Gulnaz Khabibullina - 1001555830
Nishad Aherrao - 1001351291
CSE 4340-001

Final Project Report

Security Camera: Facial Recognition

Team Collaboration Tools and Documentation -

Github Repository (Public) GNU GPL License -
https://github.com/nishad10/faceRecognitionSecurity

Youtube Link for a demo presentation -
https://youtu.be/ikXQcRfVvYs

System requirements -

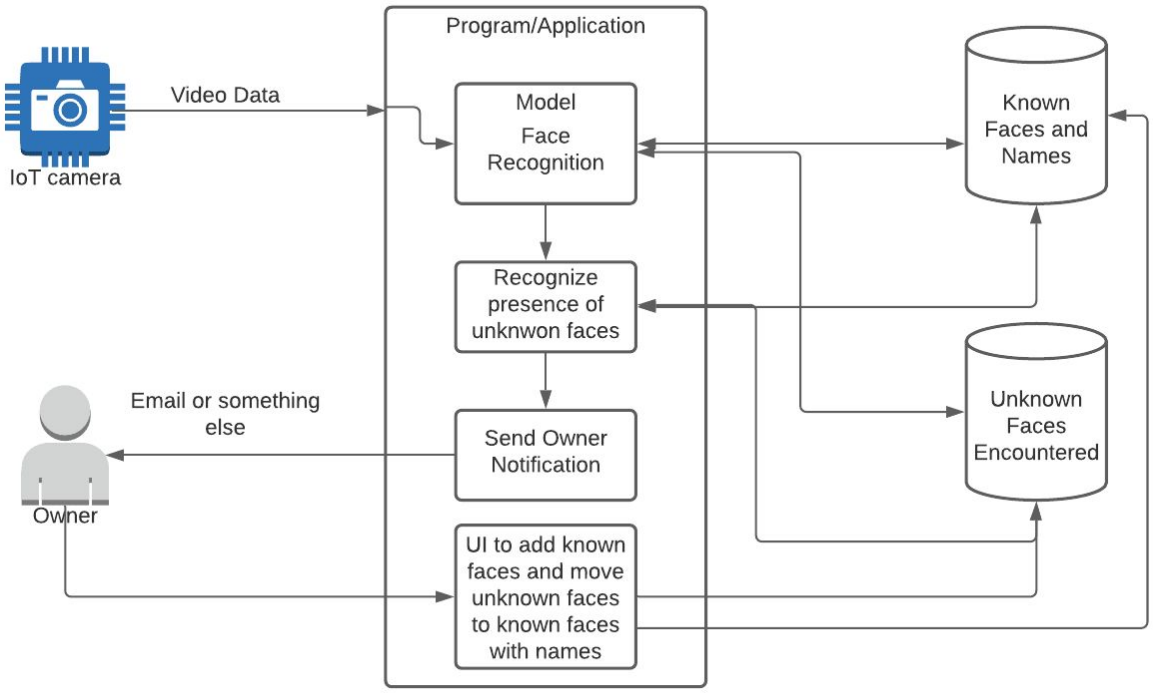
Task	Due Date	Progress
Requirements Planning	October 25, 2020	Completed
Designed Application (High Level)	October 27, 2020	Completed
Decided on exact model to use along with programming language and UI/UX specifics	October 29, 2020	Completed
Program can now identify faces	November 5, 2020	Completed
~~Webcam/Camera can transfer feed to our program~~ Use webcam instead of IOT camera	November 12, 2020	Skipped/Modified
Integrated the program so that it can recognize faces from a video feed from camera	November 23, 2020	Completed

UI/UX minimal or console based commands to let user add faces to known list or move faces from unknown to known list	December 1, 2020	Completed
Identify presence of unknown faces and notify user	December 3, 2020	Completed

List Version

1. Camera is able to capture faces that appear in front of it.
2. Recognize known faces and distinguish from unknown ones.
3. Collect and store the list of known faces.
4. Allow to add new faces into the existing known list.
5. Provide a notification to the owner when an unknown face appears. (Email)

Workflow and System Design -



Overview -

Our program will serve the list of requirements defined above. The program has a basic UI that can be used to control it. On running the program you are given a startup window where you can choose to set up your email settings, add your/friends pictures to the list of known faces etc. Once that is done you can run the program itself so that it can start monitoring. If it sees an unknown face in the feed it will send you an alert. We have set it up so that it sends you an alert every 30 seconds. This can be changed by changing the *timethreshold* value.

Accuracy and Performance -

Built using dlib's state-of-the-art face recognition built with deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark.

