“AI based Smart Attendance Management System”

Team Members:

Prason Poudel 19BCE2550

Sashank Rijal 19BCE2484

Mickey Kumar Rouniyar 19BCE2520

Submitted to:

Prof. Geraldine Bessie Amali, SCOPE

School of Computer Science and Engineering

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Contents

Abstract

Face recognition is an essential prerequisite for an AI based Smart Attendance Management System. Previously automating an attendance process requires a real time system to quickly detect a face and mark the attendance of each staff by recognizing their faces. However, such systems are not able to distinguish between the real faces and the fake ones and as a result the employees can easily bypass their attendance by showing their pictures to the camera using their phones. So, such systems are not considered reliable enough as they are vulnerable to spoofing attacks. In our system, we have proposed a face liveness detection method using transfer learning on our custom datasets to prevent the attendance system from such spoofing attacks. We have used Mobilenet V2 network architecture for training a face liveness detector. Face detection is done by using the Viola–Jones object detection framework due to its high real time performance. The face liveness detection is done prior to face recognition. Face recognition is done by using a two way process. In the first process, the facial embeddings are extracted from a Facenet model which uses triplet loss to output a 128-d feature vector called an embedding. Then, in the second process, we train a SVM classifier to a list of embeddings to create a recognition model. The attendance process is then automated by using a recognition model to recognize the face of each staff precisely and automatically marking their attendance. This project is a biometric software application with interactive user interface by using a standard GUI library Tkinter. This project research work involves the disciplines from computer vision, deep learning, machine learning, image processing, calculus and statistics.

Keywords: Facenet, embedding, 128-d, Viola-Jones algorithm, SVM classifier, Mobilenet V2, transfer learning, liveness, spoofing attacks, facial embeddings, triplet loss.

1. Introduction

Over the past years attendance is done by calling the names or by passing the sheets and by signing the signature on the attendance paper which is time or effort consuming. As time gradually passes the new way of keeping attendance records of staff has been shifted to the technology that uses biometric which requires eye, face or fingerprint as a user input. Among all biometrics, Face recognition is an ongoing research topic in the field of computer vision and is widely adopted due to its contactless process.

In face recognition, the identification of an individual is done by comparing the captured images with the stored images of that person in real time. These recognition systems are in the rapid development phase and are accumulated with a new strong algorithm that improves the system day by day. However, these systems are facing many security issues as frauds are increasing on a daily basis and there is a need to upgrade these systems to make them more secure, reliable and automatic. So by observing all these things we are inspired to build our project "AI based Smart Attendance Management System" which not only improves the existing system but also adapts to some of the security challenges making this system more secure and reliable.

2. Hardware / Software Requirements

Hardware Used	Software Used
1. Web Camera	1. Visual Studio Code
2. Laptop	2. Anaconda

3. Methodology

The system development process was done by adopting the agile methodology in our project “AI Based Smart Attendance Management System". The whole system was broken down into small, and manageable increments. Then we experimented quickly by iterating over different approaches for solving a small problem. After each iteration, we examined our progress frequently and decided the next course of action. If a chosen course of action wasn’t delivering the results we expected, we changed the direction quickly else we stuck with the same path and continued further by making increments in the system.

Our system had to go through three phases: Registration phase, Training phase and Testing phase. In the registration phase, the system had to take in user inputs and make use of a camera sensor i.e webcam to gather the datasets which can later be referenced for the recognition purpose. The camera accepts the facial inputs from the staff and the Viola-Jones algorithm detects the facial region from the captured frames of each staff. Then the detected faces are stored in a separate folder which makes the training datasets and the user inputs are stored in a separate database generating a unique user id which completes our registration process. In the training phase, the training datasets are provided as inputs to the pretrained FaceNet model which uses Inception ResNet V1 network architecture for extracting a list of 128-d feature vectors called embeddings. Then we create a recognition model by training a SVM classifier to a list of embeddings. The embeddings and the trained model is serialized and stored in pickle format so that it can be deserialized and used during the testing phase. In the testing phase, the detected faces have to go through a liveness detection method which checks whether the face comes from a live person or not. If it comes from a live person then it is sent to the recognition pipeline for the recognition process. The first process in the recognition pipeline is to extract a 128-d feature vector from each detected face by using a FaceNet model. Then, each 128-d feature vector is taken as input to the recognizer model and the attendance process is automated by recognizing the face of each staff precisely and automatically marking their attendance.

3.1 Limitation

We have faced some problems during the development of our projects which are mentioned below:

- The most challenging part of this system is to distinguish between identical twins. This situation is still a challenge to biometric systems especially facial recognition technology so the main limitation of our system is to detect identical twins.

- System is unable to prevent 3d mask attack as our system does not consist of 3d sensors.

- System is unable to recognize faces in low light conditions as we have trained our model with high resolution images and in a good lighting condition.

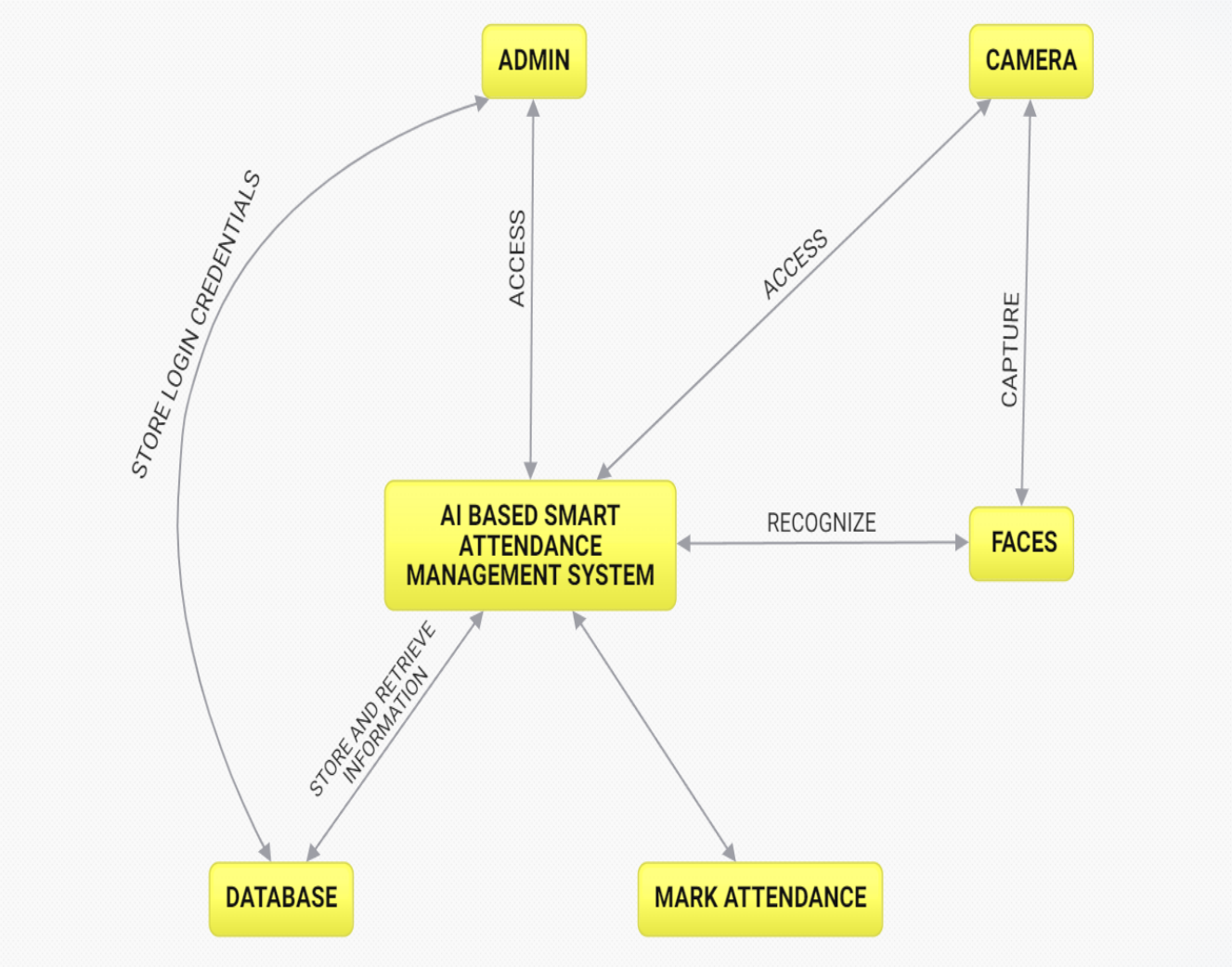
- The anti-spoofing model works well with high resolution camera inputs but the model performance drops down rapidly when the model is tested with low resolution camera inputs.

