MINI PROJECT

(2020-2021)

FACE RECOGNITION ATTENDANCE MANAGEMENT SYSTEM

FINAL REPORT

Institute of Engineering & Technology

Submitted by-

Suraj Garg	**181500735**
Dharmendra Sharma	**181500215**
Astik Singh	**181500158**
Adarsh Kumar Dixit	**181500038**

Supervised by :-

Mr. Vinay Agrawal
(Assistant Professor)

Table of Contents

Acknowledgement

Presentation, inspiration and motivation have always played a key role in success of any venture.

I pay my deep sense of gratitude to **Mr. Sharad Gupta** Sir who encouraged all our team members to the highest peak and provided us the opportunity to work on this project.

I would also like to express my sincere thanks to our project mentor and Assistant Professor **Mr. Vinay Agarwal** Sir, who in spite of being extraordinary busy with their duties, took time to give invaluable advice and guidance throughout the development of this project. In addition, we are immensely obliged to our friends for their constant support, elevating inspiration and encouragement throughout this project.

Last, but not the least, our parents are also our inspiration. So, with due regards, we express our gratitude to them also.

Abstract

Face is the representation of one’s identity. Hence, we have proposed an automated student attendance system based on face recognition. Face recognition system is very useful in life applications especially in security control systems. The airport protection system uses face recognition to identify suspects and FBI (Federal Bureau of Investigation) uses face recognition for criminal investigations. In our proposed approach, firstly, video framing is performed by activating the camera through a user friendly interface. As we all know in this today’s world where technology is changing really quick. Everyone is switching from desktops to mobile devices. Those days are gone when people used to send text messages and has to pay for it; also there is not much time to waste to call via PC’s video calling software’s, as no one can carry PC’s and laptops with themselves everywhere. Nowadays everyone gets switched to mobile. People are engaging more in mobile applications rather than using PC’s software.

So, keeping all this in mind, we decided that user need not to waste his precious time in downloading or keep waiting for others as what happens in desktop’s software. He or she can get in touch with it using this application from anywhere. So the idea behind this is simple, it’s just to connect people at once.

Introduction

The main objective of this project is to develop face recognition based automated student attendance system. In order to achieve better performance, the test images and training images of this proposed approach are limited to frontal and upright facial images that consist of a single face only. The test images and training images have to be captured by using the same device to ensure no quality difference. In addition, the students have to register in the database to be recognized. The enrolment can be done on the spot through the user-friendly interface.

We made a student's facial attendance managing application. As we all know that there are multiple attendance applications available in market today. So what is different in our application as our name says Facial Attendance basically our main idea behind making this application is just to provide user a hassle free experience. No adds, No privacy concern. We just want to provide user a friendly environment.

This awesome facial managing application will help users to simplifying their needs.

Problem Statement :-

Traditional student attendance marking technique is often facing a lot of trouble. The face recognition student attendance system emphasizes its simplicity by eliminating classical student attendance marking technique such as calling student names or checking respective identification cards. There are not only disturbing the teaching process but also causes distraction for students during exam sessions. Apart from calling names, attendance sheet is passed around the classroom during the lecture sessions. The lecture class especially the class with a large number of students might find it difficult to have the attendance sheet being passed around the class. Thus, face recognition student attendance system is proposed in order to replace the manual signing of the presence of students which are burdensome and causes students get distracted in order to sign for their attendance. Furthermore, the face recognition based automated student attendance system able to overcome the problem of fraudulent approach and lecturer's does not have to count the number of students several times to ensure the presence of the students.

Hence, there is a need to develop a real time operating student attendance system which means the identification process must be done within defined time constraints to prevent omission. The extracted features from facial images which represent the identity of the students have to be consistent towards a change in background, illumination, pose and expression. High accuracy and fast computation time will be the evaluation points of the performance.

Aims and Objective :-

The objective of this project is to develop face recognition based automated student attendance system. Expected achievements in order to fulfil the objectives are:-

- To detect the face segment from the video frame.
- To extract the useful features from the face detected.
- To classify the features in order to recognize the face detected.
- To record the attendance of the identified student.

A facial recognition system is a technology capable of matching a human face from a digital image or a video frame against a database of faces. Researchers are currently developing multiple methods in which facial recognition systems work. The most advanced face recognition method, which is also employed to authenticate users through ID verification services, works by pinpointing and measuring facial features from a given image.

While initially a form of computer application, facial recognition systems have seen wider uses in recent times on smartphones and in other forms of technology, such as robotics. Because computerized facial recognition involves the measurement of a human's physiological characteristics facial recognition systems are categorised as biometrics. Although the accuracy of facial recognition systems as a biometric technology is lower than iris recognition and fingerprint recognition, it is widely adopted due to its contactless and non-invasive process. Facial recognition systems have been deployed in advanced human-computer interaction, video surveillance and automatic indexing of images.

Student Attendance System:-

People mentioned disadvantages of RFID (Radio Frequency Identification) card system, fingerprint system and iris recognition system. RFID card system is implemented due to its simplicity. However, the user tends to help their friends to check in as long as they have their friend's ID card. The fingerprint system is indeed effective but not efficient because it takes time for the verification process so the user has to line up and perform the verification one by one. However for face recognition, the human face is always exposed and contain less information compared to iris. Iris recognition system which contains more detail might invade the privacy of the user. Voice recognition is available, but it is less accurate compared to other methods. Hence, face recognition system is suggested to be implemented in the student attendance system.

Face Detection :-

Difference between face detection and face recognition are often misunderstood. Face detection is to determine only the face segment or face region from image, whereas face recognition is to identify the owner of the facial image. These factors consist of background, illumination, pose, expression, occlusion, rotation, scaling and translation.

For the application of face recognition, detection of face is very important and the first step. After detecting face the face recognition algorithm can only be functional. Face detection itself involves some complexities for example surroundings, postures, enlightenment etc.

There are some existing methodologies for detection of face. Some of them are skin color based, characteristic or feature based (feature like mouth, nose and eyes) and neural network based. Among the above techniques, the skin based procedure is well thought-out as simplest one. The approach premeditated and applied in this thesis is the skin color based face detection method. The algorithm is pretty dynamic as numerous people face can be detected at one time from an image containing many people. In this project YCbCrcolor model is used to detect the skin of human being.

Segmentation based on color :-

The Segmentation can be defined as the conception of subdividing a given image into its constituent region. Segmentation based on the color of skin is picking up its supremacy in current time. Skin based segmentation is being studied for the reason that of its dynamic research in content- based picture illustration. In the case of face detection, segmentation is used to find locate the face boundary of face region in an image. Once the face region is found, we can apply various processing like image editing, various coding, and image indexing and client intuitiveness intention. Moreover, face detection is the first step required for the purpose of recognition of face and its expression using various processes. Color of Skin of an individual depends on various biochemical components like the melanin content, pigmentation of skin and much more. The skin color is belongs to certain range in the total color space. Consideration should be taken into account that the skin should not be abnormal. Used algorithm in this project takes the benefit of face color association to limit the face search to areas of an input image that have at least the accurate color components. There are many existing algorithm for segmentation but used algorithm is the simplest one.

A. RGB Color Space :-

Red, green, blue are the three color component that constitute the RGB color space. These red, green, and blue colors combine in a definite proportion to yield a different color. RGB color model can be represented by a 3-dimentional cube with the three colors at the corner of the cube and in each axis as shown in Figure. At the origin of this cube black color is present. White color is present at the opposite corner of the 3-dimesional cube i.e., at the opposite diagonal of the cube. Graycolor scale is represented by the line from black color at the origin to the white color at opposite corner. Red is (255, 0, 0) when we consider a 24-bit color graphics system with a color bit of 8-bit per color channel. When we consider the 3-dimensional colorcube, red is at (1, 0, 0). This model simplifies the design of the pc graphics frameworks yet is not perfect for all 13 type of application applications. These three color component are very closely related to each other, which makes it hard to implement some of the algorithm for image processing.

B. HSV Color Model :-

There are some problems which are associated with the RGB color space model. One important issue connected with RGB is that it doesn’t consider the effect of light on the color of skin, which generally cause some wrong in turn. This problem can we solved by using HSV model for skin color. Here H stand for hue which is depth of color, S stand for saturation or the purity of color and V stand for the value of intensity of light or the brightness of color. The model is shown in figure. The red, blue and yellow color is represented by hue and which has the range from 0 to 3600 . While using HSV color space we don’t need the information of the proportion of blue and green color needed to produce a different color. Only thing needed is that by altering the hue we can get the desired color of our interest. The purity of the color is represented by the saturation value, which has the range of 0 to 100%. Lets take an example of getting pink color from that of the dark red color, which can be done by simply modifying the saturation value .value gives idea about the brightness of the given color and it gives the achromatic thought of the given color. The range of value extend from zero to hundred. ‘H’ and ‘S’ gives the required information about the color of skin. The pixel value of skin color should satisfy the below condition in order to consider that as skin.

C. YCbCrColor Model :-

This color model has more advantage than the upper two model as discussed above and it uses the chrominance value to extract the skin color region of an given image. ‘YCbCr’ or ‘Y’CbCr’ colorspace are generally used in the digital image processing. Y is the luminance, luma component is represented by Y’ while Cb and Cr are the blue difference and red difference of the chroma component respectively. YCbCr is not a real color space. This is just another way of encoding the RGB color space. The YCbCr values can only be obtained only if the original RGB information of the image is available.

D. OpenCV (Open Source Computer Vision Library)

Open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV

makes it easy for businesses to utilize and modify the code.The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many start-ups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDAand OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

OpenCV Python -

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.
This OpenCV tutorial will help you learn the Image-processing from Basics to Advance, like operations on Images, Videos using a huge set of Opencv-programs and projects.

Reading an image in OpenCV using Python

Using OpenCV
Read an image
Use the function(cv2.imread()) to read an image. The image should be in the working

Real Time Image Processing Using Python & OpenCV

OpenCV is a library of cross platform programming functions aimed at real time Computer Vision. IT was designed for computational efficiency and with a strong focus on real-time applications, video and image processing. Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code

readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C.
The language provides constructs intended to enable clear programs on both a small and large scale. The growing demand of integrating OpenCV with python promises clear cut solutions to image processing problems. Since the tools are open source, researchers can exploit the freedom and possibilities of expansion. Wide spread applications in the field of robotics underlines the scope of OpenCV for image processing. Image classification, segmentation, feature extraction etc are made with suitable libraries and it can be invoked through many of the programming languages.
Here in Raspberry Pi Opencv is invoked through Python. So we need the updated version of both Python and Opencv.
Click here to get the detailed instructions to install Opencv and Python in Raspberry Pi 2.
For an example let us move on to a python program that runs a real time video from webcam using OpenCv interface. After the installation of OpenCv a new folder will be created in the home. In this folder there are some python sample codes based on image processing.

