DEVELOPMENT OF ATTENDANCE SYSTEM BASED ON REAL TIME ADVANCED FACIAL RECOGNITION CASE STUDY: VTI JOINT STOCK COMPANY INTERNSHIP

UNDERGRADUATE THESIS

Submitted to the International

Program Department of Industrial

Engineering the Requirement for the

Degree of Sarjana Teknik Industri

at

Universitas Islam Indonesia

By

Ryan Gerhan Canndrika (17522226)

INTERNATIONAL PROGRAM

DEPARTMENT OF INDUSTRIAL ENGINEERING

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2022

AUTHENTICITY STATEMENT

In the name of Allah SWT, I, the undersigned, Ryan Gerhan Canndrika, declare and admit that this research is completely my work. This research was conducted originally by me as the Researcher to fulfill the objectives of the research itself and to fulfill the requirement bachelor's degree in Undergraduate Industrial Engineering International Program Universitas Islam Indonesia. Furthermore, I declare this research has never been submitted to any other academic institution. However, even though this was originally created by me, there are several references and writings or experts that being used in this research and all of that has been properly acknowledged and has been well cited. If someday in the future, my confession is proven to be contradictory, wrong or untrue and commits the violation of laws and regulations of intellectual property. If so, then I understand and am ready that my degree will be revoked or withdrawn by Universitas Islam Indonesia.

Yogyakarta, May 2022

SEPULUH RIBU RUPIAH 10000 METERAI TEMPEL E8CFFAJX820556229

[illegible]

Ryan Gerhan Canndrika

THESIS APPROVAL LETTER OF SUPERVISION

Development of Attendance System Based On Real Time Advanced Facial Recognition Case Study: VTI Joint Stock Company Internship

Thesis

By:

Name	: Ryan Gerhan Canndrika
Student Number	: 17522226

Yogyakarta, May 2022

[illegible]

Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph. D.

THESIS APPROVAL OF EXAMINATION COMMITTEE

**DEVELOPMENT OF ATTENDANCE SYSTEM
BASED ON REAL TIME
ADVANCED FACIAL RECOGNITION
CASE STUDY: VTI JOINT STOCK COMPANY INTERNSHIP**

By

Name	: Ryan Gerhan Canndrika
Student Number	: 17522226

Had been defended in front of Examination Committee in Partial Fulfillment of the Requirement for Bachelor Degree of Industrial Engineering Department

Universitas Islam Indonesia

Yogyakarta, July 11th 2022
Examination Committee

Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D Examination Committee Chair	[illegible]
Dr. R.M. Sisdarmanto Adinandra, S.T., M.Sc. Member I	[illegible]
Risdiyono, S.T., M.Eng., Ph.D. Member II	[illegible]

Acknowledged by,
Head of Undergraduate Program-Department of Industrial
Engineering Faculty of Industrial Technology
Universitas Islam Indonesia

UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
FAKULTAS TEKNOLOGI INDUSTRI

[illegible]

(Dr. Taufiq Immawan, ST., MM.)

DEDICATION PAGE

This undergraduate thesis is wholeheartedly dedicated to my beloved parent, Winarni and Budi Mulyono.T who have been my source of motivation, inspiration and vision. My heroes who always taught me to get up when everything is dragged me down, who always give their spiritual, emotional, knowledge, moral, financial and any other support for me to live this life. Moreover, I dedicate this research to my brothers, my sister, my special person and my friends, who always give support to me during my study.

ACKNOWLEDGEMENT

Bismillahirrahmanirrahim. *Al-hamdu lillahi rabbil 'alamin.*

First and foremost gratitude praised to Allah SWT. Allah the Almighty, the Most Gracious, and the Most Merciful for His blessing given to me during my study and all along to complete my thesis, entitled "Development of Attendance System Based on Real-Time Advanced Facial Recognition" to acquiring my undergraduate degree. May Allah's blessing goes to His final Prophet Muhammad, his family, and his companions. Besides Allah SWT, author also wants to give so much gratitude to:

1. Prof. Fathul Wahid, S.T., M.Sc., Ph.D. as Rector of Universitas Islam Indonesia
2. Hari Purnomo, Prof., Dr., Ir., M.T., IPU as Dean of Faculty of Industrial Technology Universitas Islam Indonesia
3. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph. D. as Head of Industrial Engineering Program Universitas Islam Indonesia, author's mentor, supervisor, and lecturer, for the knowledge, motivation, figure, insight, tips, and tricks
4. Dr. Taufiq Immawan, S.T., M.M. as Head of Undergraduate Industrial Engineering Program Universitas Islam Indonesia.
5. Ir. Ira Promasanti Rachmadewi, M.Eng. as Secretary of Industrial Engineering International Program Universitas Islam Indonesia
6. La Familia, MyHappyFamily. Consists of my beloved parents, Budi Mulyono and Winarni, for such wonderful heroes figures in life. My lovely sister, Ashti Youvica Cinnidia, for all the support and care. My motivator brother, Lingga Renggana Cannagia, for all the motivation and determination. My mentor and my partner to share ideas with, Agie Mashilga Canndista, for being bros in every situation, for the tutors and ideas.
7. Salwa Fitri Ariemurty, for all of the presence, the story, the journey, wonderful moments, and support
8. All of my classmates in Industrial Engineering International Program batch 17 for historic saga, tremendous effort, and togetherness, for all the support and encouragement.
9. All of the laboratory and all of Industrial Organizations members, the committee for becoming a place to share industrial knowledge, insight, and ideas.

10. Mrs. Devy Nurrahmah, S.Kom. and Mrs. Diana, S.T. as Academic Staffs, who help Researcher for any academic affair during study.
11. For all of parties, civitas academic and many more who cannot be mentioned one by one.

ABSTRACT

As the covid-19 pandemic goes on, terms of attendance in education are slowly shifting. To support online learning, attendance in education has a new concept which is "remote attendance". Advance attendance system is required to support remote attendance procedures. Therefore, this research aims to design and develop an advanced facial attendance system based on a real-time scenario. Three kinds of modules will be featured. The first one is liveness detection, used for detecting whether the face is real or fake. Real-Time attendance system based on facial recognition is the main system that also can record when exactly people conduct their attendance in real-time. Picture capture with time intervals is the last part of the whole system, and this module allows the camera to have the control to take several pictures in between predefined time intervals. The result of this research gives the Researcher the description on how the advance facial recognition attendance system works and developed.

Keywords: Facial Recognition, Advance Facial Attendance System, Remote Attendance

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

In this chapter, the whole background of this research will be explained. Several discussion will be discussed. Study background consists of the review of the researched area and addresses it to become background topic discussion in this research. A problem later will be figured out and formulated to a problem formulation. Based on that problem, the researcher tried to find out the problem solution, and it became the objectives of this research. After that, even if the researcher might solve the problem, yet, the problem solver itself may have a limitation. Thus, for better understanding, the researcher explained this at the introduction chapter down below.

1.1 Study Background

Education has long been recognized as a fundamental human right and an important building block in the development of children and broader society. The educational aspect has a significant impact in terms of developing the quality of way people think, understand, integrate, and prove. An analysis of education word reveals that education aims at providing a learner or a child a nourishing environment to bring out and develop the latent potentiality hidden inside him. (Dagar, Rao, Dash, & Joseph, 2020)

Attendance is defined as participation in a program of educational activities arranged and agreed upon by the educational institution such as school, campus or university, academy or institute. Attendance is nonetheless credited as an important component of school success. Students with better attendance records are cited as having stronger test performance. Oppositely, the low attendance may have serious implications for the academic growth of learners. Low attendance levels can reflect more general disaffection with school and can be associated with early school leaving, academic underperformance, and more restricted opportunities in terms of further education, training, and the labor market (Malcolm, Wilson, Davidson, & Kirk, 2003)

In early 2020, the world is facing the greatest problem in the past few decades. Corona virus disease, known as COVID-19, became pandemic. Since the world becomes increasingly interconnected, and Covid-19 becomes pandemic, then the risk does the same. Covid-19 affected people regardless of race, economic status, nationality, gender, or even educational background. Covid-19 significantly impacts many sectors, and the education sector is no exception. The COVID-19 pandemic has also had a severe impact on higher education as universities closed in response to lockdown measures. Although higher education institutions were quick to replace face-to-face lectures with online learning, these closures affected learning and examinations as well as the safety and legal status of students. (Schleicher, 2020)

As the covid-19 pandemic goes on, terms of attendance in education are slowly shifting. To support online learning, attendance in education has a new concept which is “remote attendance”, which was introduced for the first time in the 2020-21 school year. Remote attendance basically has the same meaning as general attendance, yet it can be done remotely and not conventionally conducted in-person or face-to-face. Remote attendance can be done using several digital platforms, synchronous or asynchronous, real-time or scheduled. Yet, while conducting online class sometimes occur ineffective and inefficient results in terms of attendance and lectures delivered. Most students conduct ethical violation while participating in an online class. Students often put the camera off to avoid face-to-face meetings. In extreme conditions, students probably just conduct attendance in an early class, then turn the camera off and continue their activities without even considering that they are in a lecture.

According to all of the explanations above, this research will be discussed creating an advance system of attendance. This system uses biometric recognition, which is face recognition, to conduct the attendance checking of students. Therefore, to prevent misconduct during online classes; the system will initiate the sampling method by automatically taking a picture of the streamed camera every certain minute. This kind of activity hopefully can prevent students from conducting cheating during an online class.

1.2 Problem Formulation

According to the study background of this research, the formulated problems are how to design the advanced real-time attendance system based on facial recognition and how to develop the advanced real-time attendance system based on facial recognition.

1.3 Research Objectives

Based on problem formulation, the objectives of this research are to develop and fulfill the required face attendance system that can prevent cheating while in an online class.

1.4 Research Limitation

Limitations of this research are:

1. This research only provides a prototype
2. This research was conducted on a local host personal computer

CHAPTER II

LITERATURE REVIEW

In this chapter, the literature that is significant to this research is reviewed. This chapter also discusses the other research that has already been contributed by someone in this topics field. Moreover, by reviewing the literature, the researcher might have a better understanding, because the literature being reviewed, referred to scholarly papers, journals, and any other research related to this research. Thus, the literature that is being reviewed will be discussed in detail down below.

2.1 Related Studies

Many research or studies has been conducted in the automation attendance system study field. Every study has a different approach and different methods, and so it distinguishes one study from another. For the automation in the attendance system study field itself, the researcher found varied studies. Research conducted by (Ummi, Andrew, & Risanti, 2018), proposed an NFC-based attendance system facial authorization using raspberry pi and a cloud server. In that study, the researcher designed an attendance system that can read NFC on a mobile device installed on Pi server. Once the attendance is recorded, the Pi server will store the data temporarily. If an internet connection is available, the server will upload and sync with the local storage database and also with the cloud base. Moreover, the authority, the lecturer and the student can recap the attendance through the application on a public/cloud server. It can be accessed through a web-based interface, and that attendance file can be downloaded through that. Research that the researcher conducted was developed using Balsamiq mockup and RFID technology.

The other research in the same field was conducted by (Kiran, Reddy, Ninan, Krishnan, Aravindhar, & Geetha, 2020). In this research, the researcher developed an attendance system in the classroom using the face recognition technique. The system will independently mark the attendance of students in a classroom without distracting the teacher, allowing the teacher to focus on other aspects of their teaching. The research of this attendance system used PCA and LDA as the method. PCA is a method for simplifying the complexity of generating an eigenvalue and containing eigenvector representation to obtain a consistent representation. The dimension space must be decreased in order to provide rapid and reliable object detection. Furthermore, PCA maintains the data's original information.

The PCA basis is used in the Eigenface algorithm. (Nurulhuda & Idayu, 2009). The most typically used method for face detection is the Eigenface-based approach. Eigenface is well-known for its ease of use, reduced sensitivity in poses, and improved performance with small databases or training sets. (Ripal, Nidhi, & Ami, 2012). The existence of eyes, nose, and mouth on a face, as well as the relative distances between these objects, are used in this method. In the facial domain, this distinguishing feature is referred to as Eigenfaces. (Pissarenko, 2002). LDA is also known as Fisher’s Linear Discriminant (FLD). The FLD approach is used to minimize the dimension space. The FLD technique makes use of within-class information to reduce variation within each class while increasing class separation. The Fisherface algorithm is a development of the eigenface algorithm that accounts for variations in illumination. (Shang-Hung, 2000). Fisherface is also one of the most extensively used methods for extracting features from face photos. This method aims to discover the projection direction in which images from various classes are separated as much as possible. (Yalamanchili & Paladugu, 2007)

The other research was conducted by (Dmello, Yerremreddy, Basu, Bhitle, Kokate, & Gharpure, 2019). Researcher explained the automated facial recognition attendance system with IoT Camera. Dmello et al. created an application for students and teachers to conduct attendance automatically using raspberry pi. For that research, the researcher used CNN. CNN are neural networks that work with images as a picture, however, CNN uses a Convolutional layer which basically applies image filters. The convolutional layer is parameters consisting of a set of learnable filters. Every filter is small in size (in terms of width and height), yet it covers the entire depth of the input volume. (Matthew & Rob, 2014).

For another automated attendance system of CNN-based facial recognition, (Shailender, Dhruv, Dipen, & Mandeep, 2020) proposed an automated attendance system using facial recognition. Faces should be detectable in each frame of a real-time video by the system. It should also be able to indicate the attendance of pupils whose faces are recognized by the system after recognizing the detected faces. The system's main feature is that it only records attendance for students who have attended more than 60% of the class; the remainder of the students are marked absent. The other research conducted was by (Smitha & Afshin, 2020), research conducted by the Researcher was Face Recognition based Attendance Management System. The system can conduct the attendance by live stream video of the classroom, and at the end, the system can send an email for notification. The researcher did not create an application interface for the research project. Instead, the researcher used haar-cascade and LBPH algorithm.

Another research conducted by (Clyde, Sagar, Tanmay, & Dipti, 2020) proposed a class attendance management system using facial recognition. Haar-cascade classifier algorithm and LBPH algorithm were the algorithms that were used by the authors. After conducting attendance, it will be marked in excel sheets. Another haar-cascade classifier and LBPH algorithm were used in the facial recognition attendance system were conducted by (Jenif D, Jothi, & Chandrasekar, 2019). (Hajri, Hafeez, & V, 2019) ResearcherThe Researcher was created a fully automated attendance system with facial recognition embedded in the customized mobile app. CNN algorithm was used in this mobile app research project. Table 2.1 shows a comparison of the previous studies.

Table 2.1 Related Studies Literature Review

No	Researchers	Year	Method or Algorithm						Proposed System
			Haar-Cascade	CNN	LBPH	PCA	Web-Based FR	LDA/FLD	
1	Ummi, et al.	2018					√		Web-Based, SMART-FR and NFC
2	Kiran, et al.	2020				√		√	Regular Facial Recognition Attendance System
3	Dmello, et al.	2019		√					IoT Attendance System
4	Shailender, et al.	2020	√	√					Wamp Server SQL Attendance System
5	Smitha, et al.	2020	√		√				Email Notification Marked Attendance System
6	Clyde, et al.	2020	√		√				Regular Facial Recognition Attendance System
7	Jenif D,et al.	2019	√		√				Regular Facial Recognition Attendance System
8	Hajri, et al.	2019		√					Mobile App with RFID Attendance System

Based on previous research related to this research, haar-cascade and LBPH were the most commonly used algorithms. Therefore, the reason why researcher uses Haar-Cascade and LBPH as the algorithm because haar-cascade and LBPH are the most common algorithm to be used as face detection and face recognition algorithms. Haar-cascade and LBPH algorithms are relatively easy to use for student-user. The haar-cascade classifier package is also already available in OpenCV. The accuracy of LBPH is also better comparing to other algorithms, also LBPH can still recognizes faces even if there are expression changes, as explained by (Mutiara & Prastyo, 2019). Every previous research also used a different approach and method also an algorithm. Furthermore, even if several researchers used the same approach or method, yet each researcher also provides many kinds of features or proposed systems. Each system provides advantages and disadvantages, and there is no such best system.

Thus, here, the researcher proposed a new kind of feature or proposed a new system regarding the automated attendance system. Using the haar-cascade classifier and LBPH algorithm, the researcher proposed a freshly different feature for the automated attendance-based facial recognition system. The liveness detection and picture capture by timer are factors that differentiate this research from previous research. Liveness detection is used libfaceid by (Richmondu, 2019) to detect an attack or spoofing while conducting attendance. Moreover, the picture capture timer is used to capture pictures within a determined interval. This research also can mark the completed attendance into csv file in excel sheets. By using the real-time scenario, the recorded attendance file can be recapped by the exact time when the attendance was conducted.

