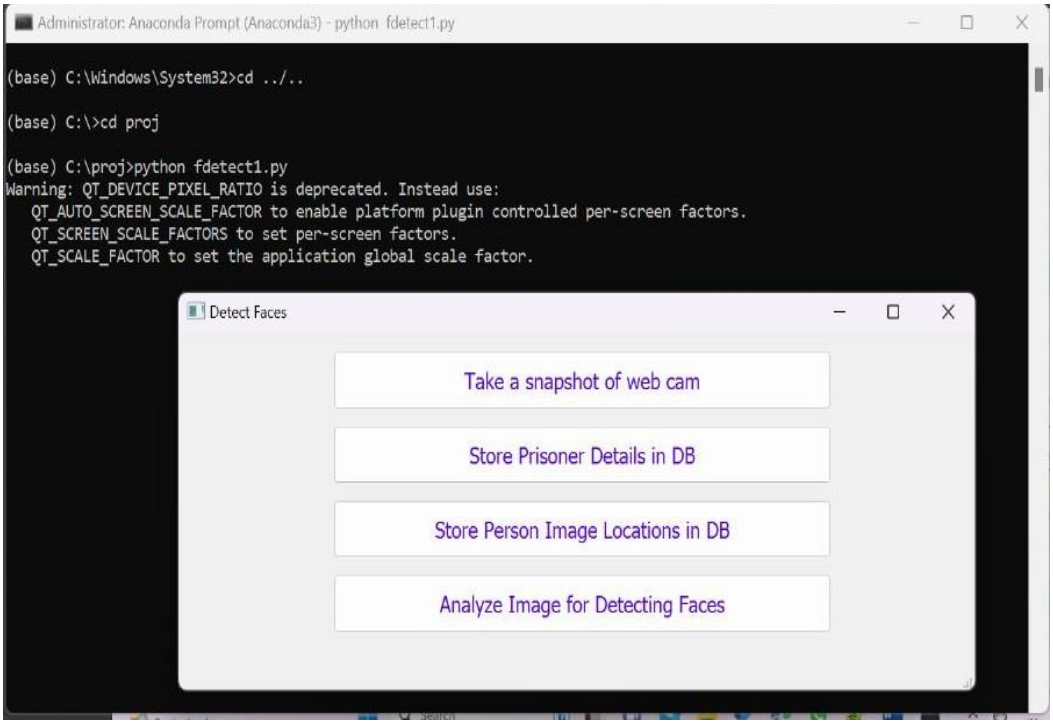
In the code implementation, we have used python language where the main logic of the system involves in the utilization of Squared Differences Based Image Matching technique to compare the faces in the given main image, with the images in the file system, one by one using an iterative structure.

To open the Home GUI screen, First open anaconda prompt as run as administration and type the following commands.

(base) C:\Windows\System32> cd ../..

