**Faculty of Computers**
**& Artificial Intelligence**

**Benha University**

# Face Recognition Attendance System

A senior project submitted in partial fulfillment of the requirements for the degree of Bachelor of Computers and Artificial Intelligence.

**Computer Science Departement,**

## *Project Team*

1- Abdallah Ramdan Mohammed
2- Nourhan Taher Mohammed
3- Nagham Sherif Abo Elmakrem
4- Hadeer Ahmed Sami
5- Honey Ahmed Mohammed

## *Under Supervision of*

**Dr. Shadia Sarhan**
**Eng Hisham Gamil**
**Eng Abdelwahab El-mestekawy**

Benha, July 2021

J u l y – 2 0 2 1

# DEDICATION

The family is the great entity that support and assist successful work. In my ups and downs times, our parents have always been there for me. They have shown me how to take responsibility as a person. I wanted to express my love and appreciation to our parents since they have been there for us over these years, helping us and striving to see us succeed. We wouldn't have succeeded without them, so we would like to dedicate this work and success to our parents.

# ACKNOWLEDGMENT

After thanking Allah for giving us the will to finish this project, we need to thank people who were very helpful and never spared no information from us. Without the effort and cooperation of our group members, this project will not be finished. We would like to express our gratitude to our supervisor, Dr. Shadia Sarhan for her guiding, motivation and support. We also want to thank our teaching assistants Eng. Adelwahab Al-Mestekawy and Eng. Hesham Gamil for their helping us going forward.

# DECLARATION

We hereby certify that this material, which we now submit for assessment on the program of study leading to the award of Bachelor of Computers and Artificial Intelligence in ( Computer Science ) is entirely our own work, that we have exercised reasonable care to ensure that the work is original, and does not to the best of our knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of our work.

**Signed:** _____

**Date:** 13, July 2021.

# ABSTRACT

Attendance management is one of the most important processes of many work environments. Educational organizations need to manage the record of students present daily over the whole semesters. Traditional attendance management systems have been used for many years and it was good enough to be reliably used but that does not negate, it was time consuming and it was there always a chance for proxy attendance. New technologies always help us to improve the daily approaches. Fortunately, new technologies have led to development of an automatic attendance system that produces significantly successful results. The purpose of the automation of attendance system in educational institutes is to make the process of recording the attendance more optimized and efficient. The core definition of the system can be summed up in identifying the students by their faces to mark up as attendant. Our system identifies multiple faces in real time by video capturing the lecture hall. First step after capturing live video, face detection is done face recognition happens then the system searches for every face match in the database and the result at the end is an excel sheet with names of these students and some other data the instructor needs to know. We implemented face detection part by OpenCv and deep learning the next step was implemented to extract embbeddings from the images using pre-trained OpenFace model.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF ACRONYMS/ABBREVIATIONS

| | |
|---|---|
| **ML** | Machine Learning |
| **DL** | Deep Learning |
| **AI** | Acritical Intelligence |
| **CNN** | Convolutional Neural Networks |
| **DNN** | Deep Neural Networks |
| **RNN** | Recurrent Neural Networks |
| **ReLU** | Rectified Linear Unit |
| **IOT** | Internet Of Things |
| **ROI** | Region of interest |

*C h a p t e r  O n e*

# 1 INTRODUCTION

Educational organizations require a stable students attendance recording system. Attendance is important for the instructor and the student as well, so the problem was found in the conventional method in the lecture hall. Calling out the students by their names or giving them the attendance sheet to sign is not only has a problem of time consumption but also is not guaranteed, Thus, Automated attendance system can solve the traditional system problems.

The system uses two main operations which are face detection and face recognition. The face detection step is used to find the faces regions and mark them in boxes and face recognition is used for identifying the person whose face detected. The live camera captures a video of students then face detection process detect the faces found and crop the faces. Face recognition process maps the faces with the dataset we created to recognize whether the faces found in dataset or not.

Post the face recognition process is done the system retrieves from the database more details about each student that recognized by their name and the output of the system is a sheet with all the attendees. Our system which is Face Recognition Attendance System used in many secure applications like banks, airports etc. It follows a tallying procedure to record attendance and stores it in the system.

## 1.1 PROBLEM DEFINITION

Previously, the attendance in the educational organizations was done manually by an attendance sheet and there were many problems to get the attendance which are:

1. Waste of time and effort.
2. Proxy attendance may happen.
3. Too much paperwork is done.

## 1.2 EXISTING ATTENDANCE SYSTEMS

There are many attendance systems found we will mention some of them.

**a. Paper Based Attendance System:**

The Paper Based Attendance System is the manual attendance system. Attendance can be taken in any format, and it is documented on a piece of paper by writing either the absentees or merely the attendees. Faculty members frequently write the roll numbers of students who are absent or present.

**b. Token Based Attendance System:**

Token-based attendance requires displaying a security token when required to confirm attendance. A security token is a small piece of hardware that the owner carries about to approve access to a network service. This could take the shape of a smart card or be integrated in a common object. In case of educational organization, the token may be an identification card.

**c. Swipe Card Attendance System:**

The Swipe Card Attendance System works by a person swiping their card at the gate as they enter and exit, and the attendance is recorded. Before any entrance can take place, a swipe card must make touch with the associated card reader. When a card's magnetic stripe passes through a gate's console, the entrance becomes active.

## 1.3 DRAWBACKS OF EXISTING ATTENDANCE SYSTEMS

**a. Accuracy:**

Manually taking the attendance will lead to low accuracy as many scenarios many happen such as if a student has illegible handwriting, it could make it difficult to ensure his presence. The organization is essentially depending on the morality system when it comes to manual reporting. This method has the power to be misused, resulting in time theft.

**b. Efficiency:**

Manual attendance system has problems in the efficiency it consumes too much time to end the process and it wastes resources in comparison of any automated system.

## 1.4 OBJECTIVE

The proposed system is Automated Attendance System using face recognition overcomes the problems of the existing systems as mentioned previously. It mainly identifies Facial Recognition to mark student's attendance into the database. This will not be time consuming, and it will be human error free also there is not any chance for proxy attendance for students and lecture time will be only for teaching and no wasting time for taking attendance or calling students names. Accuracy of face recognition attendance system is great, and it is fast as well. Our goal is to take advantage of recognition technologies to improve the educational process as much as possible.

## 1.5 DOCUMENTION ORGANIZTION

In this research project, we are using face recognition using deep learning to help the educational systems to take the attendance by automated system with human error free. The subsequence chapters are organized as follows: Chapter 1 discusses introduction, problem definition, existing attendance systems, drawbacks of existing attendance systems and objective about project. Chapter 2 is discussing previous related work.
Chapter 3 discusses system analysis. Chapter 4 explain the project implementation in detail. Chapter 5 contains conclusion and drawbacks of our project.

# 2 LITERATURE REVIEW

## 2.1 DEEP LEARNING

Deep Learning is subset of Machine Learning which is subset of Artificial Intelligence. We can define Artificial Intelligence as some techniques that simulate human behaviors. Machine Learning represents a set of algorithms trained on data that make mimicking humans possible. Deep Learning is a subfield of machine learning involved with algorithms inspired by the structure and function of the brain neurons called **artificial neural networks**. Deep learning algorithms collect data with a specified logical structure to reach valid results as humans. It achieves this by employing a multi-layered structure of algorithms known as neural networks. The function of deep learning appears when dealing with unstructured data, deep learning's capacity to process huge amounts of features makes it incredibly strong. Deep learning techniques may be not functional for less complicated tasks because they require massive amounts of data to be effective.

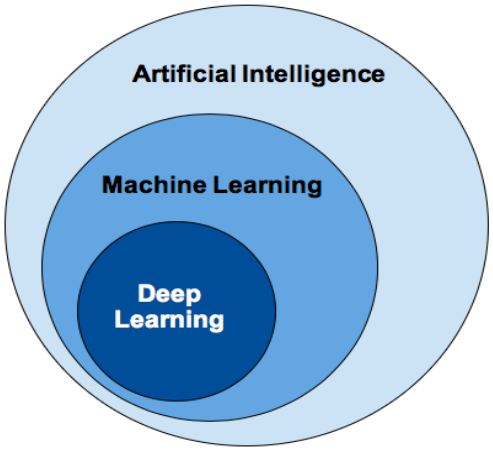



Figure 2-1: Concentric circles show relation between AI, ML & Deep Learning. [1]

### 2.1.1 Where is deep learning being used today? [2]

Deep learning has been applied to a variety of applications like computer vision, speech recognition, education, self-driving cars, industry, and many other fields.

#### 2.1.1.1 Self-driving cars

"Autonomous cars are no longer beholden to Hollywood sci-fi films" — Elon Musk, the founder of Tesla Inc. Self-driving cars are the most popular issues in science and technology research. Deep learning is one of the most important fields of artificial intelligence study right now. The development of self-driving cars has been aided by artificial intelligence (AI). Self-driving cats are becoming more popular in terms of market demand and economic value. Furthermore, businesses and scientific research organisations are investing in this sector. Google, Tesla, Apple, Nissan, Audi, General Motors, BMW, Ford, Honda, Toyota, Mercedes, Nvidia, and Volkswagen have participated in the research and development of self-driving cars [3].


**Figure 2-2.2: Self-driving cars of Google and Tesla: (a) Google's self-driving car; (b) Tesla's self-driving car. [4]**

#### 2.1.1.2 Medical Field

In medical science, there are many research sectors available, where computers play a key role. The most significant applications are health monitoring, medical informatics, bioinformatics, medical imaging, and so on. Deep learning in medical field enables doctors to accurately analyse any condition and effectively treat it, resulting in improved medical judgments.

1. **Drug Discovery**: Machine learning and deep learning algorithms have been implemented in several drug discovery processes such as peptide synthesis, structure-based virtual screening, ligand-based virtual screening, toxicity prediction, drug monitoring and release, pharmacophore modeling, quantitative structure–activity relationship, drug repositioning, poly pharmacology, and physiochemical activity. [5]

2. **Medical Imaging**: Heart disease, cancer, and brain tumors are diagnosed using medical imaging procedures such as MRI scans, CT scans, and ECG. As a result, deep learning assists doctors in better analyzing diseases and providing the best therapy for patients.

3. **Insurance Fraud**: Fraud detection is an important topic of research in healthcare systems due to the financial costs of fraud, which include investigative expenses, revenue losses, and reputational damage. Most businesses use Machine Learning and/or Deep Learning-based fraud detection algorithms to combat this. Healthcare systems function better when fraud detection methods are effective. Medical insurance fraud reports are analysed using deep learning. It can identify fraud claims that are likely to occur in the future using predictive analytics. Deep learning also aids the insurance sector in sending discounts and offers to its target patients. [6]
4. **Alzheimer's disease**: Alzheimer's disease is one of the major issues facing the medical profession. Alzheimer's disease is detected early using a deep learning approach.

5. **Genome**: Deep learning approach is used to understand a genome and assist patients in gaining understanding diseases that may impact them. In genetics and the insurance business, deep learning has a bright future.

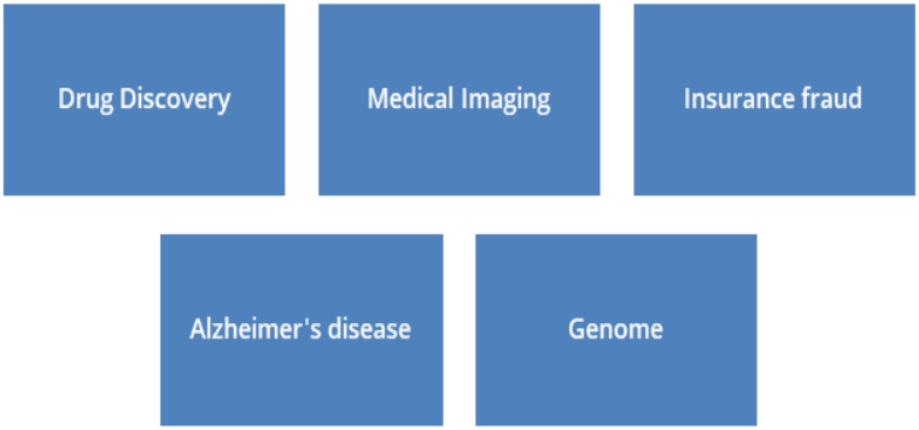



**Figure 2-3: Applications of deep learning in medical field [7]**

### 2.1.1.3 Speech Recognition

The ability of a computer or software to recognise words and phrases in spoken language and convert them to a machine-readable format is known as speech recognition. Today, there are several speech recognition apps available, including voice dialling, basic data entry, and speech-to-text. Its done by deep learning algorithms.

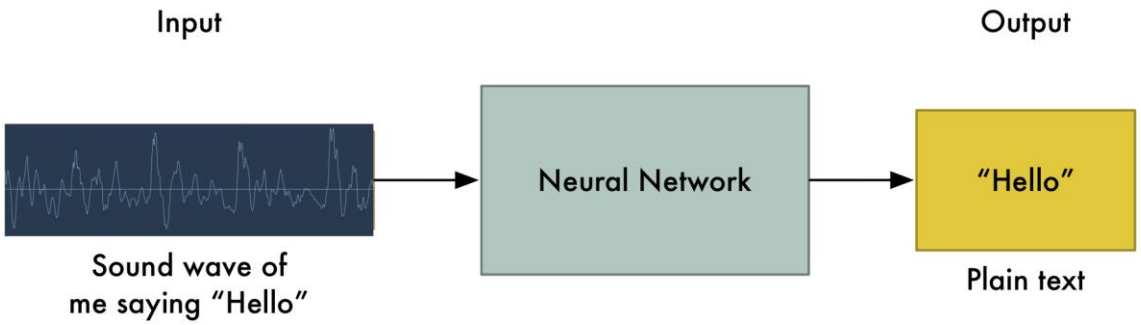



**Figure 2-4:Speech recognition using deep learning. [8]**

## Industrial Field

Several businesses in many different disciplines are looking at deep learning to capitalise on the insights in their data, enhance their competitiveness, open fresh business opportunities, or solve problems that were previously considered to be insurmountable. The vast size of the systems in which deep learning is employed, as well as the necessity to protect the privacy of the data.

### 2.1.2 Deep Learning Methods and Developments

#### 2.1.2.1 Convolutional Neural Network

A convolutional neural network (CNN) is a form of deep neural network that is frequently used in deep learning to assess visual images. When most people think of neural networks, they think of matrix multiplications, but that is not the case with CNN. It makes use of a technique known as Convolution. Convolution is a mathematical procedure performed on two functions that produces a third function that explains how the shape of one is modified by the other.

##### 2.1.2.1.1 CNN Architecture [9]

CNN image classifications take an input image, process it and classify it under certain categories (E.g., Dog, Cat, Tiger, and Lion). Computers see an input image as an array of pixels, and it depends on the image resolution. Based on the image resolution, it will see h x w x d (h = Height, w = Width, d = Dimension). E.g., An image of $6 * 6 * 3$ array of matrix of RGB (3 refers to RGB values) and an image of $4 * 4 * 1$ array of matrix of grayscale image.