2.2 Deductive Study

1. Artificial Intelligence, Machine Learning, and Deep Learning

Artificial Intelligence (AI) is a branch of computer science that refers to simulated human intelligence performed by computers and machines. It involves developing computer programs to complete tasks that would otherwise require human intelligence. In Artificial Intelligence, the terms “problem solving and searching” refer to a large body of core ideas that deal with deduction, inference, planning, commonsense reasoning, theorem proving, and related processes. (Edward & Barr, 1979). AI is arguably one of the most important issues facing us in the future. AI offers many benefits and opportunities, as well as many challenges. To maximize the usage of AI, a prominent and informed public on AI are needed. Since the 1950's, this kind of AI has been categorized as Narrow or Weak AI, which means an AI that is programmed to perform or conduct only one single task.

In terms of establishing or creating a program to conduct tasks using AI, algorithms as the sets of code are needed. Algorithms are very crucial to AI, because AI results when machines or programs perform tasks based on the algorithms themselves in the form of an "intelligent" manner. While the process of practicing the algorithms statistically to parse data, learn from it and determine or predict something called as Machine Learning (ML). So rather than hand-coding software routines with a specific set of instructions to accomplish a particular task, the machine is 'trained' using large amounts of data and algorithms that give it the ability to learn how to perform the task. As stated by (Wayne, Maya, & Charles, 2019) that much early, rule-based AI involves writing in advance the steps that the computer will take to complete a task and rules that will be followed exactly. Machine learning, on the other hand, is about getting computers to act without being given every step in advance. Instead of the algorithms being programmed what to do, broadly speaking, they have the ability to learn what to do.

Machine Learning model needs to be told how it should make accurate predictions (by feeding it more data), a deep learning model is able to learn that through its own 'brain' computing. Deep learning model is designed to continually analyze data with a logic structure similar to how a human would draw conclusions. Therefore, to achieve this, deep learning applications use a layered structure of algorithms called an artificial neural network. (Grossfeld, 2021). If the functionality of deep learning is working as intended as programmed, deep learning is often received as a scientific marvel. Therefore, deep learning plays a role model in terms of being the backbone of true artificial intelligence because deep learning is what powers the most human-like artificial intelligence.

Deep learning applies multiple processing layers to learn representations of data with multiple levels of feature extraction. Deep Learning presents significant discoveries in solving the problems that have faced many struggles of the machine learning and artificial intelligence community in the past. A more recent trend is deep learning, which arose as a promising framework that provides the latest image processing performance. Furthermore, deep neural networks showed important conclusions to solve problems of recognition in the domain of image processing. The idea of deep learning which is searching for the main features to be capable of distinguishing between all the images given and figuring out the unique features that through low levels features and high-level features that are extracted by the convolution layer without using feature algorithms. (Mohamed, Mohamed, Khalil, & Mohra, 2019)

2. Face Detection and Face Recognition

One of the examples of deep learning applications is facial detection and recognition. The main approach taken in face recognition is the comparison between the information present in a database and the probe data obtained from the person to be recognized. An automatic face recognition system is usually a procedure of four main stages. In most cases, these four stages or blocks are namely: pre-processing, face detection, feature extraction, and finally, classification. Face recognition is the process of labeling a face that has been recognized. Face verification is when the images in data training match with data testing. Face recognition is applied in any sectors, including inthe education sector. By using biometric recognition, face recognition has grown rapidly and is very effective. Facial recognition has attracted particular interest as it provides a discreet, non-intrusive means of detection, identification, and verification, without the need for the subject's knowledge or consent. (Shepley, 2019). The process of face detection and face recognition can be shown in Figure 2.1

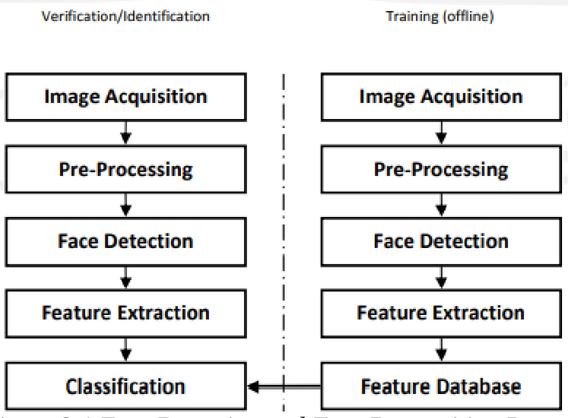


Figure 2.1 Face Detection and Face Recognition Process

Image acquisition is the process of obtaining image(s) from devices. Pre-processing is the process to cancel or remove the noise from the taken image, normalize the color, and fix the illumination. Then faces should be located and segmented in input images;

therefore face detection algorithm is presented in this step. The next step would be to extract some predefined features in order to make a feature vector. These features must include distinctive information about each person in the database so that they can recognize the individual based on these features. And finally, the last stage is the classifier, which intends to recognize an unknown sample by assigning a class to its feature vector based on the database of features that we have from previously seen samples. (Nabatchian, 2011)

a) Image Acquisition

Before any recognition can even be attempted, the system must acquire an image of the subject with sufficient quality and resolution to detect and recognize the face. The issues examined in this section are the sensor issues in lighting, image/sensor resolution issues, the field-of view, the depth of field, and effects of motion blur. (Boult & Scheirer, 2009)

Understanding the impact of illumination variation, or normalizing to reduce it, is by far the most well-studied of the issues associated with lighting and face recognition. Resolution is about ensuring enough pixels on the face to support recognition and sufficiently above the minimum needed for recognition to deal with the loss of resolution due to atmospheric turbulence. Field-of-view is defined by the combination of the sensor resolution and the focal length. Depth of field defines the ranges around the focus distance where subjects will be the sharp focus. Motion blur can occur because of platform motion, including vibrations, as well as because of subject motion. (Boult & Scheirer, 2009)

b) Pre-Processing

The first step in most face recognition systems as well as many other computer vision applications, is preprocessing or image quality enhancement. Some pre-processing could also be done to adapt the input image to the algorithm prerequisites. The input images acquired via still or video cameras might suffer from noise, bad illumination, or unrealistic color. Therefore, noise removal might be a necessary block in some cases. Histogram equalization is the most common method used for image enhancement when images have illumination variations.

c) Face Detection

Another step in the face recognition process is to locate the face in an input image. Face detection is the process of locating faces or faces in an image without identification. Face detection is a concept that includes many sub-problems. Some systems detect and locate faces at the same time. Others first perform a detection routine, and then, if positive, they try to locate the face. Then, this requires the algorithm to locate a single face with a known scale and orientation. Face detection at frame rate is an impressive goal that has an apparent application to practical face tracking and real-time face recognition. (Nabatchian, 2011). Face detection is, therefore, a two-class problem where we have to decide if there is a face or not in a picture. This approach can be seen as a simplified face recognition problem. (Ion, 2010)

d) Feature Extraction

Extraction of applicable features from the human face images is an important part of the recognition. Therefore choosing the proper feature extractor is crucial when designing a face recognition system with a high recognition rate. Feature extraction involves several steps - dimensionality reduction, feature extraction and feature selection. This feature extraction process can be defined as the procedure of extracting relevant information from a face image. This information must be valuable to the later step of identifying the subject with an acceptable error rate. The feature extraction process must be efficient in terms of computing time and memory usage. The output should also be optimized for the classification step.

e) Classification

This approach is intuitive and simple. Similar patterns should belong to the same class. This approach has been used in the face recognition algorithms implemented later. The idea is to establish a metric that defines similarity and a representation of the same-class samples. For example, the metric can be the Euclidean distance. Some classifiers are built based on a probabilistic approach. Bayes decision rule is often used. The rule can be modified to take into account different factors that could lead to miss- classification.

CHAPTER III

RESEARCH METHODOLOGY

This chapter discusses the methodology of this research, consisting of the research subject, research object, data types, and research flow. Research methodology shows how this scientific research is performed systematically. Moreover, it revealed the framework and logical scheme that coherently can guide the entire research that is being conducted. Thus, to enlarge the point of view of how this research was carried out by the researcher, more detail explanation is presented, as follows.

3.1 Research Subject

The subject of this research is the attendance system based on the facial recognition system.

3.2 Research Object

The object of this research is the advance of attendance systems using real-time facial recognition. This research conducted is to develop an attendance system based on real-time facial recognition.

3.3 Data Types

The data from this research is data training and data testing. Those data were collected from respondents by collecting given photos of their faces of several types of photos.

3.4 Research Flow

This research is described in several steps. Those steps are shown in Figure 3.1

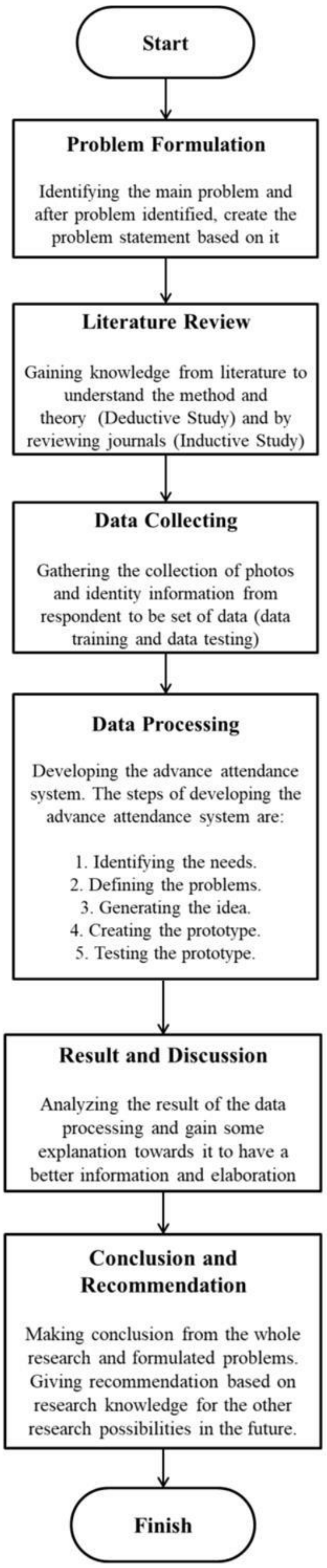

Start
Problem Formulation
Identifying the main problem and after problem identified, create the problem statement based on it
Literature Review
Gaining knowledge from literature to understand the method and theory (Deductive Study) and by reviewing journals (Inductive Study)
Data Collecting
Gathering the collection of photos and identity information from respondent to be set of data (data training and data testing)
Data Processing
Developing the advance attendance system. The steps of developing the advance attendance system are:
1. Identifying the needs.
2. Defining the problems.
3. Generating the idea.
4. Creating the prototype.
5. Testing the prototype.
Result and Discussion
Analyzing the result of the data processing and gain some explanation towards it to have a better information and elaboration
Conclusion and Recommendation
Making conclusion from the whole research and formulated problems. Giving recommendation based on research knowledge for the other research possibilities in the future.
Finish

Figure 3.1 Research Flow

1. Problem Formulation.

Problem Formulation generates the initiation of this research to be conducted. Right before formulating a problem, the researcher tries to identify the problem first. The idea of this process is to know the purpose and goal of this research and clarify the limitation field while conducting the research so that researcher knows and can decide what kind of action should be taken in terms of achieving the formulated problem itself. The problem identification process was conducted based on the researcher's point of view. After that, the problem formulation is generated.

2. Literature Review

Literature Review is fundamental to be done, by gaining some knowledge of the current research. Literature review guides researcher in terms of theoretical basis of this research.

3. Data Collection

Process of collecting required data by asking for respondents' photos to be collected. Those photos will be used as data training and data testing. Therefore, there are several types of photos that are being collected.

4. Data Processing

This step is the process of making the advanced attendance system by processing based on the collected data with the definition of the formulated problem, supported by the literature review. Data processing is the key component of this research; solving the formulated problem by processing the data and creating the advanced attendance system required several steps to be conducted, which are:

1. Identifying the needs.
   Identifying the needs is the beginning of the whole process. This process can determine what kind of problem that is happening in the current situation. In this research, the researcher identifies the problem purely from the researcher point of view. During the COVID-19 pandemic, regular face-to-face meeting in the educational sector has been replaced by online meeting/ online learning. While identifying the needs, the researcher tries to Understand the ethical violation phenomenon of online

learning. During online learning, most the students often put the camera off to avoid face-to-face online meetings. Moreover some of them only conduct attendance at the beginning of class, then turn the camera off and continue their activities without even considering that they are in lecture.

2. Defining the problems.

   From the researcher point of view, the action when students often put the camera off and continue their activities without even considering that they are in a lecture are actions that may lead to academic misconduct. Therefore, this kind of problem needs to be solved. To solve the problems, the researcher needs to define the problems in the problem statement. The problem statements are how to design and how to develop system that can minimize or even avoid that academic misconduct.

3. Generating the idea.

   Once the problem statement has been defined, the researcher tries to solve it. So, the next step that the researcher did was generate the idea. The purpose of this idea is to answer the problems that have been defined. The idea is to develop an attendance system based on real-time advanced facial recognition. In terms of solving the problem statement of how to design the system that can minimize or even avoid academic misconduct by adding several features, such as liveness detection, a real-time attendance system based on facial recognition, and picture capture by timer. This idea of developing the advanced attendance system helps the researcher to solve the problem related to the design.

   In terms of solving the problem statement of how to develop the system that can prevent academic misconduct is by making an attendance application system on researcher's local host personal computer. Using a programming language, the researcher used its libraries, modules, and virtual environment to create a liveness detection module, a real-time attendance system based on facial recognition module, and a picture captured by timer module.

4. Creating the prototype

   After idea formulation, researchers use one of programming language which is python programming language. With python, researchers use its libraries and modules such as OpenCV, NumPy, Pandas, Pillow, Libfaceid, Tkinter, Shutil, CSV, Argparse, sys, Datetime, and Time. The module consists of 3 parts that run one to another. In liveness detection modules, there are face detectors, face encoders, and liveness detection. Once the liveness detection is completed, the real-time attendance system based on facial recognition begins to run. In this module, there are interface functions, set date, and real-time function, storing path function, face detection function, and facial recognition function. Once it has been done, the next feature, which is the picture capture by timer feature, begins to run. In this module, there are frame set commands, image counter commands, timer commands and storing path commands.

5. Testing the prototype.

   For the final step, which is testing the prototype, the researcher was assisted by two other respondents in testing the system. Liveness detection runs at the beginning, and liveness detection can detect whether the facial area of the user is real or fake. This occurred by counting the eyes blinking (EAR/eyes aspect ratio) and mouth opening (MAR/mouth aspect ratio) and calculating the area threshold of the facial area user. When the systems can detect the facial area and conduct calculations, as a result, systems will notify the user about the identified unique faces. When the system can count how many framess of identified unique faces, it can conclude whether the facial is real or fake. The facial box also will show the percentage of similarities between trained data and facial images that are currently captured from real-time liveness detection.

   Real-time attendance based on facial recognition begins immediately after the liveness detection has been finished. Firstly there will be a user interface (UI) that appears that needs to be filled if the user has not completed the attendance. Still, if the user has been conducting the attendance, the user only needs to conduct the attendance directly. This system in simple explanation works by comparing the trained images and the real-time images captured from the user facial area. Once it’s matched, the system will save the exact date and time and store it in the attendance database. After that, picture capture by timer begins. This feature will capture any image of every certain defined time. For example, in 5 minutes, the camera will capture any images from the stream camera every 5 minutes. Finally, the captured images are stored in the database.

5. Analysis and Discussion

After the system has been made, can run well and can be operated. The advance system of face recognition attendance system will be analyzed to gain the knowledge and information of the recent study and research. Those discussed knowledge and information help researcher to make the conclusion of the whole research formulated problem.

6. Conclusion and Recommendation

Last but not least, the whole research will be concluded and answer the formulated problem. Finally, the recommendations are given for future research.

CHAPTER IV

DATA COLLECTING AND PROCESS

Data collecting and process is the most important thing in this research. This chapter differentiated this current research and the other research. The way how data is collected and processed by the researcher is discussed in this chapter. Data collecting is a systematically and scientific process of gathering data to establish reliable and credible research. After that, data will be analyzed, processed, and translated from raw data to be useful information such as reports, charts, or any other information through the developing system. Therefore, the researcher will explain the data collecting and process below in terms to have a better understanding.

4.1 Data Collecting

Before the attendance system based on facial recognition can work, there are a set of data needs to be inputted into the system. Data collected in this research consists of two types. Photos and identity information. Photos are divided into two types, data training and data testing. Identity information is only submitted once at the beginning of the process. Once submitted, the data will be recorded in database. Same as identity information, data training is only recorded once along with identity submission. While data testing gains every time people conduct facial recognition in a real-time condition.