- The system is not capable of figuring out the registration of the same staff with different names.

- The liveness detection model is trained and validated on high resolution images but the model is tested on low resolution images captured from webcam due to which the liveness detection model is not giving good results.

4. Proposed Model

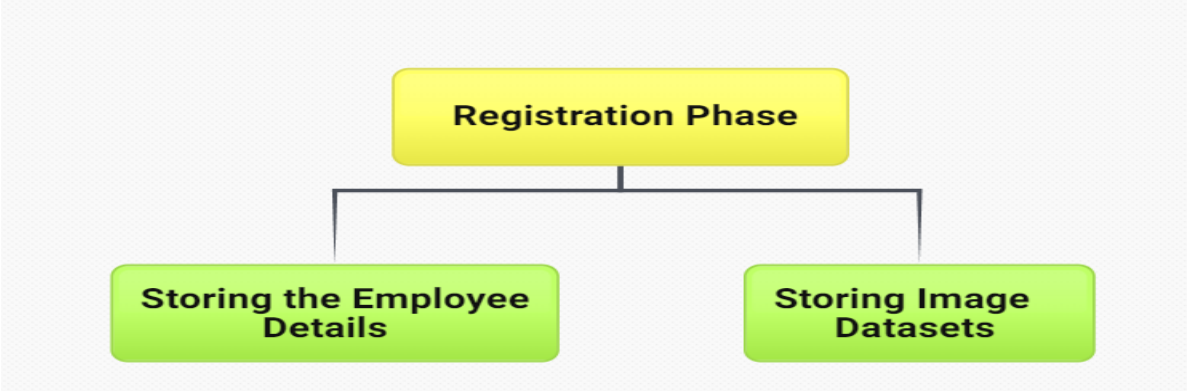
4.1 Design



4.2 Module Wise Description

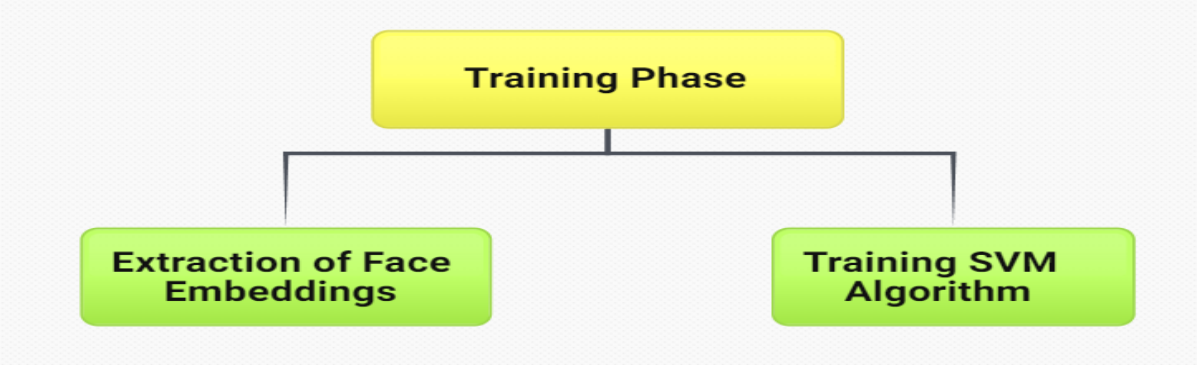
Our system had to go through three phases:

- Registration phase
- Training phase
- Testing phase



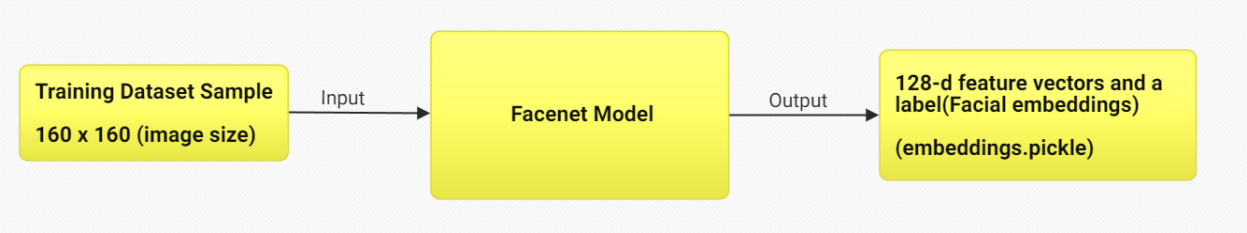
During the Registration phase, at first employee will fill the details in the given form and then there is a button called add which will initialized the web camera when it gets pressed and captures 50 images from the Employee.

Then on those capture images we have to apply face detection algorithm. So here we have used OpenCV Haar Cascade Classifier as a face detection algorithm. Then that detected face is resized to 160 x 160 pixels. Then after capturing the 50 images web camera will gets destroy and later the images are stored in the local directory. At last all the employee details which were filled gets inserted in the database.



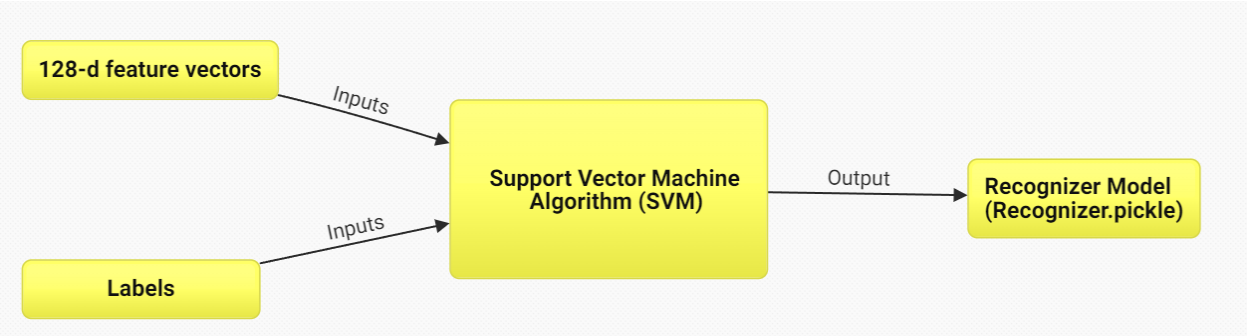
Extraction of Face Embeddings

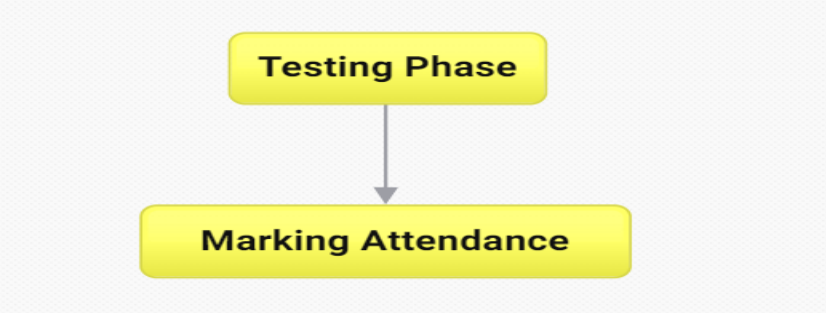
At first, the training datasets which have been generated from the registration process is provided as an input to the FaceNet model. Here the FaceNet model acts as feature extractor for extracting 128 dimension feature vector and a label called facial embeddings. Then the output is stored as embeddings.pickle file.