OpenCV is the most popular library for computer vision. Originally written in C/C++, it now provides bindings for Python.

OpenCV uses machine learning algorithms to search for faces within a picture. Because faces are so complicated, there isn't one simple test that will tell you if it found a face or not. Instead, there are thousands of small patterns and features that must be matched. The algorithms break the task of identifying the face into thousands of smaller, bite-sized tasks, each of which is easy to solve. These tasks are also called classifiers.

For something like a face, you might have 6,000 or more classifiers, all of which must match for a face to be detected (within error limits, of course). But therein lies the problem: for face detection, the algorithm starts at the top left of a picture and moves down across small blocks of data, looking at each block, constantly asking, “Is this a face? … Is this a face? … Is this a face?” Since there are 6,000 or more tests per block, you might have millions of calculations to do, which will grind your computer to a halt.

To get around this, OpenCV uses cascades. What’s a cascade? The best answer can be found in the dictionary: “a waterfall or series of waterfalls.”

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 algorithm may have 30 to 50 of these stages or cascades, and it will only detect a face if all stages pass.

The advantage is that the majority of the picture will return a negative during the first few stages, which means the algorithm won’t waste time testing all 6,000 features on it. Instead of taking hours, face detection can now be done in real time.

Cascades in Practice –

Though the theory may sound complicated, in practice it is quite easy. The cascades themselves are just a bunch of XML files that contain OpenCV data used to detect objects. You initialize your code with the cascade you want, and then it does the work for you.

Since face detection is such a common case, OpenCV comes with a number of built-in cascades for detecting everything from faces to eyes to hands to legs. There are even cascades for non-human things. For example, if you run a banana shop and want to track people stealing bananas, this guy has built one for that!

Installing OpenCV –

First, you need to find the correct setup file for your operating system.

I found that installing OpenCV was the hardest part of the task. If you get strange unexplainable errors, it could be due to library clashes, 32/64 bit differences, and so on. I found it easiest to just use a Linux virtual machine and install OpenCV from scratch.

Once you have completed the installation, you can test whether or not it works by firing up a Python session and typing:

```
>>>
>>> import cv2
>>>
```

If you don't get any errors, you can move on to the next part.

Understanding the Code

Let's break down the actual code, which you can download from the repo. Grab the **face_detect.py** script, the **abba.png** pic, and the **haarcascade_frontalface_default.xml**.

```
# Get user supplied values
imagePath = sys.argv[1]
cascPath = sys.argv[2]
```

You first pass in the image and cascade names as command-line arguments. We'll use the ABBA image as well as the default cascade for detecting faces provided by OpenCV.

```
# Create the haar cascade
faceCascade = cv2.CascadeClassifier(cascPath)
```

Now we create the cascade and initialize it with our face cascade. This loads the face cascade into memory so it's ready for use. Remember, the cascade is just an XML file that contains the data to detect faces.

```
# Read the image
image = cv2.imread(imagePath)
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

Here we read the image and convert it to grayscale. Many operations in OpenCV are done in grayscale.

```
# Detect faces in the image
faces = faceCascade.detectMultiScale(
    gray,
    scaleFactor=1.1,
    minNeighbors=5,
    minSize=(30, 30),
    flags = cv2.cv.CV_HAAR_SCALE_IMAGE
)
```

This function detects the actual face and is the key part of our code, so let's go over the options:

1. The detectMultiScale function is a general function that detects objects. Since we are calling it on the face cascade, that's what it detects.
2. The first option is the grayscale image.
3. The second is the scaleFactor. Since some faces may be closer to the camera, they would appear bigger than the faces in the back. The scale factor compensates for this.

4. The detection algorithm uses a moving window to detect objects. minNeighbors defines how many objects are detected near the current one before it declares the face found. minSize, meanwhile, gives the size of each window.

Note: I took commonly used values for these fields. In real life, you would experiment with different values for the window size, scale factor, and so on until you found one that works best for you.

The function returns a list of rectangles in which it believes it found a face. Next, we will loop over where it thinks it found something.

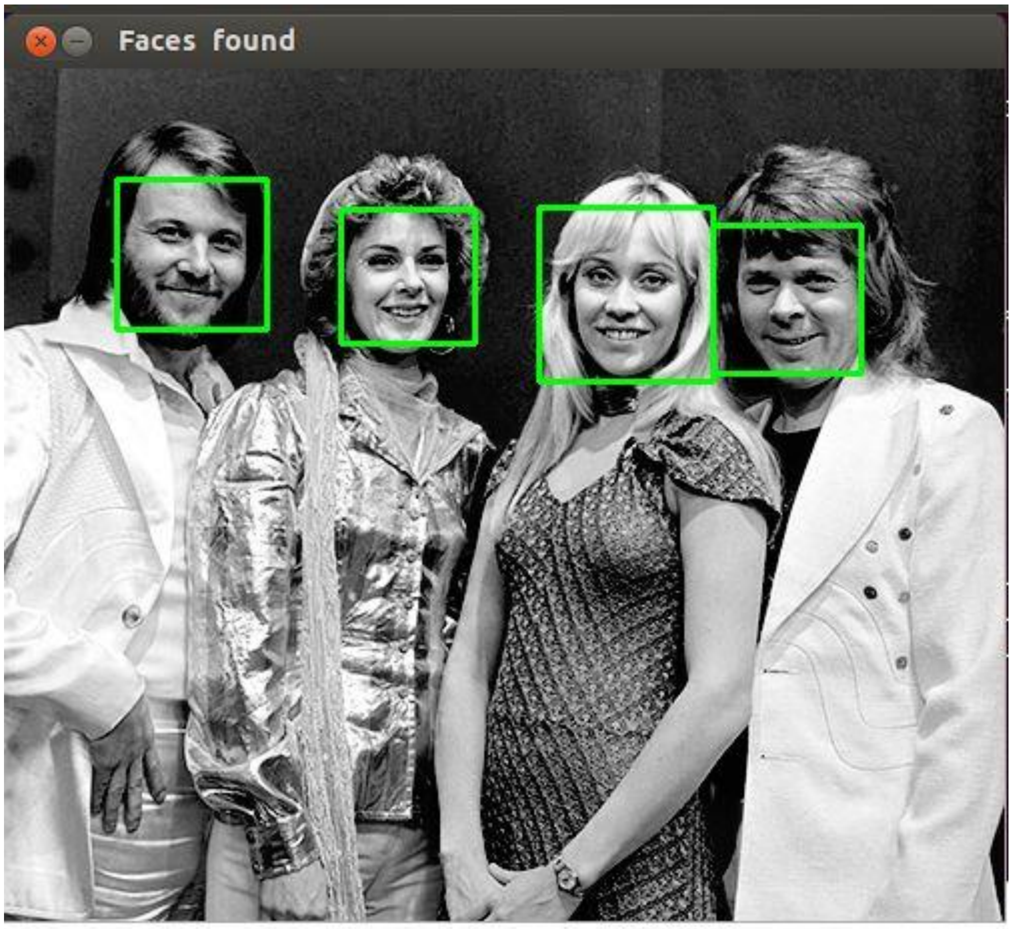
```
print "Found {0} faces!".format(len(faces))