As we are using a prebuilt module for face recognition we achieve a high accuracy of 99.38% as shown above. Please take a look at Caveats/Issues on their repository to understand this accuracy further.

We are also using a tolerance level of 0.6 as we want to be able to end up with false positives instead of missing out on an actual alert however this can be changed if wanted.

The module we used for face recognition has been tested on a Raspberry Pi so we expect similar performance levels given that we do not use any substantial amount of extra resources/computation power. However we have not tried running this on a Raspberry Pi. Given that we have a standardized code it should work without modifications on any linux distribution but we haven't tried it.

Setup / Installation -

You can refer to our github page for this information as well.

Here is a simple installation guide that worked for us (It might change for you depending on what programs/libraries you are missing)

Make sure you go through preparation before moving on to Library/Module installation

Preparation

1. Make sure you have python v3.0+
2. Make sure you have cmake installed.
3. Make sure you have dlib installed

Depending on your operating system the instructions might vary

For python check out - https://www.python.org/downloads/

For cmake check out - https://cmake.org/install/

There are precompiled binaries for windows as well as mac/linux that you can download and run to install cmake.

- For pip one thing to keep in mind is your path should be properly setup so that pip is recognized if you face issues try doing this - python3 -m pip install or python -m pip install instead of doing pip install

For dlib installation do pip install dlib

Now that we have met all requirements we will download and install all libraries used in the program.

Libraries/Modules Installation

Use pip3 instead of pip if that's what you have setup for v3.0+

Install face_recognition library pip install face_recognition

Install cv2 pip install opencv-python

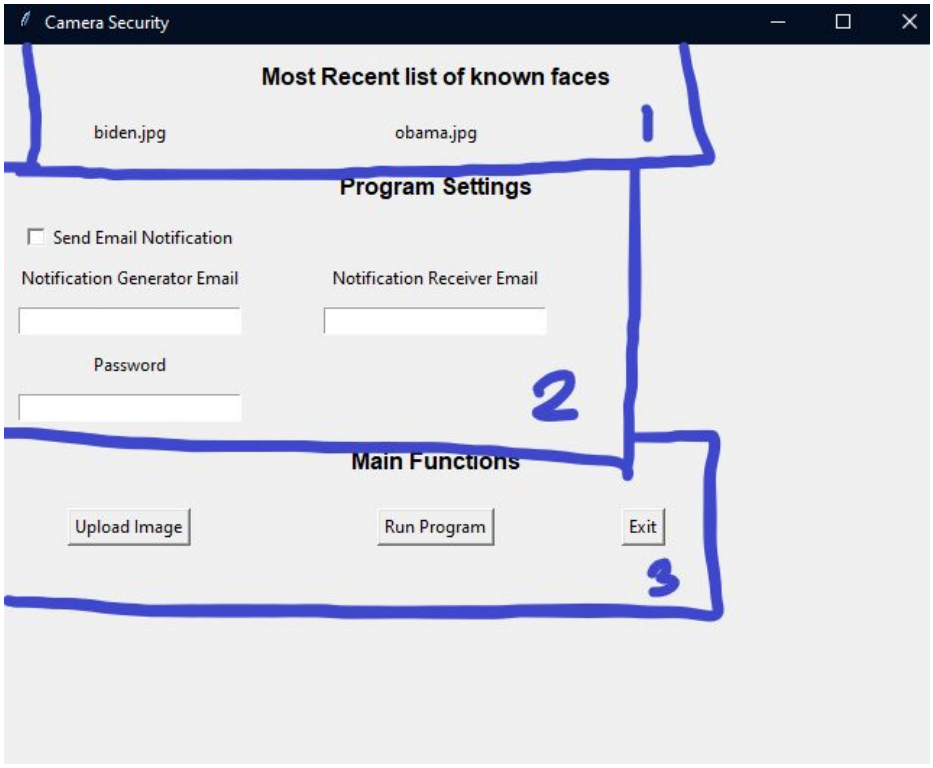
Install numpy pip install numpy

We have a few other modules imported in the program, these are and should be a part of your standard library and should come installed by default. If you face any errors after running the above errors please take a look at what library you are missing and refer to docs to install it using pip

Continued below

How to Use the Program -

On cloning and installation, when you run the program this is what you should see.



You need to add your picture to known faces or the program won't recognize you. Also we need to set up Email settings if we want to receive email notifications.