Training SVM Algorithm

Here for training the support vector machine algorithm (SVM), we provide 128-d feature vectors and the labels as inputs to the SVM Algorithm. So then we train the SVM algorithm by using supervised machine learning approach to output a Recognizer model, which is stored as a recognizer.pickle file

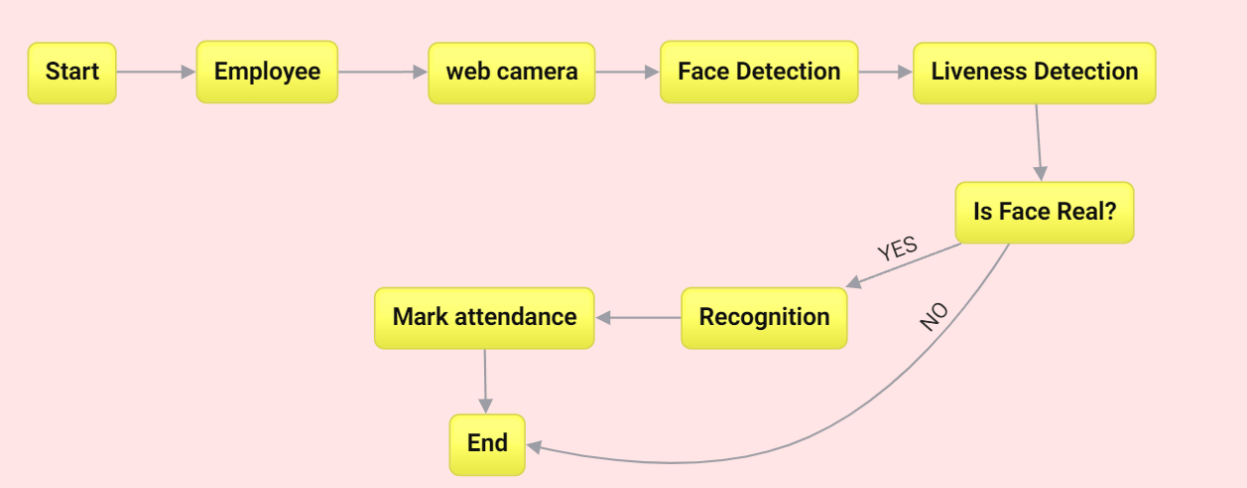




During the Testing Phase, at first employee will face the web camera then the camera will gets initialized and capture the frames and then on those captured frames we apply face detection algorithm. So here we have used OpenCV Haar cascade classifier algorithm as a face detection algorithm.

Now those detected faces are forwarded to liveness detection method. In the liveness detection method our system checks whether the detected face comes from the live person or a fake person. So if that comes from a live person then it is forwarded to the recognition pipeline.

In the Recognition pipeline the recognizer model that has been trained in the training phase gets loaded up and then it recognizes the identity of each staff automatically and mark their attendance as present.



Liveness Detection

Face liveness detection is an essential prerequisite of face recognition systems. Attendance systems which rely on face recognition technology are still vulnerable and open to attacks where a staff can easily fool a system security by showing facial pictures such as portrait photographs, mobile photos and videos. User photos can easily be found through social media and used to spoof facial recognition software. This way, users can easily bypass the system security and fool the system to fake the attendance of staff. That's why it is important to have anti-spoofing systems in place to reduce theft, and mitigate frauds in the attendance process. These systems enhance existing facial recognition solutions by improving their ability to detect fraud. This is where the need for anti-spoofing solutions comes into play. We rely on liveness detection to validate an individual's identity. These checks can verify whether an individual is actually present or using a photo to spoof the system.

Then, the training process is done by using a transfer learning technique on this dataset to create a liveness detection model. We have used Mobilenet V2 network architecture for training a face liveness detector. The liveness detection model is a binary classification model which can classify whether the detected face comes from a live person or not. If it comes from a live person then we can classify as live and if it doesn't come from a live person then we can classify as spoof.

After training, we saved the pretrained weights and the architecture of a model so that we can load and use the trained model for inference. The weights file is saved in h5 format and the configuration of the model is saved in json format. The weights file and the configuration file of the trained model is saved under the 'models' directory.

Models

1. Antispoofing_model.h5
2. Antispoofing_model.json

During the testing phase, the detected faces from a webcam have to go through a liveness detection method where the weights file and the configuration file of the model is first loaded from the directory and then the liveness check is performed to identify whether the detected face comes from a live person or not. If it comes from a live person then it is sent to the recognition pipeline for the recognition process otherwise an error or warning is shown to the user for faking the attendance process.

4.3 Implementation

The proposed method is implemented in the Python programming language with complementary tools like Tkinter for making GUI software applications and Apache server for starting MySql database server.

Software Packages and Libraries

- **OpenCV-Python:** OpenCV-Python is a Python wrapper around the original C++ implementation. It supports a lot of algorithms related to Computer Vision and Machine Learning and it is expanding day-by-day. In our project, we have used this library to gather the staff dataset from a live stream, preprocess the captured images and apply face detection and recognition algorithms to a live video.

- **NumPy:** NumPy stands for Numerical Python. It is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. Images are the collection of pixels and can be represented in an array of pixel values so to compute any image processing operations we require this package. In our project, we need to apply various image processing operations such as resizing image dimensions, expanding image dimensions, reading the image, cropping the facial region from an image so, for getting all those things done we have used this library.