4.1.1 The Data

Requirement for photos, the distance between user and camera must be at least 25 cm for minimum distance. The data that the Researcher collected are some photos of the respondent that were requested by the Researcher to be used for research purposes only. The photos needed to be used as data in this study are photos of the respondent's faces with a high level of clarity, meeting the required number of photo quotas, and with the right angle. The first procedure of portrait acquisition can be done by using the camera to capture individual faces. If there is more than one user who wants to conduct the face detection for data train, the user should conduct one-by-one. There will be a facial box on the monitor screen while the camera turned on to capture faces. The box will detect the face in a certain space between the camera and the face of the respondent. The photos are taken in portrait orientation; no matter the expression the user express, the facial box will detect the face every 0.5 seconds for 30 seconds long. If the respondent’s face cannot be clearly seen or if the camera cannot detect the presence of the face, the box will detect as soon as the face clearly appears. For a certain time, the camera will be turned off automatically. At the end of the capturing images to be data, the systems will store several pictures. Those pictures will be divided up to 60 pictures. The system already stored the data in the database. Below here are the examples of face pictures taken to be data training. Figure 4.1 below show the example of data collection (photos)

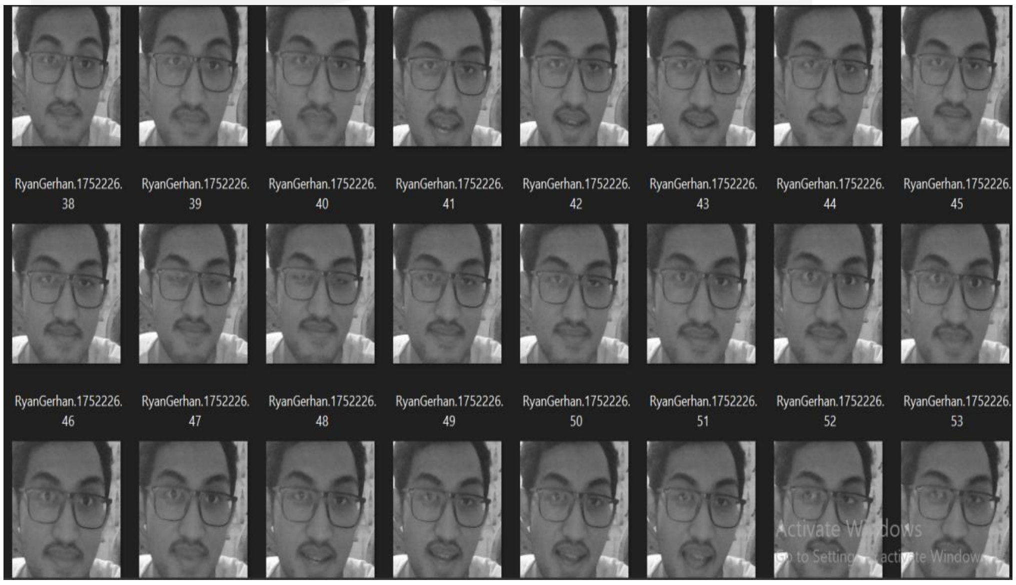
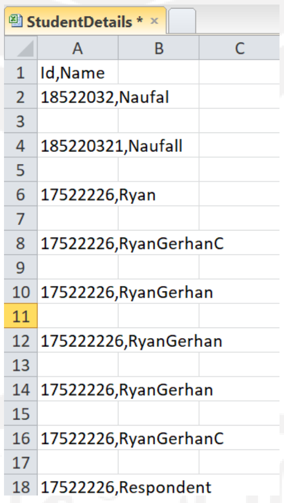


Figure 4.1 Data Collection

4.1.2 The identity information

Data is being used in this research is not only photos but other information may also concern. ID and students' name contains important information about students’ identity. This information helps the attendance system match the real people with their photos as the training data. This identity data only needs to be fulfilled once at the beginning of the attendance process. After that, students only need to conduct the attendance system during the course. Down below is the example of identity data that will be recorded once at the beginning of facial attendance. For the ID field, user needs to enter a numeric variable. For the Name field, the user needs to enter an alphabetic variable. For a name that more than one word should not be separated by spaces, spaces cannot be processed, so the user may input a capital letter for every first letter of the user’s name.


StudentDetails *

	A	B	C
1	Id,Name		
2	18522032,Naufal		
3			
4	185220321,Naufall		
5			
6	17522226,Ryan		
7			
8	17522226,RyanGerhanC		
9			
10	17522226,RyanGerhan		
11			
12	175222226,RyanGerhan		
13			
14	17522226,RyanGerhan		
15			
16	17522226,RyanGerhanC		
17			
18	17522226,Respondent		

Figure 4.2 Identity Information

4.2 The Data Processing

Data that has already been collected will be processed by a developed attendance system based on facial recognition. The attendance system based on facial recognition will be explained by understanding the pipeline or flowchart of the system. After that will be explained by creating, discussing the code details. The system itself consists of three parts, which are liveness detection, real-time attendance system based on facial recognition, and pictures captured at a time interval.

Liveness detection is used to detect whether the person face is a real face or fake. This is conducted by detecting the center point of both eyes in the input face image. Using detected both eyes, normalize face region and extract eye regions. After binarizing extracted eye regions, and then compare each binarized eye region and calculate variation. If the result exceeds the threshold, the input image is recognized as the live face. If not, it is separated as the photograph. Liveness detection also uses mouth detection to strengthen the performance to testify the real face picture taken. Several pictures will be taken, and counts how many times the eyes also mouth open and shut. Liveness detection is supposed to create several protections of spoofing or attack. Therefore, the researcher tried to develop liveness detection in this proposed new system.

Real-Time attendance system based on facial recognition is the main system is a concern in this research. The procedure of this attendance system is by recognizing the biometrics of the student's faces. After conducting attendance, the system will store several data in database. Information such as name and student's ID will be recorded and also can record when exactly people conduct the attendance in real time. The recorded data will be converted to an excel file to ease the checking.

The reason why researcher used facial recognition is that research conducted by (Dhaw, Moammer H., Amer R, Nabel K., & M., 2008) those researchers compared fingerprint and facial recognition ,currently very popular to be biometric authentication, and came up with the result that effectivity and efficiency of facial recognition are greater than the fingerprint. Facial recognition comes up with a 100% rate, and fingerprint comes up with 90%. The researcher compares by extracting the significant information represented by facial recognition and fingerprint recognition. For another reason, the researcher thinks this kind of biometric is suitable with the condition when this research was conducted which is during covid-19 when everybody practically have their own camera to conduct the facial recognition authentication.

Picture capture with time interval is the last part of the whole system. Basically, the definition of this part is the camera will have the control to take several pictures in between predefined time intervals. For instance, when time intervals once predefine, i.e 5 minutes, so the camera will capture a single picture every 5 minutes. The taken picture will be stored to the database and can be seen by opening the file storage. The reason why researchers use this picture capture feature is to enhance the statistics number. This statistics variable will add some information related the attendance percentage conducted by students.

4.1.3 System Pipeline

The system pipeline of the advanced attendance system based on facial recognition is shown in Figure 4.3 – Figure 4.5

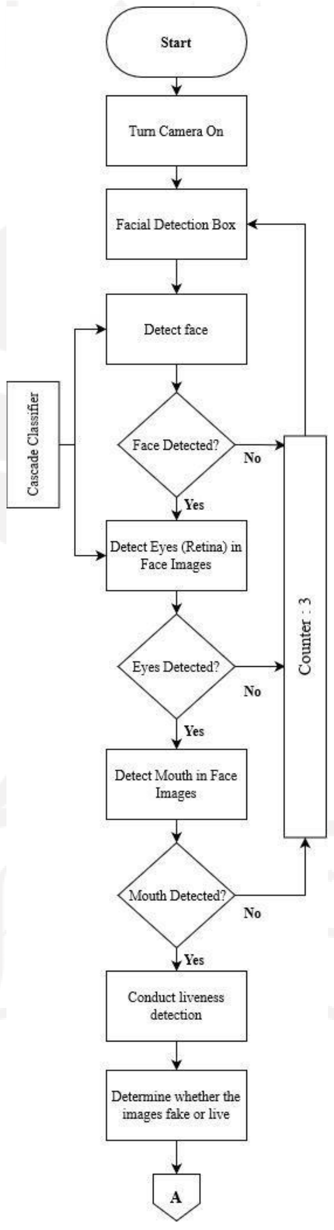


Figure 4.3 System Pipeline Diagrams 1

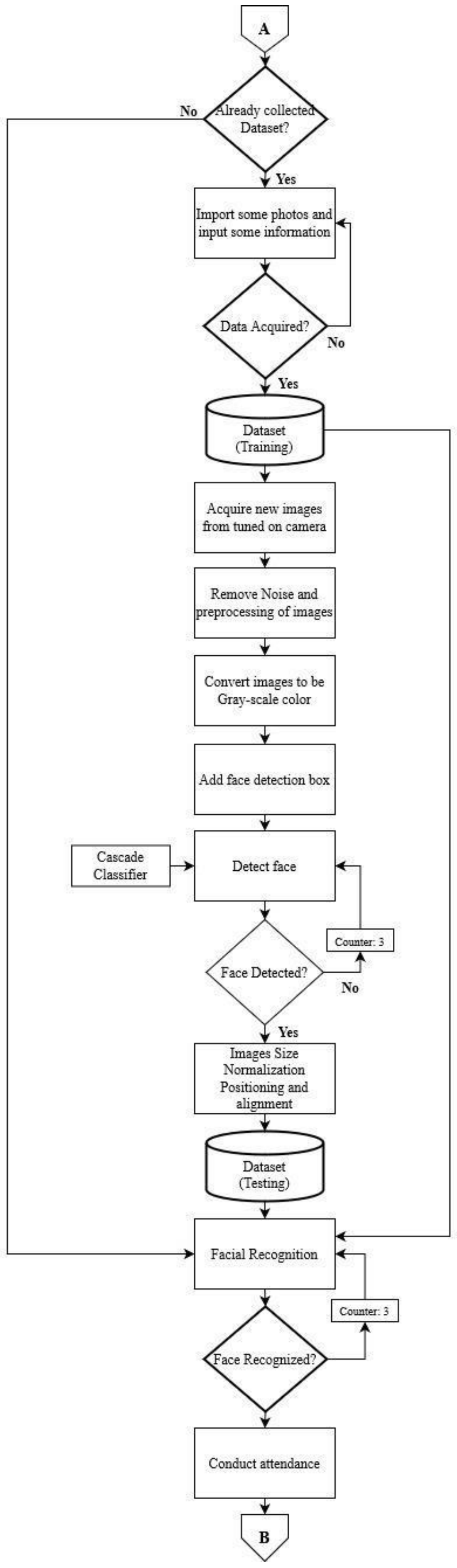

A
Already collected Dataset?
No
Yes
Import some photos and input some information
Data Acquired?
No
Yes
Dataset (Training)
Acquire new images from tuned on camera
Remove Noise and preprocessing of images
Convert images to be Gray-scale color
Add face detection box
Cascade Classifier
Detect face
Counter: 3
Face Detected?
No
Yes
Images Size Normalization Positioning and alignment
Dataset (Testing)
Facial Recognition
Counter: 3
Face Recognized?
Conduct attendance
B

Figure 4.4 System Pipeline Diagrams 2

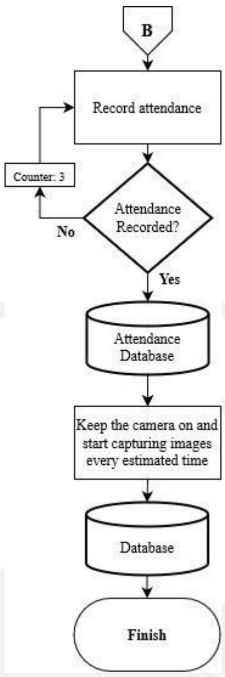


Figure 4.5 System Pipeline Diagrams 3

The system of pipeline above shows how the attendance system based on facial recognition works. As mentioned, the system itself consists of 3 parts, which are liveness detection, an attendance system based on facial recognition, and pictures captured at time intervals. Liveness detection is shown in a red rectangle, the attendance system based on facial recognition is shown in blue shape, and last but not least, the picture capture with time interval shown in green shape.

4.1.4 Developing the advanced attendance system based on facial recognition

Based on the system pipeline above, the whole system will be divided into 3 parts. Therefore, in this research, the code of those 3 parts will be explained in detail. Those code parts were developed with python language.

4.2.2.1 Python Libraries and Modules

Python is a high-level, general-purpose, interpreted object-oriented programming language. Since python uses the English keyword, and it is an open-source, so python is relatively easy to use, read, and simple to implement.

There are several modules in this research and several libraries development that being used. For the libraries, the researcher uses OpenCV, NumPy, Pandas, Pillow, and Libfaceid. For modules, the researcher uses Tkinter, Shutil, CSV, Argparse, sys, Datetime, and Time. Moreover, 3 parts of the whole system it was also created in module files.

OpenCV stands for Open Source Computer Vision. It is a free computer vision library that allows you to manipulate images and videos to accomplish various tasks, from displaying the feed of a webcam to potentially teaching a robot to recognize real-life objects. (Minichino & Howse, 2015). NumPy is a package that defines a multi-dimensional array object and associated fast math functions that operate on it. It also provides simple routines for linear algebra and fft and sophisticated random-number generation. NumPy replaces both Numeric and Numarray (Suresh, 2019). Pandas is an open-source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays. As one of the most popular data wrangling packages, Pandas works well with many other data science modules inside the Python ecosystem. (State, 2021).

Python Pillow module is built on top of PIL (Python Image Library). It is the essential module for image processing in Python. Anything is possible in using the pillow module on the digital images processing. For instance, filtering images, creating a thumbnail, merging images, cropping images, blurring an image, resizing an image, creating a watermark, and many other operations. However, PIL it is not supported by Python 3. But, it can be use with the Python 3.x versions as PIL. It supports the variability of images such as jpeg, png, bmp, gif, ppm, and tiff. (javatpoint, 2022).

Libfaceid is a Python library for facial recognition that seamlessly integrates multiple face detection and face recognition models. Libfaceid enables beginners to learn various models and simplifies prototyping of facial recognition solutions by providing a comprehensive list of models to choose from. libfaceid is designed so that it is easy to use, modular and robust. Selection of model is done via the constructors while the expose function is simply detect() or estimate() making usage very easy. (Richmondu, 2019)

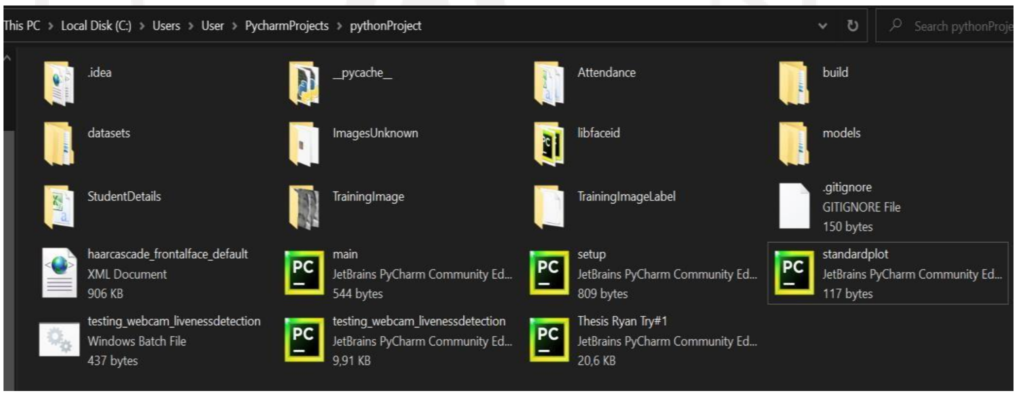
Tkinter is the inbuilt python module that is used to create GUI applications. It is one of the most commonly used modules for creating GUI applications in Python as it is Tkinter is the standard GUI library for Python that simple and easy to work with. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. There are currently 15 types of widgets in Tkinter. Widgets in Tkinter are the elements of GUI application which provides various controls (such as Labels, Buttons, ComboBoxes, CheckBoxes, MenuBars, RadioButtons and many more) to users to interact with the application. All Tkinter widgets have access to specific geometry management methods, which have the purpose of organizing widgets throughout the parent widget area. Tkinter exposes the following geometry manager classes: pack, grid, and place. (Kumaravnish, 2020).

Shutil module offers high-level operation on a file like a copy, create, and remote operation on the file. It comes under Python's standard utility modules. This module helps in automating the process of copying and removal of files and directories. (Satyam, 2021). CSV (Comma Separated Values) file is a form of plain text document which uses a particular format to organize tabular information. CSV file format is a bounded text document that uses a comma to distinguish the values. Every row in the document is a data log. Each log is composed of one or more fields, divided by commas. It is the most popular file format for importing and exporting spreadsheets and databases. (Saha, 2021). Argparse module in Python helps create a program in a command-line-environment in a way that appears not only easy to code but also improves interaction. Argparse module also automatically generates help and usage messages and issues errors when users give the program invalid arguments. (Kagoyal, 2020).

Sys module provides functions and variables used to manipulate different parts of the Python runtime environment. This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter. (Aggarwal, Command Line Arguments in Python :Geeksforgeeks, 2021). Datetime module in python supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times, and time intervals. Date and DateTime are objects in Python, so when it is being manipulate, it means that it is actually manipulating objects and not strings or timestamps. (Aggarwal, Python datetime module: Geeksforgeeks, 2021). Python has defined a module, “time,” which allows us to handle various operations regarding time, its conversions, and representations, which find its use in various applications in life. The beginning of time is started measuring from 1 January, 12:00 am, 1970 and this very time is termed as “epoch” in Python. (Geeksforgeeks, 2022).