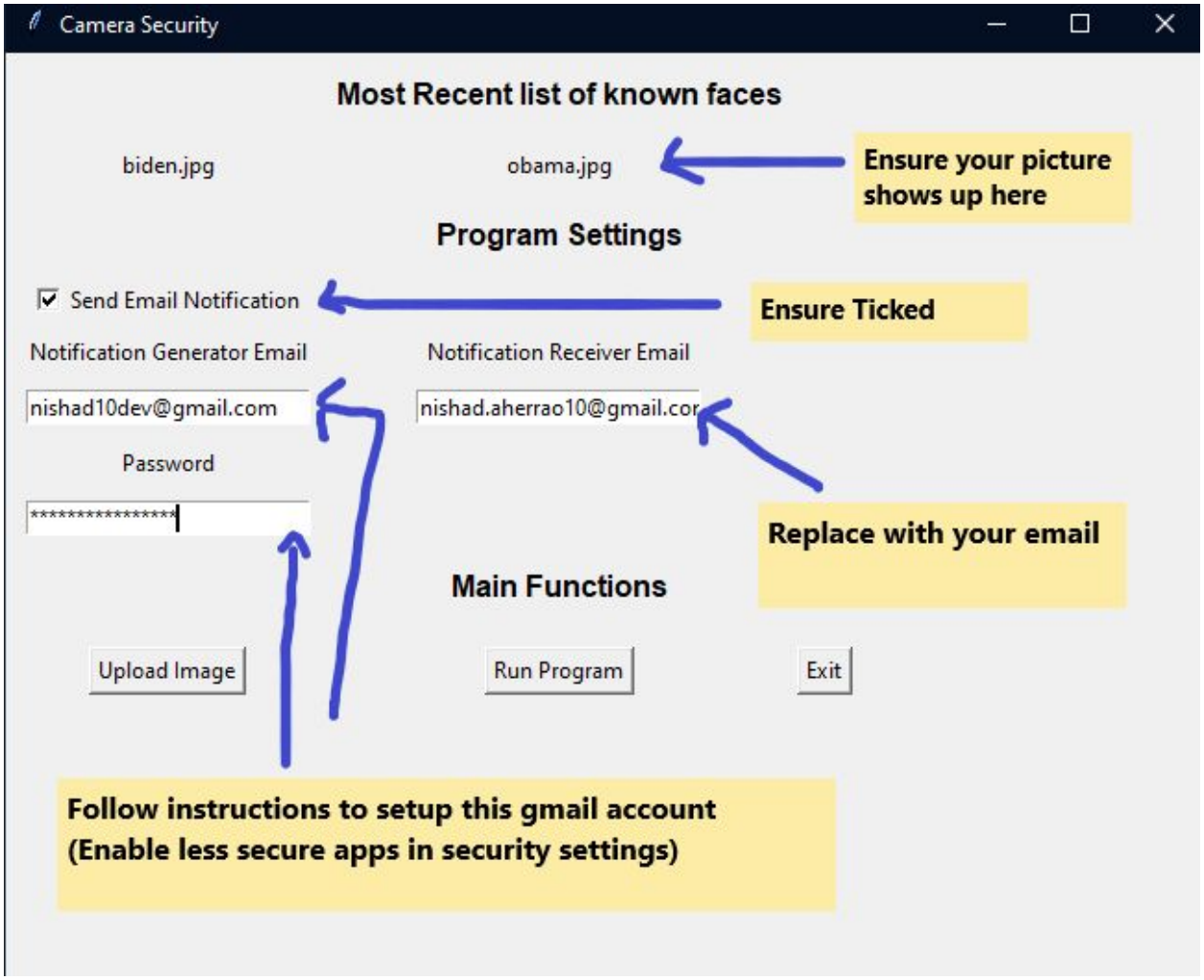
Get ready to upload your picture to list of known images and setup email

EMAIL IMPORTANT

To make sure you can send email properly the gmail account used to send email notifications need to have less secure apps enabled. To do this