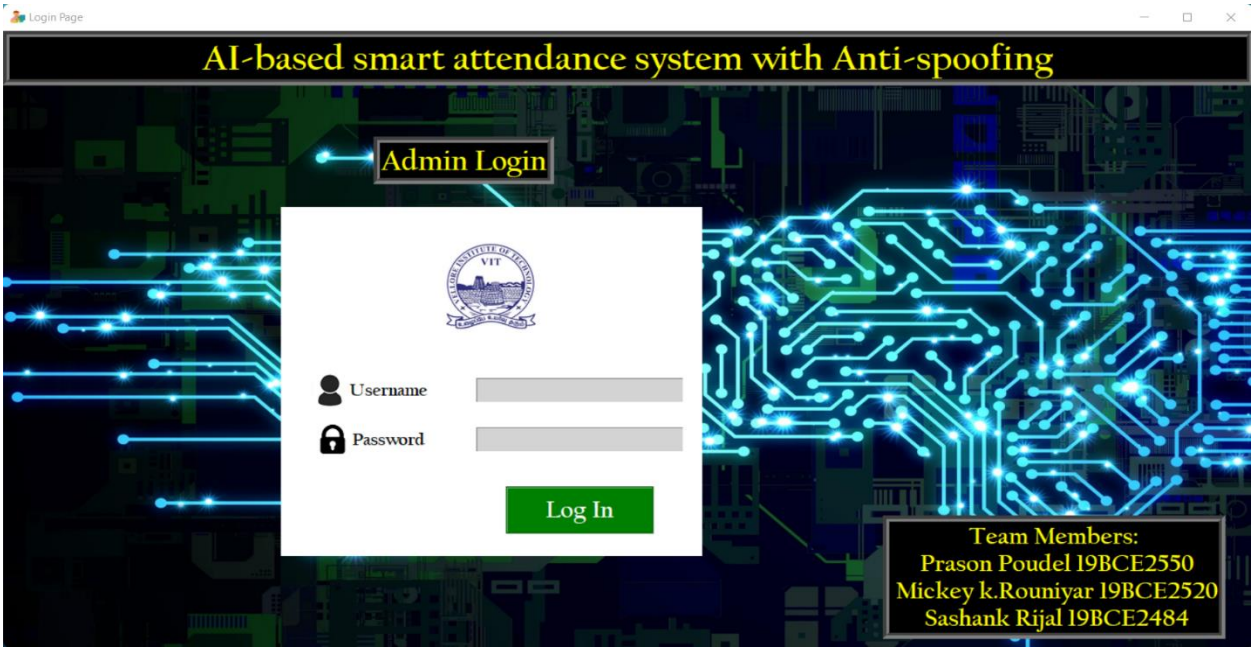
- **Scikit Learn:** Scikit-learn is a free machine learning library for python programming language. It features various classification, regression and clustering algorithms like SVM, KNN, Random Forest, Linear Regression and is designed to interoperate with the Python numerical and scientific libraries Numpy and Scipy.

- **Tensorflow:** Tensorflow is an open source library for numerical computation and large scale machine learning.

- **Keras**
- **Pickle**
- **Tkinter**
- **PyMySql**
- **Pillow**
- **Os**
- **Random**
- **Sys**

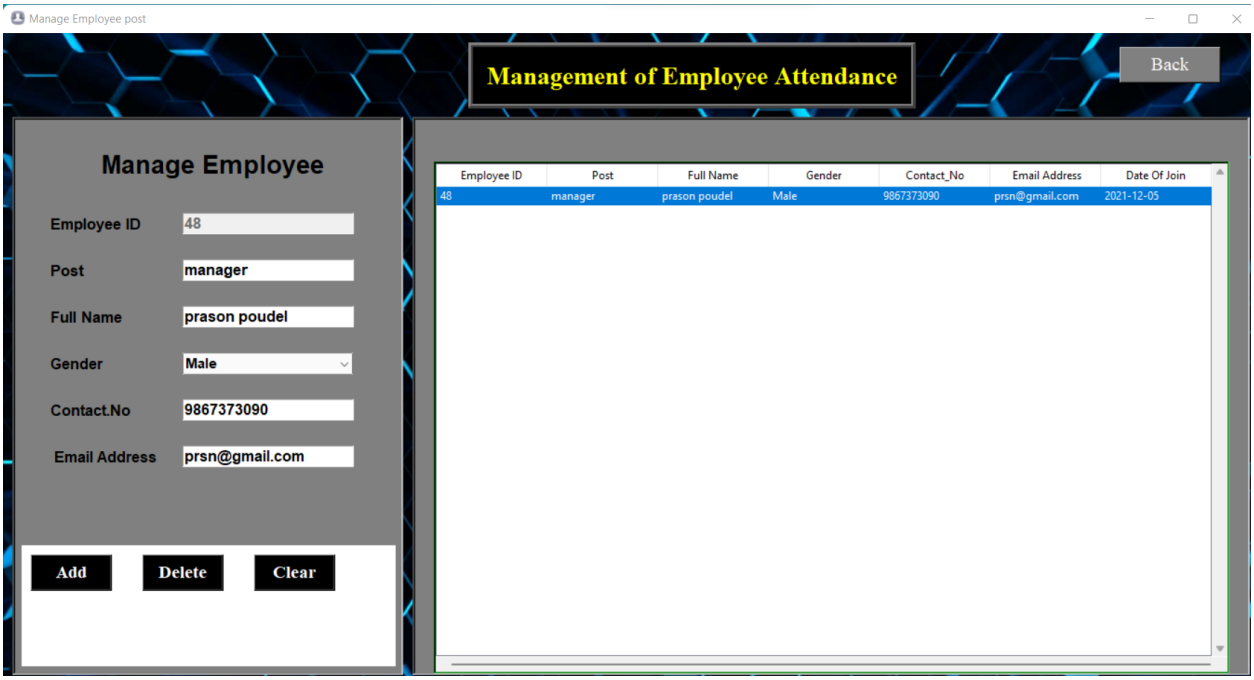
5. Results and Discussion

Initially system start with the Graphical User Interface Login page in which we need Login credentials i.e. Username and Password. If the provided login credentials is correct then admin can enter into the main system otherwise not.

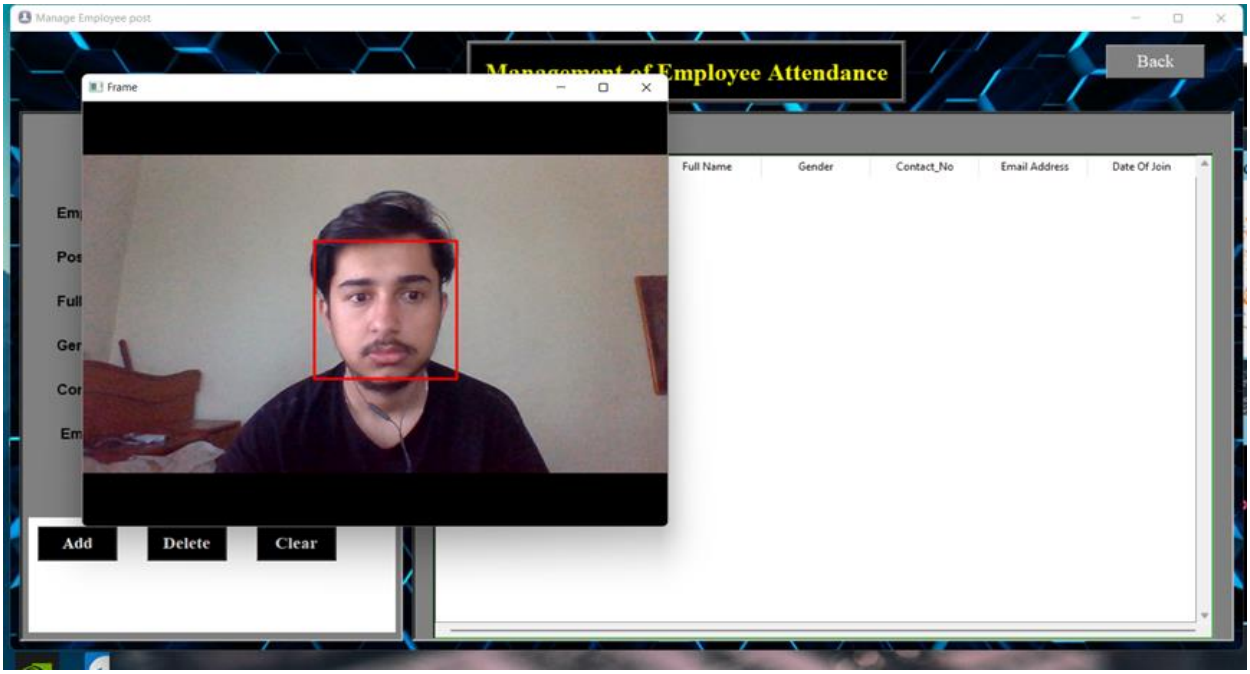


This is our system main page. In this admin can perform all the activities that are required to manage all the details of employees and the system.