# Draw a rectangle around the faces
for (x, y, w, h) in faces:
    cv2.rectangle(image, (x, y), (x+w, y+h), (0, 255, 0), 2)
```

This function returns 4 values: the x and y location of the rectangle, and the rectangle's width and height (w , h).We use these values to draw a rectangle using the built-in rectangle() function.

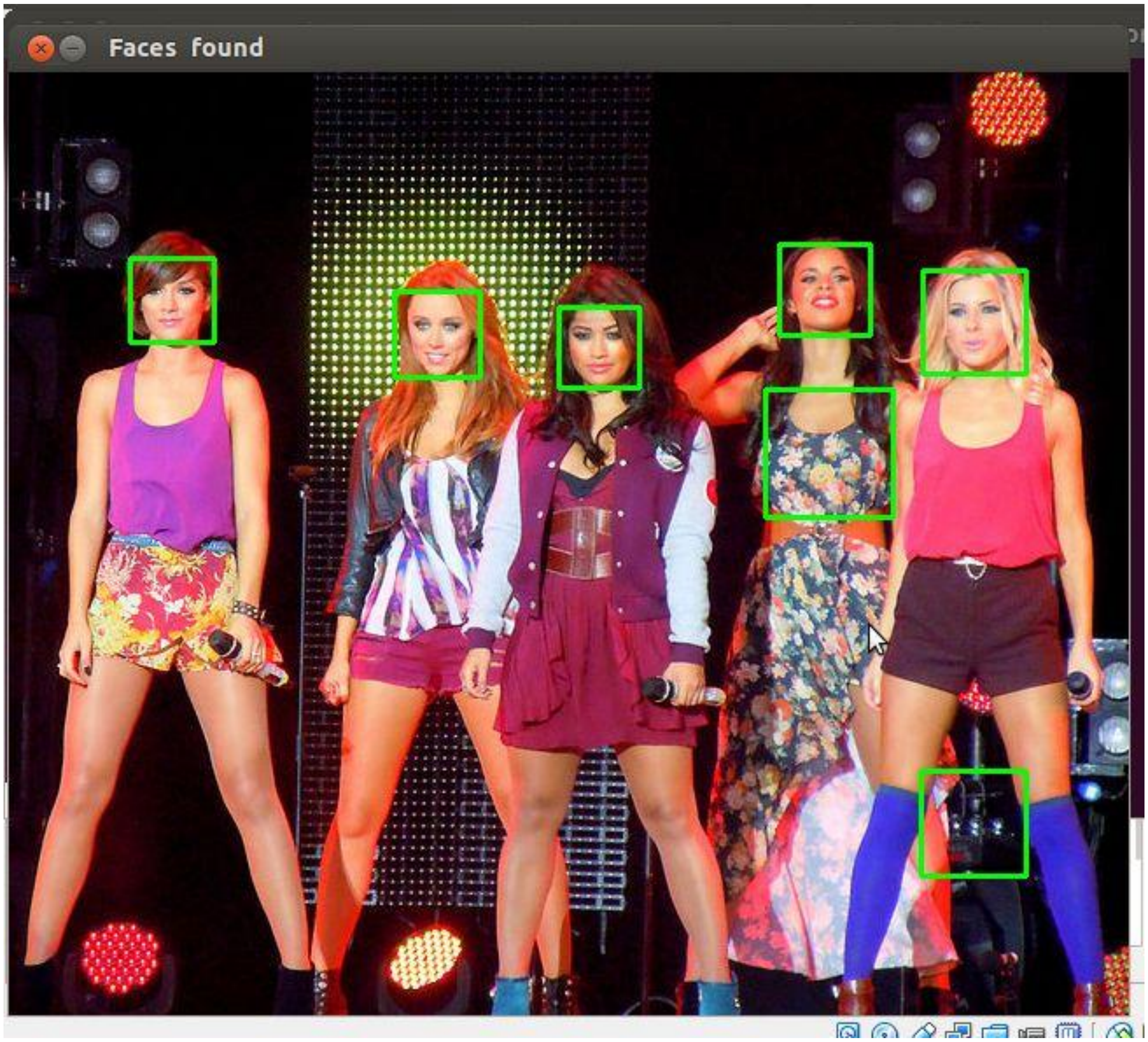
```
cv2.imshow("Faces found", image)
cv2.waitKey(0)
```

In the end, we display the image and wait for the user to press a key.

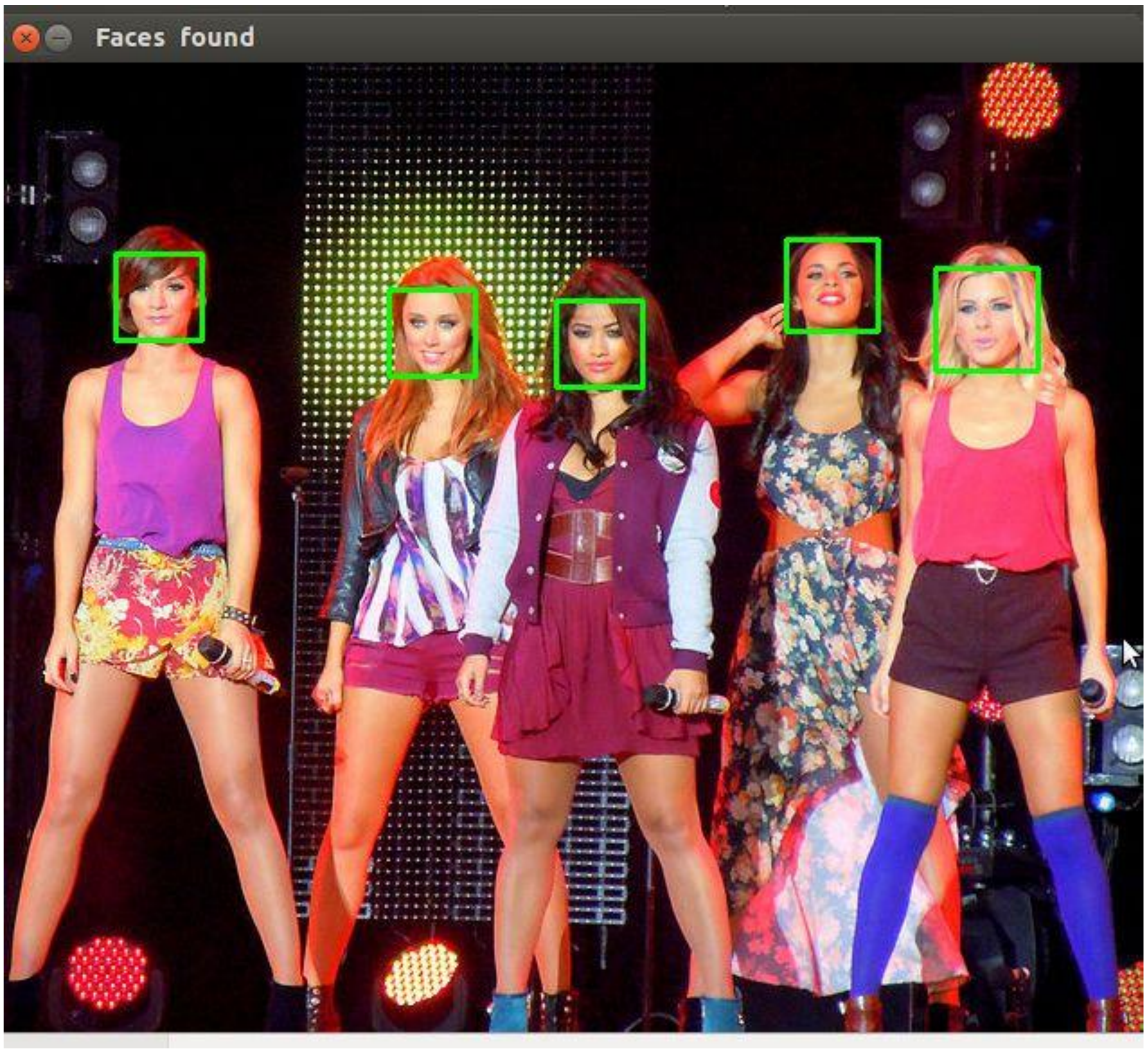
```
$ python face_detect.py abba.png haarcascade_frontalface_default.xml
```



That worked. How about another photo:



That … is not a face. Let’s try again. I changed the parameters and found that setting the scaleFactor to 1.2 got rid of the wrong face.



What Happened?

Well, the first photo was taken fairly close up with a high quality camera. The second one seems to have been taken from afar and possibly with a mobile phone. This is why the scaleFactor had to be modified. As I said, you’ll have to set up the algorithm on a case-by-case basis to avoid false positives.

Be warned though that since this is based on machine learning, the results will never be 100% accurate. You will get good enough results in most cases, but occasionally the algorithm will identify incorrect objects as faces.

The final code can be found here.

Extending to a Webcam –

What if you want to use a webcam? OpenCV grabs each frame from the webcam, and you can then detect faces by processing each frame. You will need a powerful computer, but my five-year-old laptop seems to cope fine, as long as I don’t dance around too much.

Full Stack Python

MySQL:-

MySQL (/ˌmaɪˌɛsˌkjuːˈɛl/)[5] is an open-source relational database management system (RDBMS).[5][6] Its name is a combination of "My", the name of co-founder Michael Widenius's daughter,[7] and "SQL", the abbreviation for Structured Query Language.
A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. **MySQL** was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation).[8] In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.[9]

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for *Linux, Apache, MySQL, Perl/PHP/Python*. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter, and YouTube.

Overview –