(base) C:\>cd proj

(base) C:\proj>python fdetect1.py

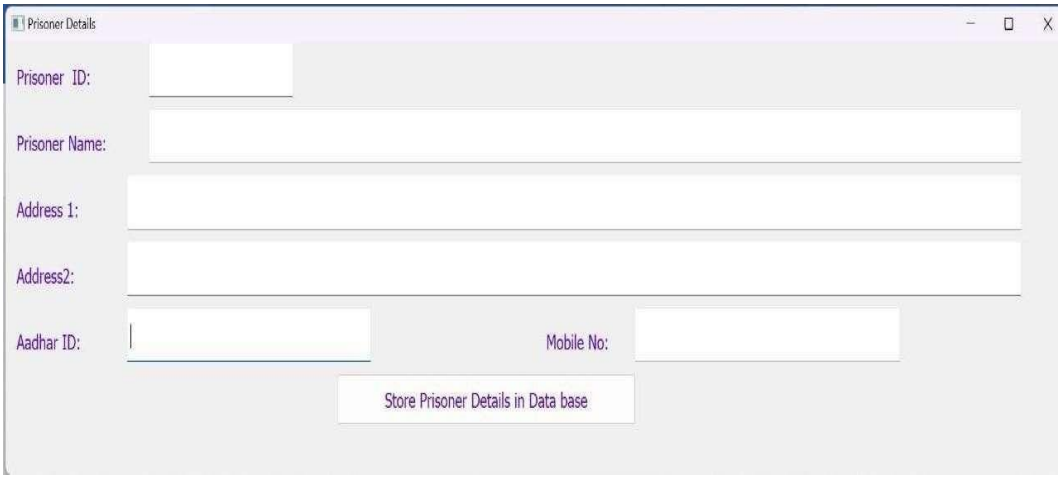


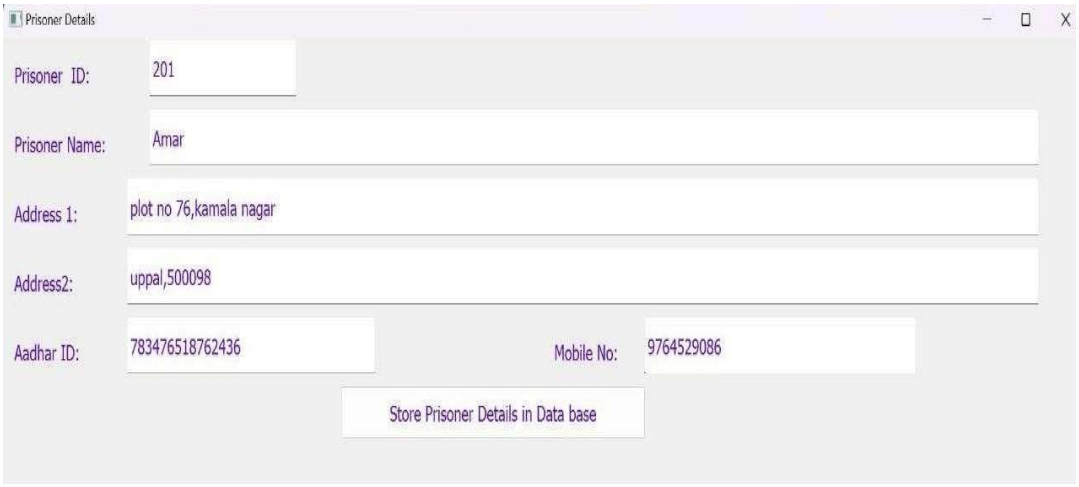
By typing the above commands in anaconda prompt, the home GUI screen pops up.

The Home GUI screen of the project consists of 4 buttons, which are Take a snapshot of web cam, Store Prisoner Details in database, Store Person Image Locations in database and Analyze Image for Detecting Faces.

When the take a snapshot of web cam button is pressed the system directly connects to the web cam where we are able to take a snapshot of the individual. The snapshot taken by the webcam is stored in the database file system.

As our aim is to identify the individual image in a group image and generating a report of efficient individuals identified, firstly we have to save the individual images and their details into the database. To save their details, click on Store Person Details in Database button where a new window opens with empty text boxes. With filling the details like Prisoner ID, Prisoner Name, Address 1, Address 2, Aadhar ID and Mobile No the information is saved in database.





To store person images in database click on Store Person Image Location in database where a new window pops up with Name of the person(individual) and the name of the image file name.

After entering the details click on the store details button where all the given information is stored into data base.

To check whether is the information stored or not we can check in SQlite 3 by following commands.

```
Sqlite> .open wbcam1
Sqlite>  .tables
persons  prisoner
```

Now available tables appears which are persons and prisoner

To view the tables follow below commands

```
Sqlite> Select * from persons;
```

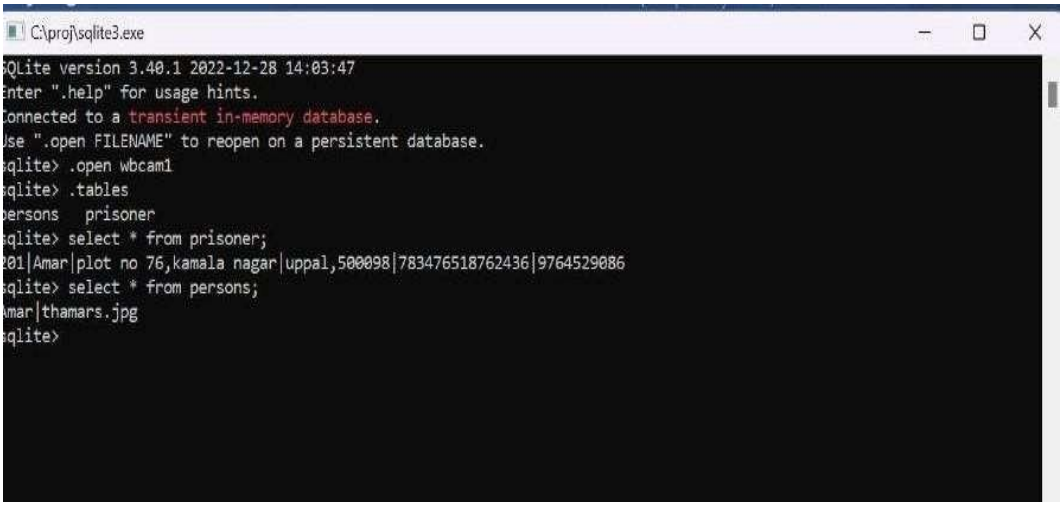
The details are saved according to the details filled in the gui screen.

```
Amar|thamars.jpg
```