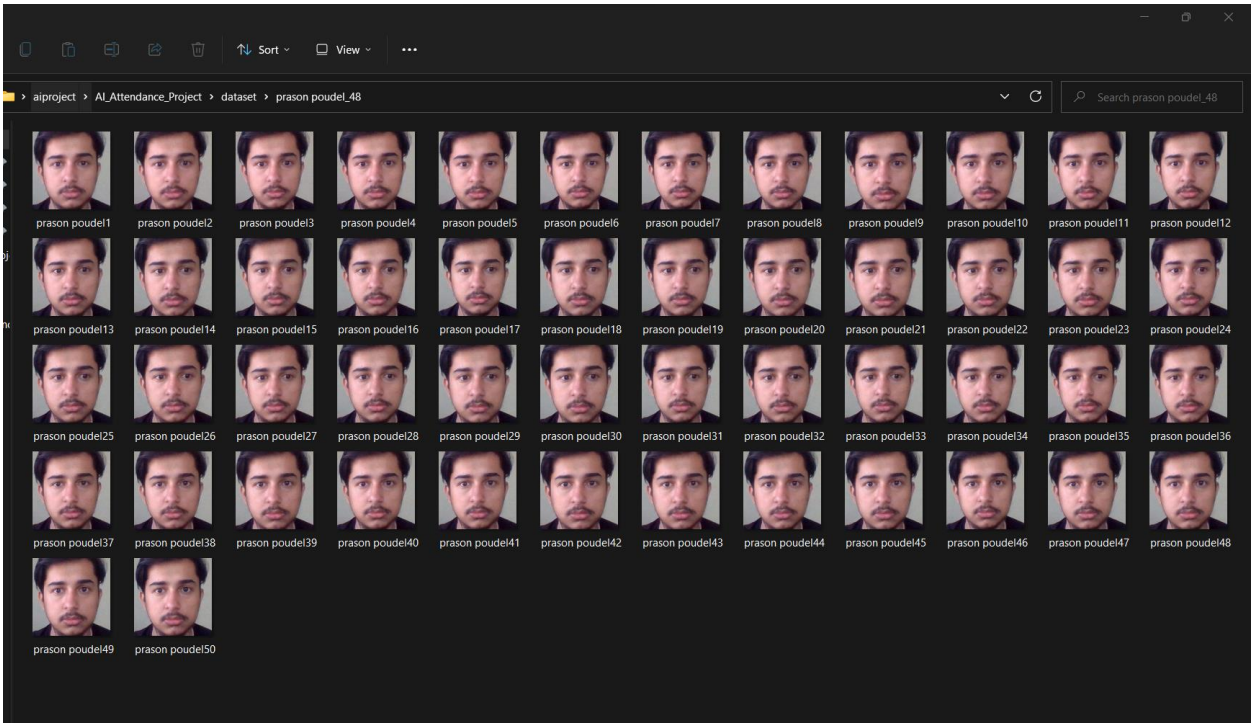
Now we will see how the admin manages all the details of the employee by viewing the Employee Management Page. To register the employee we need to provide all the data of the employee as given in the figure.



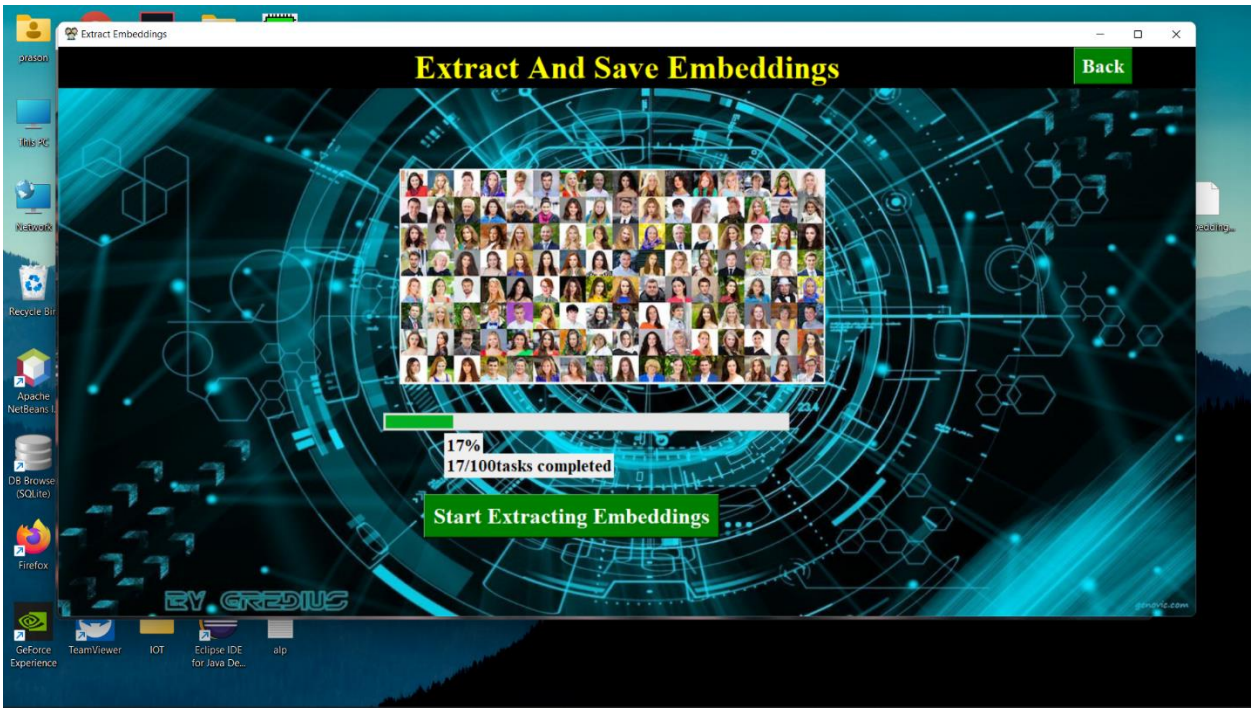
After we click the add button the data is stored in the database and displayed in the right frame of Management of Employee Attendance page. Also after registering the data the photo samples of employees are added by clicking Add Photo Sample.