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer.[16] MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, ArcaOS, eComStation, i5/OS, IRIX, Linux, macOS, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists.[17]

The MySQL server software itself and the client libraries use dual-licensing distribution. They are offered under GPL version 2, or a proprietary license.[18]

Support can be obtained from the official manual.[19] Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organisations exist to provide support and services.

MySQL has received positive reviews, and reviewers noticed it "performs extremely well in the average case" and that the "developer interfaces are there, and the documentation (not to mention feedback in the real world via Web sites and the like) is very, very good".[20] It has also been tested to be a "fast, stable and true multi-user, multi-threaded SQL database server.

Features –

MySQL is offered under two different editions: the open source MySQL Community Server[76] and the proprietary Enterprise Server.[77] MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

Major features as available in MySQL 5.6:

- A broad subset of ANSI SQL 99, as well as extensions
- Cross-platform support
- Stored procedures, using a procedural language that closely adheres to SQL/PSM[78]
- Triggers
- Cursors
- Updatable views
- Online Data Definition Language (DDL) when using the InnoDB Storage Engine.
- Information schema
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.[79]
- A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines[80]
- SSL support
- Query caching

- Sub-SELECTs (i.e. nested SELECTs)
- Built-in replication support
  - Asynchronous replication: master-slave from one master to many slaves[81][82] or many masters to one slave[83]
  - Semi synchronous replication: Master to slave replication where the master waits on replication[84][85]
  - Synchronous replication: Multi-master replication is provided in MySQL Cluster.[86]
  - Virtual Synchronous: Self managed groups of MySQL servers with multi master support can be done using: Galera Cluster[87] or the built in Group Replication plugin[88]
- Full-text indexing and searching[b]
- Embedded database library
- Unicode support[a]
- Partitioned tables with pruning of partitions in optimizer