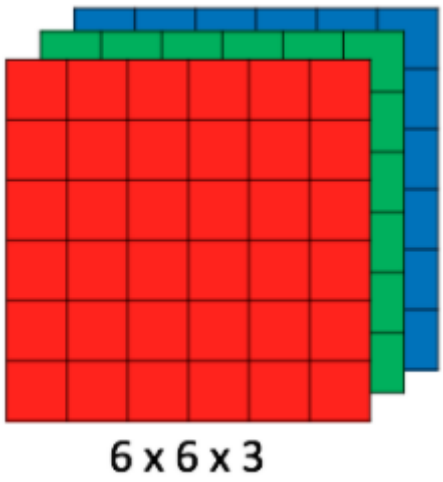



**Figure 2-5: Array of RGB Matrix**

Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernals), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values

between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values.

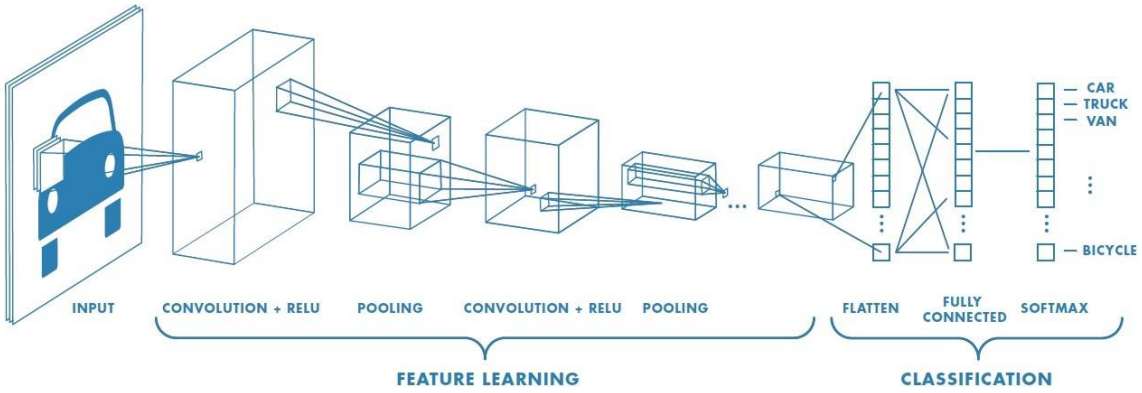



**Figure 2-6:Neural network with many convolutional layers**

### 2.1.2.1.2 Convolution Layer [9]

Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel.

- An image matrix (volume) of dimension **(h x w x d)**
- A filter **($f_h$ x $f_w$ x d)**
- Outputs a volume dimension **(h - $f_h$ + 1) x (w - $f_w$ + 1) x 1**

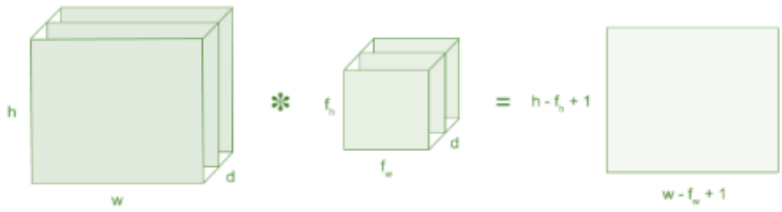



**Figure 2-7:Image matrix multiplies kernel or filter matrix.**

Consider a 5 x 5 whose image pixel values are 0, 1 and filter matrix 3 x 3 as shown in below.

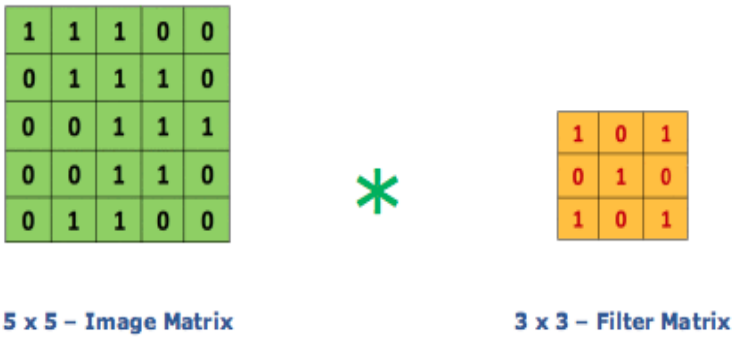



**Figure 2-8: Image matrix multiplies kernel or filter matrix**

Then the convolution of 5 x 5 image matrix multiplies with 3 x 3 filter matrix which is called “Feature Map” as output.

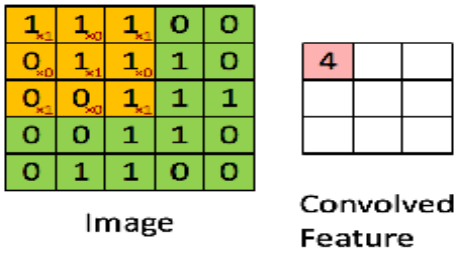



**Figure 2-9: 3*3 output matrix.**

Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters. The below example shows various convolution image after applying different types of filters (Kernels).

| Operation | Filter | Convolved Image |
|---|---|---|
| Identity | $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ |  |
| Edge detection | $\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$ |  |
| | $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ |  |
| | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur (normalized) | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur (approximation) | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |

**Figure 2-10: Some popular filters.**

- **Strides**

Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on. The below figure shows convolution would work with a stride of 2.

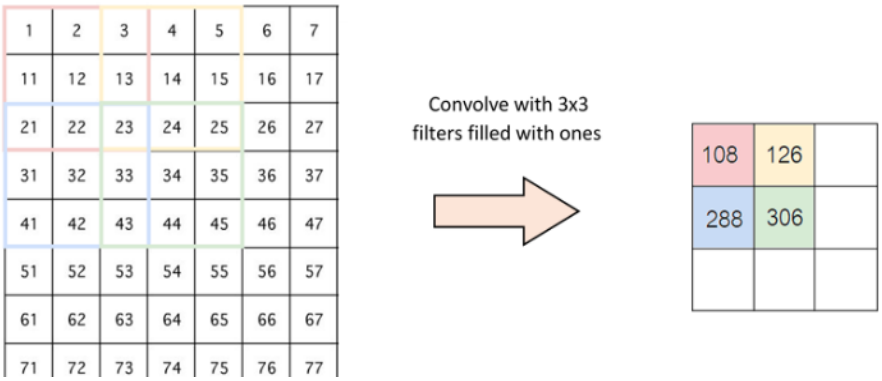



**Figure 2-11: Stride of two pixels**

- **Padding**

Sometimes filter does not fit perfectly fit the input image. We have two options:

1. Pad the picture with zeros (zero-padding) so that it fits.
2. Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image.

- **Non-Linearity**

ReLU for a non-linear operation. The output is $f(x) = \max(0, x)$.

Why ReLU is important: ReLU's purpose is to introduce non-linearity in our CNN. Since, the real-world data would want our CNN to learn would be non-negative linear values.

There are other nonlinear functions such as tanh or sigmoid that can also be used instead of ReLU. Most of the data scientists use ReLU since performance wise ReLU is better than the other two as shown in the figure below.

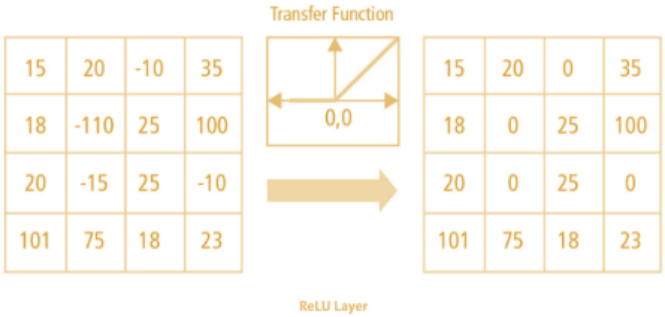



**Figure 2-12:ReLU operation.**

- **Pooling Layer**

Pooling layers section would reduce the number of parameters when the images are too large. Spatial pooling also called subsampling or down sampling which reduces the dimensionality of each map but retains important information. Spatial pooling can be of different types:

1. Max Pooling
2. Average Pooling
3. Sum Pooling

Max pooling figure below takes the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling [9].

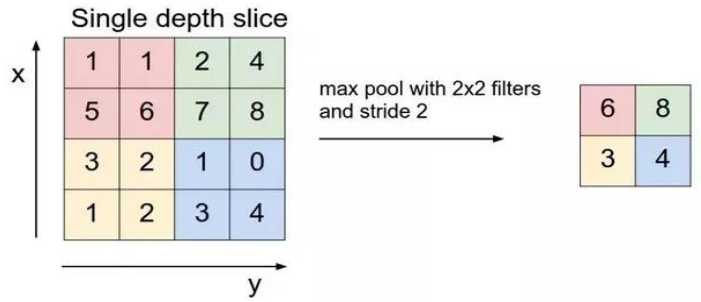



**Figure 2-13:Max pooling**

- **Fully Connected Layer**

The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like a neural network as shown in the figure below.

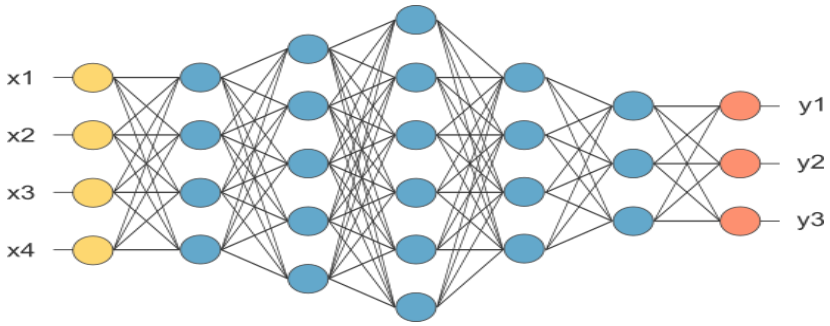



**Figure 2-14: After pooling layer, flattened as FC layer.**

The feature map matrix will be converted as vector (x1, x2, x3, --). With the fully connected layers, we combined these features together to create a model. Finally, we have an activation function such as SoftMax or sigmoid to classify the outputs as cat, dog, car, truck etc.

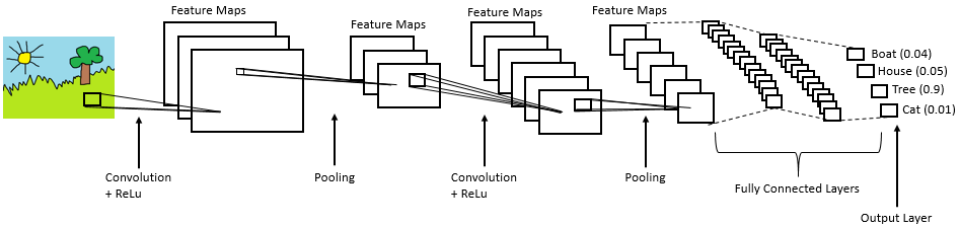



**Figure 2-15: Complete CNN architecture.**

- **Output Layer**

**SoftMax Function**

Softmax extends this idea into a multi-class world. That is, Softmax assigns decimal probabilities to each class in a multi-class problem. ... Softmax is implemented through a neural network layer just before the output layer. The Softmax layer must have the same number of nodes as the output layer.

**Output SoftMax**

Our output for the Softmax function is the ratio of the exponential of the parameter and the sum of exponential parameter. θ, on a high level is the sum of the score of each occurring element in the vector. In a generalized form we say that θ is the transpose of the weights matrix w, multiplied by the feature matrix x.

$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

#### 2.1.2.1.3 CNN applications [10]

1. Decoding Facial Recognition:

Face recognition is a technique for identifying or verifying an individual's identification by using their face. Face recognition has been used in a variety of applications, including an automatic classroom attendance management system. CNN divides facial recognition into the following key components. - Identifying each face in the image regardless of external circumstances such as light, perspective, posture, and so forth, focusing on each face. Identifying distinguishing characteristics to match a face with a name, all acquired data is

compared to data already in the database. A similar procedure is used for scene lab.

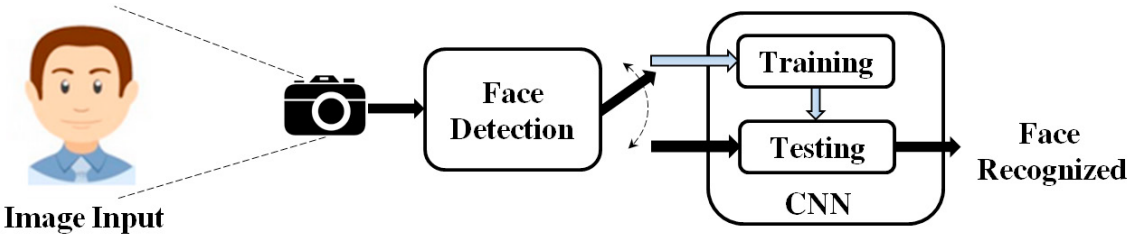



**Figure 2-16: Block diagram of proposed real-time face recognition system. [10]**

2. Advertising

CNNs have already made a significant influence in advertising by introducing programmatic purchase and data-driven targeted advertising.

3. Understanding Climate

CNNs have the potential to play a significant role in the battle against climate change, particularly in understanding why we are seeing such severe shifts and how we may experiment with mitigating the effect. It is claimed that the data in such natural history collections can also give better social and scientific insights; however, this would necessitate competent human resources, such as researchers, who can physically visit these sorts of archives.

### 2.1.2.2 Deep Neural Network

Deep Neural Networks (DNNs) are typically Feed Forward Networks (FFNNs) in which data flows from the input layer to the output layer without going backward[3] and the links between the layers are one way which is in the forward direction, and they never touch a node again.

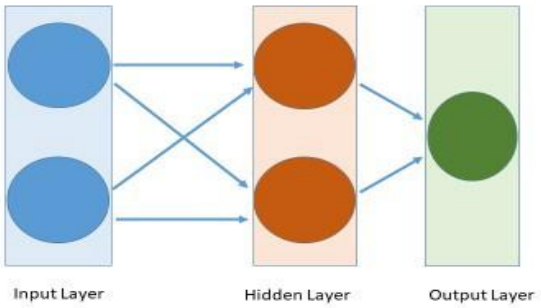



**Figure 2-17: Feed Forward Neural network.**

Back propagation is used to acquire the outputs from supervised learning using datasets of certain information depending on ‘what we want.' Assume you go to a restaurant and the chef gives you an idea of the components for your dish. FFNNs operate in the same manner that you will have the flavour of those specific components while eating but will forget what you have eaten shortly after finishing your meal. If the chef serves you the same dish with the same components again and you cannot recognise the ingredients, you will have to start again since you have no recollection of it. However, the human brain does not work like this.

### 2.1.2.3 Recurrent Neural Network

A Recurrent Neural Network (RNN) addresses this issue which is a FFNN with a time twist. This neural network is not stateless, has connections between passes and connections through time. They are a class of artificial neural network where connections between nodes form a directed graph along a sequence like features links from a layer to previous layers, allowing information to flow back into the previous parts of the network thus each model in the layers depends on past events, allowing information to persist.