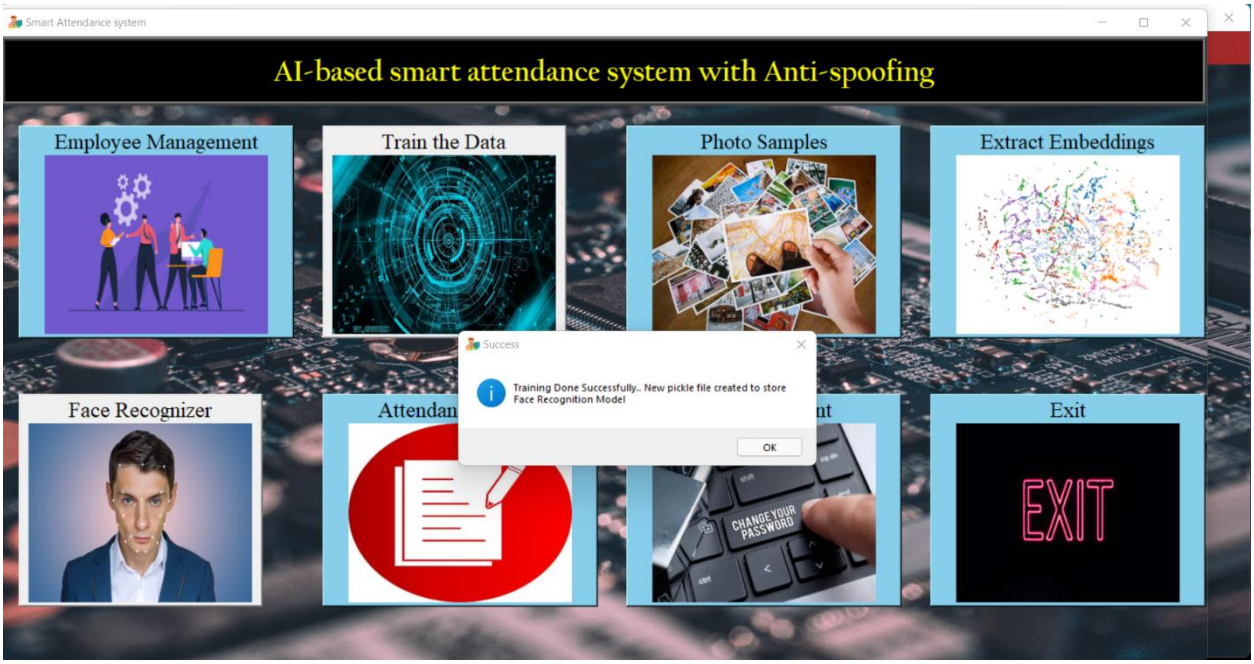
We can also view all the photo samples by clicking the Photo Samples button.



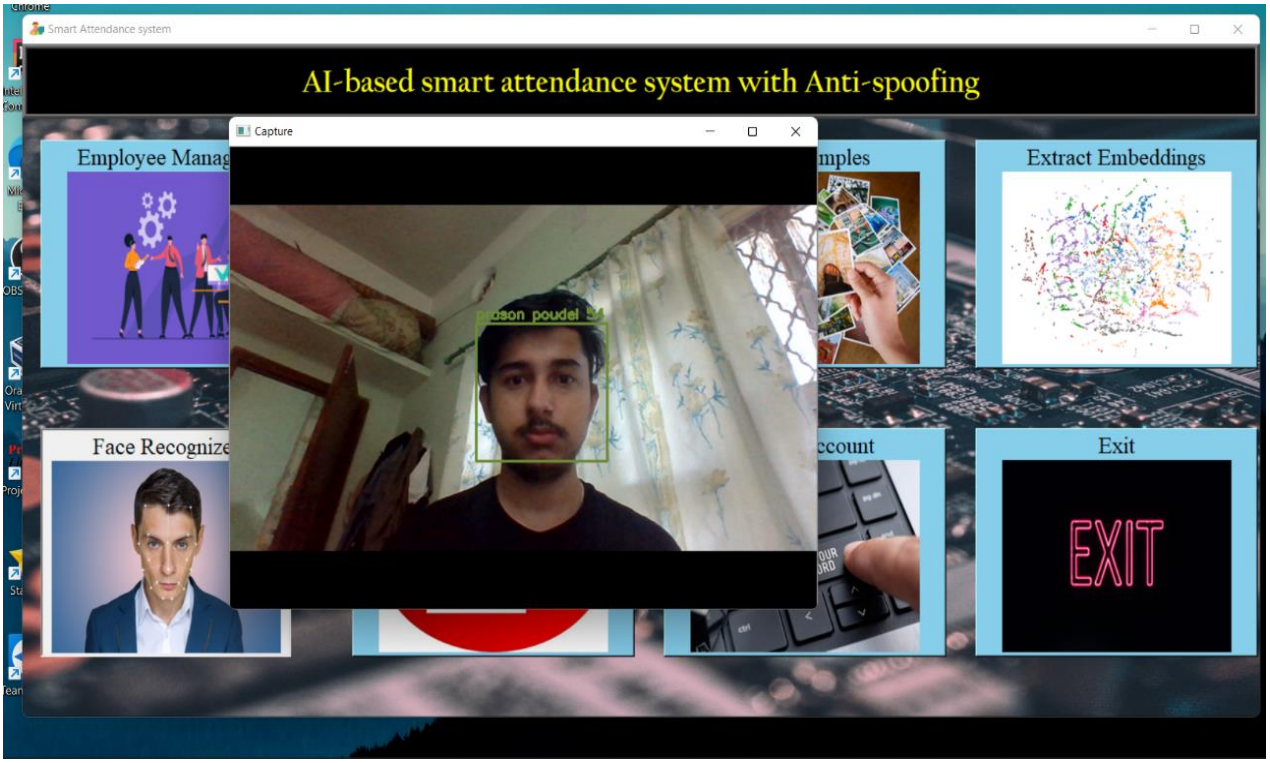
After registering the employee now we have to extract embeddings of photo samples so for this we have to navigate to the open embeddings page

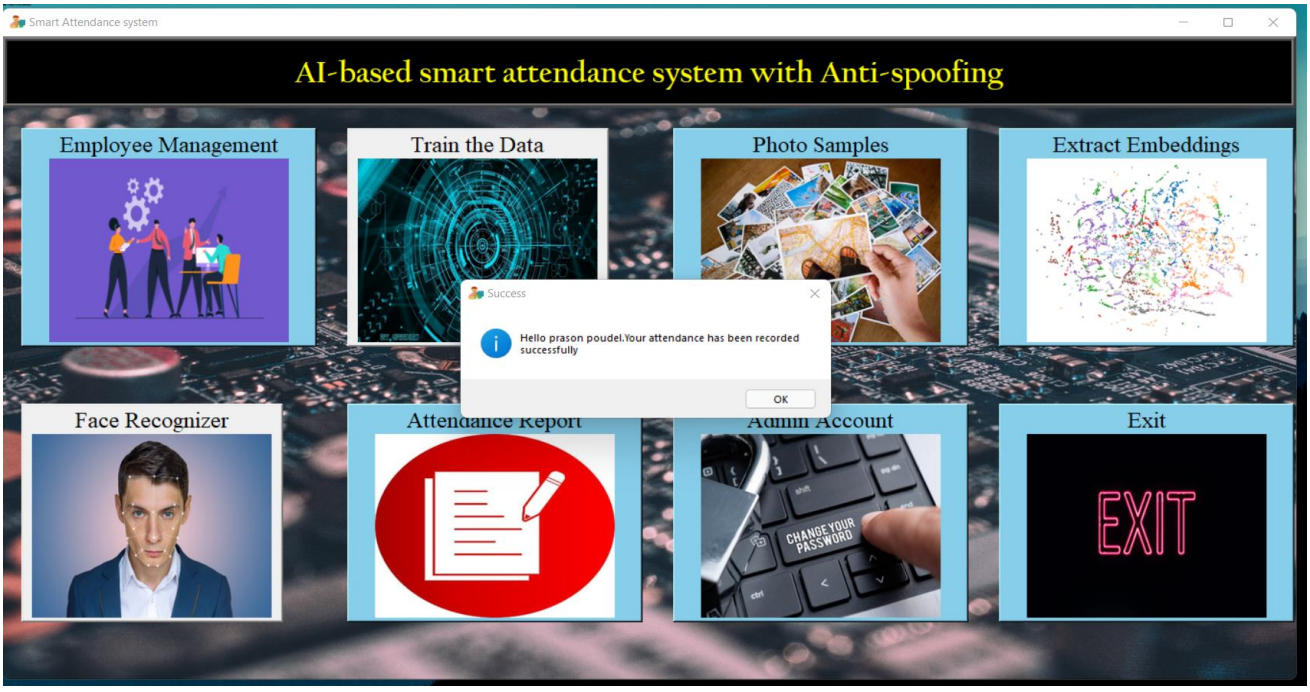


After that we have to train the photo samples of employees by taking registered photo samples.