Take a look at documentation on it

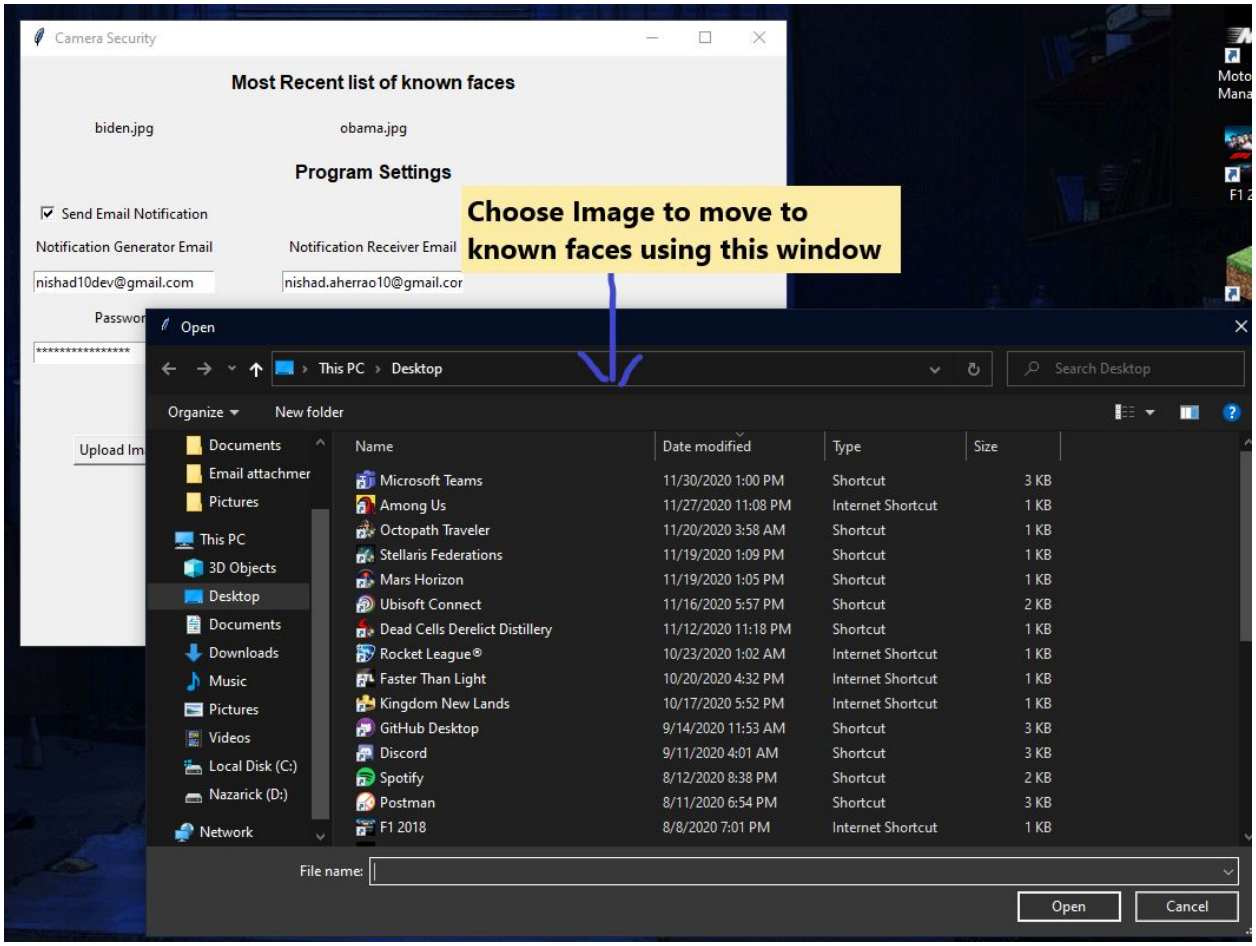
Enable less secure apps



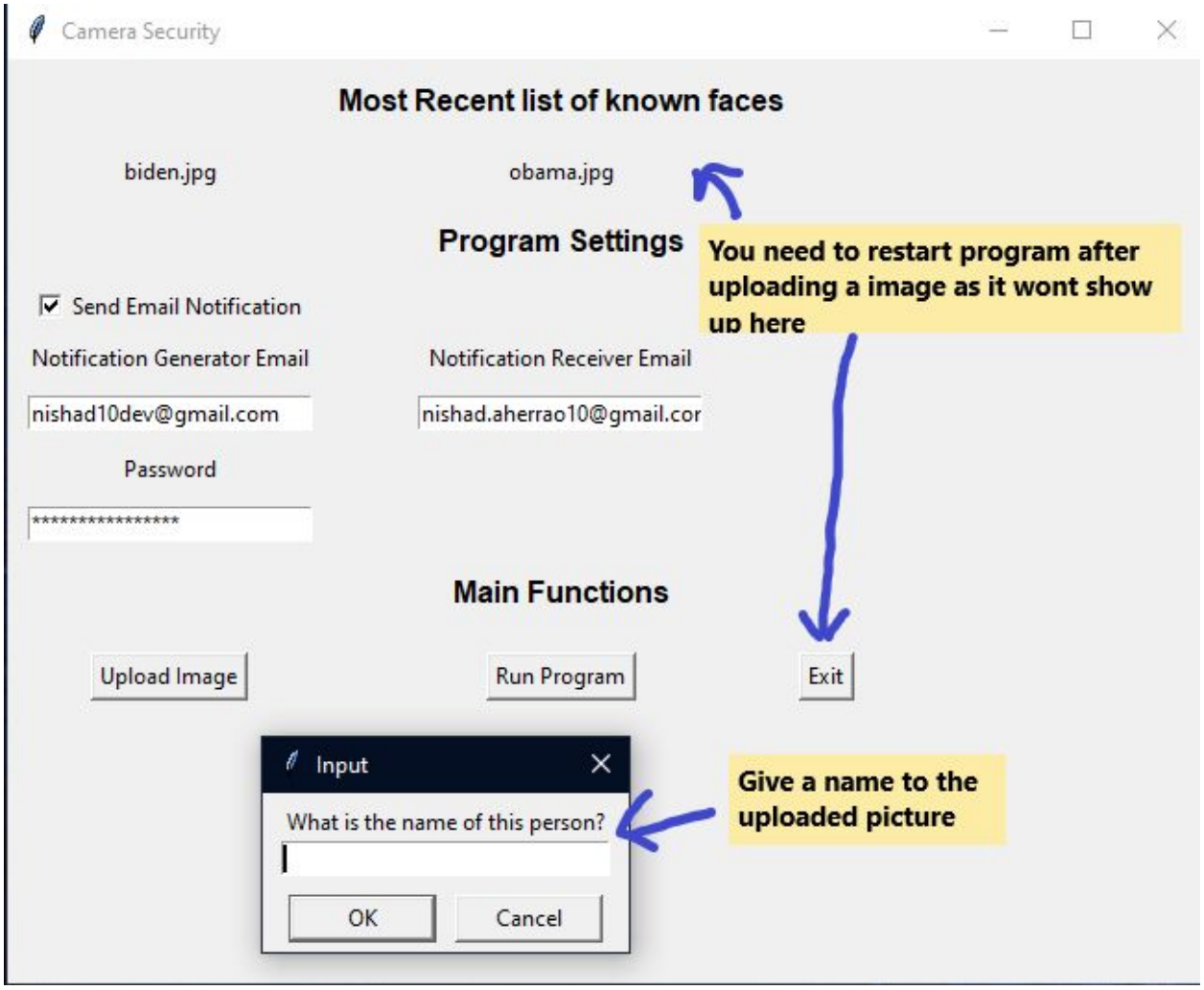
Continued Below

Upload your image to known faces

Click the upload image button and then choose a picture to upload. After that give that picture a name!



Continued Below



UPLOAD IMPORTANT

Right now the program does not take into account uploaded images immediately. You will have to quit using the Exit button and then restart the program so that the new uploaded image can be taken into consideration.

Running the program

Now that we have everything setup we can click `Run Program` button to run the actual program. The threshold right now is setup to be 30 seconds so if there is an unknown face on screen you should get an email every 30 seconds. Make sure you change this in the program before running to avoid spam. The picture should be recognized with your name or else a Unknown name will be shown below the image.

Distinguishing the FaceRecognition Library from our program -

The face recognition library that we used helps us recognize faces in a picture and helps us label them based on a list of known pictures. This is the only part of the program that the library helps us do. We have used this to integrate it with a program that can do much more.

What we built:

1. A UI to control the program easily.
2. Setup upload options in UI so you do not have to modify code or copy paste images to code repository.
3. Use the library/module on a live video feed.
4. Process the video feed, by splitting it into frames (images) and process every other frame for faces.

5. Setup email notifications, so that the program can notify user when an unrecognized face is in front of the camera for more than a timethreshold (30 seconds).
6. Setup Email options in UI so that you don't have to change any code and can change/setup with any email you want.

Technology/ Solutions Stack Details -

In order to implement our version of face recognition environment we have used the existing sources that are available on GitHub.We used the following repository with implementation of our own customized features and functions.

Python - https://github.com/ageitgey/face_recognition

We are using this built in standard module in python to help us send email notifications.

https://docs.python.org/3/library/email.html#module-email to send the user any notifications.

For our UI we are using another standard module from python called tkinter.

https://docs.python.org/3/library/tkinter.html

As for processing the video and frames and displaying a modified version of the video with names under faces we are using OpenCV.

https://pypi.org/project/opencv-python/