- Shared-nothing clustering through MySQL Cluster
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.[c]
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

The developers release minor updates of the MySQL Server approximately every two months. The sources can be obtained from MySQL's website or from MySQL's GitHub repository, both under the GPL license.

Face Recognition :-

The recognition of face of human is challenging in computer-human interaction. The face is our essential center of consideration in societal life playing a critical part in assigning identification and emotion of the person. We can perceive various appearances adapted all through our lifespan and distinguish faces initially even following quite a while of detachment. This expertise is very vigorous notwithstanding of substantial varieties in visual boost because of evolving condition, maturing and diversions, for example, facial hair, glasses or changes in haircut. Computational models of face acknowledgment are fascinating in light of the fact that they can contribute to hypothetical learning as well as to functional applications. PCs that identify and recognizes the face could be connected to a broad assortment of undertakings together with criminal recognizable proof, security framework, image and film handling, identity confirmation and human-PC interaction. Tragically, adding to a computational model of face recognition and acknowledgment is very troublesome in light of the fact that faces are perplexing, multidimensional and important visual stimuli. This project uses the principal component analysis (PCA) for face recognition. There are many other face recognition algorithm, but principal component analysis (PCA) based face recognition is the simplest one for face recognition.

<u>**Attendance Registering :-**</u>

Class attendance is very important aspects for the students studying in the colleges or schools. For an organization to be successful, it needs precise and quick method for recording the performance of the
individuals inside this organization. Attendance gives the data of the individual whether that particular person is physically present or absent. The traditional method of calling roll number or name of the student for marking attendance is time consuming or wastage of time during the class hour. In this project an automated student attendance system is made, which is based on face recognition. The recognized face of the individual is used for marking the attendance.