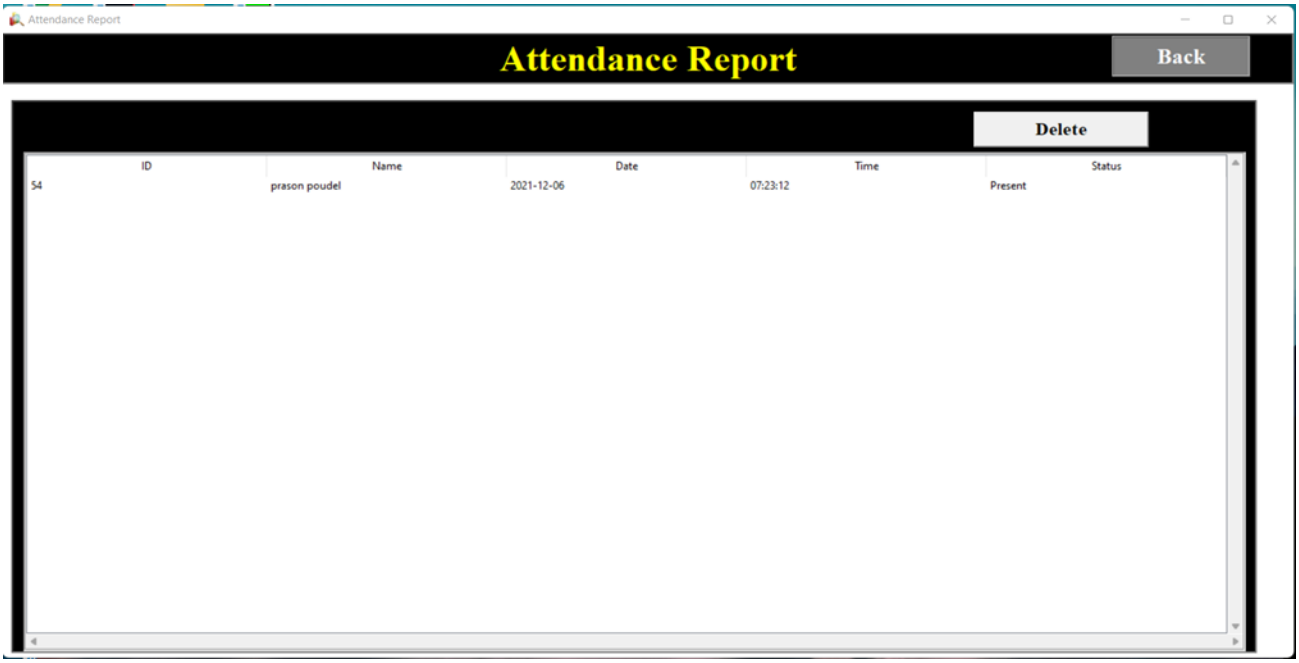


Finally the system can recognize registered employees through a recognizer model which is applied in the proposed system.

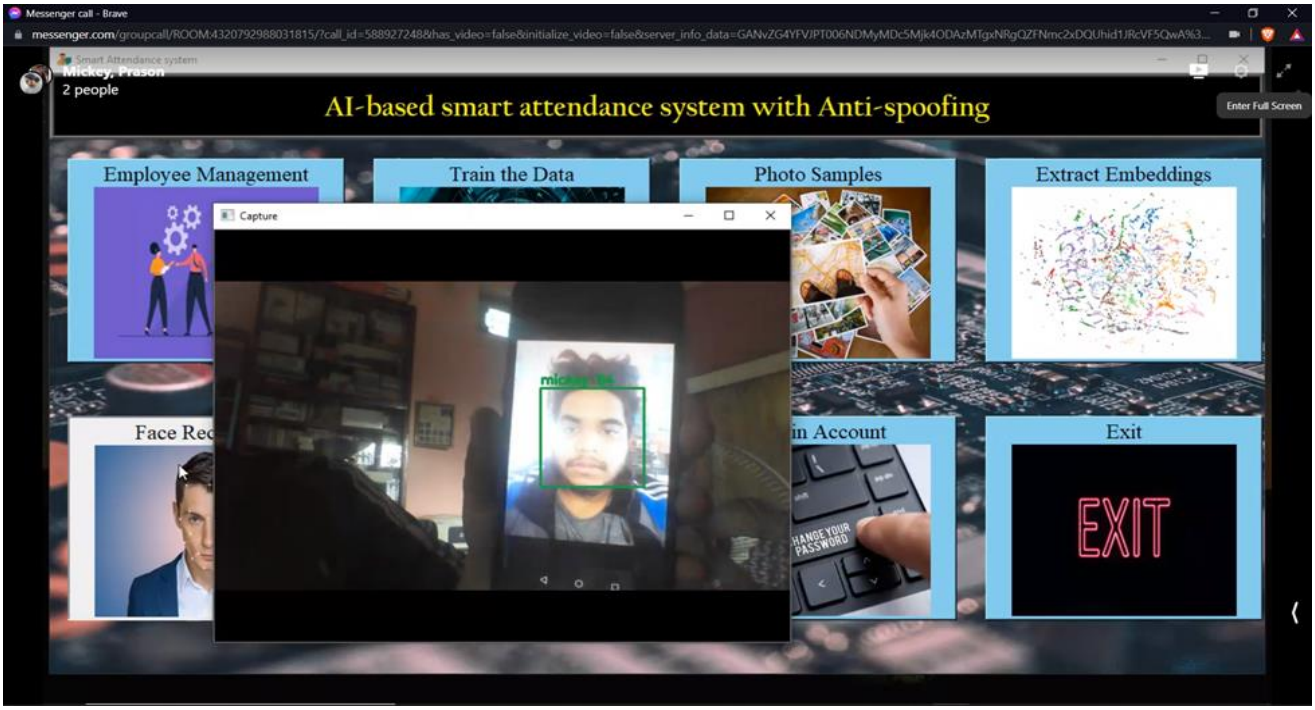




When the system recognizes the employee then attendance will be registered in the Attendance Report Page where we can view all the details of present and absent employees with their date and time.