4.2.2.2 Folders and Files

In this research, the files are mostly saved in python format (.py). Image data that is being used in this research is in jpg format. Identity information data in this research is stored in csv format. All of those data, files, and the required things stored in one folder. Moreover, the folder contains many sub-folder and each of that sub-folder represent an individual where all those data, files, and required things exist. The whole process of acquiring data, processing the data, gathering the identity information, creating an environment of required things, and storing mechanism of all files into one folder is done by the scripted code. Figure 4.6 shows the files and folder


This PC > Local Disk (C:) > Users > User > PycharmProjects > pythonProject

Search pythonProje

.idea
__pycache__
Attendance
build
datasets
ImagesUnknown
libfaceid
models
StudentDetails
TrainingImage
TrainingImageLabel
.gitignore
GITIGNORE File
150 bytes
haarcascade_frontalface_default
XML Document
906 KB
main
JetBrains PyCharm Community Ed...
544 bytes
setup
JetBrains PyCharm Community Ed...
809 bytes
standardplot
JetBrains PyCharm Community Ed...
117 bytes
testing_webcam_livenessdetection
Windows Batch File
437 bytes
testing_webcam_livenessdetection
JetBrains PyCharm Community Ed...
9,91 KB
Thesis Ryan Try#1
JetBrains PyCharm Community Ed...
20,6 KB

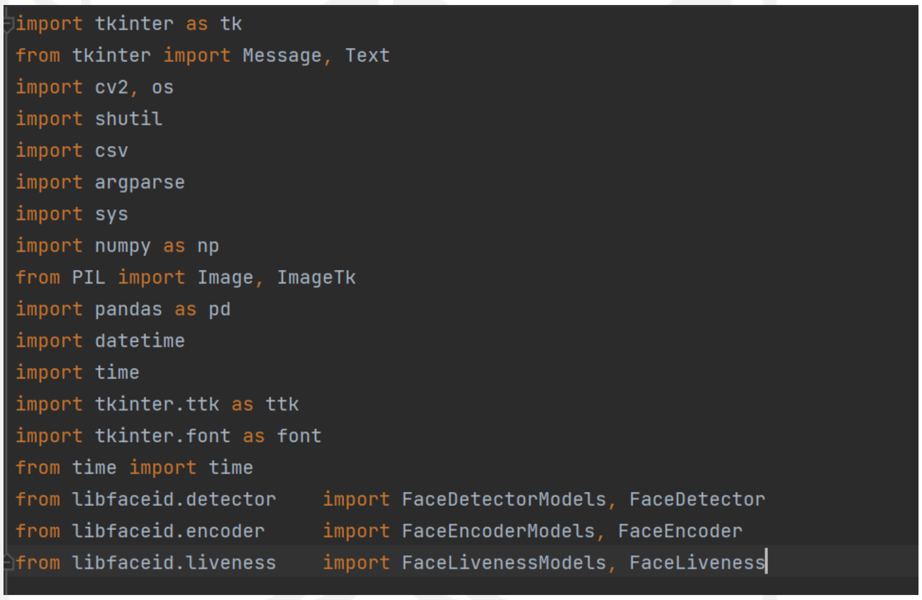
Figure 4.6 Files and Folder

4.2.2.3 Programming Module of Advanced Facial Recognition Feature

In case to have a better understanding of the overview of the whole system, this section will be divided into several particular details. Those details explain how the system was made.

1. Code for Importing, Set Directories, and Set Resolution

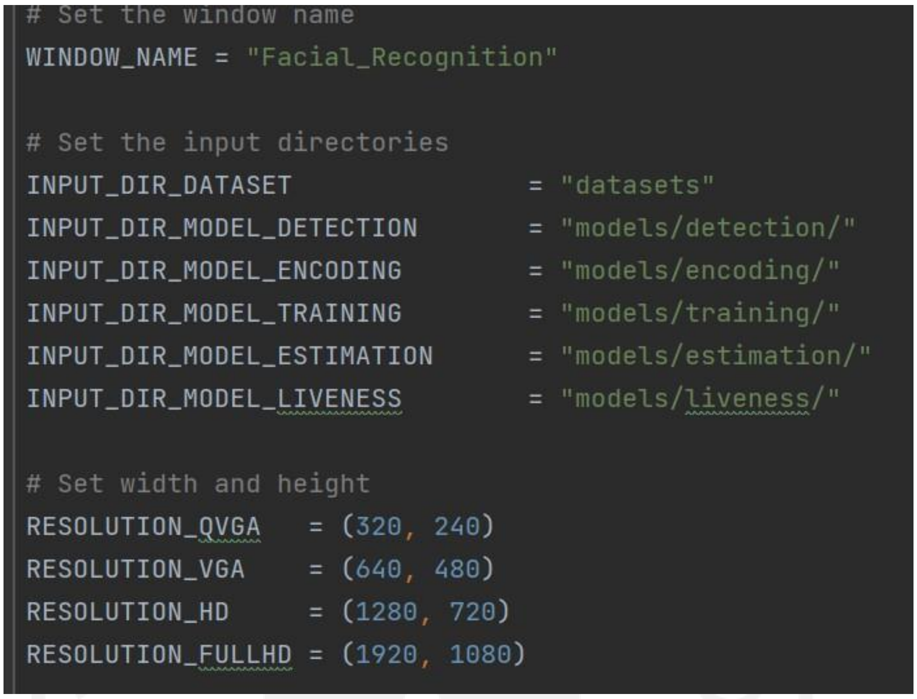
In this particular detail, will be explained about the code that constructs the whole system. Code is one of the most important things in this research. The first code that will be explained is importing code. Figure 4.7 shows the code for importing modules and libraries.



```
import tkinter as tk
from tkinter import Message, Text
import cv2, os
import shutil
import csv
import argparse
import sys
import numpy as np
from PIL import Image, ImageTk
import pandas as pd
import datetime
import time
import tkinter.ttk as ttk
import tkinter.font as font
from time import time
from libfaceid.detector    import FaceDetectorModels, FaceDetector
from libfaceid.encoder     import FaceEncoderModels, FaceEncoder
from libfaceid.liveness    import FaceLivenessModels, FaceLiveness
```

Figure 4.7 Import Code

The above code importing is conducted at the beginning of the entire process of code. It is designated to construct the environment that will be used in this research. Importing libraries and modules required installation first. Those modules and libraries developed the architecture of the whole system. Set the directories and set the resolution code shown in Figure 4.8.



```
# Set the window name
WINDOW_NAME = "Facial_Recognition"

# Set the input directories
INPUT_DIR_DATASET                = "datasets"
INPUT_DIR_MODEL_DETECTION        = "models/detection/"
INPUT_DIR_MODEL_ENCODING         = "models/encoding/"
INPUT_DIR_MODEL_TRAINING         = "models/training/"
INPUT_DIR_MODEL_ESTIMATION       = "models/estimation/"
INPUT_DIR_MODEL_LIVENESS         = "models/liveness/"

# Set width and height
RESOLUTION_QVGA   = (320, 240)
RESOLUTION_VGA    = (640, 480)
RESOLUTION_HD     = (1280, 720)
RESOLUTION_FULLHD = (1920, 1080)
```

Figure 4.8 Set Directories and Set Resolution Code

2. Module Code of Liveness Detection Feature

Code that is shown in Figure 4.8 is code to set the window name, the input directories, and the resolutions. The window name is in the form of strings, which represents the name of the running code window title. The input directories code is for a path to where the validation scripts' location. The file list for processing includes any files that meet the pattern defined in the file pattern. Last but not least, the resolution code is for the resolution setting when the camera wants to take a picture. So when the camera has been set, the ratio aspect of height and width of the taken picture has been stated. Figure 4.9 shows the camera initially and labeled face code

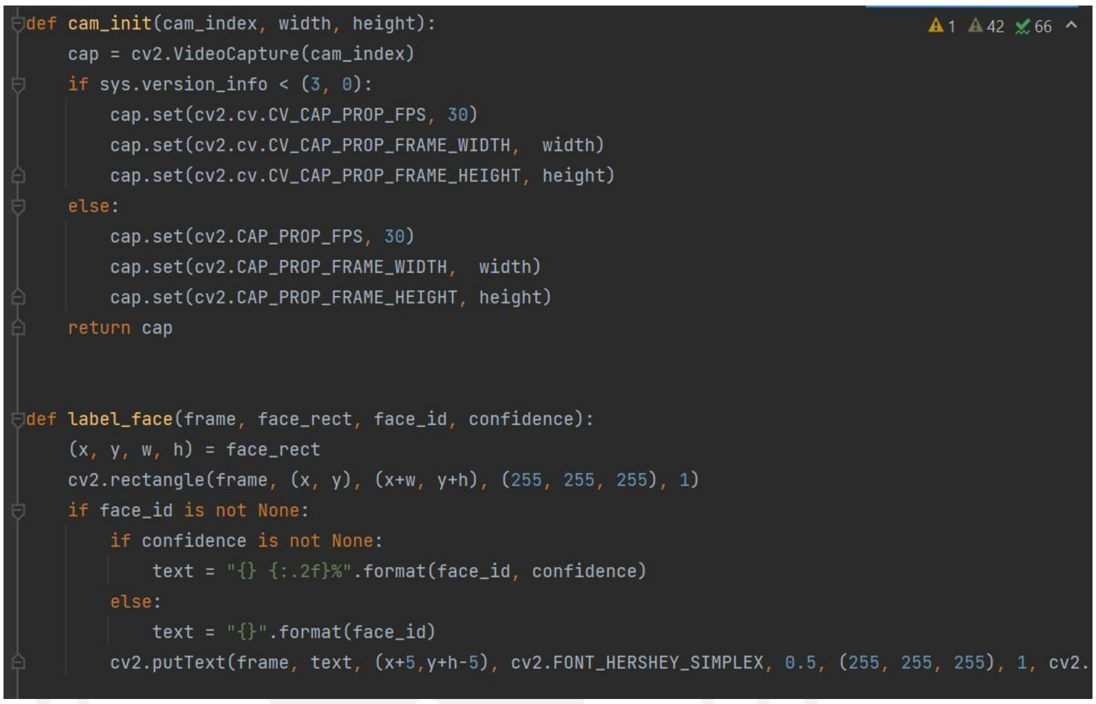
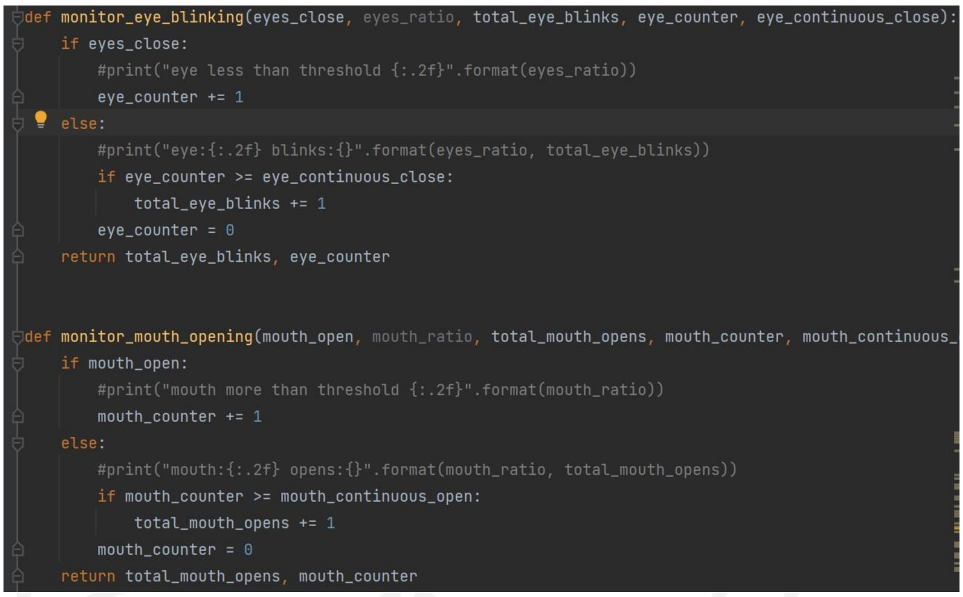

```python
def cam_init(cam_index, width, height):
    cap = cv2.VideoCapture(cam_index)
    if sys.version_info < (3, 0):
        cap.set(cv2.cv.CV_CAP_PROP_FPS, 30)
        cap.set(cv2.cv.CV_CAP_PROP_FRAME_WIDTH,  width)
        cap.set(cv2.cv.CV_CAP_PROP_FRAME_HEIGHT, height)
    else:
        cap.set(cv2.CAP_PROP_FPS, 30)
        cap.set(cv2.CAP_PROP_FRAME_WIDTH,  width)
        cap.set(cv2.CAP_PROP_FRAME_HEIGHT, height)
    return cap


def label_face(frame, face_rect, face_id, confidence):
    (x, y, w, h) = face_rect
    cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 255, 255), 1)
    if face_id is not None:
        if confidence is not None:
            text = "{} {:.2f}%".format(face_id, confidence)
        else:
            text = "{}".format(face_id)
        cv2.putText(frame, text, (x+5,y+h-5), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 1, cv2.
```

Figure 4.9 Camera Initial and Label Face Code

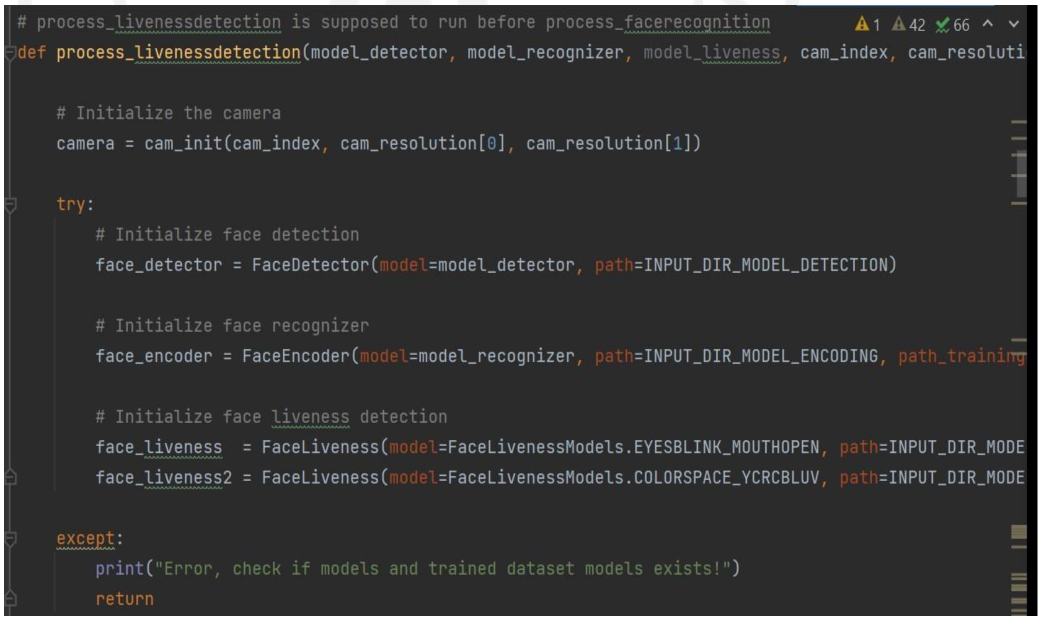
The code shown in Figure 4.9 is to define the initial camera and code for the face labeling. So the def means function definitions. A function definition is an executable statement that defines a user-defined function object. Here the def are def cam_init and def label_face. Def cam_init helps the user to define the camera resolution, the ratio aspect of width, and height if the camera is turned on (Video Captured). Def label_face in a function to help the user in terms of detecting faces and give a classification to the camera if the face is detected. Monitor eyes blinking and monitor mouth opening code shown in Figure 4.10



```
def monitor_eye_blinking(eyes_close, eyes_ratio, total_eye_blinks, eye_counter, eye_continuous_close):
    if eyes_close:
        #print("eye less than threshold {:.2f}".format(eyes_ratio))
        eye_counter += 1
    else:
        #print("eye:{:.2f} blinks:{}".format(eyes_ratio, total_eye_blinks))
        if eye_counter >= eye_continuous_close:
            total_eye_blinks += 1
        eye_counter = 0
    return total_eye_blinks, eye_counter


def monitor_mouth_opening(mouth_open, mouth_ratio, total_mouth_opens, mouth_counter, mouth_continuous_
    if mouth_open:
        #print("mouth more than threshold {:.2f}".format(mouth_ratio))
        mouth_counter += 1
    else:
        #print("mouth:{:.2f} opens:{}".format(mouth_ratio, total_mouth_opens))
        if mouth_counter >= mouth_continuous_open:
            total_mouth_opens += 1
        mouth_counter = 0
    return total_mouth_opens, mouth_counter
```