<u>**Advantages of facial recognition based attendance system –**</u>

<u>**Automated time tracking system -**</u>

Offices or workplaces or even just public places where the entry and exit times of employees or a person are strictly noted down will have a ready-made automated system to record the entry and exit time of each person for a given time. It won’t even need the person to stop and click a photo, the software’s are advanced enough to record the data from a continuous reel also. This means the flow won’t get hampered, or you won’t have to stop and smile or something like that. Just enter or exit the place effortlessly like you do everyday and boom! Your attendance will be recorded without any fuss!

Cost-effective -

Since the whole process will be done by a computer, it means the total attendance registration and calculation will be automated and done by the system itself, therefore, saving us the money which would have been otherwise spent on the labour cost to do that.

Increased Security -

Face recognition based attendance system won’t just calculate attendance but also note down the entry and exits of visitors in the place. At times when there is a situation where the identity and time of entry and exit of a specific person need to be noted, this system would become handy as it will easily show you when he/she came in and what are the places he/she went to a very precise level. All of this means, you will have a much higher security level in your workplace.

Time Saving -

The whole world is suffering from COVID19 and it is high time we must give heed to social distancing. Having a safe distance with others has become a necessity nowadays. Times like this can be problematic if you have manual attendance system, Having a Face recognition based attendance system will not only allow you to register the attendance of a person but also keep you at a safe distance from them as you can work remotely and still see who all are coming and going. This calls for the point that, this whole system is a much safer, time-saving, and faster method to record attendance.

Easy to manage -