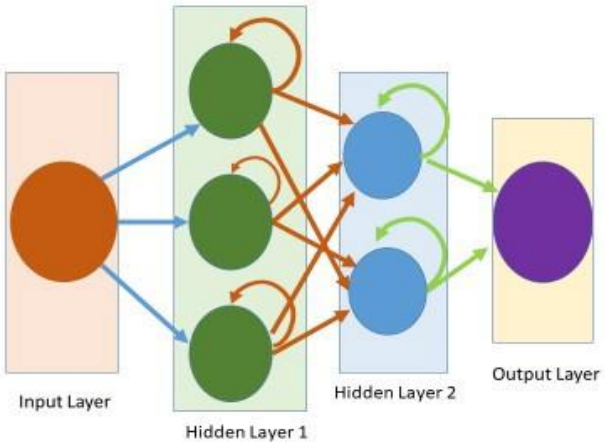



**Figure 2-18: Recurrent Neural network.**

## 2.2 LITERATURE SUVREY

Many techniques were developed to deal with disparity in pictures subject to variations in lighting, and these approaches were applied in object recognition systems as well as systems that were specialized to faces. Among the approaches are the following: -

**In [11],** A Counterpart Approach to Attendance and Feedback System using Machine Learning Techniques. In this paper, the idea of two technologies namely Student Attendance and Feedback system has been implemented with a machine learning approach. This system automatically detects the student performance and maintains the student's records like attendance and their feedback on the subjects like Science, English, etc. Therefore, the attendance of the student can be made available by recognizing the face. On recognizing, the attendance details, and details about the marks of the student is obtained as feedback.

**In [12],** Real time locating system using RFID for IOT, in the present attendance techniques are usually supplemented manually, because the number of university students is increasing within training institutes, the problem with getting hold of a hand

requires human effort to report and maintain student attendance. Consequently, human errors are common in this process. In recent years, there has been an increase in the number of applications based on RFID (Radio Frequency Identification) systems. RFID technology facilitates automatic wave identification using passive and active passive electronic labels with convenient readers. In this paper, an attempt has been made to address the problem of continuous attendance of lectures in developing countries and to find the location of special students using RFID technology. The implementation of RFID for attending student attendance as developed and deployed in this study can eliminate lost time during manual attendance gathering and an opportunity for education administrators to capture classroom statistics for sharing appropriate outcomes Attendance and for further managerial decisions**.**

**In [13],** An Evaluation of Face Recognition Algorithms and Accuracy based on Video in Unconstrained Factors There are three well-known algorithms that this paper will compare Eigenfaces, Fisherfaces, and LBPH by using a database that contains a face of persons with a variety of position and expression. According to the experiment results, LBPH got the highest accuracy on the possible external factors like light exposure, noise, and the video resolution. However, this algorithm has limitation due to the negative light exposure and high noise level more than the other statistical methods. The recognition accuracy also tested with three various video resolutions that are 720p, 480p, and 360p. The results show LBPH got the highest accuracy in 720p while the others got the highest accuracy in 360p video resolution. LBPH can give reliable recognition accuracy hence it uses a histogram similarity, but it was sensitive in some cases.

**In [14],** The main usage of this system is to generate the student attendance automatically. The main advantage of this system is every forty minutes the presence of the students will be identified. In the in between time the students are missing it will mark as the absent status. The attendance status immediately sends to the parents or concerned people. Mainly this system reduces the large number of human errors. This technology is one of the interesting research areas. This system is mainly used in airport for security purpose to identify the criminals easily.

**In [15],** This system shows automatic attendance using facial recognition which focuses on saving time and effort. Our system is design to overcome traditional system which is taken on pen and paper or file system. The main advantage of this system is it does not take long time and effort when huge number of students is present in a classroom. It takes snapshot of classroom then identify the faces and mark the attendance in excel sheet. Face recognition has many applications like it can be used in surveillance, security purpose, law enforcement etc. We have used LBPH algorithm for recognition which is simpler and more efficient than Eigen face and fisher face. LBPH algorithm is used for recognizing frontal and slightly tilt face up to 30 degrees. This algorithm still has some drawback but in future face recognition can be more accurate with the help of deep learning.

**In [16],** This system aims to build an effective class attendance system using face recognition techniques. The proposed system will be able to mark the attendance via face Id. It will detect faces via webcam and then recognize the faces. After recognition, it will mark the attendance of the recognized student and update the attendance record.

**In [17],** The system successfully simulated attendance recording both at lectures and examinations. The prototype successfully captured new fingerprints to be stored in the database; scanned fingerprints placed on the device sensor and compared them against those stored in the database successfully. The performance of the system was acceptable and would be considered for full implementation especially because of its short execution time and reports generation. Everyone who tested the system was pleased and interested in the product being developed for use in schools.

**In [18],** They have implemented an attendance management system for student's attendance. It helps to reduce time and effort, especially in the case of large number of students marked attendance. The whole system is implemented in Python programming language. Facial recognition techniques used for the purpose of the student attendance. And this record of student attendance can further be used mainly in exam related issues like who are attending the exams and who are not attending. On this project, there is some further works remained to do like installing the system in the classrooms. It can be constructed using a camera and computer.

**In [19],** The design and implementation of the attendance system in this paper have been described minutely. The developers have worked on the programs to make them quite easy to use. By sharing information timely, proctors, teachers and dorm supervisors now could work together to make sure that effective measures could be taken in time to prevent things from going worse. So, the system has achieved high-efficiency and synergetic. Since the system doesn't have too much hardware requirements, the cost of the system is extremely low. However, the performance of the system still needs to be improved, and some teachers have given valuable suggestions, dealing with all these problems and expanding the using of the system are the focus of next step.

Chapter Three

# 3 ANALYSIS AND DESIGN

## 3.1 USE CASE DIAGRAM

A use case is a software and system engineering term that describes how a user uses a system to accomplish a particular goal. It acts as a software modeling technique that defines the features to be implemented and the resolution of any errors that may be encountered.

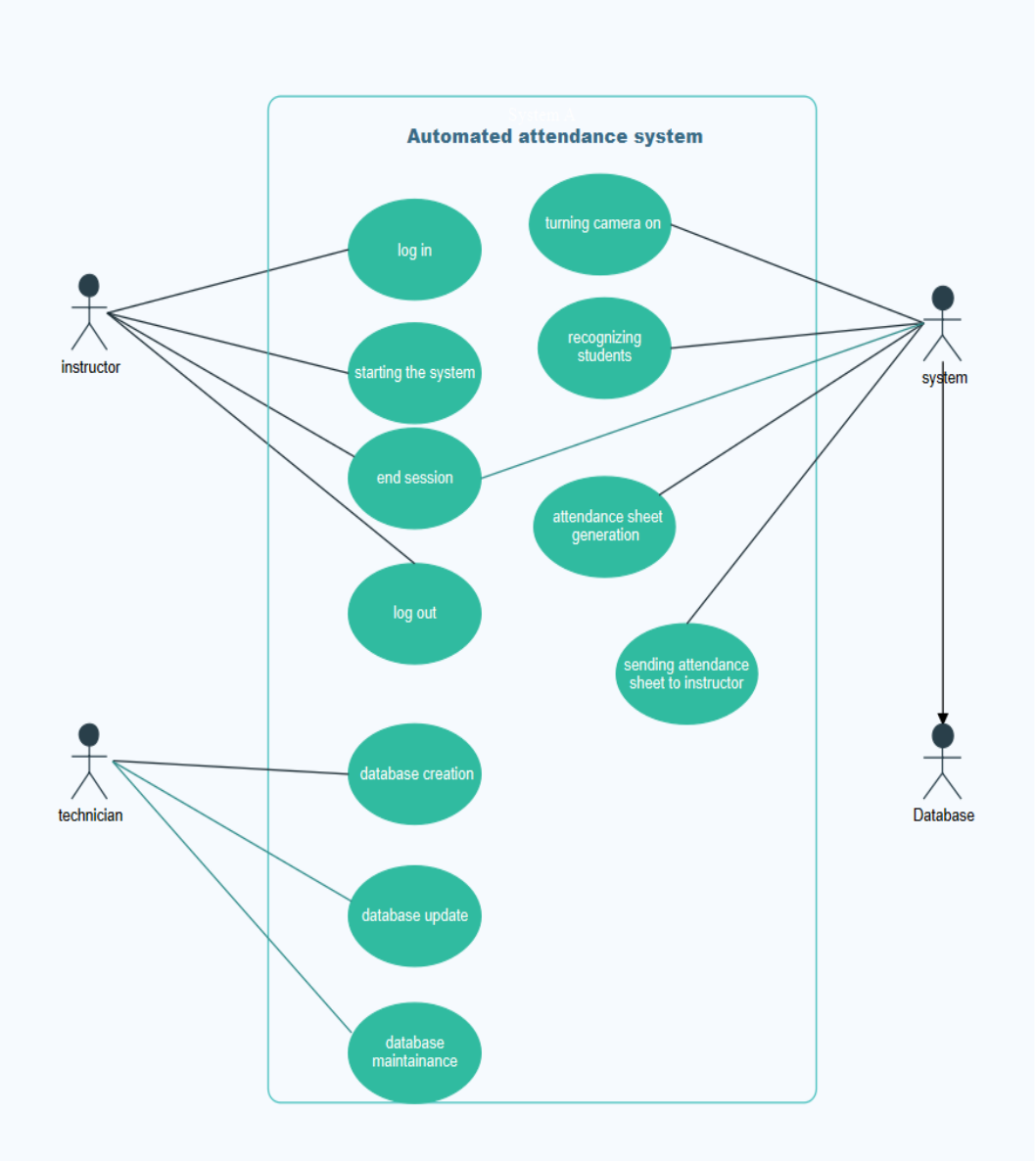



**Figure 3-1: Use Case Diagram.**

**Main Factors**

1) **Instructor**

   - Log in: opens the system and logs in
   - Starts the session: the instructor gives order to the system to start the session and do its work.

2) **Technician**

   - Creates the database: he is the responsible of creating the database that the system uses to generate data sheet.
   - Update and maintain database: he is the one who updates database each year and fix database if an error happened.

3) **System**

   - Checks authentication: digs into database to check instructor authentication.
   - Activate camera and recognize students: sends request to camera to start capturing frames and recognizes students in each frame.
   - Generates attendance sheet: it uses database to retrieve students required information, generates attendance sheet and sends it to the instructor via email.

## 3.2 CLASS DIAGRAM

Shows static structure of classifiers in a system Diagram provides basic notation for other structure diagrams prescribed by UML. Helpful for developers and other team members too Business Analysts can use class diagrams to model systems from business perspective.

A UML class diagram is made up of:

•A set of classes and.

•A set of relationships between classes.

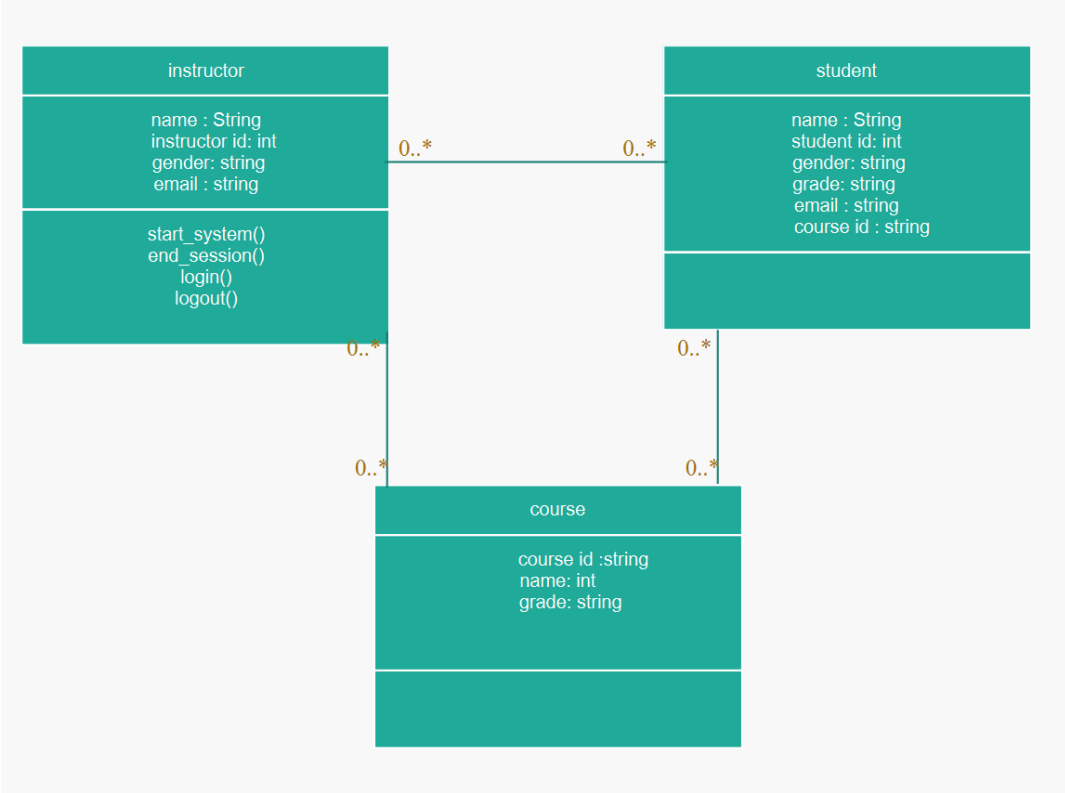



**Figure 3-2: Class Diagram**

1. instructor class: logs in and out, starts and ends the session.
2. student class: has a primary key (student id), name, grade, gender,email and course id.
3. Course class: has course id attribute as a primary key, name, and grade.

## 3.3 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

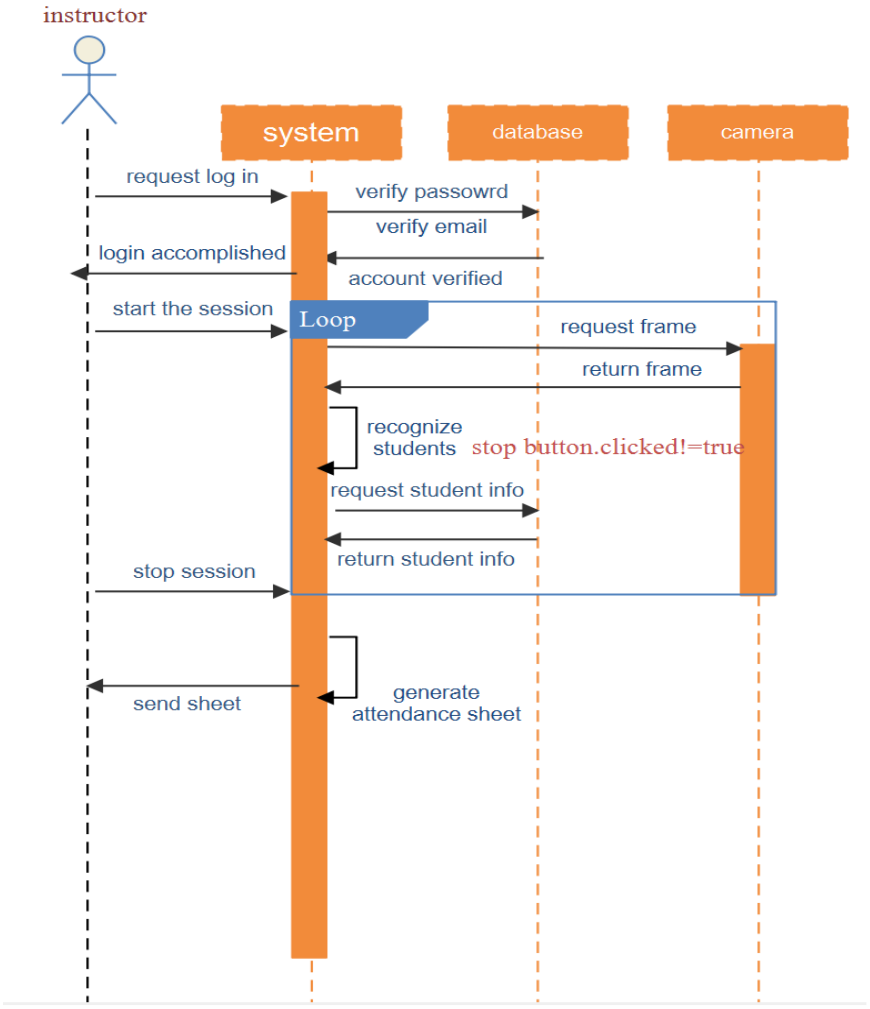



**Figure 3-3: Sequence Diagram.**

## 3.4 ENTITY RELATIONSHIP DIAGRAM (ERD)

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database.