Figure 4.10 Monitor Eyes Blinking and Monitor Mouth Opening Code

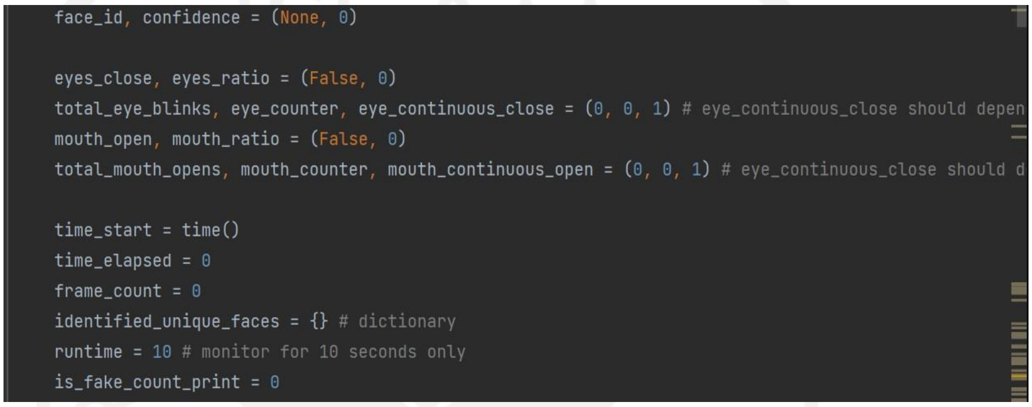
Here, there are two types of def, which are def monitor_eye_blinking and def monitor_mouth_opening. Def monitor_eye_blinking is to help the user to define the code for the eye monitor. If the eye is blinking, the system will count with eye_counter. Def monitor_mouth_opening helps the user to define the counter for monitoring the mouth opening. If the mouth starts opening, the system would start to count with mouth_counter. Figure 4.11 shows the process of liveness detection code



```
# process_livenessdetection is supposed to run before process_facerecognition
def process_livenessdetection(model_detector, model_recognizer, model_liveness, cam_index, cam_resoluti

    # Initialize the camera
    camera = cam_init(cam_index, cam_resolution[0], cam_resolution[1])

    try:
        # Initialize face detection
        face_detector = FaceDetector(model=model_detector, path=INPUT_DIR_MODEL_DETECTION)

        # Initialize face recognizer
        face_encoder = FaceEncoder(model=model_recognizer, path=INPUT_DIR_MODEL_ENCODING, path_training

        # Initialize face liveness detection
        face_liveness  = FaceLiveness(model=FaceLivenessModels.EYESBLINK_MOUTHOPEN, path=INPUT_DIR_MODE
        face_liveness2 = FaceLiveness(model=FaceLivenessModels.COLORSPACE_YCRCBLUV, path=INPUT_DIR_MODE

    except:
        print("Error, check if models and trained dataset models exists!")
        return
```

Figure 4.11 Process Liveness Detection Code

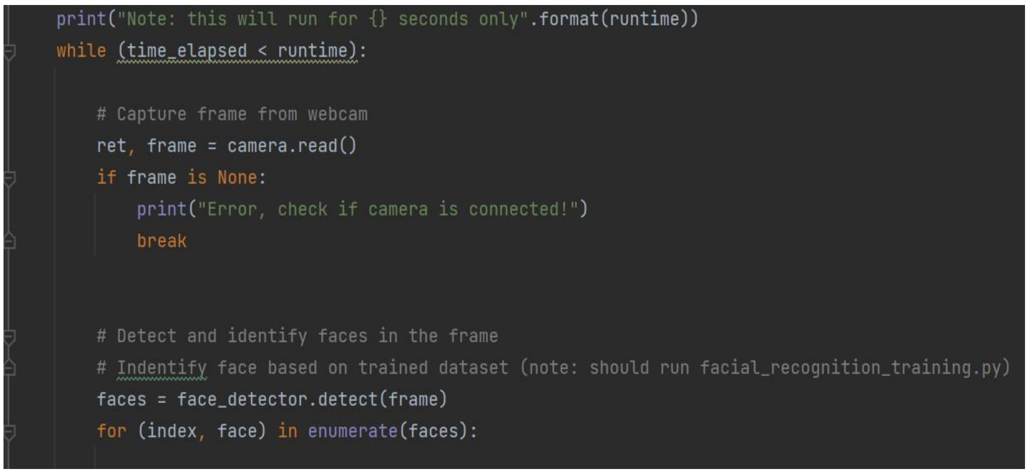
Def process_livenessdetection is to help the user to define the model of liveness detection detector. Initialize the camera code for turning the camera on based on the defined aspect ratio as stated in Figure 4.8. The initialization of face detection, face recognizer, and face liveness detection helps the user defined the model and directories path of face detection, face recognizer, and face liveness detection. Figure 4.12 shows the eye counter and mouth counter code



```
    face_id, confidence = (None, 0)

    eyes_close, eyes_ratio = (False, 0)
    total_eye_blinks, eye_counter, eye_continuous_close = (0, 0, 1) # eye_continuous_close should depen
    mouth_open, mouth_ratio = (False, 0)
    total_mouth_opens, mouth_counter, mouth_continuous_open = (0, 0, 1) # eye_continuous_close should d

    time_start = time()
    time_elapsed = 0
    frame_count = 0
    identified_unique_faces = {} # dictionary
    runtime = 10 # monitor for 10 seconds only
    is_fake_count_print = 0
```

Figure 4.12 Eye Counter and Mouth Counter Code

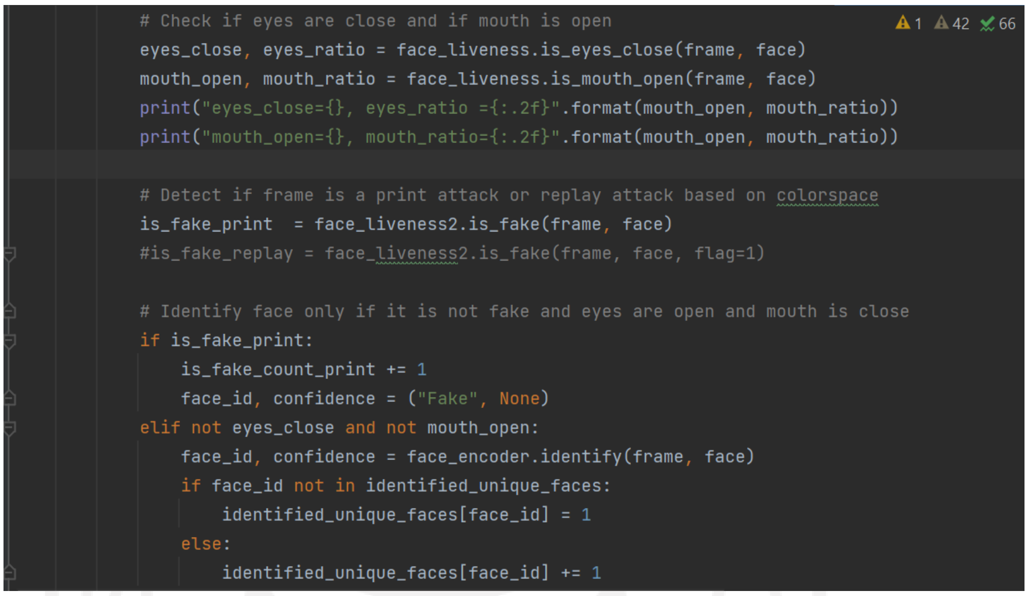
The code that shown in Figure 4.12 still consists of the face liveness detection line. The line is about the details of the def process_livenessdetection, such as the eyes ratio, mouth ratio, and timer for the liveness detection. Figure 4.13 shows the capture frame, detect and identify face for liveness detection code.



```
    print("Note: this will run for {} seconds only".format(runtime))
    while (time_elapsed < runtime):

        # Capture frame from webcam
        ret, frame = camera.read()
        if frame is None:
            print("Error, check if camera is connected!")
            break


        # Detect and identify faces in the frame
        # Indentify face based on trained dataset (note: should run facial_recognition_training.py)
        faces = face_detector.detect(frame)
        for (index, face) in enumerate(faces):
```

Figure 4.13 Capture Frame, Detect and Identify Face for Liveness Code

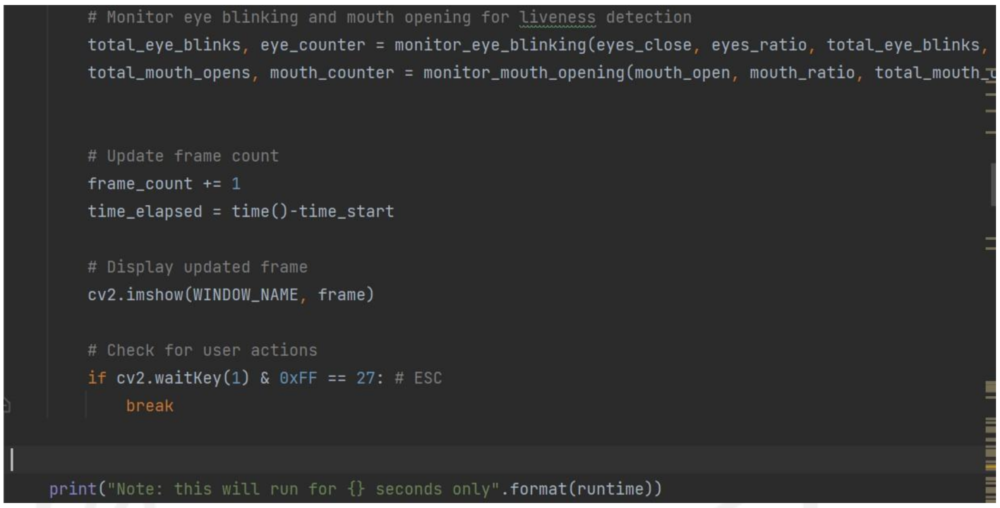
Code shown in Figure 4.11 explains the capturing frame from the webcam andidentifies the faces in the frame to proceed with the liveness detection. Figure 4.12 shows the attacks or spoofing code.



```
# Check if eyes are close and if mouth is open
eyes_close, eyes_ratio = face_liveness.is_eyes_close(frame, face)
mouth_open, mouth_ratio = face_liveness.is_mouth_open(frame, face)
print("eyes_close={}, eyes_ratio ={:.2f}".format(mouth_open, mouth_ratio))
print("mouth_open={}, mouth_ratio={:.2f}".format(mouth_open, mouth_ratio))

# Detect if frame is a print attack or replay attack based on colorspace
is_fake_print  = face_liveness2.is_fake(frame, face)
#is_fake_replay = face_liveness2.is_fake(frame, face, flag=1)

# Identify face only if it is not fake and eyes are open and mouth is close
if is_fake_print:
    is_fake_count_print += 1
    face_id, confidence = ("Fake", None)
elif not eyes_close and not mouth_open:
    face_id, confidence = face_encoder.identify(frame, face)
    if face_id not in identified_unique_faces:
        identified_unique_faces[face_id] = 1
    else:
        identified_unique_faces[face_id] += 1
```

Figure 4.14 Attacks or Spoofing Check Code

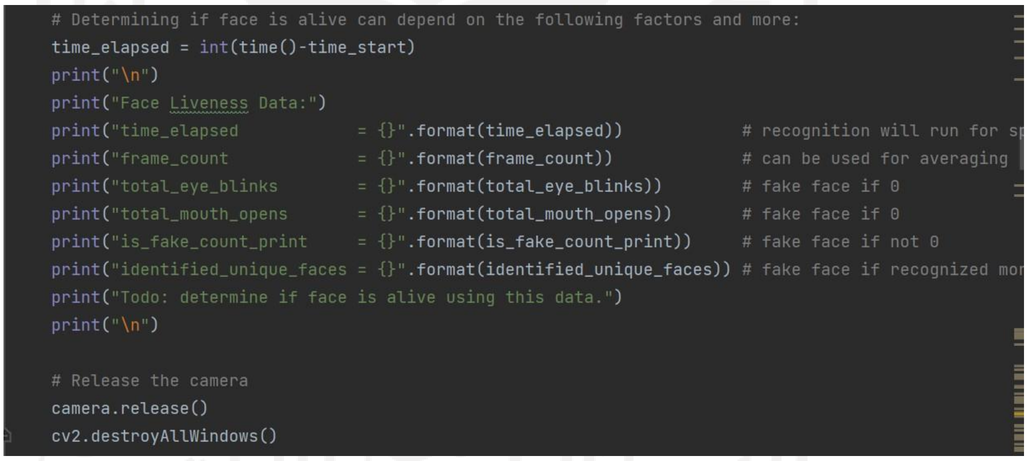
As it can be seen in Figure 4.14, the code explain how liveness detection works. The liveness detection is conducted by detecting the eyes and mouth. The system will detect closed eyes, and the opened mouth by eyes ratio and mouth ratio in the frame. After checking the eyes and mouth, the system will detect if the captured frame are attacking. The attack here means providing a false biometric sample into the system, for example, the user try to manipulate the face by showing an image instead of the real face or trying to avoid detection, those actions are categorized as spoof. Figure 4.15 shows the update counter and frame code



```
    # Monitor eye blinking and mouth opening for liveness detection
    total_eye_blinks, eye_counter = monitor_eye_blinking(eyes_close, eyes_ratio, total_eye_blinks,
    total_mouth_opens, mouth_counter = monitor_mouth_opening(mouth_open, mouth_ratio, total_mouth_


    # Update frame count
    frame_count += 1
    time_elapsed = time()-time_start

    # Display updated frame
    cv2.imshow(WINDOW_NAME, frame)

    # Check for user actions
    if cv2.waitKey(1) & 0xFF == 27: # ESC
        break


print("Note: this will run for {} seconds only".format(runtime))
```

Figure 4.15 Update Counter and Frame Count Code

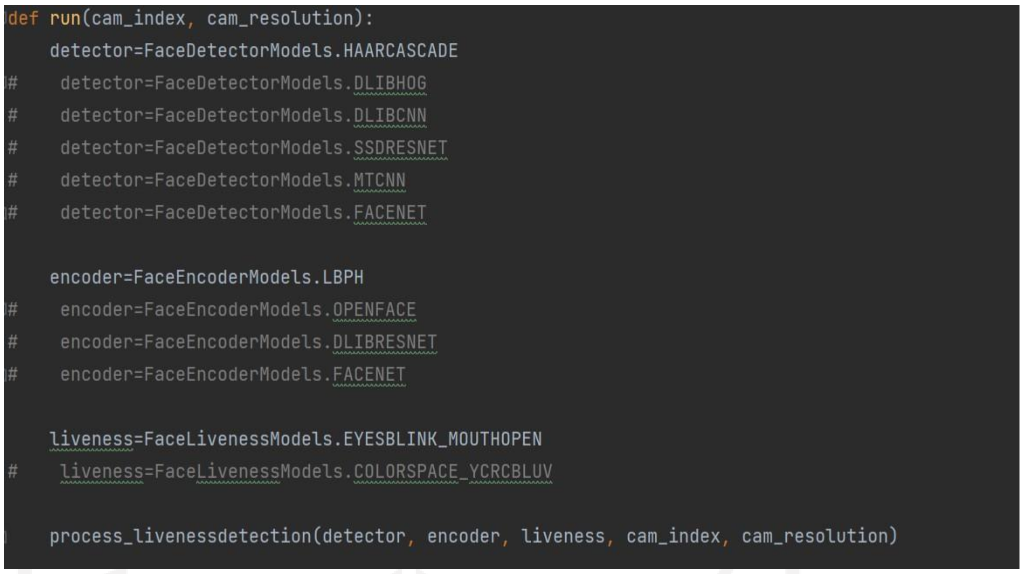
Code in Figure 4.15 is to monitor the eye blinking and mouth opening, update the frame count, display the frame, and check for user action. The print code is shown in Figure 4.16



```
# Determining if face is alive can depend on the following factors and more:
time_elapsed = int(time()-time_start)
print("\n")
print("Face Liveness Data:")
print("time_elapsed           = {}".format(time_elapsed))           # recognition will run for sp
print("frame_count            = {}".format(frame_count))            # can be used for averaging
print("total_eye_blinks       = {}".format(total_eye_blinks))       # fake face if 0
print("total_mouth_opens      = {}".format(total_mouth_opens))      # fake face if 0
print("is_fake_count_print    = {}".format(is_fake_count_print))    # fake face if not 0
print("identified_unique_faces = {}".format(identified_unique_faces)) # fake face if recognized mor
print("Todo: determine if face is alive using this data.")
print("\n")

# Release the camera
camera.release()
cv2.destroyAllWindows()
```