Since the artificial intelligence based attendance system is fully automated, managing the records and keeping a track of day to day activities will become much easier than the manual system. Everything will be done by the system. Many software are programmed in such a way that it shows the exact time of how many hours or minutes a person worked on his/her desk in the day. All this is can be done on a very large scale. Just imagine, recording the activities of a large crowd of 200 people simultaneously without any fuss and recording it at the same time in an organized manner!. Such is the power of AI in face recognition.

Conclusion

The face detection and recognition algorithms were studied thoroughly taking number of the test from different varying condition images. For face detection combination of RGB and HSV model algorithm is used. For face recognition principal component analysis method is used. Attendance of the student are marked using the recognized face of every individual student and the data is stored in an attendance sheet. The attendance of every student marked automatically by recognizing their face with the face present in the data base.

REFERENCES

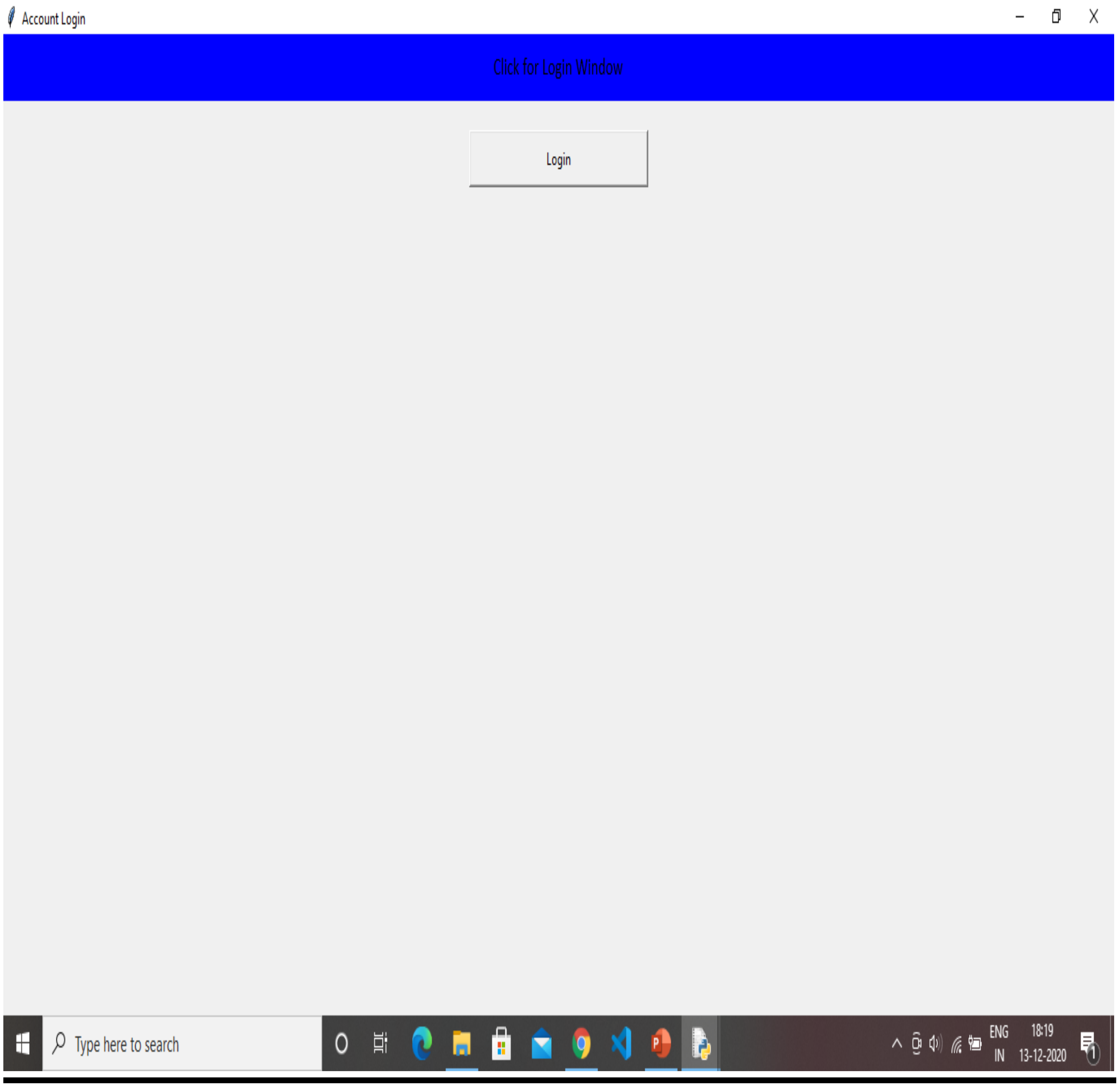
1.) https://www.wikipedia.org/

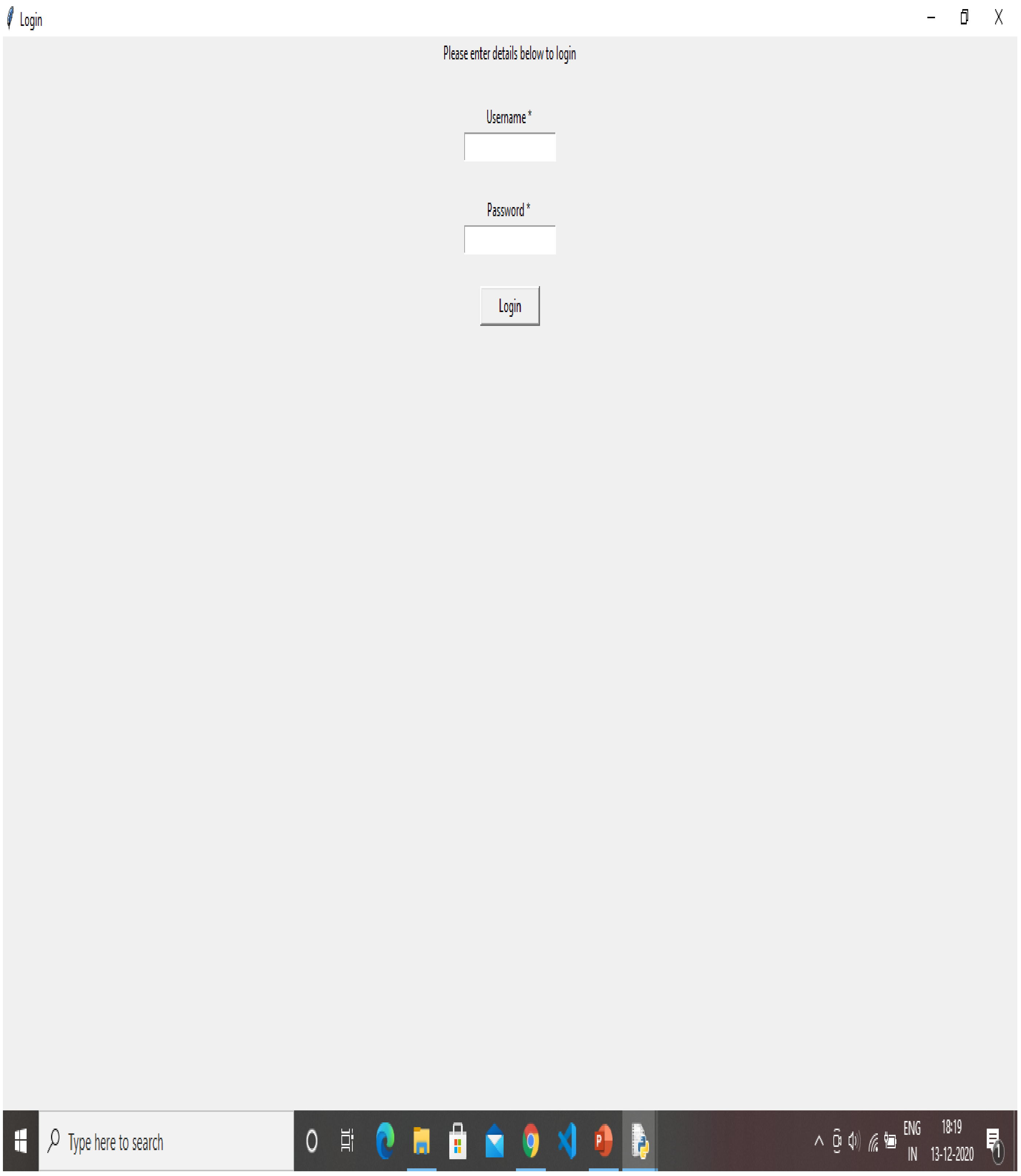
2.) https://www.coursera.org/learn/java-for-android/home/welcom

3.) http://myapp.buildfire.com

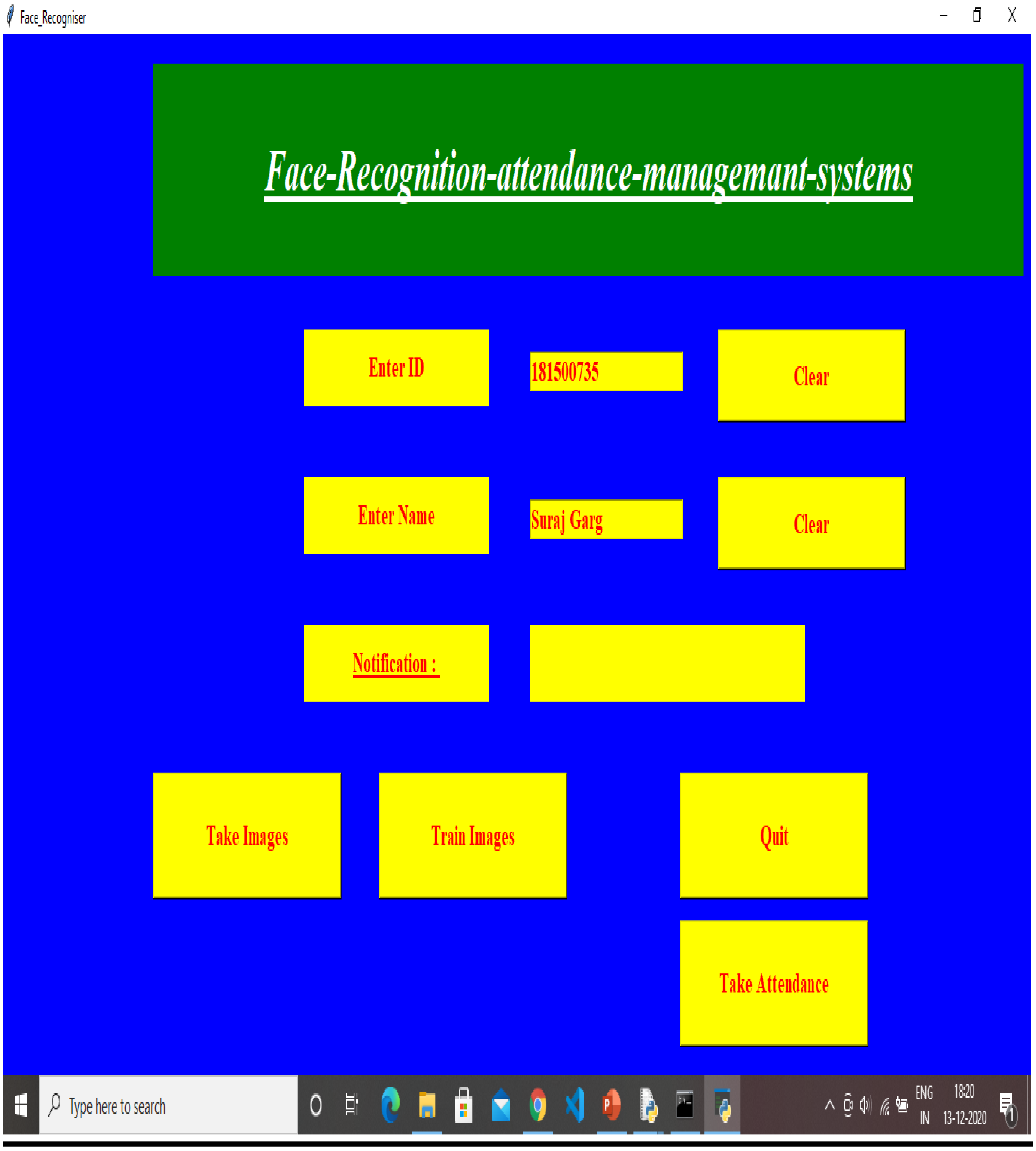
4.) https://androidstudio.googleblog.com/2020/11/android-studio-411-available.html

ScreenShots –





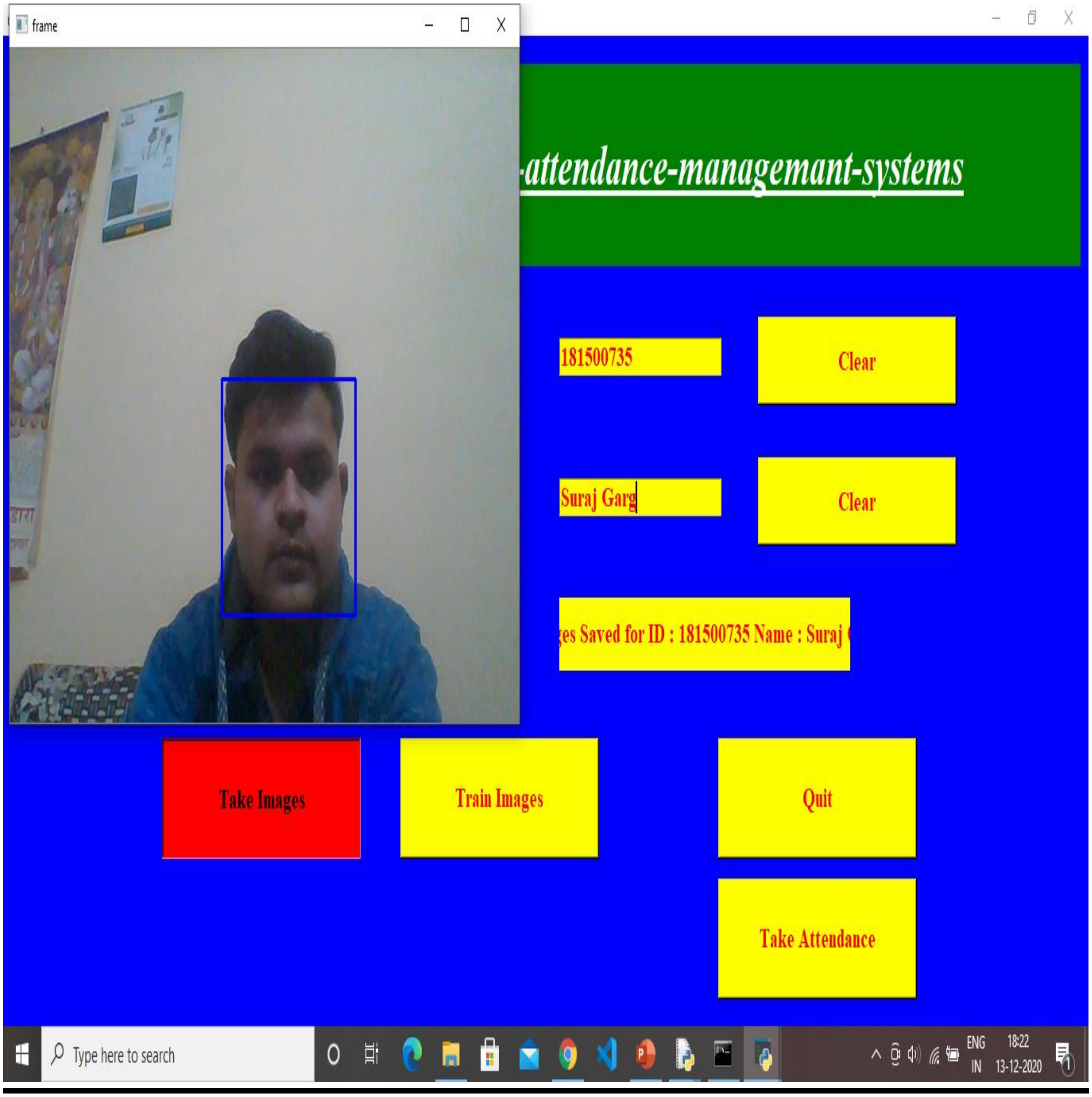
Login
Please enter details below to login
Username *
Password *
Login
Type here to search
ENG
IN
18:19
13-12-2020

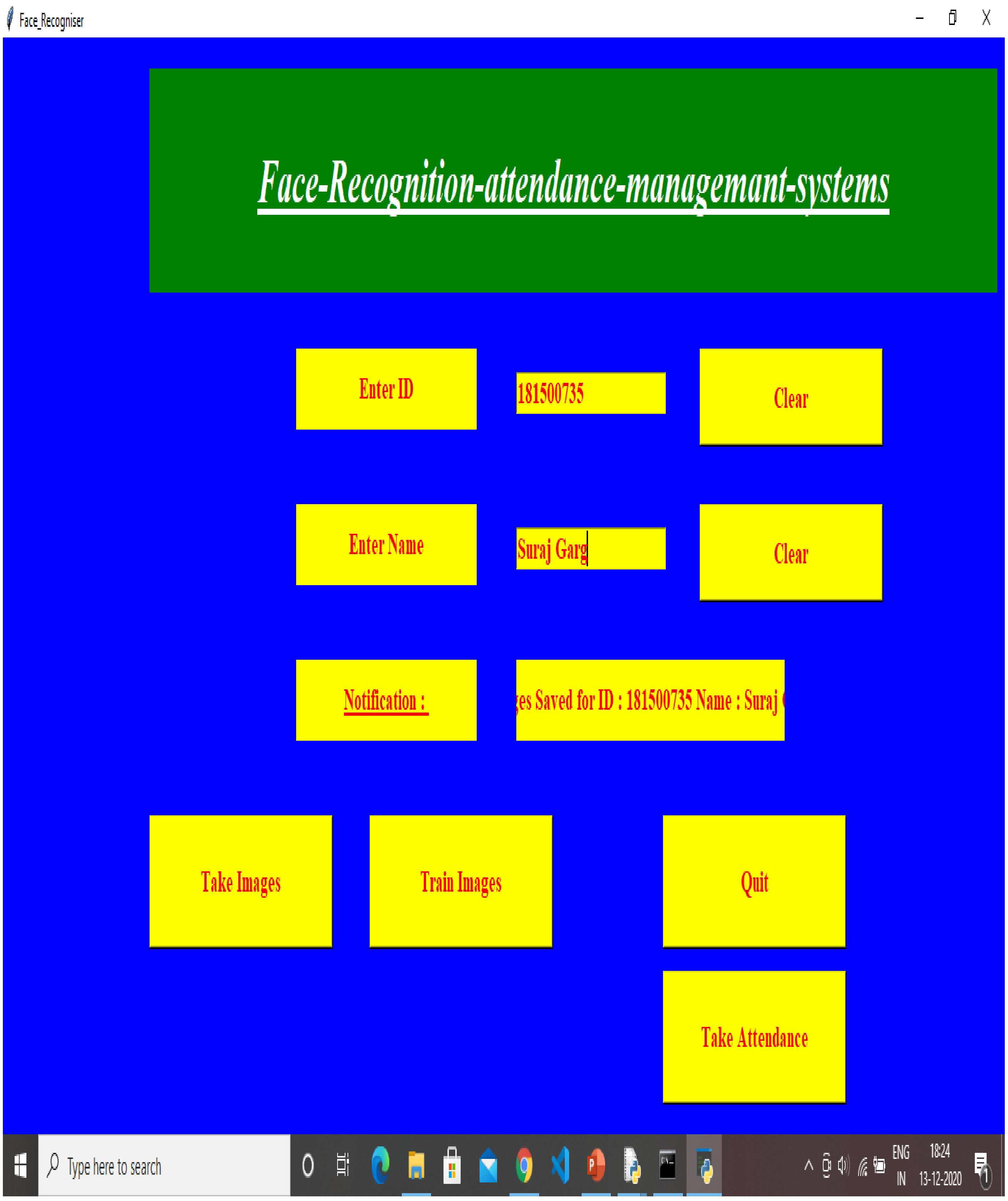


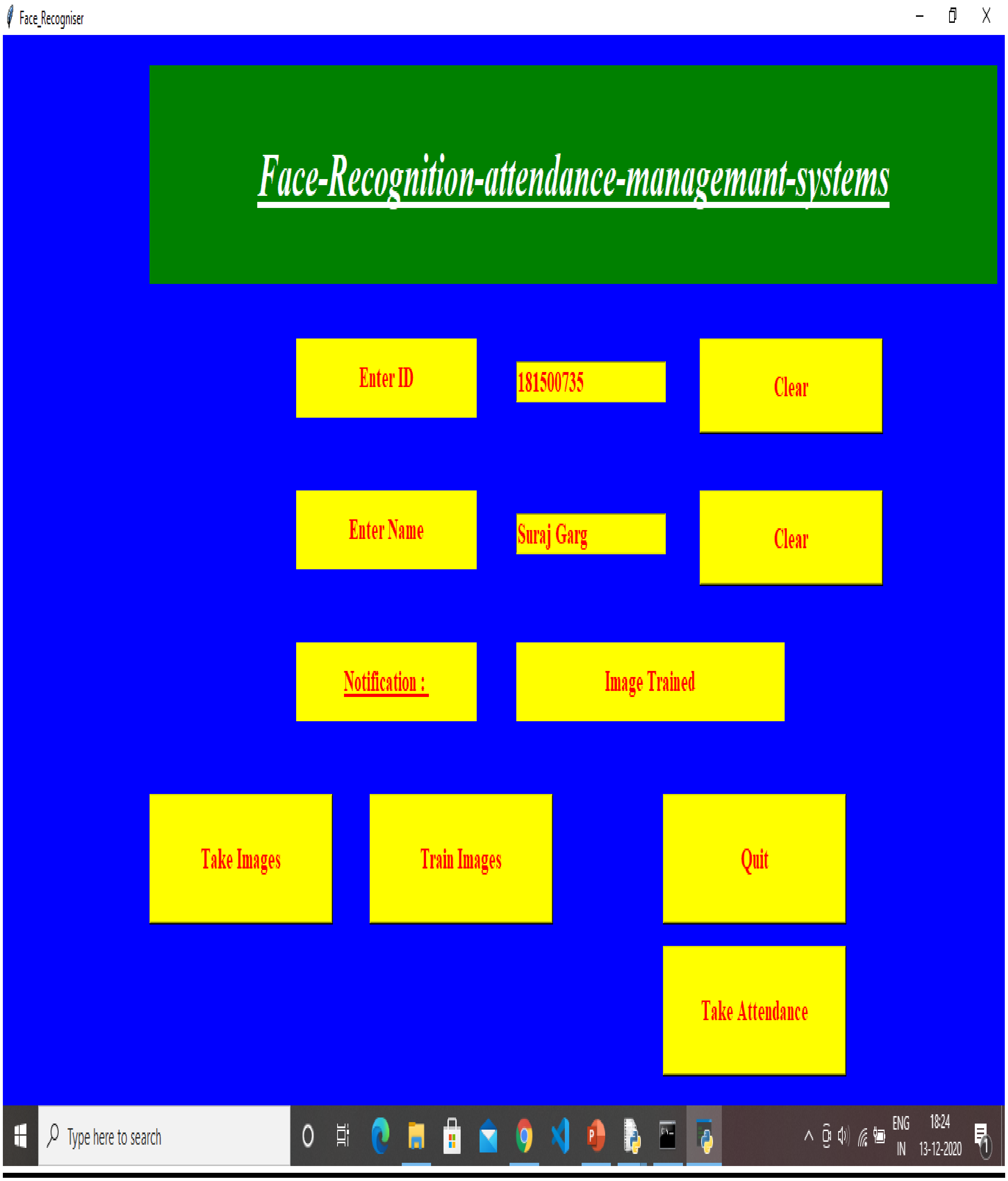
Face_Recognizer

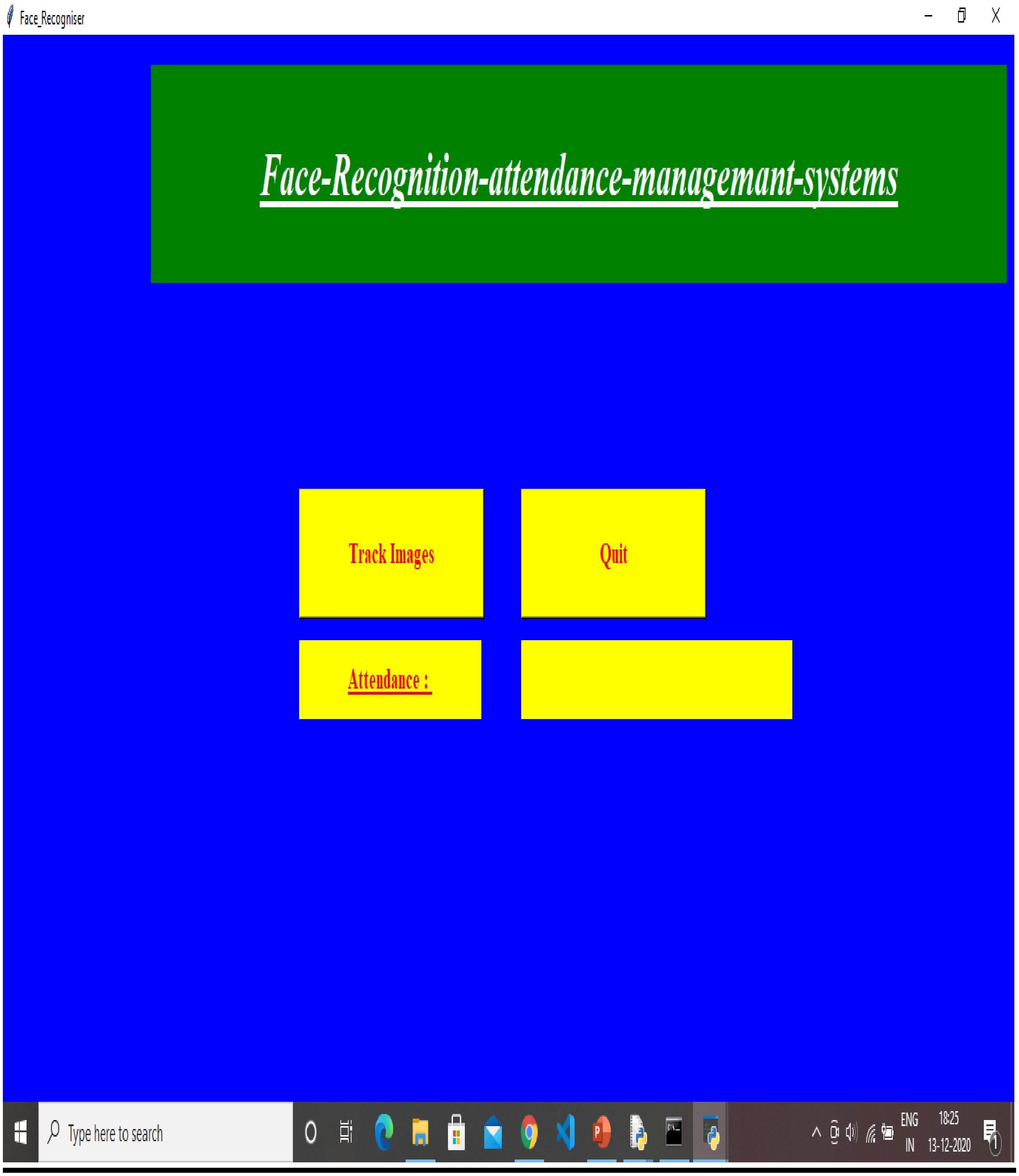
Face-Recognition-attendance-managemant-systems

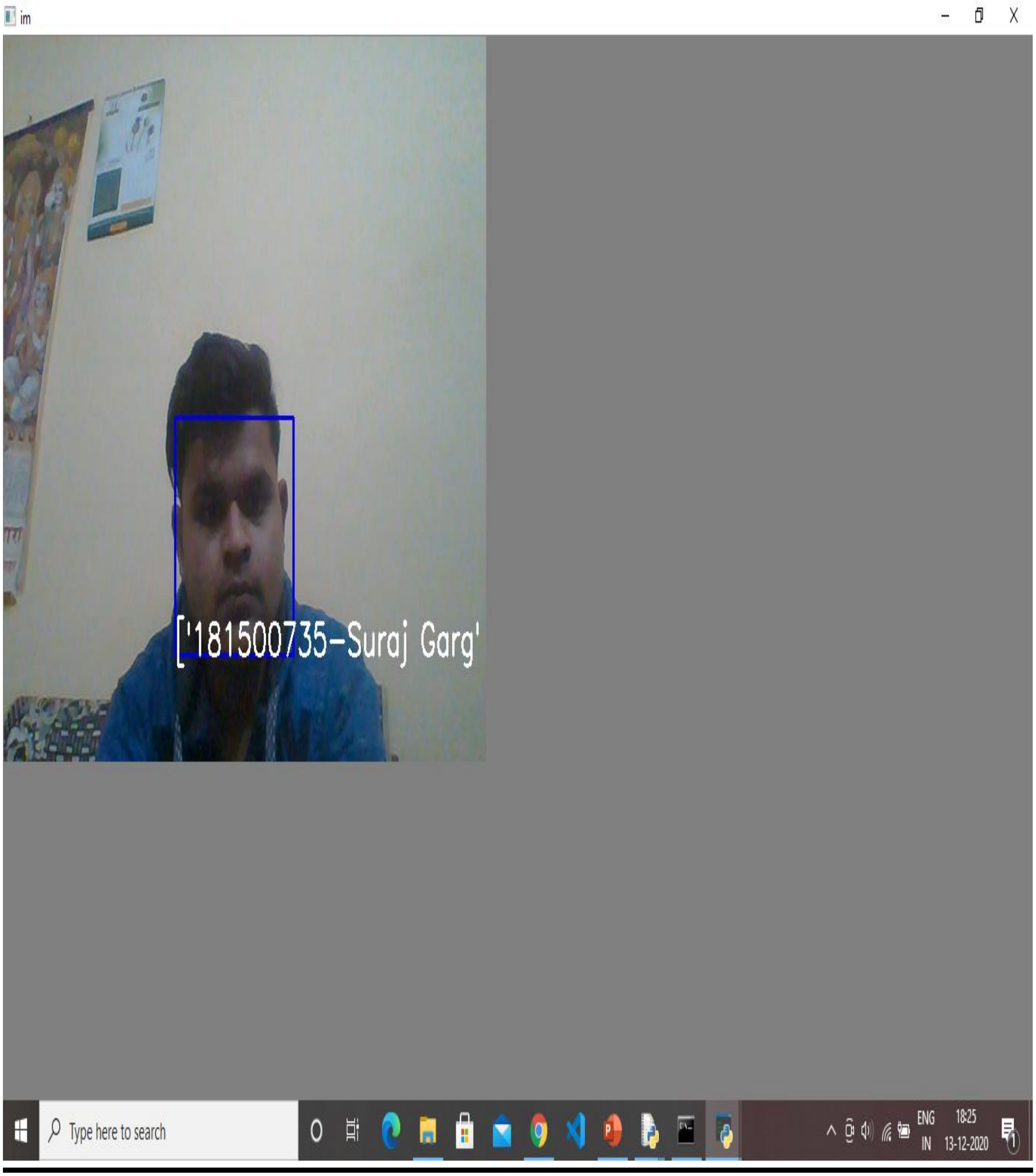
Enter ID	181500735	Clear
Enter Name	Suraj Garg	Clear
Notification :		


frame
-attendance-managemant-systems
181500735
Clear
Suraj Garg
Clear
es Saved for ID : 181500735 Name : Suraj
Take Images
Train Images
Quit
Take Attendance
Type here to search
ENG
IN
18:22
13-12-2020

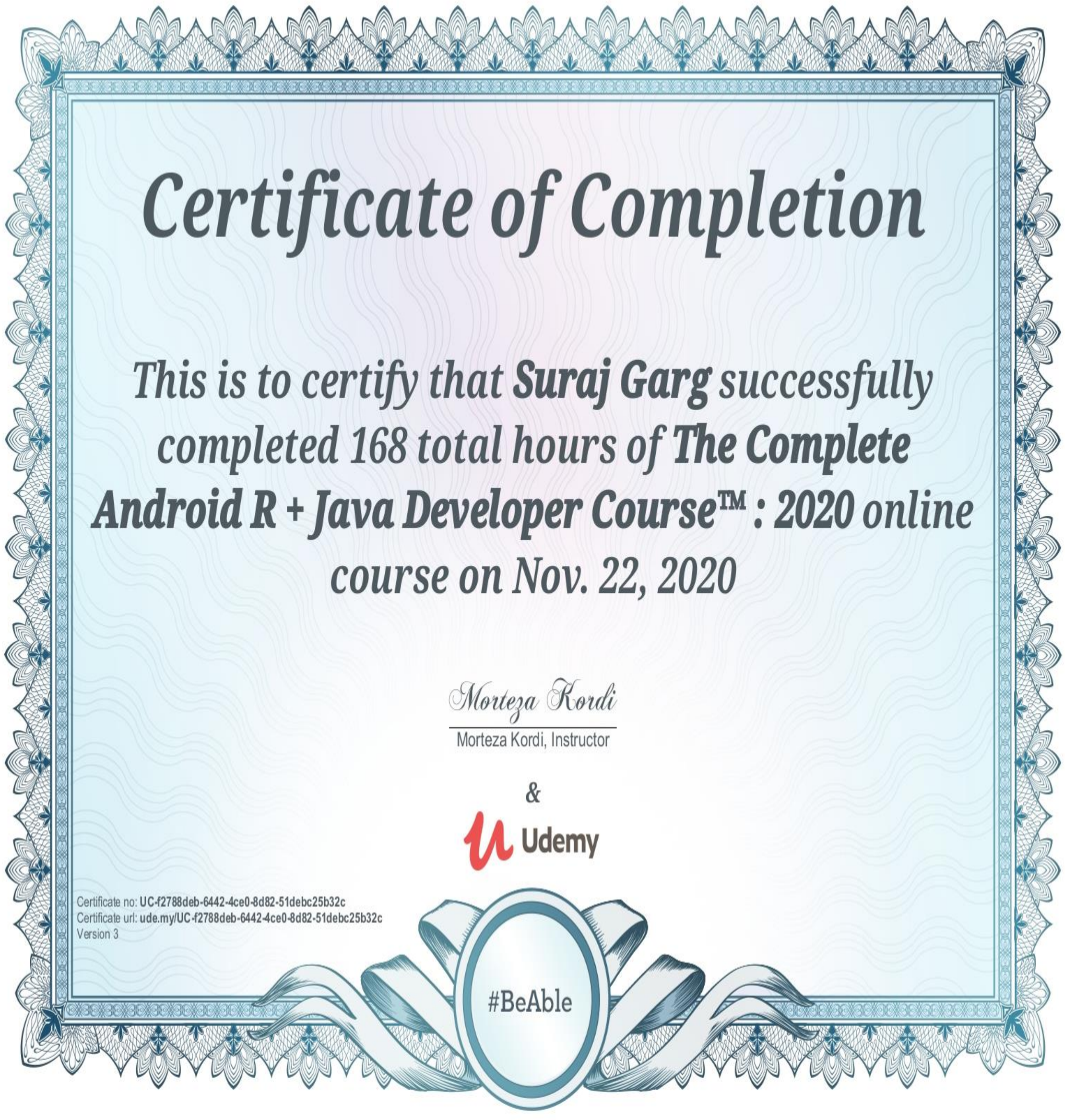

Face_Recogniser
Face-Recognition-attendance-managemant-systems
Enter ID
181500735
Clear
Enter Name
Suraj Garg
Clear
Notification :