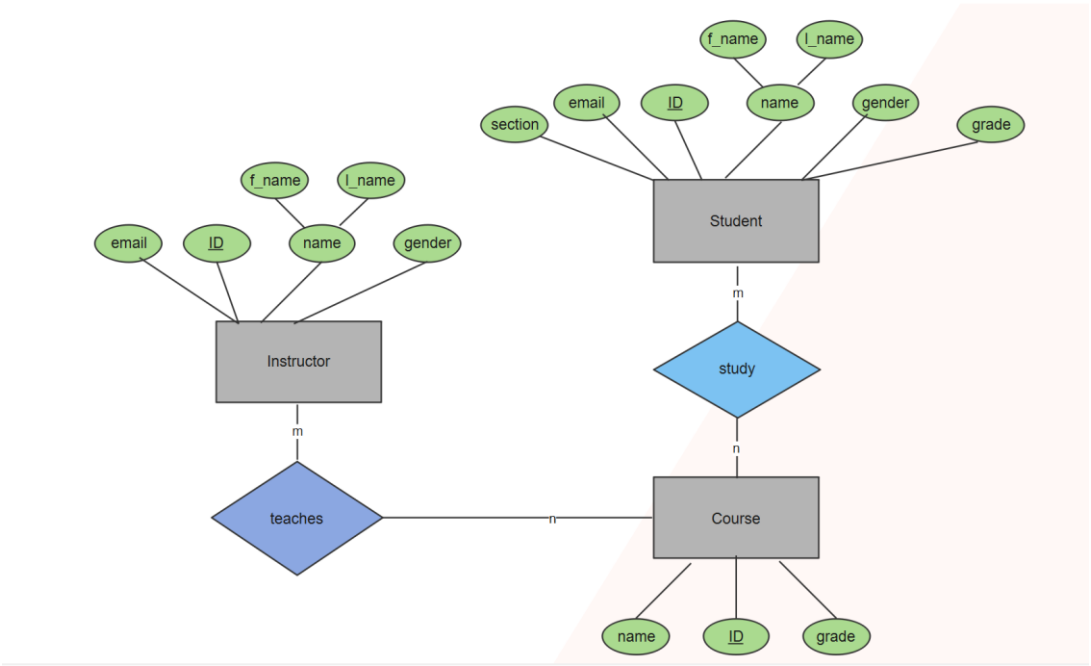



**Figure 3-4: ERD Diagram**

### a) Instructor

- M-N Relationship with course: a course can be taught by multiple instructors and an instructor can teach multiple courses.

### b) Student have

- M-N Relationship with Course: a student can have multiple courses and a course can be joined by many students.

### c) Course have

- M-N Relationship with instructor: a course can be taught by multiple instructors and an instructor can teach multiple courses.

- M-N Relationship with student: a student can have multiple courses and a course can be joined by many students.

## 3.5 CONTEXT DIAGRAM

Represent all external entities that may interact with system.

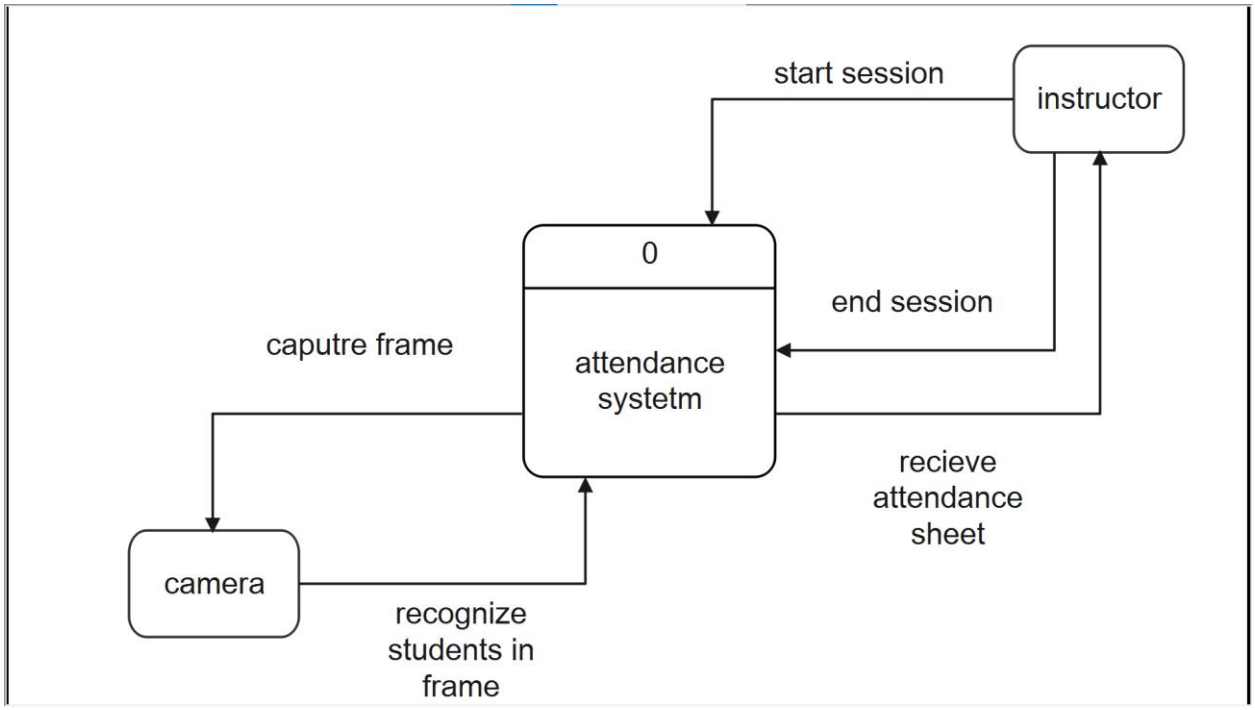



**Figure 3-5:context digram**

## 3.6 DATA FLOW DIAGRAM

Data flow diagram is a picture of the movement of data between external entities and the processes and data stores within a system.

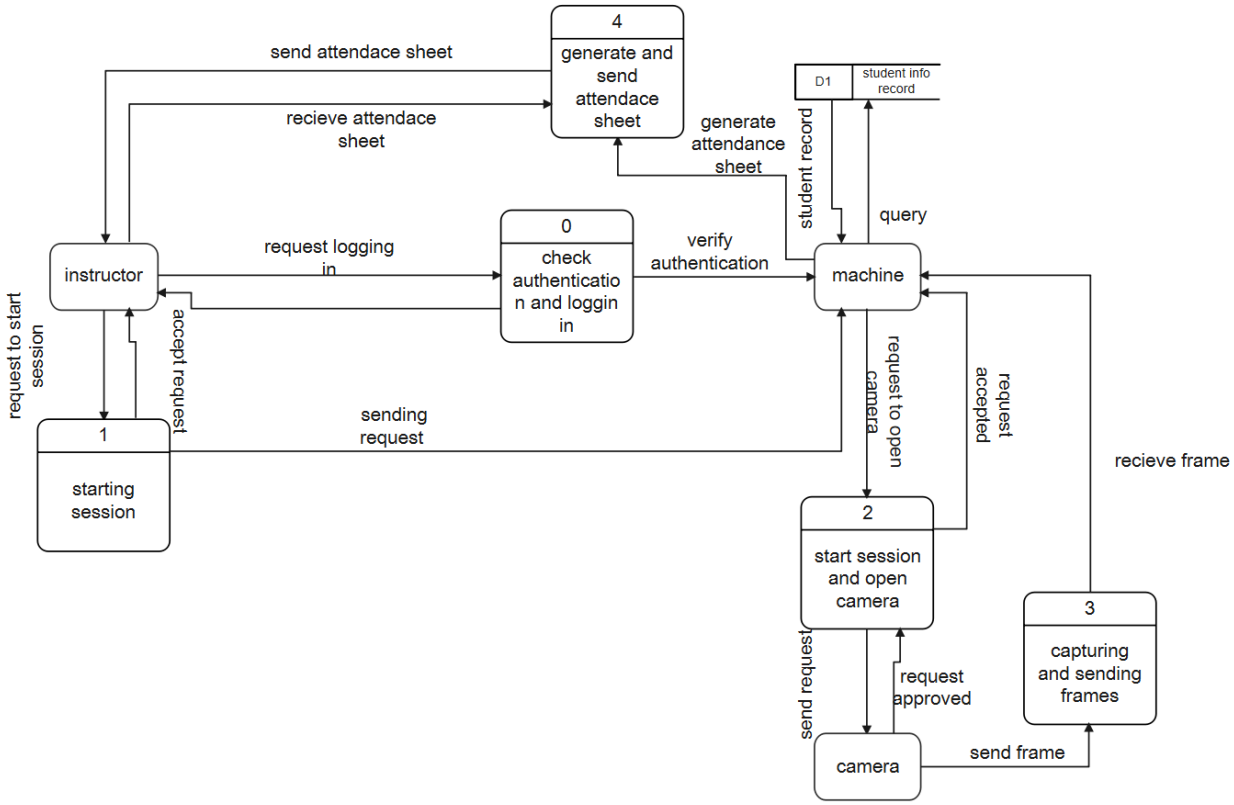



**Figure 3-6: Data Flow Diagram.**

## 3.7 PROJECT SCHEDULE ALLOCATION:

The project schedule is the core of the project plan. The project schedule is a calendar that links the tasks to be done with the resources that will do them. Schedules are used to communicate final deadlines and, in some cases, to determine resource needs. They are also used as a kind of checklist to make sure that every task necessary is performed. If a task is on the schedule.

| ID | Task Name | assignee | Start | Finish | Duration |
|---|---|---|---|---|---|
| 1 | search | team | 21/9/2020 | 3/10/2020 | 13.0 d. |
| 2 | analysis | team | 9/10/2020 | 15/10/2020 | 7.0 d. |
| 3 | opencv course | hadeer-norhan-abdallah | 20/10/2020 | 22/10/2020 | 3.0 d. |
| 4 | deep learning course | hadeer-honey-norhan | 28/10/2020 | 26/11/2020 | 30.0 d. |
| 5 | pyqt 5 course | nagham | 28/10/2020 | 26/11/2020 | 30.0 d. |
| 6 | project design | nagham | 11/12/2020 | 24/12/2020 | 14.0 d. |
| 7 | project implementation | honey-norhan-hadeer-abdallah | 12/1/2021 | 12/4/2021 | 91.0 d. |
| 8 | project testing | team | 20/5/2021 | 3/6/2021 | 15.0 d. |
| 9 | documentation | honey-hadeer-nagham-norhan | 10/6/2021 | 24/6/2021 | 15.0 d. |
| 10 | presentation | nagham | 23/6/2021 | 25/6/2021 | 3.0 d. |

**Figure 0-1: PROJECT SCHEDULE Table.**

*Chapter Four*

# 4 METHODLOGY

At first, we started to install the required libraries using commend prompt.

1- Install Keras:

Keras is a powerful and easy-to-use free open-source Python library for developing and evaluating deep learning models.

Figure: commend to install keras.

```
C:\Users\alshrooq>pip install keras
```

2- Keras depends on the following python libraries:

- Numpy
- Pandas
- Scikit-learn
- Matplotlib
- Scipy

So, we installed them using pip commend in cmd.

After that we collected our dataset and devided them into classes each class hold the name of the person inside the class

Now after we setup our environment we divided our system into six stages.

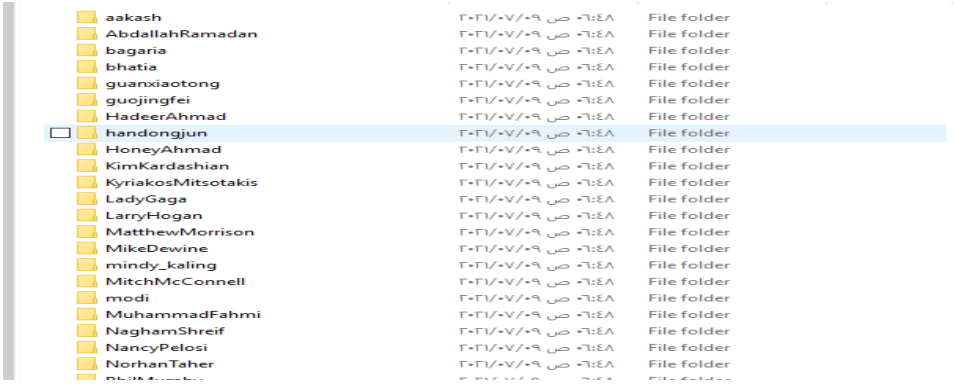



**Figure 4-1:classes of dataset**

## 4.1 EXTRACT EMBEDDINGS

This file is responsible for using a deep learning feature extractor to generate a 128-D vector describing a face. All faces in our dataset will be passed through the neural network to generate embeddings. The embedding is a generic representation for anybody's face. Unlike other face representations, this embedding has the nice property that a larger distance between two face embeddings means that the faces are likely not of the same person. This property makes clustering, similarity detection, and classification tasks easier than other face recognition techniques where the Euclidean distance between features is not meaningful.

### 4.1.1 Load Face Detector

In our project we used OpenCV's deep learning face detector it is based on the Single Shot Detector (SSD) framework with a ResNet base network.

When using OpenCV's deep neural network module with Caffe models, you will need two sets of files:

- The .prototxt file(s) which define the model architecture (i.e., the layers themselves)
- The .caffemodel file which contains the weights for the actual layers

Both files are required when using models trained using Caffe for deep learning. We use **cv2.dnn.readNetFromCaffe** to Read a network model stored in Caffe model in memory.

```
# load serialized face detector
print("Loading Face Detector...")
#.prototxt file which defines model architecture
protoPath = "face_detection_model/deploy.prototxt"

#.caffemodel file which contains the weights for the actual layers
modelPath = "face_detection_model/res10_300x300_ssd_iter_140000.caffemodel"

#load our model in detector variable using cv2.dnn.readNetFromCaffe
detector = cv2.dnn.readNetFromCaffe(protoPath, modelPath)
```

**Figure 4-2: Load Face Detector.**

### 4.1.2 Load Face Recognizer

In our system we use OpenFace deep learning model to compute a 128-d embedding that quantifies the face itself.

OpenFace is a deep learning facial recognition model from a high-level perspective, OpenFace uses Torch, a scientific computing framework to do training offline, meaning it's only done once by OpenFace, and the user doesn't have to get their hands dirty training hundreds of thousands of images themselves. Those images are then thrown into a neural net for feature extraction using Google's FaceNet model. FaceNet relies on a triplet loss function to compute the accuracy of the neural net classifying a face and can cluster faces because of the resulting measurements on a hypersphere. This trained neural net is later used in the Python implementation after new images are run through dlib's face-detection model. Once the faces are normalized by OpenCV's Affine transformation, so all faces are oriented in the same direction, they are sent through the trained neural net in a single forward pass. This results in 128 facial embeddings used for classification for matching or can even be used in a clustering algorithm for similarity detection.