Figure 4.16 Print Code

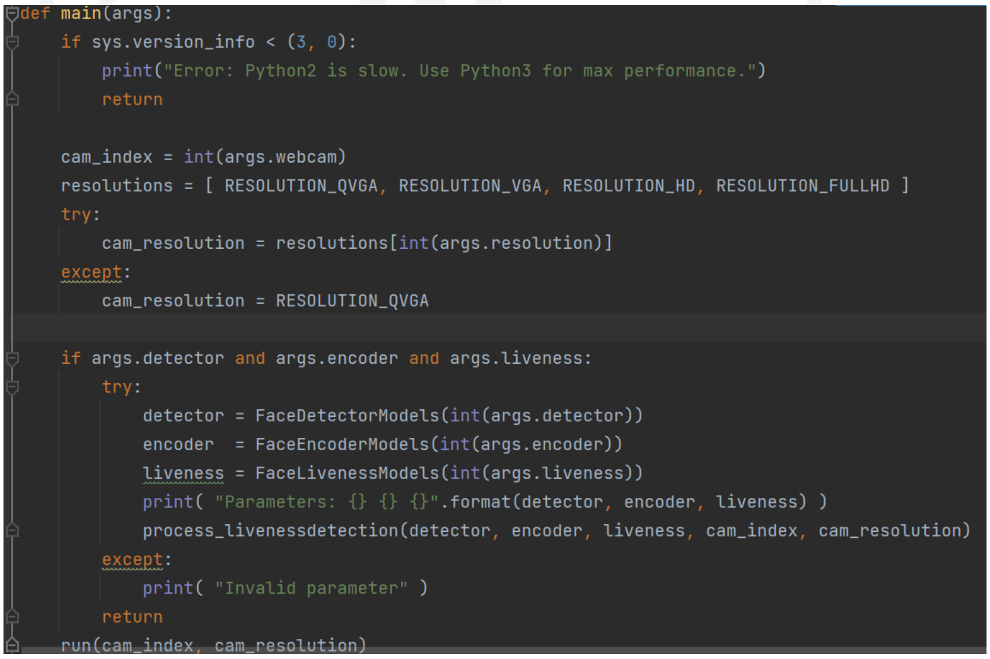
Here, code above is about print if the system can determine whether the face is alive or not based on factor dependency. Release the camera is to stop the camera operation from streaming, whether by software or by hardware. The detector, encoder, and liveness detection model code are shown in Figure 4.17



```
def run(cam_index, cam_resolution):
    detector=FaceDetectorModels.HAARCASCADE
#    detector=FaceDetectorModels.DLIBHOG
#    detector=FaceDetectorModels.DLIBCNN
#    detector=FaceDetectorModels.SSDRESNET
#    detector=FaceDetectorModels.MTCNN
#    detector=FaceDetectorModels.FACENET

    encoder=FaceEncoderModels.LBPH
#    encoder=FaceEncoderModels.OPENFACE
#    encoder=FaceEncoderModels.DLIBRESNET
#    encoder=FaceEncoderModels.FACENET

    liveness=FaceLivenessModels.EYESBLINK_MOUTHOPEN
#    liveness=FaceLivenessModels.COLORSPACE_YCRCBLUV

    process_livenessdetection(detector, encoder, liveness, cam_index, cam_resolution)
```

Figure 4.17 Detectors, Encoder and Liveness Detection Model Code

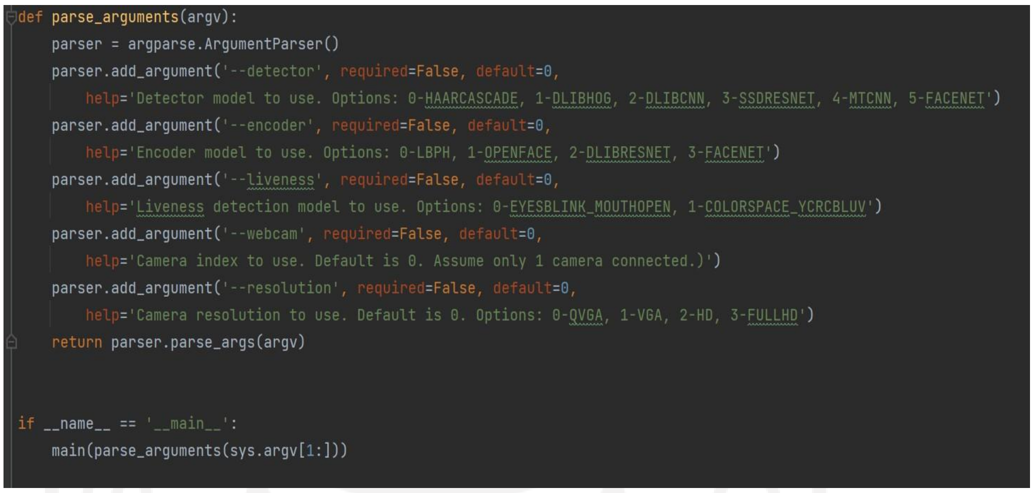
Here is the code for defining the detector, the encoder, and the liveness detection model. The face detector itself is to detect the face inside the frame image. Face Encoder is addressed for generating the mathematical representation of each face in a frame image. The face liveness detection model itself is for the counter spoofing attacks while conducting face recognition by counting the blinked eyes and opened mouth. Figure 4.18 shows the code for main liveness detection



```
def main(args):
    if sys.version_info < (3, 0):
        print("Error: Python2 is slow. Use Python3 for max performance.")
        return

    cam_index = int(args.webcam)
    resolutions = [ RESOLUTION_QVGA, RESOLUTION_VGA, RESOLUTION_HD, RESOLUTION_FULLHD ]
    try:
        cam_resolution = resolutions[int(args.resolution)]
    except:
        cam_resolution = RESOLUTION_QVGA

    if args.detector and args.encoder and args.liveness:
        try:
            detector = FaceDetectorModels(int(args.detector))
            encoder  = FaceEncoderModels(int(args.encoder))
            liveness = FaceLivenessModels(int(args.liveness))
            print( "Parameters: {} {} {}".format(detector, encoder, liveness) )
            process_livenessdetection(detector, encoder, liveness, cam_index, cam_resolution)
        except:
            print( "Invalid parameter" )
        return
    run(cam_index, cam_resolution)
```

Figure 4.18 Main Liveness Detection Code

The code shown in Figure 4.18 is the main function definition (def). It help user to define how the face detector, face encoder, and liveness detection works and also sets the resolution of the taken image. Figure 4.19 shows the Parse Argument Code

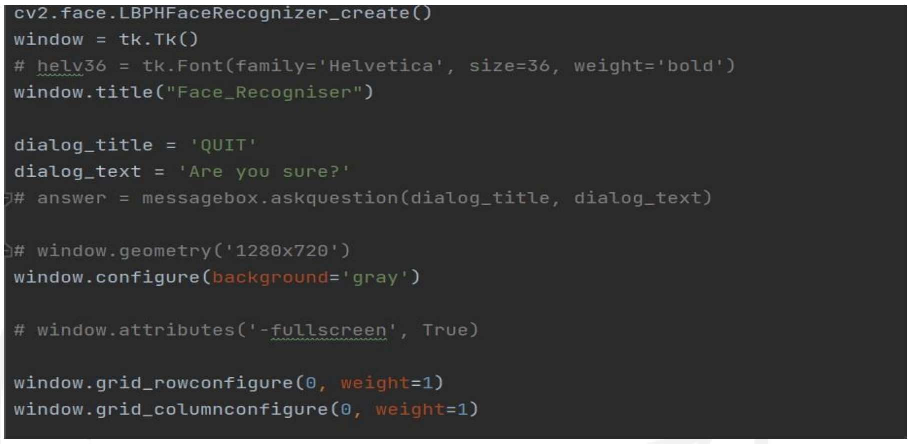


```
def parse_arguments(argv):
    parser = argparse.ArgumentParser()
    parser.add_argument('--detector', required=False, default=0,
        help='Detector model to use. Options: 0-HAARCASCADE, 1-DLIBHOG, 2-DLIBCNN, 3-SSDRESNET, 4-MTCNN, 5-FACENET')
    parser.add_argument('--encoder', required=False, default=0,
        help='Encoder model to use. Options: 0-LBPH, 1-OPENFACE, 2-DLIBRESNET, 3-FACENET')
    parser.add_argument('--liveness', required=False, default=0,
        help='Liveness detection model to use. Options: 0-EYESBLINK_MOUTHOPEN, 1-COLORSPACE_YCRCBLUV')
    parser.add_argument('--webcam', required=False, default=0,
        help='Camera index to use. Default is 0. Assume only 1 camera connected.)')
    parser.add_argument('--resolution', required=False, default=0,
        help='Camera resolution to use. Default is 0. Options: 0-QVGA, 1-VGA, 2-HD, 3-FULLHD')
    return parser.parse_args(argv)


if __name__ == '__main__':
    main(parse_arguments(sys.argv[1:]))
```

Figure 4.19 Parse Argument Code

This code shown in Figure 4.19 is about the function definition of the parse arguments. Parse arguments will figure out how to parse the command line options. This will inspect the command line, convert each argument to the appropriate type and then invoke the appropriate action. The main(parse_arguments(sys.argv[1:])) is the end of liveness of detection code.
The first parser argument is for face detector action argument, the Boolean value required=False, the default=0 it means value for HAARCASCADE. The help statement below is for default. If the default=1, it’s for DLIBHOG, default=2, it is for DLIBCNN, if default=3, it is for SSDRESNET, and so on.
The second parser argument is for the face encoder action argument, the Boolean value required=False and the default=0 for LBPH, default=1 for OPENFACE and so on. Those next parser arguments until the last parser arguments do the same action argument as described for its action.

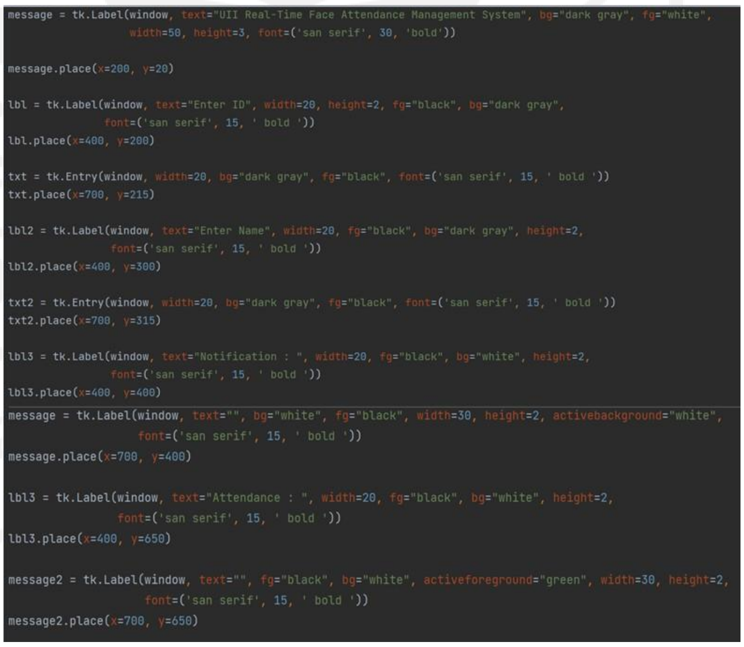
3. The Real-Time Attendance System Based on Facial Recognition Module Programming Code



```
cv2.face.LBPHFaceRecognizer_create()
window = tk.Tk()
# helv36 = tk.Font(family='Helvetica', size=36, weight='bold')
window.title("Face_Recogniser")

dialog_title = 'QUIT'
dialog_text = 'Are you sure?'
# answer = messagebox.askquestion(dialog_title, dialog_text)

# window.geometry('1280x720')
window.configure(background='gray')

# window.attributes('-fullscreen', True)

window.grid_rowconfigure(0, weight=1)
window.grid_columnconfigure(0, weight=1)
```

Figure 4.20 Create LBPH and Cofiguration of Window and Button Code

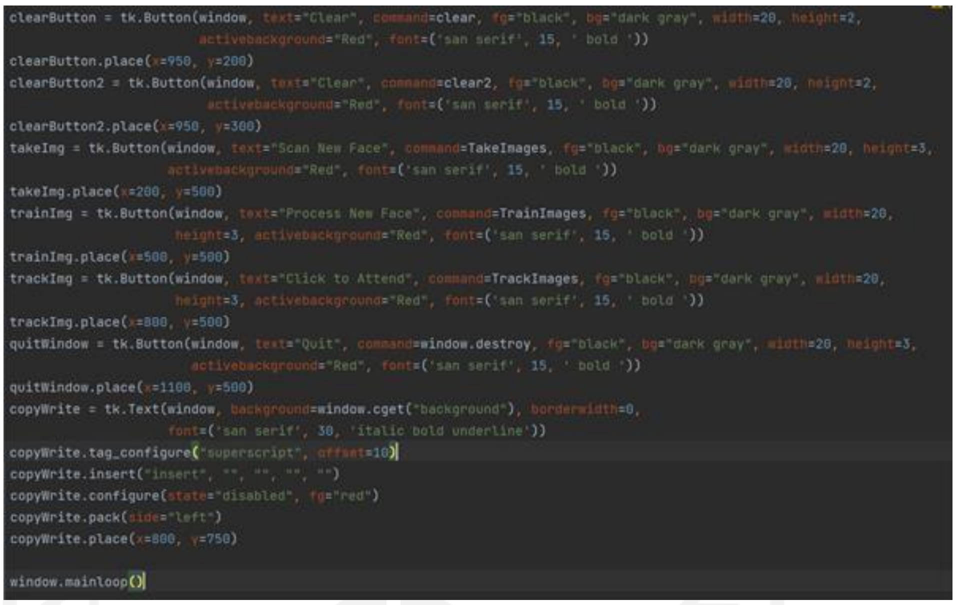
This code in Figure 4.20 shows about creation of the LBPH Face Recognizer by Opencv, to create the window title with button and message box. Figure 4.21 shows the buttons code



```
message = tk.Label(window, text="UII Real-Time Face Attendance Management System", bg="dark gray", fg="white",
                   width=50, height=3, font=('san serif', 30, 'bold'))

message.place(x=200, y=20)

lbl = tk.Label(window, text="Enter ID", width=20, height=2, fg="black", bg="dark gray",
               font=('san serif', 15, ' bold '))
lbl.place(x=400, y=200)

txt = tk.Entry(window, width=20, bg="dark gray", fg="black", font=('san serif', 15, ' bold '))
txt.place(x=700, y=215)

lbl2 = tk.Label(window, text="Enter Name", width=20, fg="black", bg="dark gray", height=2,
                font=('san serif', 15, ' bold '))
lbl2.place(x=400, y=300)

txt2 = tk.Entry(window, width=20, bg="dark gray", fg="black", font=('san serif', 15, ' bold '))
txt2.place(x=700, y=315)

lbl3 = tk.Label(window, text="Notification : ", width=20, fg="black", bg="white", height=2,
                font=('san serif', 15, ' bold '))
lbl3.place(x=400, y=400)
message = tk.Label(window, text="", bg="white", fg="black", width=30, height=2, activebackground="white",
                   font=('san serif', 15, ' bold '))
message.place(x=700, y=400)

lbl3 = tk.Label(window, text="Attendance : ", width=20, fg="black", bg="white", height=2,
                font=('san serif', 15, ' bold '))
lbl3.place(x=400, y=650)

message2 = tk.Label(window, text="", fg="black", bg="white", activeforeground="green", width=30, height=2,
                    font=('san serif', 15, ' bold '))
message2.place(x=700, y=650)
```

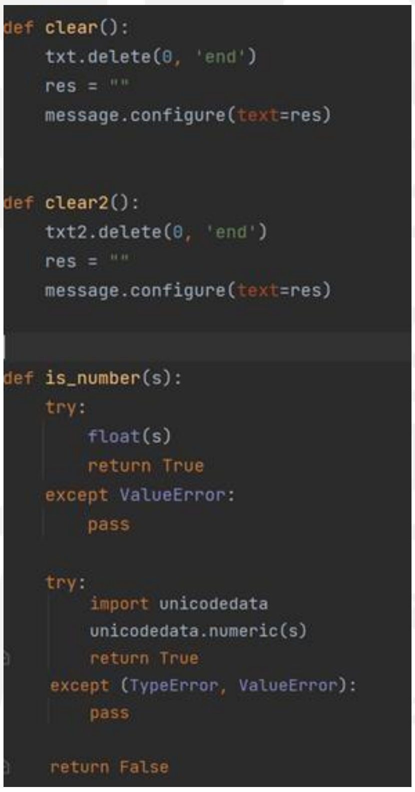
Figure 4.21 Button Code

The code in Figure 4.21 is code for attendance management interface design. Create some button with which the font, size, color, placement, and label that has been set. Figure 4.22 shows the clear button code. Figure 4.23 shows the code for the clear function of the clear button



```python
clearButton = tk.Button(window, text="Clear", command=clear, fg="black", bg="dark gray", width=20, height=2,
                        activebackground="Red", font=('san serif', 15, ' bold '))
clearButton.place(x=950, y=200)
clearButton2 = tk.Button(window, text="Clear", command=clear2, fg="black", bg="dark gray", width=20, height=2,
                         activebackground="Red", font=('san serif', 15, ' bold '))
clearButton2.place(x=950, y=300)
takeImg = tk.Button(window, text="Scan New Face", command=TakeImages, fg="black", bg="dark gray", width=20, height=3,
                    activebackground="Red", font=('san serif', 15, ' bold '))
takeImg.place(x=200, y=500)
trainImg = tk.Button(window, text="Process New Face", command=TrainImages, fg="black", bg="dark gray", width=20,
                     height=3, activebackground="Red", font=('san serif', 15, ' bold '))
trainImg.place(x=500, y=500)
trackImg = tk.Button(window, text="Click to Attend", command=TrackImages, fg="black", bg="dark gray", width=20,
                     height=3, activebackground="Red", font=('san serif', 15, ' bold '))
trackImg.place(x=800, y=500)
quitWindow = tk.Button(window, text="Quit", command=window.destroy, fg="black", bg="dark gray", width=20, height=3,
                       activebackground="Red", font=('san serif', 15, ' bold '))
quitWindow.place(x=1100, y=500)
copyWrite = tk.Text(window, background=window.cget("background"), borderwidth=0,
                    font=('san serif', 30, 'italic bold underline'))
copyWrite.tag_configure("superscript", offset=10)
copyWrite.insert("insert", "", "", "", "")
copyWrite.configure(state="disabled", fg="red")
copyWrite.pack(side="left")
copyWrite.place(x=800, y=750)

window.mainloop()
```