es Saved for ID : 181500735 Name : Suraj
Take Images
Train Images
Quit
Take Attendance
Type here to search
ENG
IN
18:24
13-12-2020


Face_Recogniser
Face-Recognition-attendance-managemant-systems
Enter ID
181500735
Clear
Enter Name
Suraj Garg
Clear
Notification :
Image Trained
Take Images
Train Images
Quit
Take Attendance
Type here to search
ENG
IN
18:24
13-12-2020


Face_Recogniser
Face-Recognition-attendance-managemant-systems
Track Images
Quit
Attendance :
Type here to search
ENG
IN
18:25
13-12-2020


im
['181500735-Suraj Garg'
Type here to search
ENG
IN
18:25
13-12-2020

Certificates –



Certificate of Completion

This is to certify that **Suraj Garg** successfully completed 168 total hours of **The Complete Android R + Java Developer Course™ : 2020** online course on Nov. 22, 2020

Morteza Kordi

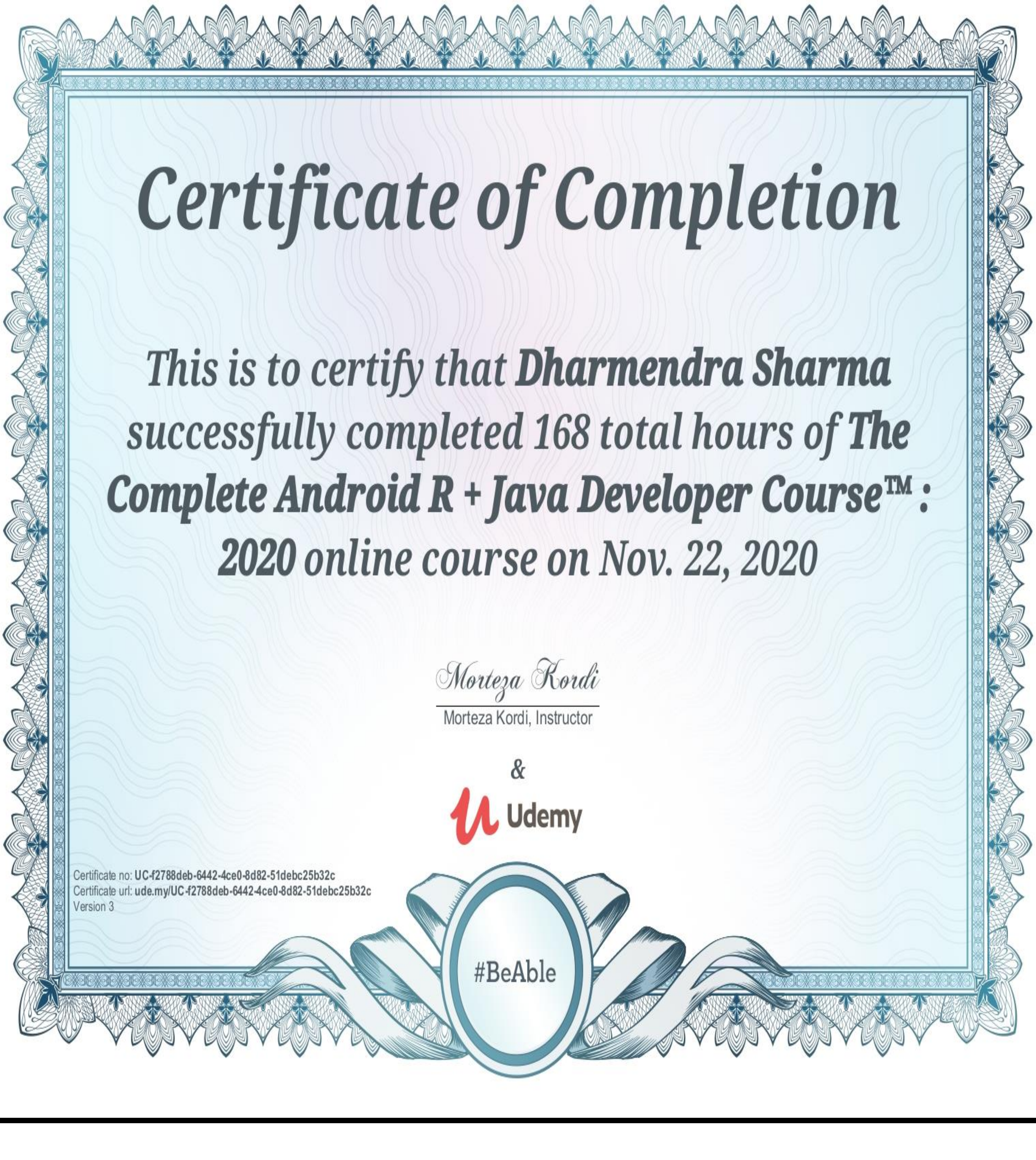
Morteza Kordi, Instructor

&

Udemy

Certificate no: UC-f2788deb-6442-4ce0-8d82-51debc25b32c
Certificate url: ude.my/UC-f2788deb-6442-4ce0-8d82-51debc25b32c
Version 3

#BeAble



Certificate of Completion

*This is to certify that **Dharmendra Sharma** successfully completed 168 total hours of **The Complete Android R + Java Developer Course™ : 2020** online course on Nov. 22, 2020*

Morteza Kordi

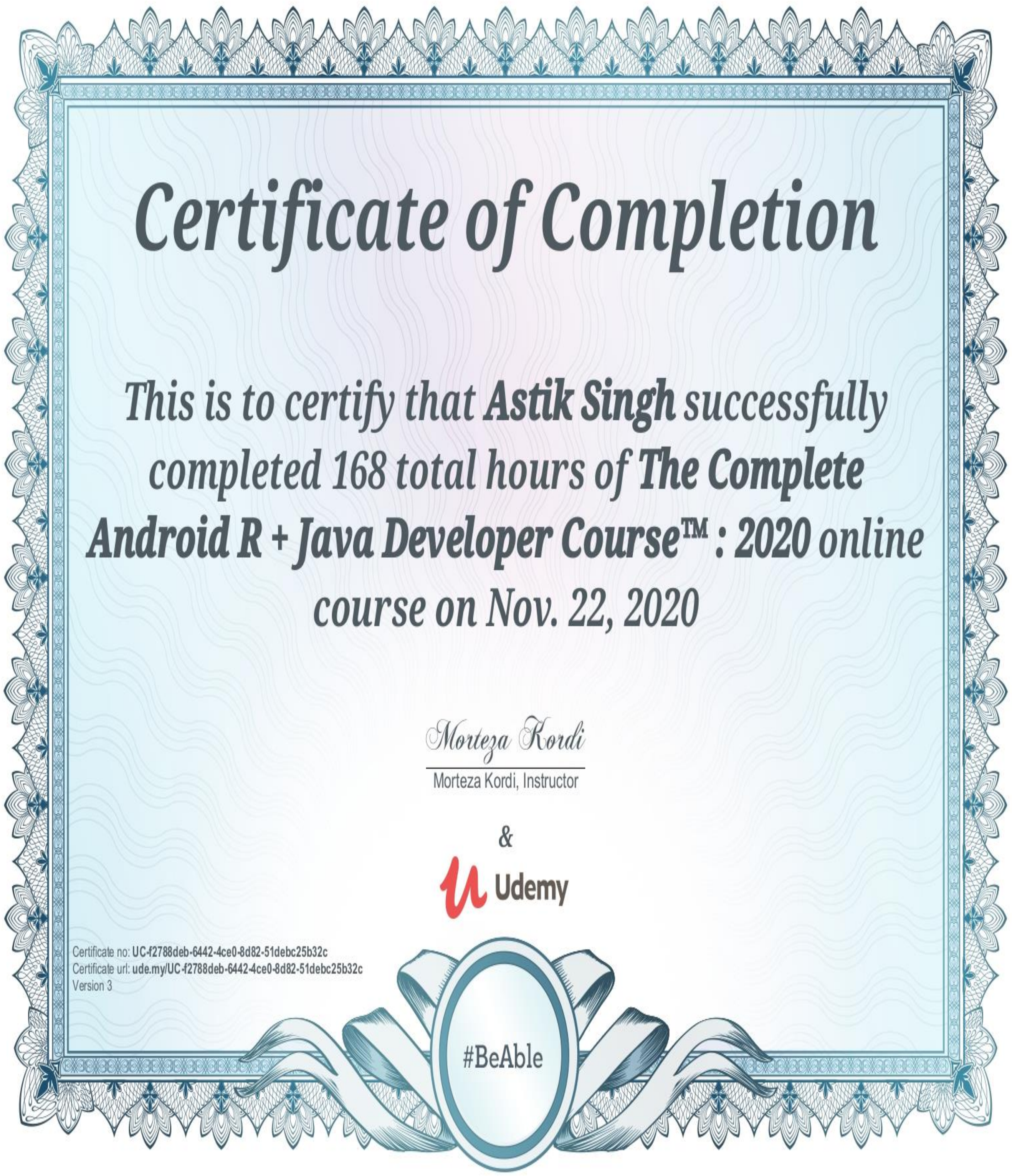
Morteza Kordi, Instructor

&

Udemy

Certificate no: **UC-f2788deb-6442-4ce0-8d82-51debc25b32c**
Certificate url: **ude.my/UC-f2788deb-6442-4ce0-8d82-51debc25b32c**
Version 3

#BeAble



Certificate of Completion

*This is to certify that **Astik Singh** successfully completed 168 total hours of **The Complete Android R + Java Developer Course™ : 2020** online course on Nov. 22, 2020*

Morteza Kordi

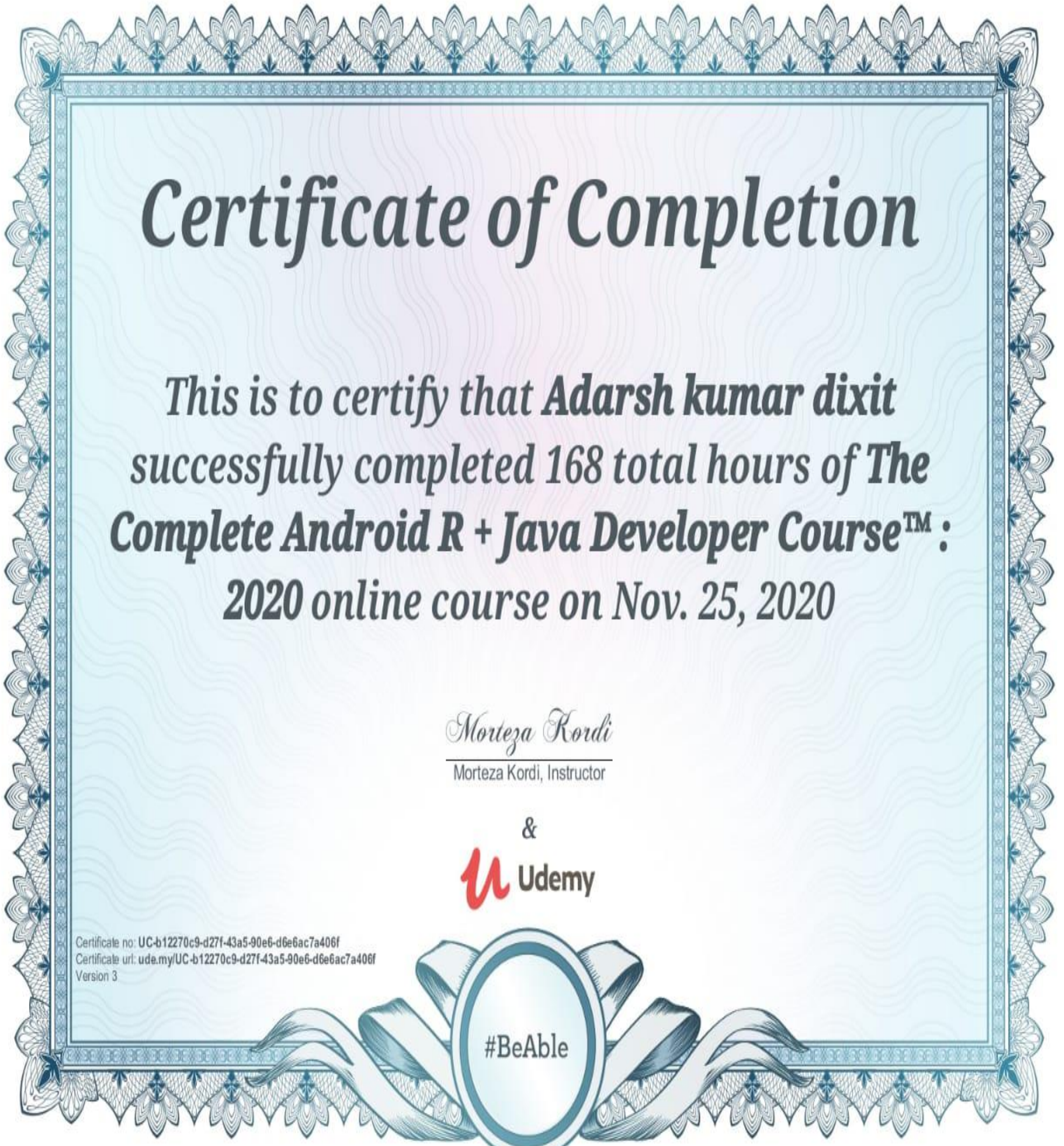
Morteza Kordi, Instructor

&

Udemy

Certificate no: **UC-f2788deb-6442-4ce0-8d82-51debc25b32c**
Certificate url: **ude.my/UC-f2788deb-6442-4ce0-8d82-51debc25b32c**
Version 3

#BeAble



Certificate of Completion

This is to certify that ***Adarsh kumar dixit*** *successfully completed 168 total hours of* ***The Complete Android R + Java Developer Course™ : 2020*** *online course on Nov. 25, 2020*

Morteza Kordi

Morteza Kordi, Instructor

&

Udemy

Certificate no: **UC-b12270c9-d27f-43a5-90e6-d6e6ac7a406f**
Certificate url: **ude.my/UC-b12270c9-d27f-43a5-90e6-d6e6ac7a406f**
Version 3

#BeAble