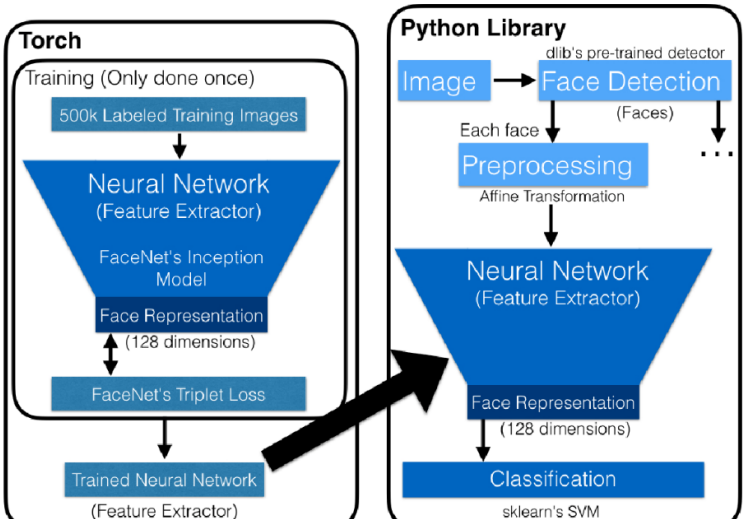



**Figure 4-3: OpenFace Diagram.**

During the training portion of the OpenFace pipeline, 500,000 images are passed through the neural net. These images are from two public datasets: CASIA-WebFace, which is comprised of 10,575 individuals for a total of 494,414 images and FaceScrub, which is made of 530 individuals with a total of 106,863 images. The point of training the neural net on all these images ahead of time is that it would not be possible on mobile or any other real-time scenario to train 500,000 images to retrieve the needed facial embeddings. OpenFace uses Google's FaceNet architecture for feature extraction and uses a triplet loss function to test how accurate the neural net classifies a face. It does this by training on three different images where one is a known face image called the anchor image, then another image of that same person has positive embeddings, while the last one is an image of a different person, which has negative embeddings. In this manner, the network can learn to quantify faces and return highly robust and discriminating embeddings suitable for face recognition.

In our system we reuse OpenFace pretrained model without going into details of the implementation and without having to explicitly train it! We use **cv2.dnn.readNetFromTorch** to Reads a network model stored in Torch7 framework's format.

```
# load serialized face embedding model
print("Loading Face Recognizer...")
embedder = cv2.dnn.readNetFromTorch("openface_nn4.small2.v1.t7")
```

**Figure 4-4:Load OpenFace Model.**

### 4.1.3 Grab The Paths

This portion of the code is responsible for grabbing the paths of the images located in every subdirectory in our dataset.

Then save the paths in imagePaths list.

```
27  print("Quantifying Faces...")
28  imagePaths = []
29  main_dir = 'dataset/train/'
30  main_dir_list = os.listdir(main_dir)
31  for name in main_dir_list:
32      name_dir = os.path.sep.join([main_dir, name])
33      name_dir_list = os.listdir(name_dir)
34      for image in name_dir_list:
35          imagePath = os.path.sep.join([main_dir, name, image])
36          imagePaths.append(imagePath)
```

**Figure 4-5: Grab images paths**

### 4.1.4 Extract Names

For every class in our dataset, we need to extract the name of each class. This is done by create loop that loop on every path in imagePaths list and extract the name of the person from the folder name.

```
48  for i in range(len(imagePaths)):
49
50      imagePath = imagePaths[i]
51      # extract the person name from the image path
52      name = imagePath.split(os.path.sep)[-2]
```

**Figure 4-6: Extract names.**

### 4.1.5 Detect Faces

Inside the loop, we used **cv2.dnn.blobFromImage** to extract every object in the image Then feed the objects to our face detector model to localize faces in the image.

```
imageBlob = cv2.dnn.blobFromImage(cv2.resize(image, (300, 300)),
                                  1.0, (300, 300),(104.0, 177.0, 123.0),
                                  swapRB=False, crop=False)

# apply OpenCV's deep learning-based face detector to localize faces in the input i
detector.setInput(imageBlob)
detections = detector.forward()
```

**Figure 4-7: Detect Faces.**

### 4.1.6 Draw Boxes Around Faces

We assume that there is only one face in the image, one by one, we iterate over all the faces detected in the images and extract their start and end points. Then, we extract the confidence of detection. If the algorithm is more than 50% confident that the detection is a face, then compute the (x, y)-coordinates of the bounding box for the face and show a rectangle around it.

```
68
69      if len(detections) > 0:
70          # we're making the assumption that each image has only ONE face,
71          #  so find the bounding box with the largest probability
72          i = np.argmax(detections[0, 0, :, 2])
73          confidence = detections[0, 0, i, 2]
74
75          # ensure that the detection with the largest probability also
76          # means our minimum probability test (thus helping filter out
77          # weak detections)
78
79          if confidence > 0.5:
80              # compute the (x, y)-coordinates of the bounding box for the face
81              box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
82              (startX, startY, endX, endY) = box.astype("int")
83
84              # extract the face ROI and grab the ROI dimensions
85
86              face = image[startY:endY, startX:endX]
87              (fH, fW) = face.shape[:2]
```

**Figure 4-8: Draw boxes around faces.**

### 4.1.7 Obtain The 128-d Quantification of The Face

We construct a blob for the face ROI, then pass the blob through our face embedding model to obtain the 128-d quantification of the face, the embedder takes the image in size 96*96 then we add the name of the person and the corresponding face embedding to their respective lists: knownNames and knownEmbeddings.

```
faceBlob = cv2.dnn.blobFromImage(face, 1.0 / 255, (96, 96), (0, 0, 0), swapRB=True, crop=False)
embedder.setInput(faceBlob)
vec = embedder.forward()

# add the name of the person + corresponding face
# embedding to their respective lists

knownNames.append(name)
knownEmbeddings.append(vec.flatten())
```

**Figure 4-9: Extract 123-d embedding.**

### 4.1.8 Save Embeddings

We arranged the embedding of the face and the corresponding names in dictionary and the save them in embeddings.joblib file.

```
110  data = {"embeddings": knownEmbeddings, "names": knownNames}
111
112  with open('output/embeddings.joblib', 'wb') as f:
113      dump(data, f)
114  f.close()
```

**Figure 4-10: Save embeddings.**

## 4.2 TRAIN THE MODEL

At this point we have extracted 128-d embeddings for each face, but we need to recognize a person based on these embeddings.

### 4.2.1 Model Architecture

So, we trained a neural network classifier with four hidden layers.The input shape of the classifier is 128-d embeddings, and we use ReLU activation function Then we add another two Dense layers with 64 neurons and ReLU as activation function The other two Dense layers with 32 neurons and ReLU as activation function The output layer produces the final result 25 predictions like the number of classes in our dataset.

```
12  model = Sequential([Dense(128, input_shape = (128,), activation = 'relu'),
13                      Dense(64, activation = 'relu', kernel_initializer = 'he_uniform'),
14                      Dense(64, activation = 'relu',  kernel_initializer = 'he_uniform'),
15                      Dense(32, activation = 'relu',  kernel_initializer = 'he_uniform'),
16                      Dense(32, activation = 'relu',  kernel_initializer = 'he_uniform'),
17                      Dense(27, activation = 'softmax')])
```

**Figure 4-11: Model Architecture.**

### 4.2.2 Model Summary

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 128)               16512
_________________________________________________________________
dense_1 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_4 (Dense)              (None, 32)                1056
_________________________________________________________________
dense_5 (Dense)              (None, 27)                891
=================================================================
Total params: 32,955
Trainable params: 32,955
Non-trainable params: 0
```

**Figure 4-12: Model summary.**

### 4.2.3 Compile Model

We must compile the model before training it. Since there are multiple classes. So, we use:

● Loss "Sparse categorical cross entropy" to compare the predicted label and true label and calculate the loss.

● Optimizer "Adam" which combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems. Adam is relatively easy to configure where the default configuration parameters do well on most problems. We used it with Learning rate 1e-3 and decay of learning rate divide the epochs which is 50.

● Metrics "accuracy" to judge the performance of the model according to the accuracy to calculate how often predictions equal labels.

```
19
20  model.compile(loss='sparse_categorical_crossentropy', optimizer = 'adam' ,metrics=['accuracy'])
21
```

**Figure 4-13: Compile model.**

### 4.2.4 Load Face Embeddings

In this portion of the code, we load the face embeddings from embeddings.joblib file. Then we used labelEncoder() to transform names labels to numerical labels.Then we transform embeddings to NumPy array.

```
print("[INFO] loading face embeddings...")
data = load("output/embeddings.joblib")

# encode the labels
print("[INFO] encoding labels...")
le = LabelEncoder()
labels = le.fit_transform(data["names"])

embeddings = data["embeddings"]
embeddings = np.array(embeddings)
```

**Figure 4-14:Transform names and embeddings.**

### 4.2.5 Training The Model

This step does not take much time It takes epochs, labels and embeddings to compare between them to calculate the accuracy and minimize the loss; this process is called empirical risk minimization.

```
h = model.fit(embeddings, labels, epochs = 50)
```

**Figure 4-15: Train the model.**

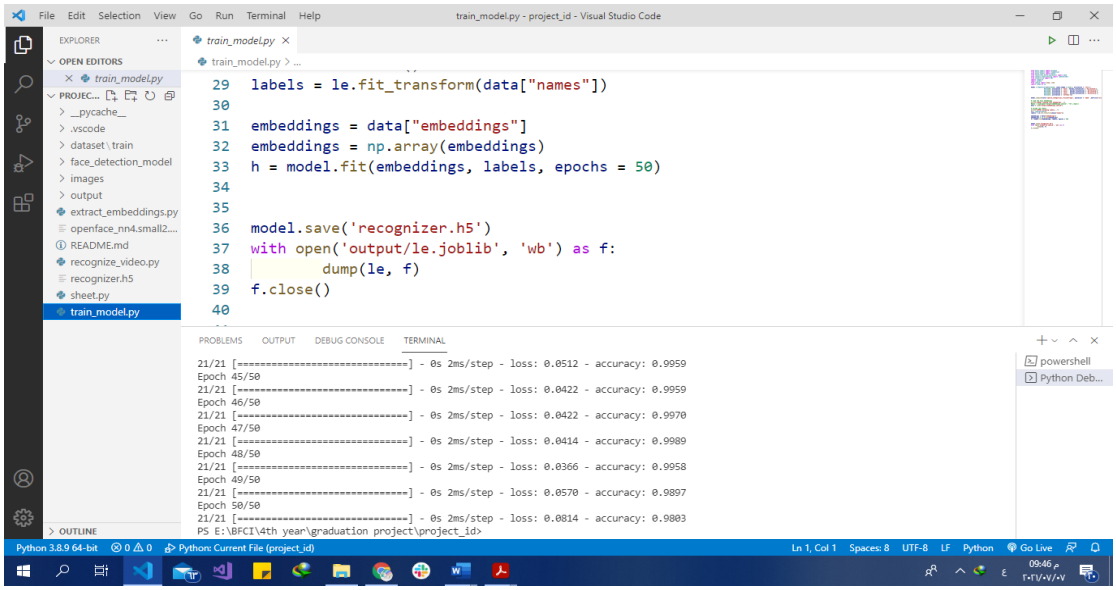



**Figure 4-16: Accuracy over 50 epochs.**

### 4.2.6 Learning Curves

Visualize training accuracy and loss function over 50 epochs.

```
36  plt.plot(h.h['accuracy'])
37  plt.title('model accuracy')
38  plt.ylabel('accuracy')
39  plt.xlabel('epoch')
40  plt.show()
41
42  plt.plot(h.h['loss'])
43  plt.title('model loss')
44  plt.ylabel('loss')
45  plt.xlabel('epoch')
46  plt.show()
```

**Figure 4-17:Visualize training accuracy and loss function over 50 epochs.**

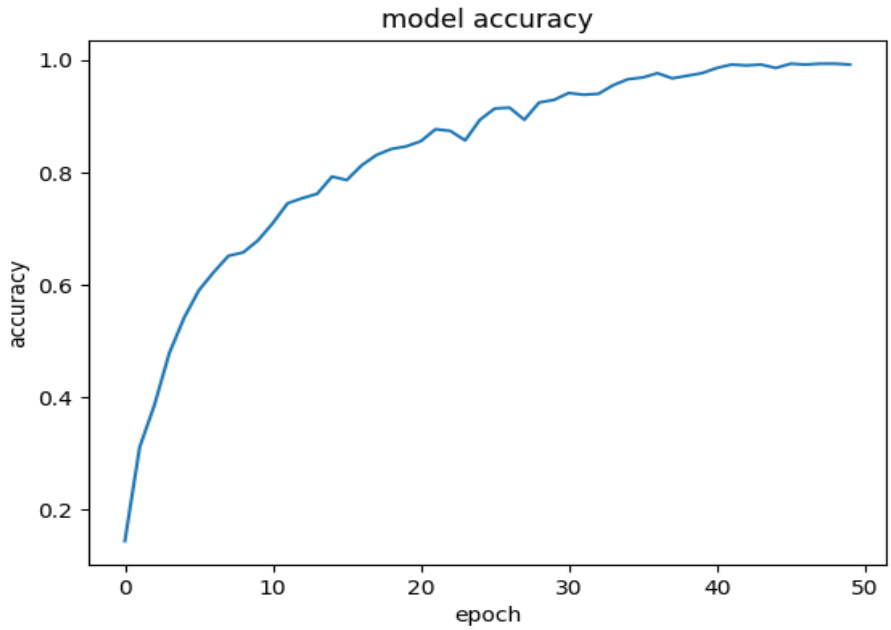



**Figure 4-18: Model accuracy.**

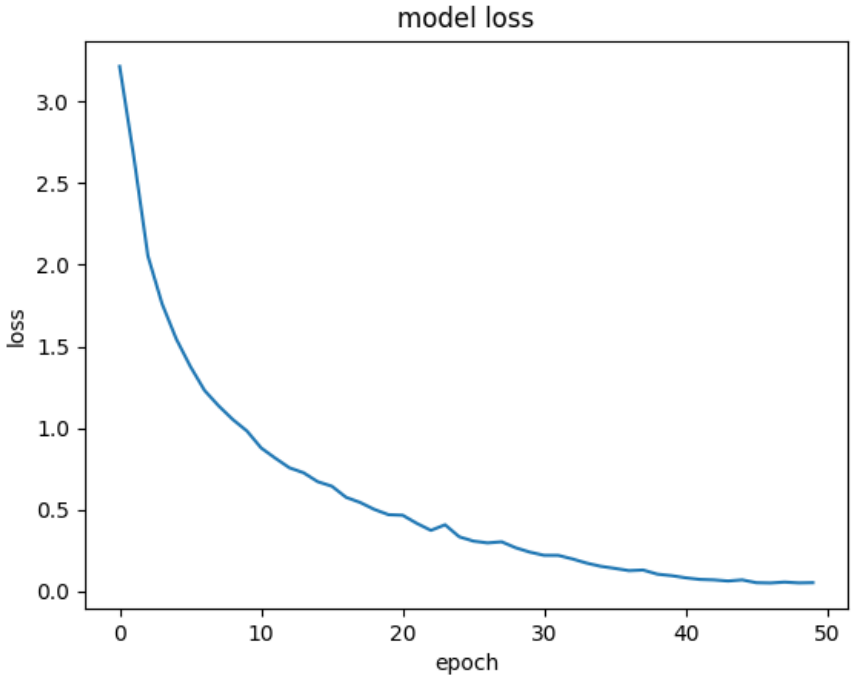



**Figure 4-19: Model loss.**

### 4.2.7 Save Model

After that we saved the model in recognizer.h5 file to reuse it in recognition.