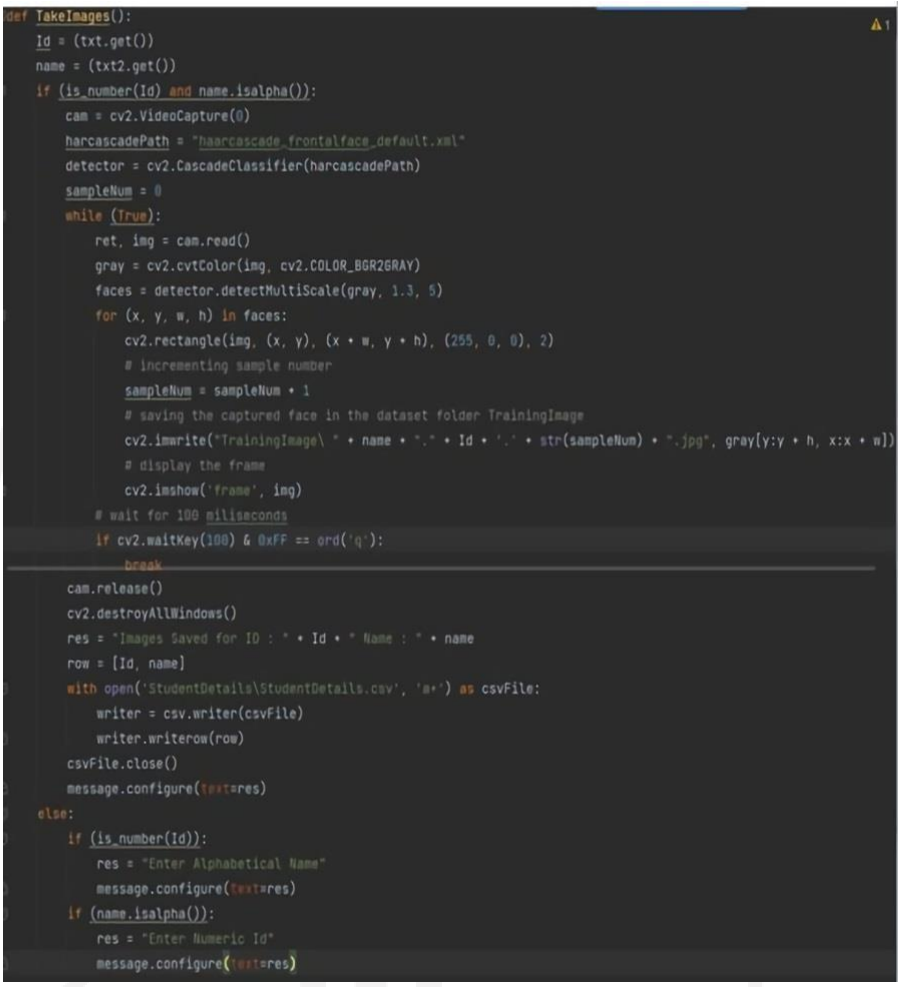
Figure 4.22 Clear Button Code



```python
def clear():
    txt.delete(0, 'end')
    res = ""
    message.configure(text=res)


def clear2():
    txt2.delete(0, 'end')
    res = ""
    message.configure(text=res)


def is_number(s):
    try:
        float(s)
        return True
    except ValueError:
        pass

    try:
        import unicodedata
        unicodedata.numeric(s)
        return True
    except (TypeError, ValueError):
        pass

    return False
```

Figure 4.23 Function of Clear Button Code

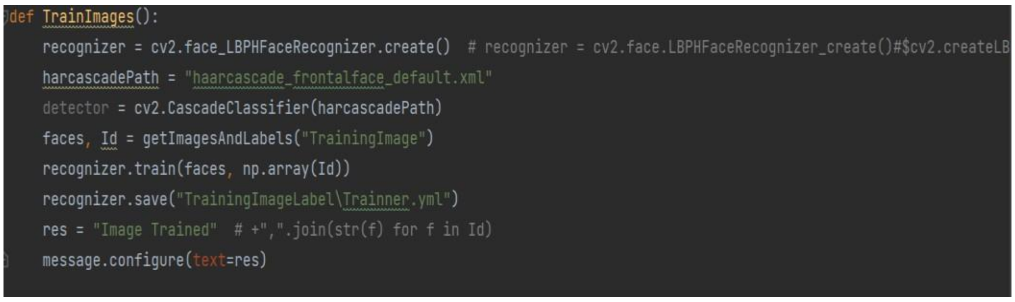
The code above have function to clear the entire field if it is/was been filled before. Figure 4.24 shows the facial detection, taking images and data storing path code.



```
def TakeImages():
    Id = (txt.get())
    name = (txt2.get())
    if (is_number(Id) and name.isalpha()):
        cam = cv2.VideoCapture(0)
        harcascadePath = "haarcascade_frontalface_default.xml"
        detector = cv2.CascadeClassifier(harcascadePath)
        sampleNum = 0
        while (True):
            ret, img = cam.read()
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray, 1.3, 5)
            for (x, y, w, h) in faces:
                cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
                # incrementing sample number
                sampleNum = sampleNum + 1
                # saving the captured face in the dataset folder TrainingImage
                cv2.imwrite("TrainingImage\ " + name + "." + Id + '.' + str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])
                # display the frame
                cv2.imshow('frame', img)
            # wait for 100 miliseconds
            if cv2.waitKey(100) & 0xFF == ord('q'):
                break
        cam.release()
        cv2.destroyAllWindows()
        res = "Images Saved for ID : " + Id + " Name : " + name
        row = [Id, name]
        with open('StudentDetails\StudentDetails.csv', 'a+') as csvFile:
            writer = csv.writer(csvFile)
            writer.writerow(row)
        csvFile.close()
        message.configure(text=res)
    else:
        if (is_number(Id)):
            res = "Enter Alphabetical Name"
            message.configure(text=res)
        if (name.isalpha()):
            res = "Enter Numeric Id"
            message.configure(text=res)
```

Figure 4.24 Facial Detection, Take Image and Data Storing Path Code

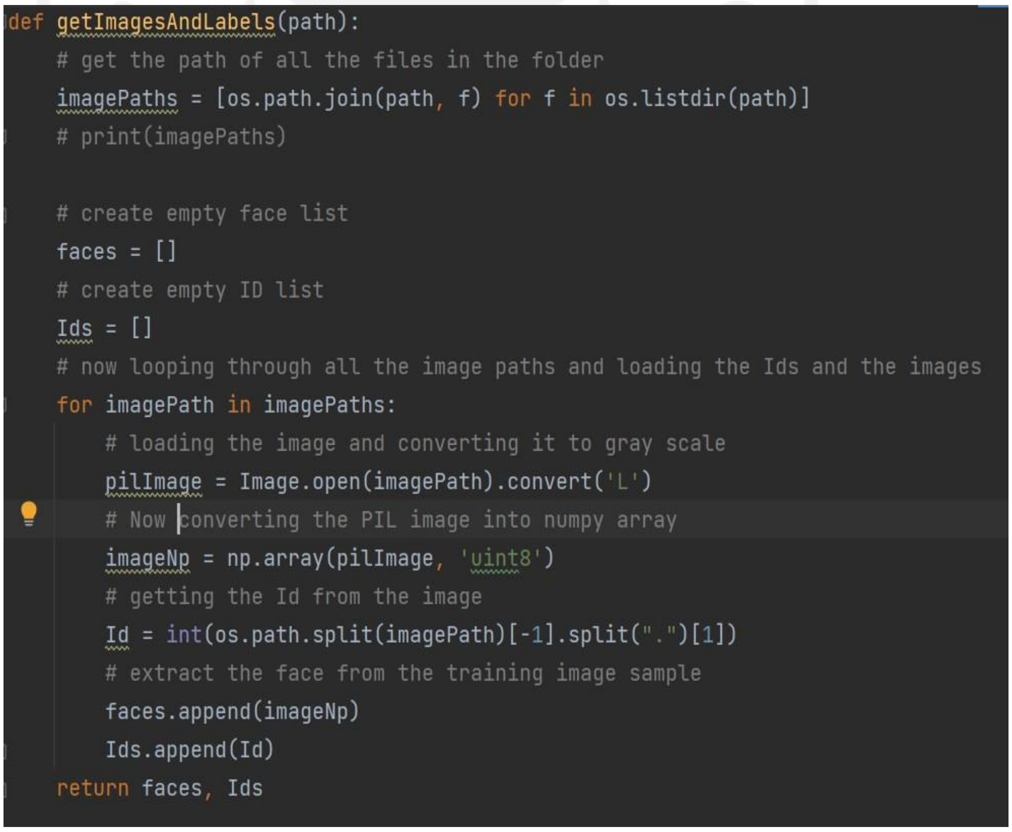
This code in Figure 4.24 shows the function definition of taking an image. This def takeimage is the command line to capture images from the camera. The detector used in this def is designated to locate and detect the face in the frame image. To analyze the face that the camera can detect is by employing the cascade classifier in opencv. After that, the images are saved and stored in a database folder called TrainingImage. The code above also generates the name and id field to be filled. The filled field of name and id will be saved into csv file called StudentDetails.csv. Once both pictures and identity are already saved and stored, the system can directly recognize the user. The train image code is shown in Figure 4.25



```
def TrainImages():
    recognizer = cv2.face_LBPHFaceRecognizer.create()  # recognizer = cv2.face.LBPHFaceRecognizer_create()#$cv2.createLB
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(harcascadePath)
    faces, Id = getImagesAndLabels("TrainingImage")
    recognizer.train(faces, np.array(Id))
    recognizer.save("TrainingImageLabel\Trainner.yml")
    res = "Image Trained"  # +",".join(str(f) for f in Id)
    message.configure(text=res)
```

Figure 4.25 Train Image Code

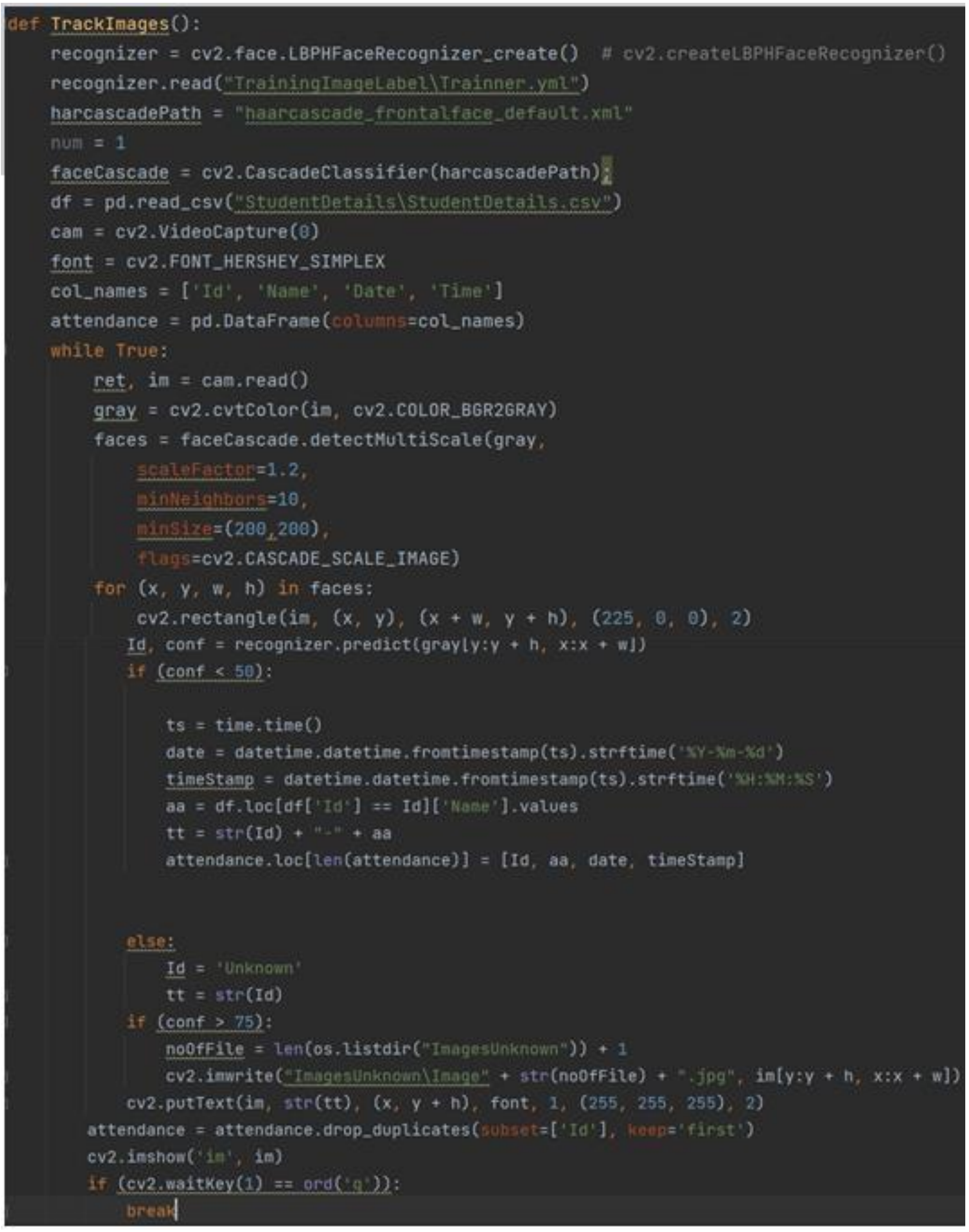
Figure 4.25 shows the training of the image code. A captured image trained with an algorithm, which is LBPH, can analyze that captured images consists of the facial of someone. Once the images are trained completely, the trainer.yml extension file is generated. This file utilized the entire process of face detection and face recognition. Figure 4.26 shows the labeling image code



```
def getImagesAndLabels(path):
    # get the path of all the files in the folder
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
    # print(imagePaths)

    # create empty face list
    faces = []
    # create empty ID list
    Ids = []
    # now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePaths:
        # loading the image and converting it to gray scale
        pilImage = Image.open(imagePath).convert('L')
        # Now converting the PIL image into numpy array
        imageNp = np.array(pilImage, 'uint8')
        # getting the Id from the image
        Id = int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
        Ids.append(Id)
    return faces, Ids
```

Figure 4.26 Labeling Image Code

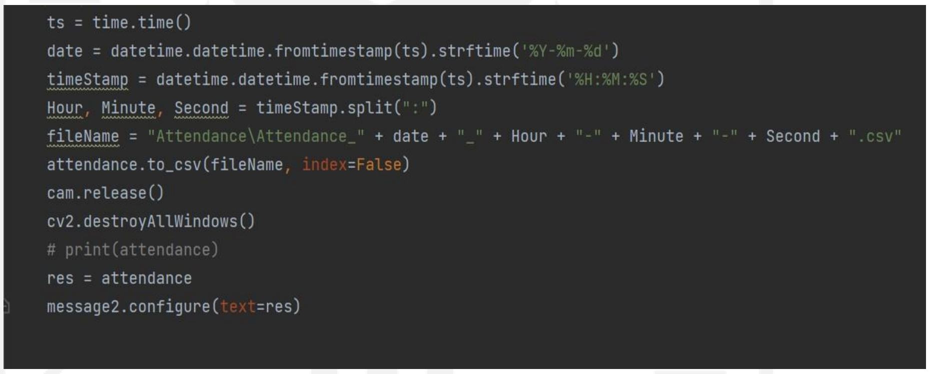
This code above contains the function definition about getting image and label. After the saved data about identity and images are already taken for trained data, the system will get those files and folder paths. After that system will loop those paths and load the identity and data images. Once the load has been completed, the images will be converted into a gray scale (black and white). The converted images will be transformed into numpy array. Figure 4.27 shows the track image and facial recognition code



```python
def TrackImages():
    recognizer = cv2.face.LBPHFaceRecognizer_create()  # cv2.createLBPHFaceRecognizer()
    recognizer.read("TrainingImageLabel\Trainner.yml")
    harcascadePath = "haarcascade_frontalface_default.xml"
    num = 1
    faceCascade = cv2.CascadeClassifier(harcascadePath);
    df = pd.read_csv("StudentDetails\StudentDetails.csv")
    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names = ['Id', 'Name', 'Date', 'Time']
    attendance = pd.DataFrame(columns=col_names)
    while True:
        ret, im = cam.read()
        gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
        faces = faceCascade.detectMultiScale(gray,
            scaleFactor=1.2,
            minNeighbors=10,
            minSize=(200,200),
            flags=cv2.CASCADE_SCALE_IMAGE)
        for (x, y, w, h) in faces:
            cv2.rectangle(im, (x, y), (x + w, y + h), (225, 0, 0), 2)
            Id, conf = recognizer.predict(gray[y:y + h, x:x + w])
            if (conf < 50):

                ts = time.time()
                date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
                timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa = df.loc[df['Id'] == Id]['Name'].values
                tt = str(Id) + "-" + aa
                attendance.loc[len(attendance)] = [Id, aa, date, timeStamp]


            else:
                Id = 'Unknown'
                tt = str(Id)
            if (conf > 75):
                noOfFile = len(os.listdir("ImagesUnknown")) + 1
                cv2.imwrite("ImagesUnknown\Image" + str(noOfFile) + ".jpg", im[y:y + h, x:x + w])
            cv2.putText(im, str(tt), (x, y + h), font, 1, (255, 255, 255), 2)
        attendance = attendance.drop_duplicates(subset=['Id'], keep='first')
        cv2.imshow('im', im)
        if (cv2.waitKey(1) == ord('q')):
            break
```

Figure 4.27 Track Image and Facial Recognition Code

Track image function defined above is the facial recognition key component. The track image code tells about how the captured images of a live-streamed in real time scenario can match with the trained images. That was when it was called facial recognition. So, to perform the facial recognition, the camera will be turned on and streamed based on real-time. After the camera can be streamed in real-time, there will be a facial box, and facial box will seek the face of the user. After that, the camera will take a picture several times, and those pictures contain the face of the user which is called data testing. Moreover, the system will be analyzed and match-make between data testing and data training. Once it is matched, the system will show the identity of the user, which means that the facial recognition is completed. Figure 4.28 shows the set data/time and storing attendance code.