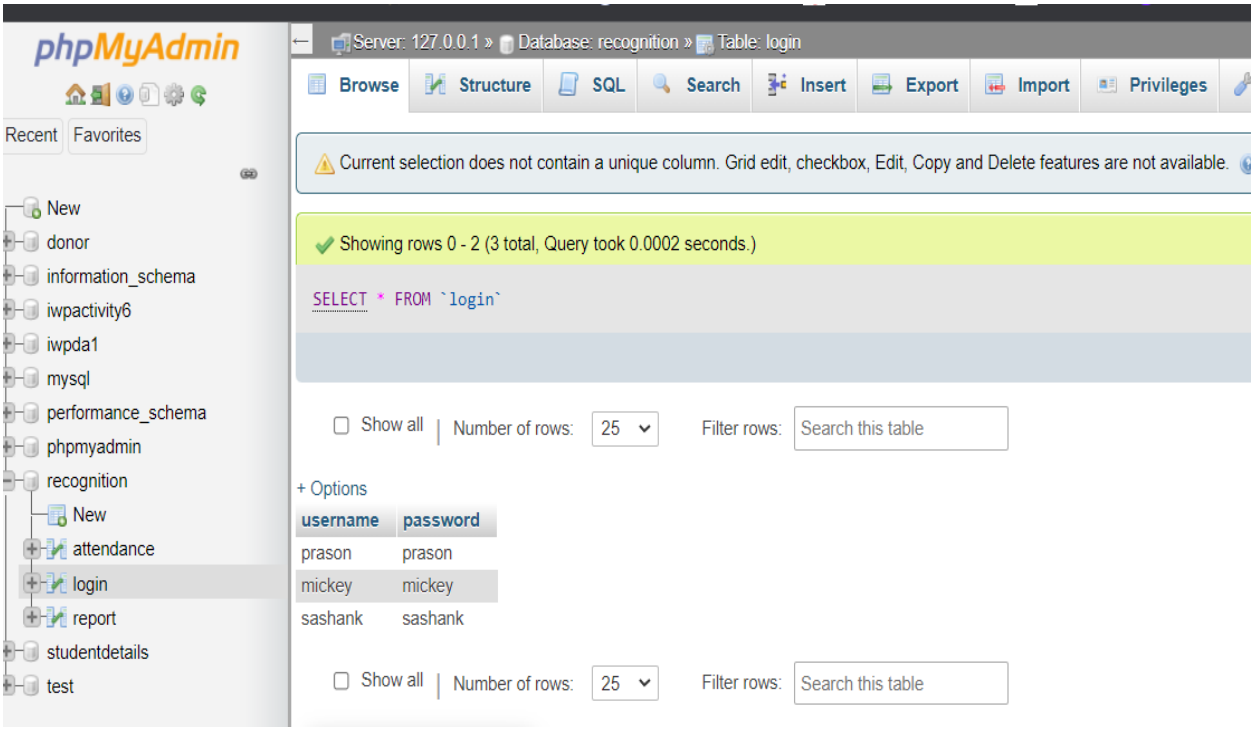


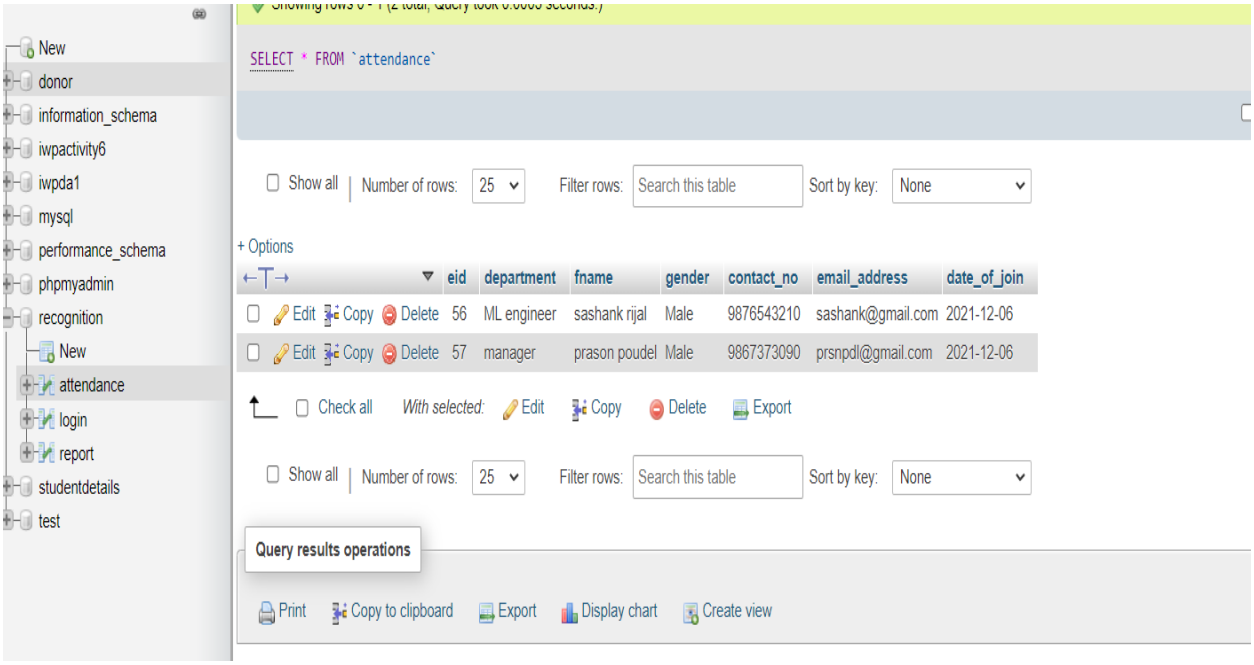
Spoofing Attempt



Spoofing Alert

Database Connection





6. Conclusion and Future Work

From scratch to a working software, carrying out real-world software projects in our academic studies helps to understand what we have achieved from the technological advancement from past decades and what knowledge we have to gain inorder to cope with the technological commercial era. And working with a team collaboratively taught us how to work as a team as almost every software project is done in a large industry of likeminded people to achieve a common goal.We have most of the industrial strategies used for the completion of projects by keeping accounts of time, quality and budget.

It was a different and most wonderful experience working on a AI based smart attendance management system where we explore the part of the computer vision in artificial intelligence i.e. image processing ,classification ,identification and recognition. We also explore different deep learning algorithms which are evolving day by day to become intelligent artificial brains.

Future Enhancement

We intend to do the following tasks in the coming future to make it more competitive.

• To make system able to detect twins.

• To prevent 3D-Mask attacks.

References

[1]. Jason Brownlee, “How to Develop a Face Recognition System Using FaceNet in Keras”, June 7, 2019.

[2]. E. Varadharajan, R. Dharani, S. Jeevitha, B. Kavinmathi and S. Hemalatha, “Automatic attendance management system using face detection”.

[3]. Ming, Zuheng et al. “A Survey on Anti-Spoofing Methods for Face Recognition with RGB Cameras of Generic Consumer Devices”.

[4]. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks”.

[5]. Schroff, F., Kalenichenko, D., & Philbin, “ FaceNet: A unified embedding for face recognition and clustering”.

Source Code

```
def face_recognize():
 print("[INFO] starting video stream...")
 training_obj = Training(embedding_path='models/facenet_embeddings.pickle')
 [label,Embeddings,labels,names] = training_obj.load_embeddings_and_labels()
 recognizer = pickle.loads(open('models/recognizer.pickle', "rb").read())
 vs = cv2.VideoCapture(0)
 predictions = [ ]
 while len(predictions) <= 10:
 try:
 (ret,frame) = vs.read()
 gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray,1.3,5)
 for (x,y,w,h) in faces:
 face = frame[y-5:y+h+5,x-5:x+w+5]
 resized_face = cv2.resize(face,(160,160))
 face_pixel = embedding_obj.normalize_pixels(imagearrays=resized_face)
 sample = np.expand_dims(face_pixel,axis=0)
 embedding = embedding_model.predict(sample)
 embedding = embedding.reshape(1,-1)
 COLORS = np.random.randint(0, 255, size=(len(label.classes_), 3),dtype="uint8")
```

```
# perform classification to recognize the face
preds = recognizer.predict_proba(embedding)[0]
p = np.argmax(preds)
proba = preds[p]
name = label.classes_[p]
if proba >= 0.5:
color = [int(c) for c in COLORS[p]]
cv2.rectangle(frame,(x,y),(x+w,y+h),color,2)
predictions.append(name)
else:
name = "NONE"
color = (255,255,0)
cv2.rectangle(frame,(x,y),(x+w,y+h),color,2)
cv2.imshow("Capture",frame)
```