```
48
49  model.save('recognizer.h5')
```

**Figure 4-20: Save Model.**

### 4.2.8 Save labelEncoder File

We use dump to write labels in **le.joblib** file.

```
50  with open('output/le.joblib', 'wb') as f:
51          dump(le, f)
52  f.close()
```

**Figure 4-21: Load labedlEncoder.**

## 4.3 FACE RECOGNITION

The actual face recognition done in this file 'recognize_vedio.py'

### 4.3.1 Image Recognize Function

This function takes the image as argument, resize it to 640*480 then take the height and the width and store them in variables we used cv2.dnn.blobFromImage to extract every object in the image. Then feed the objects to our face detector model to localize faces in the image.

Then we set the confidence to 85% to filter weak detections. After that we compute (x,y) coordinates of the bounding box and extract face ROI .All of the above we have done before in the extract embeddings files After that we resize the image to 96*96 to pass it to the embedder ,Then the recognizer model starts to make predictions about the image The next line will take the maximum predictions and output the name on the screen

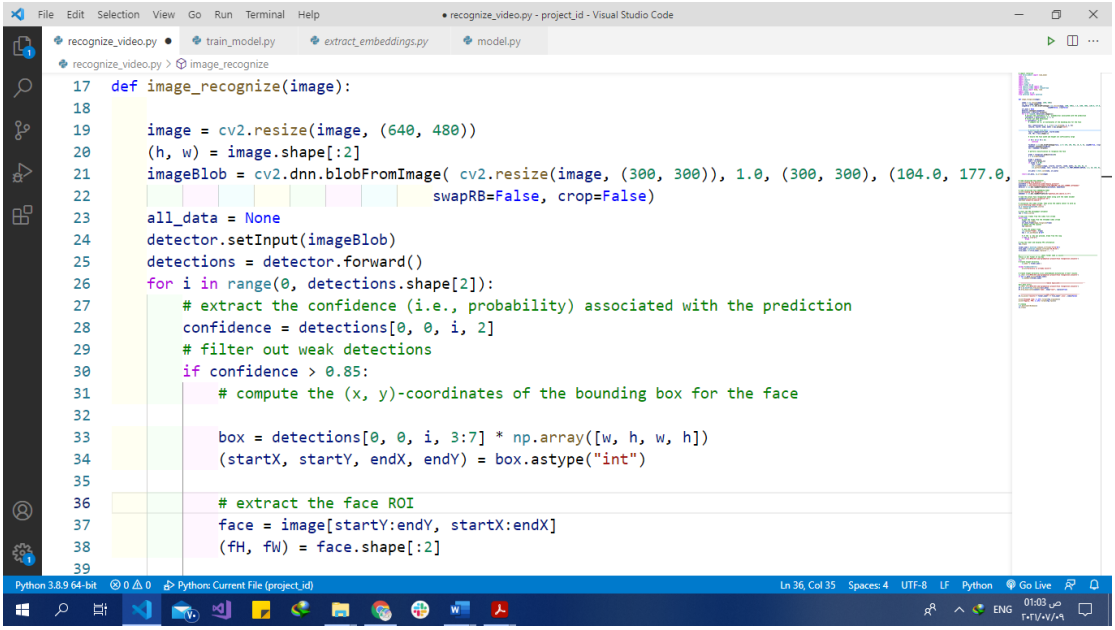



**Figure 4-22:Image recognize function.**

### 4.3.2 Video Stream

In this portion of the code we initialize the video stream and loop over frames from the video file stream then grab the frame from the threaded video

```
faceBlob = cv2.dnn.blobFromImage(face, 1.0 / 255, (96, 96), (0, 0, 0), swapRB=True, crop=False)
embedder.setInput(faceBlob)
vec = embedder.forward()
```

```
# perform classification to recognize the face

preds = recognizer.predict(vec)[0]
j = np.argmax(preds)
```

```
proba = preds[j]
name = le.classes_[j]
if proba >= 0.80:
    text = name
    cv2.rectangle(image, (startX, startY), (endX, endY), (0, 255, 0), 2)
    cv2.putText(image, text, (startX, startY), cv2.FONT_HERSHEY_DUPLEX, 1.2, (0, 255, 0), 2)
```

stream and pass the frame to image_recognize function

**Figure 4-23: Make predictions**

### 4.3.3 Result

The output of that file is video streaming in which the model could recognize faces in each frame.

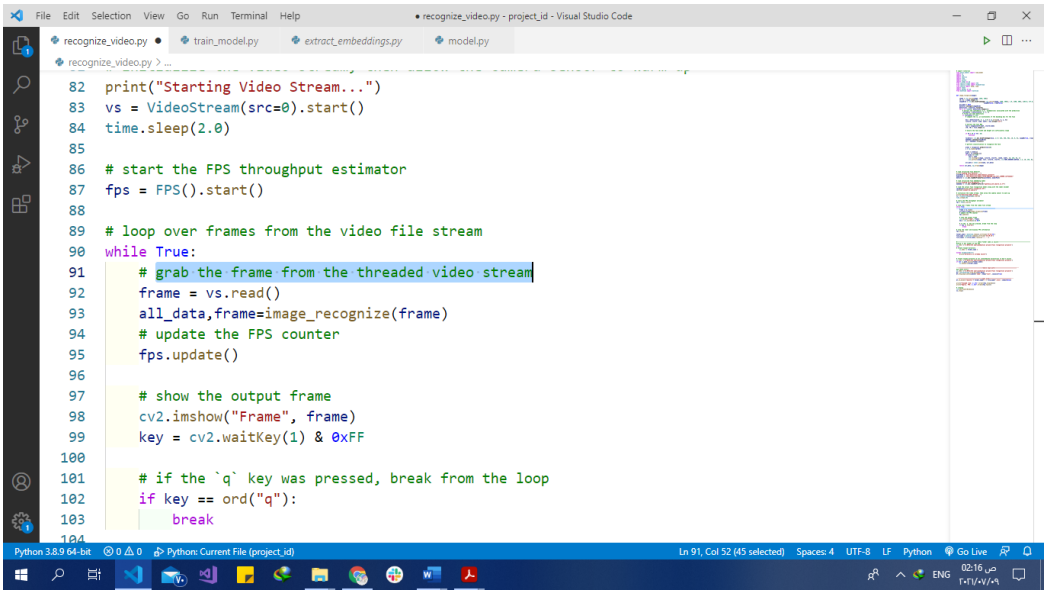



**Figure 4-24: Video stream**

### 4.3.3.1 Test 1

Input : Video stream of Norhan Taher

Output : succeeded in identifying Norhan taher

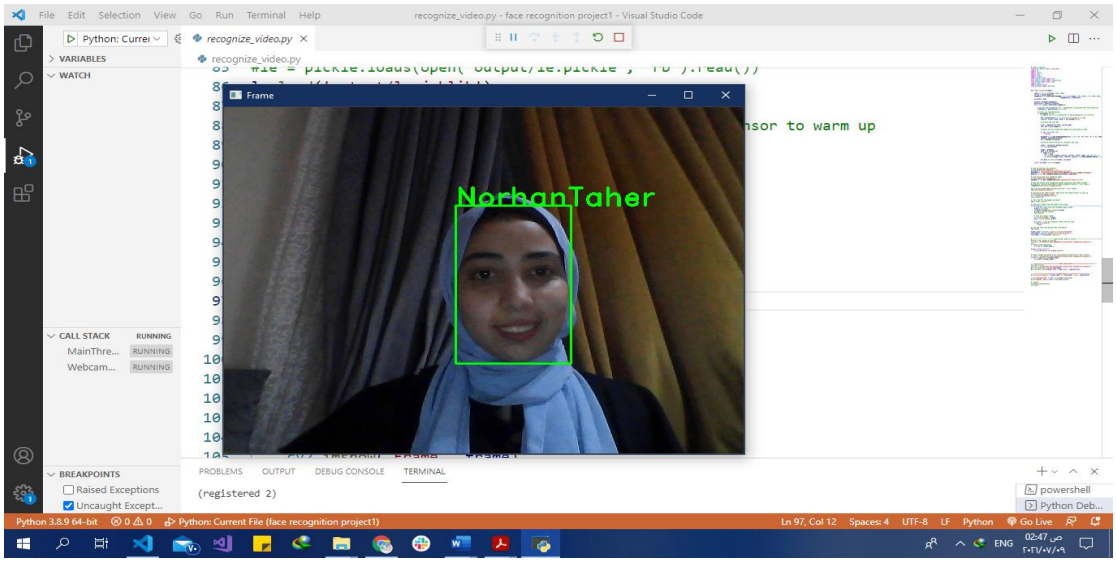



**Figure 4-25: Test 1**

### 4.3.3.2 Test 2

Input : Video stream of Norhan Taher holding image of Honey

Output : succeeded in identifying Norhan Taher and Honey Ahmad

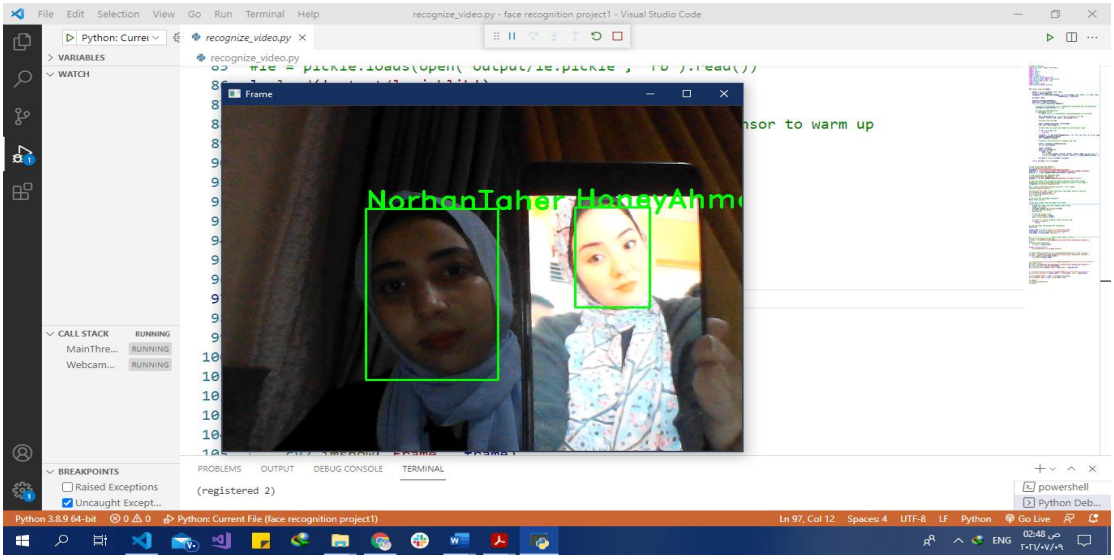



**Figure 4-26: Test 2**

### 4.3.3.2 Test 3

Input : Video stream of Nagham Sherif

Output : succeeded in identifying Nagham Sherif

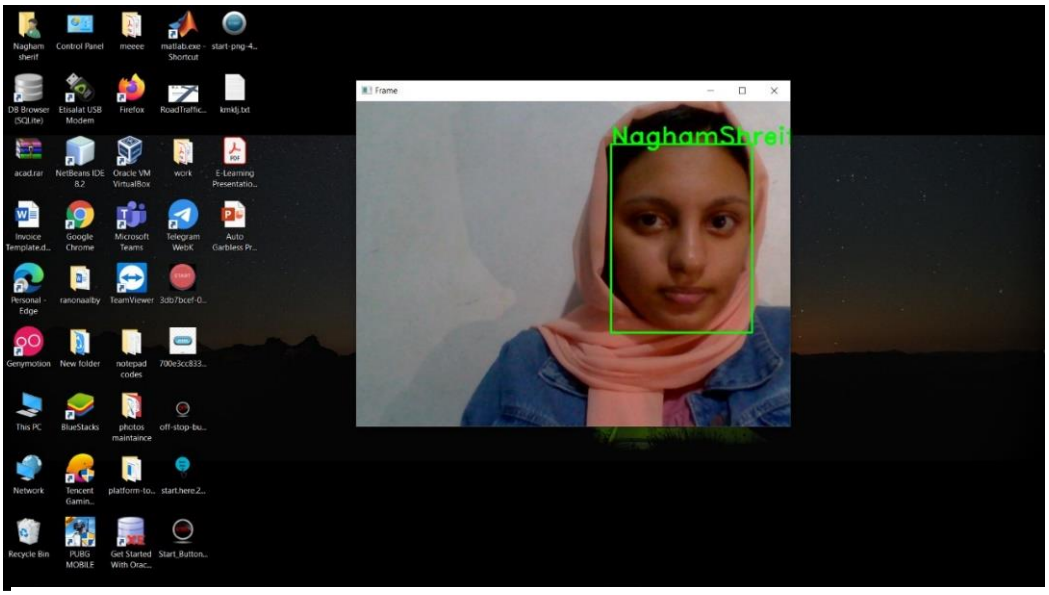



**Figure 4-25:Test 3**

### 4.3.3.3 Test 4

Input: Video stream of Hadeer Ahmad

Output: succeeded in identifying Hadeer Ahmad

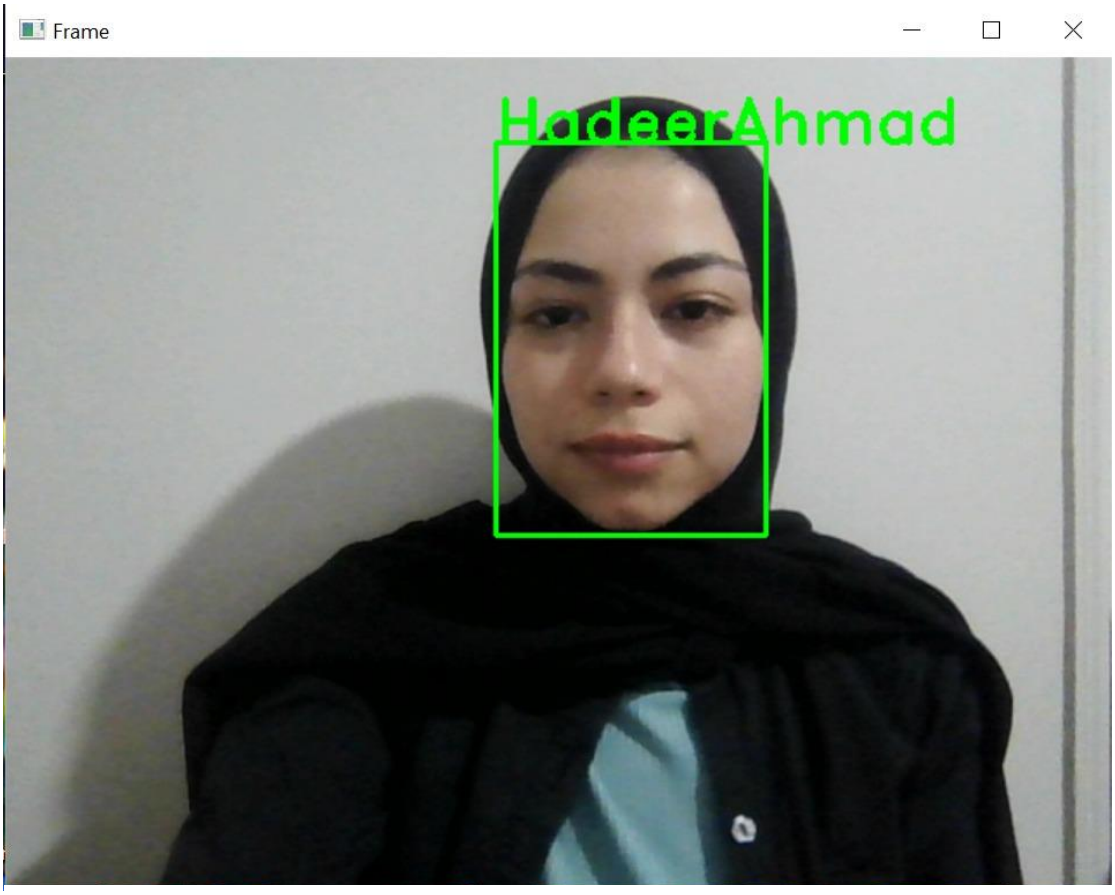



**Figure 4-26:Test 4**

## 4.4 CREATE DATABASE

We use SQLIte3 Database because it is very easy to use database engine. It is self-contained, serverless, zero-configuration and transactional. It is very fast and lightweight, and the entire database is stored in a single disk file. It is used in a lot of applications as internal data storage. The Python Standard Library includes a module called "sqlite3" intended for working with this database.

To navigate through database, we use DB browser (SQLite). Three Tables have been created (student Table, instructor Table, courses Table)