Where Amar is the person name and thamars.jpg is the file name of the image

```
Sqlite> Select * from prisoner;
```

The details are saved according to the details filled in the gui screen.



201|Amar|plotno 76,kamala nagar,uppal,500098|7834765418762436|9764529086

Where 201 is the prisoner ID, Amar is the name of the prisoner, plot no 76,kamala nagar,uppal,500098 is the address given, 7834765418762436 is the Aadhar ID of prisoner and 9764529086 is the mobile number of the prisoner.

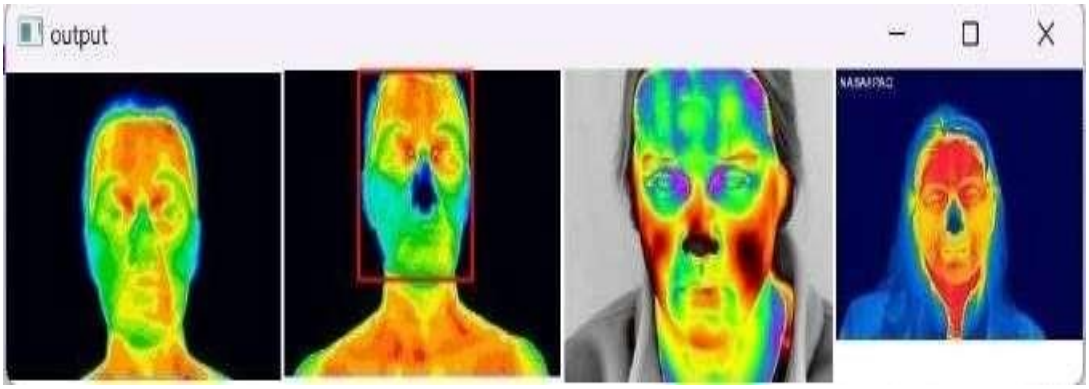
By this we can confirm that the information which is have been filled through GUI screen is safely stored into data base in a structured format without any errors.

To analyze the images click on Analyze Image for Detecting Faces button where a window pop ups asking for group image file.

After filling the name of the group image which is in jpg format we can click on analyze button to analyze and detect the image where as by clicking the generate report button a report is generated by people identified and number of individuals identified.

By using CV2.TemplateMatching with sum of squared differences we will analyze the smaller image (individual image) on larger image(group image). Template Matching is a method for searching and finding the location of a template image in a larger image. OpenCV comes with a function **cv.matchTemplate()** for this purpose. It simply slides the template image over the input image (as in 2D convolution) and compares the template and patch of input image under the template image. It returns a grayscale image, where each pixel denotes how much does the neighborhood of that pixel match with template. By using sum of squared differences we can measure the distance between the pixel to pixel and identify the smaller image in a group image. The process repeats iteratively until the image is identified. If input image is of size (WxH) and template image is of size (wxh), output image will have a size of (W-w+1, H-h+1). Once you got the result, you can use **cv.minMaxLoc()** function to find where is the maximum/minimum value. Take it as the top-left corner of rectangle and take (w,h) as width and height of the rectangle. That rectangle is your region of template.

Once the analyze button is clicked the individual image (which is smaller image) is analysed with group image(which is larger Image) as described above. When the image is identified a rectangle is drawn on the larger image which is identified with cv2.rectangle() function. By drawing the rectangle it is verified as image is identified.



To generate the report we are using ReportLab tool kit. ReportLab is an open source toolkit for creating PDF documents from Python. It is a very extensive library with many features, from small texts and geometric figures to large graphics and illustrations, all of which can be included in a PDF.

The report consists of people identified in the given image and the number of people identified. We add a count factor, so when ever a rectangle is draw it is verified as an image is analyzed and the count factor is increased. The final count factor is the number of people identified.

People identified in the given image are:, Amar
No.of people identified: 1