```
1  import sqlite3
2  con = sqlite3.connect('dbtest.db')
3  cursor = con.cursor()
4
```

**Figure 4-28:create database**

Table: students

Filter in any column

| | stu_id | f_name | l_name | email | grade | gender | section |
|---|---|---|---|---|---|---|---|
| | Filter | Filter | Filter | Filter | Filter | Filter | Filter |
| 1 | 1 | Honey | Ahmad | honeyahmed@fci.bu.edu.eg | Senior | Female | 5 |
| 2 | 2 | Norhan | Taher | norhantaher@fci.bu.edu.eg | Senior | Female | 5 |
| 3 | 3 | Nagham | Sherif | naghamsherif@fci.bu.edu.eg | Senior | Female | 5 |
| 4 | 4 | Hadeer | Ahmad | hadeerahmad@fci.bu.edu.eg | Senior | Female | 5 |
| 5 | 5 | Kyriakos | Mitsotakis | kyrikos@fci.bu.edu.eg | Senior | Male | 1 |
| 6 | 6 | Kim | Kardashian | kkardashian@fci.bu.edu.eg | Freshman | Male | 2 |
| 7 | 7 | Nancy | Pelosi | nancy@fci.bu.edu.eg | Freshman | Female | 3 |
| 8 | 8 | Matthew | Morrison | matthew@fci.bu.edu.eg | Junior | Male | 4 |
| 9 | 9 | Mike | Dewine | mike@fci.bu.edu.eg | Sophomore | Male | 3 |
| 10 | 10 | Muhammad | Fahmi | mfahmy@fci.bu.edu.eg | Sophomore | Male | 2 |
| 11 | 11 | Abdallah | Ramadan | abdallah@fci.bu.edu.eg | Senior | Male | 4 |
| 12 | 12 | Lady | Gaga | ladygaga@fci.bu.edu.eg | Freshman | Female | 1 |
| 13 | 13 | Steve | Harvey | steveharvey@fci.bu.edu.eg | Senior | Male | 4 |
| 14 | 14 | Phil | Murphy | philmurphy@fci.bu.edu.eg | Junior | Male | 3 |

**Figure 4-27:student table**

Table: courses

| | course_id | name | grade |
|---|---|---|---|
| | Filter | Filter | Filter |
| 1 | is_1234 | Information System Basics | Freshman |
| 2 | oop_1672 | Object Oriented Programming | Sophomore |
| 3 | ds_901 | Data Structures | Sophomore |
| 4 | em_234 | Embedded Systems | Junior |
| 5 | comp1_82 | Introduction to Compilers | Senior |
| 6 | dms_237 | Database Management Systems | Junior |
| 7 | net_898 | Networking | Senior |
| 8 | ml_102 | Machine Learning | Senior |
| 9 | aut_13 | Automata | Sophomore |
| 10 | fund_821 | Fundamental Programming Concepts | Sophomore |
| 11 | nn_89 | Neural Networks | Senior |
| 12 | ds_022 | Distributed Systems | Senior |
| 13 | ip_12 | Image Processing | Senior |
| 14 | hum_92 | Humanity | Freshman |

**Figure 4-29: courses table**

Table: instructors

| | inst_id | f_name | l_name | gender | email | password |
|---|---|---|---|---|---|---|
| | Filter | Filter | Filter | Filter | Filter | Filter |
| 1 | 1 | Mohammed | Ali | Male | drmohammedali764@gmail.com | 789lock789 |
| 2 | 2 | Yasser | Ahmed | Male | yasserahmed@fci.com | 20202021 |
| 3 | 3 | Mahmoud | Khalefa | Male | mahmoudkhalefa@fci.com | pw2020 |
| 4 | 4 | Amira | Eyad | Female | amiraeyad@fci.com | new2021 |
| 5 | 5 | Samir | Rady | Male | samirrady@fci.com | old2020 |
| 6 | 6 | Noha | Hesham | Female | nohahesham@fci.com | locklock |
| 7 | 7 | Yomna | Abdallah | Female | yomnaabdallah@fci.com | newnew |
| 8 | 8 | Mohamed | Salah | Male | mosalah@fci.com | oldold |

**Figure 4-30: instructors table**

the relationship between the three entities is many to many, so to connect them we made new table called student_course.

Table: studnet_course

| | st_id | c_id | ins_id | grade |
|---|---|---|---|---|
| | Filter | Filter | Filter | Filter |
| 1 | 1 | comp1_82 | 2 | Senior |
| 2 | 1 | net_898 | 3 | Senior |
| 3 | 1 | ml_102 | 5 | Senior |
| 4 | 1 | ds_022 | 1 | Senior |
| 5 | 1 | nn_89 | 8 | Senior |
| 6 | 1 | ip_12 | 4 | Senior |
| 7 | 2 | comp1_82 | 2 | Senior |
| 8 | 2 | ds_022 | 1 | Senior |
| 9 | 2 | ip_12 | 4 | Senior |
| 10 | 2 | net_898 | 3 | Senior |
| 11 | 2 | ml_102 | 5 | Senior |
| 12 | 2 | nn_89 | 8 | Senior |
| 13 | 3 | comp1_82 | 2 | Senior |
| 14 | 3 | net_898 | 3 | Senior |
| 15 | 3 | ml_102 | 5 | Senior |
| 16 | 3 | ds_022 | 1 | Senior |

**Figure 4-31: student_course table**

## 4.5 GUI OF THE SYSTEM

- Our system is designed to be desktop application programmed by python.
- We used QT designer to design the program interface
- The program started with splash screen for 7 seconds
- and the size of started screen is 780 * 600

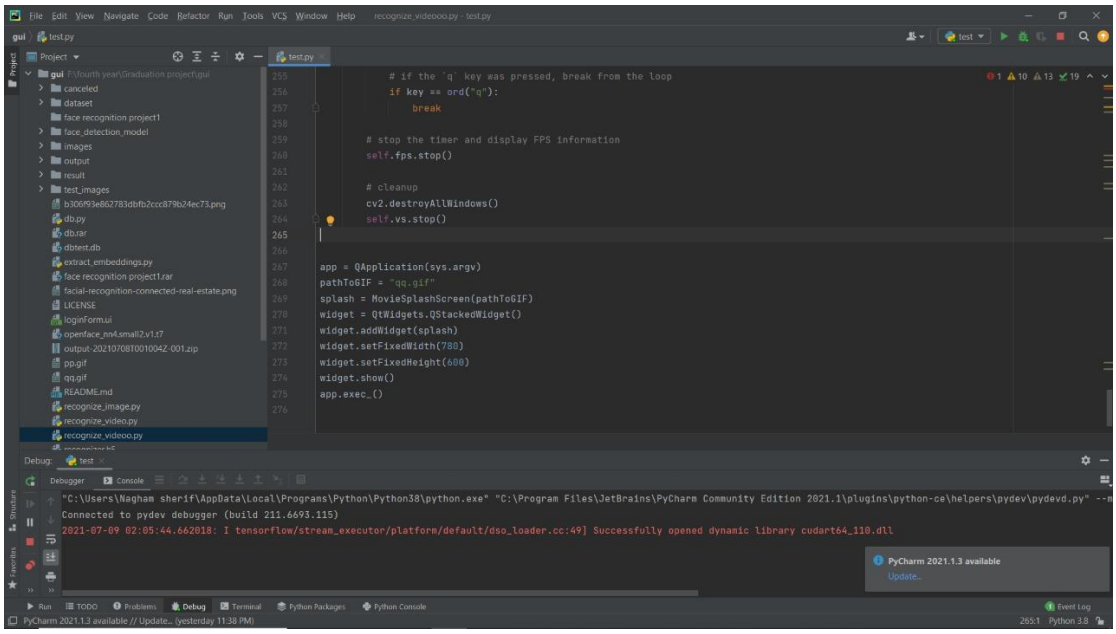



**Figure 4-32:Code to start the system**

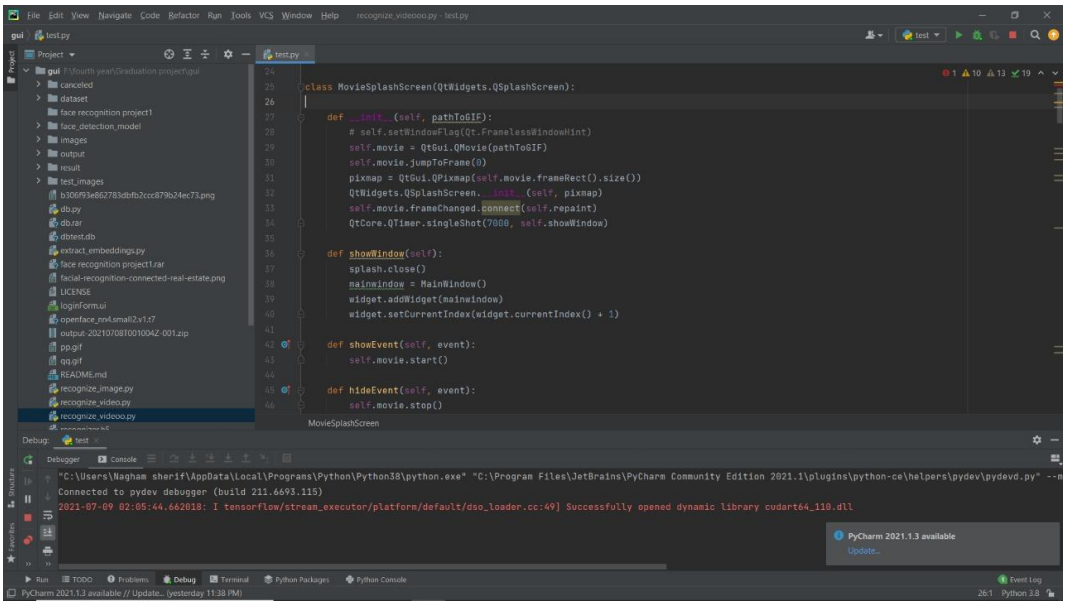



**Figure 4-33: Splash screen code**

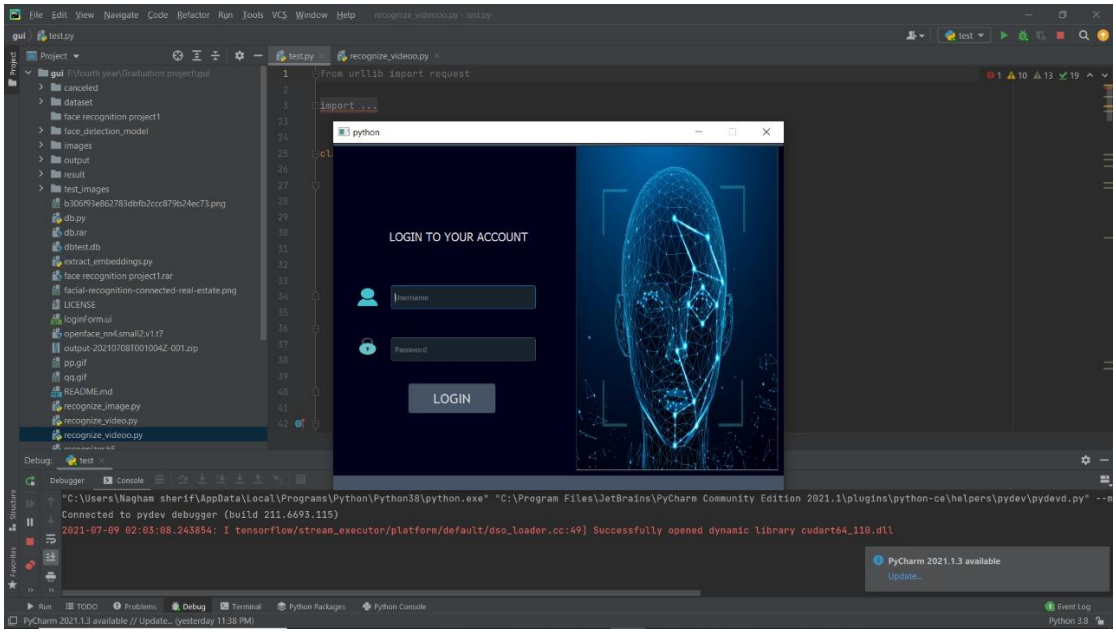
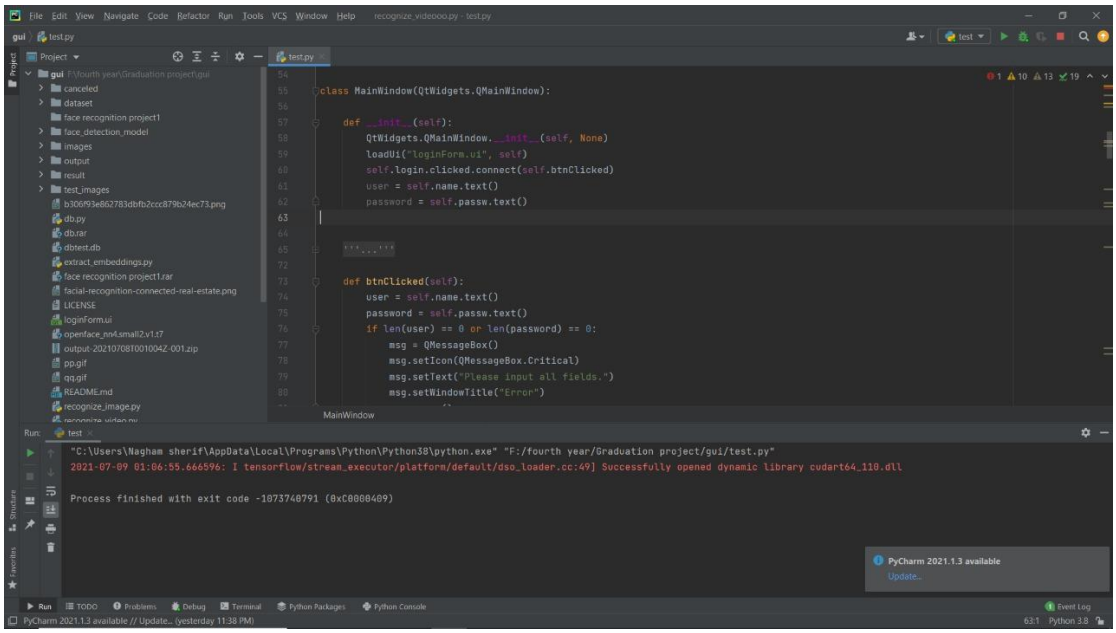



**Figure 4-34:Login form**



```
class MainWindow(QtWidgets.QMainWindow):

    def __init__(self):
        QtWidgets.QMainWindow.__init__(self, None)
        loadUi("loginForm.ui", self)
        self.login.clicked.connect(self.btnClicked)
        user = self.name.text()
        password = self.passw.text()

    '''...'''

    def btnClicked(self):
        user = self.name.text()
        password = self.passw.text()
        if len(user) == 0 or len(password) == 0:
            msg = QMessageBox()
            msg.setIcon(QMessageBox.Critical)
            msg.setText("Please input all fields.")
            msg.setWindowTitle("Error")
```

**Figure 4-35:Login form code**

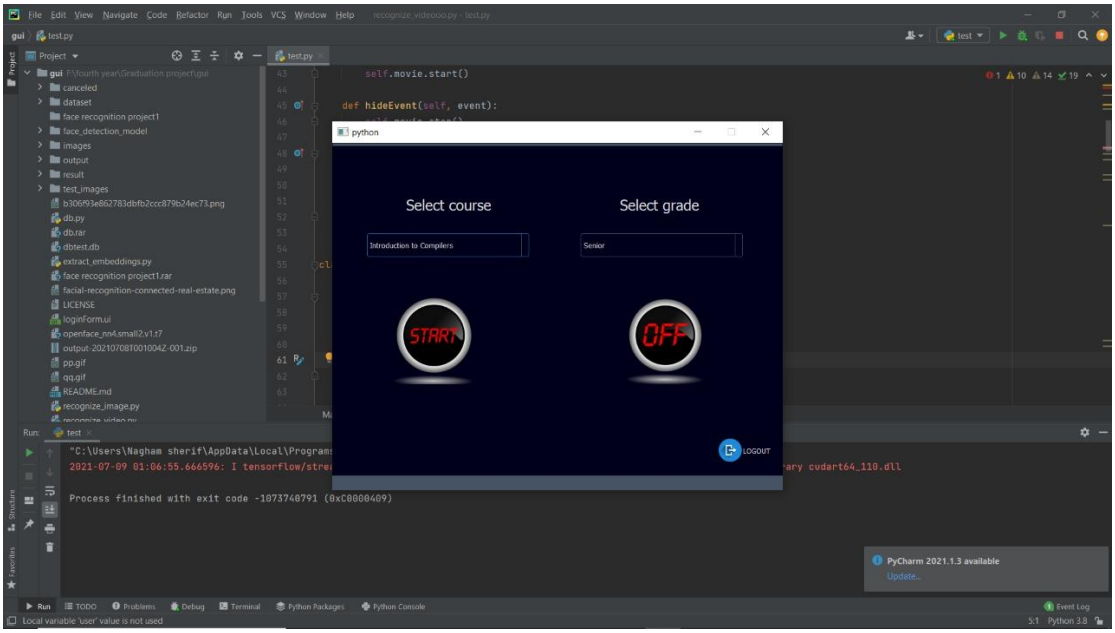



**Figure 4-36: Start and stop session form**

After creating the GUI, we connected our face recognition model with it.

But we changed the name of the classes in dataset with the corresponding id In the database, so that every recognized face is labeled with its id not name to make it easier to search in database with id


**Figure 4-37: Dataset with id**

## 4.6 GENERATE SHEET AND EMAIL IT

### 4.6.1 Sheet Module

```python
from datetime import datetime
from sklearn.preprocessing import LabelEncoder
import os
def save(folder_name, all_data=None):
    time = datetime.today().strftime('%Y-%m-%d-%H:%M:%S')
    #time = datetime.datetime.now()
    if all_data is None:
        all_data = {
            'name': [folder_name],
            'time': [time],}

    else:
            all_data['name'].append(folder_name)
            all_data['time'].append(time)

    return all_data
```

**Figure 4-38: Sheet module**

### 4.6.2 Connect With Database

```python
try:
    conn = sqlite3.connect('E:/BFCI/4th year/graduation project/project_id/db/dbtest.db')
    cursor = conn.cursor()
    columns_wanted = [ 'stu_id','f_name', 'l_name','section']
    qs = "SELECT {} FROM students WHERE  stu_id  =?".format(','.join(columns_wanted))
    for pid in all_data.get('person_id'):
        cursor.execute(qs,(pid,))
        values = cursor.fetchall()
        if len(values) > 0:
            values = values[0]
        else:
            values = ['    ']*len(columns_wanted)
        print('Values:', values,)
        for i, v in enumerate(values):
            all_data.setdefault(columns_wanted[i], []).append(v)
        print ("Opend database successfully")
except sqlite3.Error as error:
    print("Error while creating a sqlite table", error)
```

**Figure 4-39:Connect with database**

### 4.6.3 Generate Sheet

the generated sheet is named with the date of the day we stored the ids which have been extracted from the recognizer in array and iterate over them, remove the duplicates and grab the name and the section from the database then write them in excel sheet after that the system will send email to the instructor with the generated sheet

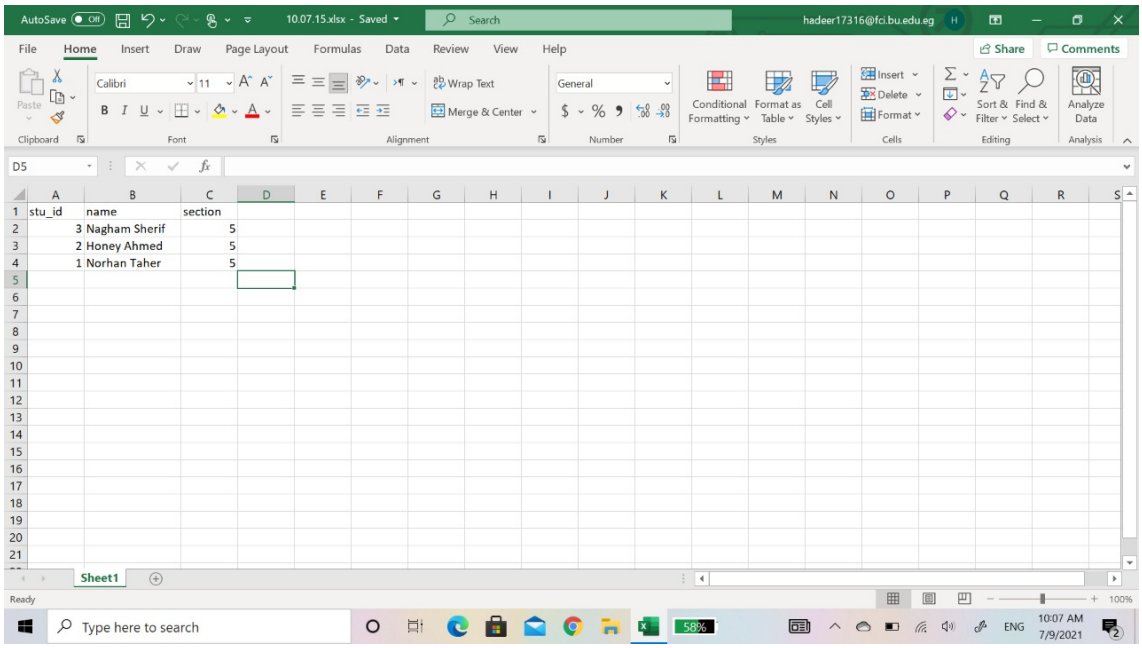



**Figure 4-40:Attendance sheet**

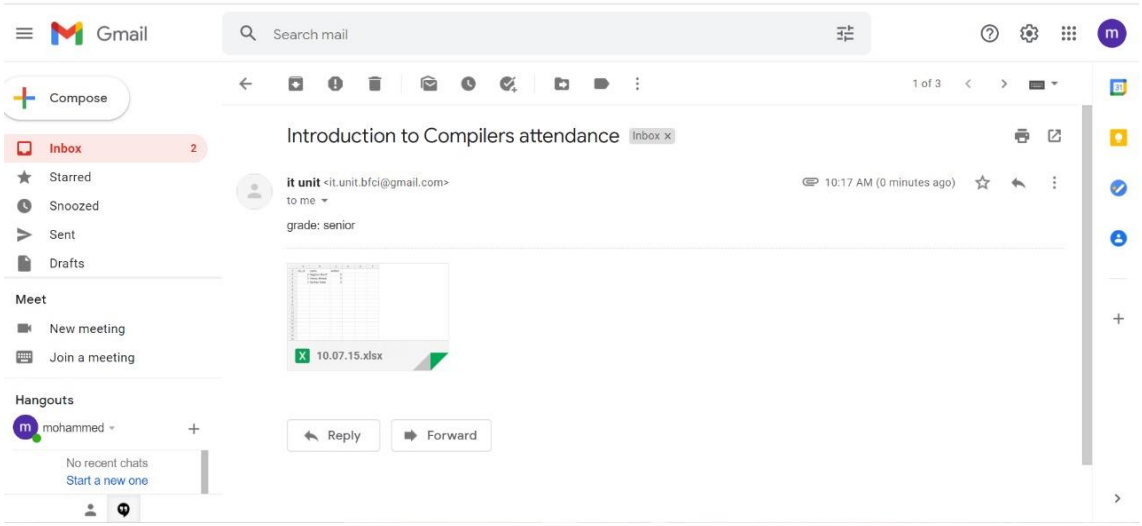



**Figure 4-41: Email the sheet**

*Chapter Five*

# 5 CONCLUSION

In this project we have met our main purpose and provided a reliable system that could facilitate the educational operation and save time and resources. First, our idea was to enhance authentication during the process of taking attendance in lectures. We have done that by using machine learning and training a deep learning model that can recognize people with an accuracy up to 96%. The model trains on the embeddings resulting from the OpenFace model. To train the model, we used a dataset of 25 person each have a range of images from 24 to 40 images. We also provided a user-friendly interactive interface using pyqt3 to make it easy to the instructor to deal with the system. After the instructor gives the order to the system to end the session, the system generates an attendance excel sheet and sends a copy to the instructor via email.

## 5.1 SYSTEM DRAWBACKS

- **Easy to be fooled:** It can recognize a 2d picture of the person which threaten the truthfulness of the system.
- **Generates false true outputs sometime:** After experimenting the model on different people from the dataset it turned out that it failed to correctly identify 2 of the 25 people, it means that the model needs to be trained on more images of those people.

## 5.2 FUTURE WORK

We will work to improve the accuracy of the recognition model and add more features to the system. Adding these features will require more search and time. Some of these features that we will work to add in the future are:

1. Display number of absence times for each of the recognized students.
2. Try to solve the 2d image problem to improve the truthfulness of the system and make it more reliable.
3. Calculate number of absents and number of students presents.

4. Trying to develop a feature that makes an administrator responsible of adding new students and instructors information into database

*Chapter Six*

# REFERENCES

[1] Dangare, S. (2019, May 16). Data processing using ML Supervised classification algorithm to find accuracy. Knoldus Blogs. https://blog.knoldus.com/data-processing-and-using-ml-supervised-classification-algorithm-and-find-accuracy/ W.-K. Chen, *Linear Networks and Systems*. Belmont, CA: Wadsworth, 1993, pp. 123–135.

[2] What Is Deep Learning? | How It Works, Techniques & Applications. (2020). MATLAB & Simulink. https://www.mathworks.com/discovery/deep-learning.htmlNi, J. (2020, April 11). A Survey on Theories and Applications for Self-Driving Cars Based on Deep Learning Methods. MDPI. https://www.mdpi.com/2076-3417/10/8/2749.

[3] How Machine Learning Algorithms Make Self-Driving Cars a Reality. (2018, October11).Intellias. https://www.intellias.com/how-machine-learning-algorithms-make-self-driving-cars-a-reality/

[4] Self-driving-cars. (2020,April).ResearchGate. https://www.researchgate.net/figure/Self-driving-cars-of-Google-and-Tesla-a-Googles-self-driving-car-b-Teslas_fig1_340682566E. P. Wigner, “Theory of travelling-wave optical laser,” *Phys. Rev.*, vol. 134, pp. A635–A646, Dec. 1965

[5][7] Top 5 applications of Deep Learning in healthcare | Artificial Intelligence |. (2018, April 16). Allerin. https://www.allerin.com/blog/top-5-applications-of-deep-learning-in-healthcare

[6] Detecting Insurance Fraud with Machine Learning. (2020, July 9). Plug and Play Tech Center. https://www.plugandplaytechcenter.com/resources/detecting-insurance-fraud-machine-learning/

[8] Geitgey, A. (2020, September 24). Machine Learning is Fun Part 6: How to do Speech Recognition with Deep Learning. Medium. https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a

[9] Wikipedia contributors. (2021, July 8). Convolutional neural network. Wikipedia. https://en.wikipedia.org/wiki/Convolutional_neural_network

[10] 7 Applications of Convolutional Neural Networks - FWS. (n.d.). Flatworld. Retrieved June 13, 2021, from https://www.flatworldsolutions.com/data-science/articles/7-applications-of-convolutional-neural-networks.php

[11] Attendance System with Face Recognition. (2019). International Journal of Engineering and Advanced Technology, 9(4), 574–577. https://doi.org/10.35940/ijeat.d7710.049420

[12] MAHATMA GANDHI INSTITUTE OF TECHNOLOGY. (2019, April). http://www.yukti.mgit.ac.in/CSE-Wasef%20_Report%20-Part1-2020.pdf. Department of Computer Science and Engineering.

[13] International Journal of Advanced Research in Computer and Communication Engineering. (2017, September). A Survey on Face Recognition based Students Attendance System. BINYAM TESFAHUN.

[14] Kanth, P. L., & Biswal, S. (2019). Attendance Marking System Using Face Recognition. Indian Journal of Science and Technology, 12(48), 1–4. https://doi.org/10.17485/ijst/2019/v12i48/145821

[15] Sharma, P. (2021). Attendance Monitoring System using Multiple Facial Recognition. International Journal for Research in Applied Science and Engineering Technology, 9(5), 2049–2056. https://doi.org/10.22214/ijraset.2021.34755

[16] Ghalib Al-Muhaidhri. (2019). Smart Attendance System using Face Recognition. International Journal of Engineering Research And, V8(12). https://doi.org/10.17577/ijertv8is120046

[17] American Journal of Computing Research Repository. (2014, September). Biometric-based Attendance System: LASU Epe Campus as Case Study. https://doi.org/10.12691/ajcrr-2-1-2

[18] R.S., D. (2020). Attendance Authentication System Using Face Recognition. Journal of Advanced Research in Dynamical and Control Systems, 12(SP4), 1235–1248. https://doi.org/10.5373/jardcs/v12sp4/20201599

[19] Pandia, H. (2017). DESIGN OF ATTENDANCE RECORDING SYSTEM FOR ELEMENTARY AND HIGH SCHOOL USING MOBILE PHONE APP. TeIKa, 7(1), 1–11. https://doi.org/10.36342/teika.v7i1.1000