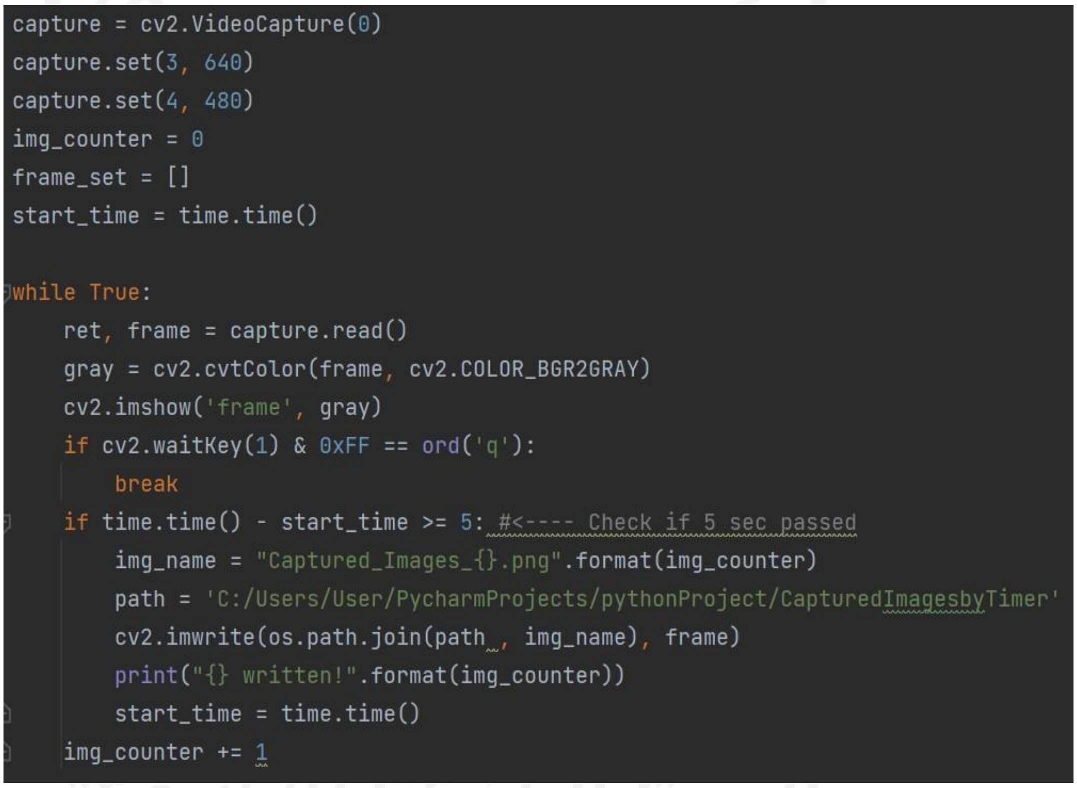
```
ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
Hour, Minute, Second = timeStamp.split(":")
fileName = "Attendance\Attendance_" + date + "_" + Hour + "-" + Minute + "-" + Second + ".csv"
attendance.to_csv(fileName, index=False)
cam.release()
cv2.destroyAllWindows()
# print(attendance)
res = attendance
message2.configure(text=res)
```

Figure 4.28 Set Date/Time and Storing Attendance Code

Once the facial recognition has been completed, the system will record the exact time when the facial recognition was done. Recorded data contains the date, time, id, and name of the person. It is stored in csv files in the attendance folder, and the file name is based on the date and time on real time.

4. Programming Code of Picture Capture by Timer Feature

After the attendance management is done, these commands will be executed. Figure 4.29 shows the command of picture capture by timer. In this code, the system will capture the image for every determined 5 seconds interval. After the images have been captured, images will be stored in the database. The folder for those taken images is “Captured Images by Timer”. The picture captured by timer code is shown in Figure 4.29



```
capture = cv2.VideoCapture(0)
capture.set(3, 640)
capture.set(4, 480)
img_counter = 0
frame_set = []
start_time = time.time()

while True:
    ret, frame = capture.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    cv2.imshow('frame', gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    if time.time() - start_time >= 5: #<---- Check if 5 sec passed
        img_name = "Captured_Images_{}.png".format(img_counter)
        path = 'C:/Users/User/PycharmProjects/pythonProject/CapturedImagesbyTimer'
        cv2.imwrite(os.path.join(path , img_name), frame)
        print("{} written!".format(img_counter))
        start_time = time.time()
    img_counter += 1
```

Figure 4.29 Picture Capture by Timer Code

4.2.2.4 The Algorithm of Face Detection and Face Recognition

Discussing the algorithm, the researcher used the common algorithm in this research, which are Haar-Cascade and LBPH. To understand the Haar-Cascade LBPH algorithm as a whole, the researcher will explain both algorithms. Face detection is to find the location and size of the face in the image, and Haar-Cascade is used to extract the face within the image. Face Recognition finds the characteristics and describes the facials in a picture, and the LBPH algorithm is used for feature extraction in the picture that has already been converted from colored images into gray-scale for face recognition. (Ahmed, Guo, Ali, & Deeba, 2018). Face Recognition, in simple words, compares the trained data with the testing data.

Haar-cascade is a machine learning technique for identifying individual faces in video or image captures. The process has been improved over a large number of positive and negative images. It is then used to recognize human faces in photographs. In general, the system can recognize the human face and parts of it. The haar-cascade algorithm was utilized to detect a human's face in this task. Therefore to train the classifier, the suggested system collects a large number of positive and negative images.

The system must then extract features from the data. The haar is derived from the training data. Consider surrounding rectangular rectangles in the group image detection window at a specific location. Eyes, noses, and cheeks are common features on all human faces. The face attributes are compared in this study utilizing a color variation of brightness. It was the first technique for real-time object detection in computer vision. It was, however, mostly employed to detect faces. In order to recognize faces, the method uses four phases: feature selection, integral image, Adaboost training, and cascade classifier. (Pandey, 2018)

The classifier begins by collecting Haar features from each image after a massive amount of training data (in the form of photos) is supplied into the system. Haar Features are a type of convolution kernel that is used to determine whether or not a suitable feature is present on an image. The following are some examples of Haar features: (Viola & Jones, 2001) shown in Figure 4.30

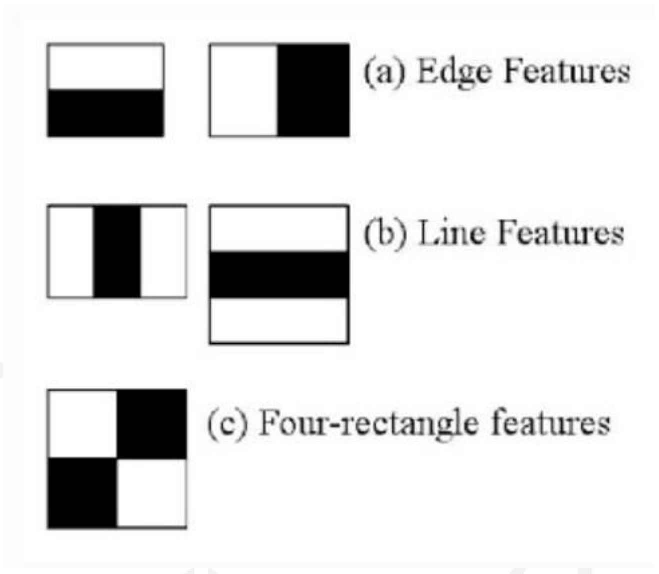


Figure 4.30 Haar-Cascade Computation Feature

These Haar Features act as windows on images, allowing a single feature to be computed. Their characteristic is essentially a single number derived by subtracting the sum of pixels in the white and black regions. In Figure 4.31 below, the process of Haar-Cascade computation feature illustration is easily visualized. (Pandey, 2018)

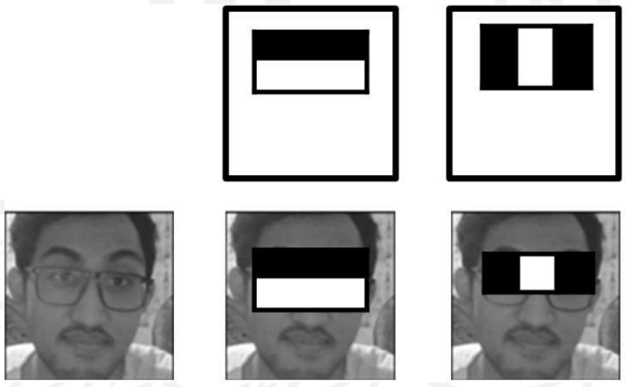


Figure 4.31 Haar-Cascade Computation Feature Illustration

Let's pretend the researcher is only extracting two features for the sake of demonstration, thus, there are only two windows here. The first distinguishes itself by the fact that the eye area is darker than the adjacent cheeks and nose. The second distinguishing trait is that the eyes are slightly darker than the bridge of the nose. As a result, when the feature window moves over the eyes, a single value is calculated. This figure will then be compared to a threshold, and if it passes, it will be concluded that there is a benefit or advantage here.

Integral Image (Summed Area Table) is an intermediate representation for the image for computing rectangle features very rapidly. An Integral image at location x,y contains pixels above and to the left of x,y. (Taqqiyah, Lawi, & Galsan, 2019). The integral images Viola Jones' technique has a 24X24 base window size, which means that more than 180,000 features would be calculated in this window. (Maneetsingh, 2014). Consider computing the pixel difference for each feature separately. The Integral Image concept was proposed as a solution for this computationally intensive technique. The integral image means that we only need the four corner values to find the sum of all pixels beneath any rectangle. (Pandey, 2018) shown in Figure 4.32

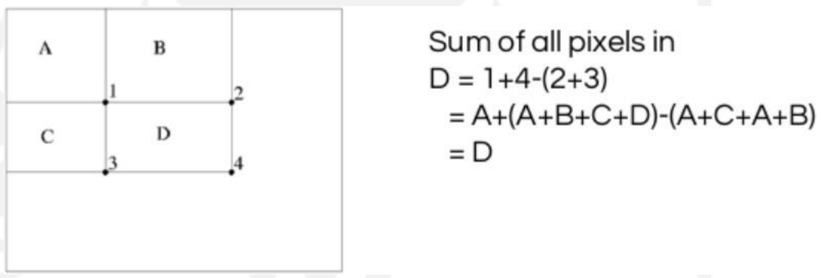


Figure 4.32 Integral Image Computation Illustrations

Within a 24X24 window, there are over 180,000 feature values, as mentioned previously. However, not all features can be used to identify a person's face. A machine learning algorithm called Adaboost is used to only select the best feature from the full block. It essentially selects only those features that assist in improving the classifier's accuracy. It accomplishes this by building a strong classifier that is a linear combination of several weak classifiers. It dramatically reduces the number of characteristics from over 180,000 to around 6000. (Pandey, 2018)

The cascade classifier essentially consists of stages, where each stage consists of a strong classifier. Each stage in the cascade reduces the false positive rate and decreases the detection rate. A target is selected for the minimum reduction in false positives and the maximum decrease in detection. Each stage is trained by adding features until the target detection and false positive rates are met (these rates are determined by testing the detector on a validation set). Stages are added until the overall target for false positive and detection rate is met. The process can be understood with the help of the diagram by (Mooseop, Deokgyu, & Ki-Young, 2015) that shown in Figure 4.33

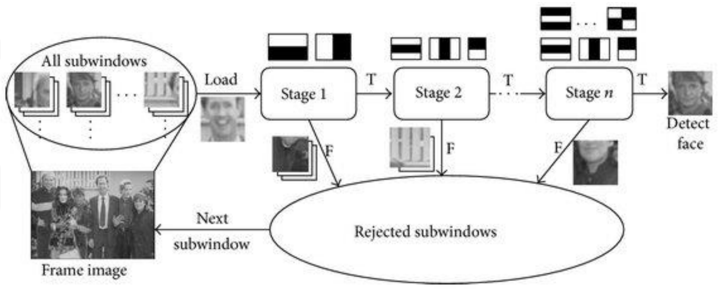


Figure 4.33 Haar-Cascade Process Diagram

The Local Binary Patterns Histograms (LBPH) was introduced in 2006. (Ahonen, T, & M.Pietikinen, 2006). LBPH algorithm was commonly used for facial recognition. This algorithm is based on the local- binary-operator (P.M & Hardwood, 1996), broadly implemented in face recognition due to its discriminating strength and calculation easiness (Pietikinen, 2010). So, basically, LBPH is the combination of Local Binary Pattern (LBP) and Histograms of Oriented Gradients (HOG) descriptor. LBP itself has since been found to be a powerful feature for texture classification. On some datasets, it has also been discovered that combining LBP with the histograms of oriented gradients (HOG) descriptor boosts detection performance significantly. The Local Binary Pattern Histogram texture operator is a significant and simple texture operator. The LBP is applied in a range of biometrics applications, including eye localization, iris and palm print identification, and age categorization.

Moreover, that combination has a goal to detect and locate the facials in the images with a simple data vector. (Kumar, Ravi, & Dinesh, 2014). The whole idea is designed to texture analysis for the gray-scale image. Thus, to detect faces in an RGB (Red, Green, and Blue) or colored photos by converting it first into the black-and-white or gray-scale image. (Deeba, Ahmed, Memon, Dharejo, & Ghaffar, 2019). There are several steps to identify in detail about the LBPH algorithm. For example, parameters, training the algorithm, applying the LBP operation, and extracting the histogram. The histogram has the human face description with three levels of the locality. Each label comprises pixel-level, region level, and overall information about the face. (Guoying & Pietikainen, 2007)

There are 4 parameters that are being used in LBPH recording to (Salton, 2017). Those parameters are radius, neighbors, grid-x, and grid-y. Radius represents the radius enclosing the central pixel and is used to generate the circular local binary pattern. Normally, it is set to 1. Neighbor is the number of sample points required to construct a circular local binary pattern. Keep in mind that the higher the computational cost, the more sample points you include. Normally, it is set to 8. Grid-X is the number of cells in the horizontal direction. The higher the dimensionality of the generated feature vector, the more cells there are in the grid and the finer the grid is. It's generally set about 8. Grid-Y is the number of cells in the vertical direction. Same with grid-x, the higher the dimensionality of the generated feature vector, the more cells there are in the grid, and the finer the grid is.

To train the algorithm firstly is by collecting a dataset, consisting of identity information, which can be a name or ID and pictures of someone’s facial who wants to be recognized. That picture will be trained, while the identity information will be used as the label for the inputted image. The labels serve as image IDs, thus if there are multiple images of the same subject or same person. Those pictures should all have the same labels (Kelvin S, 2017). The LBP's first computational step is to build an intermediate image that better describes the original image by highlighting the facial features. Therefore, in order to do so, the algorithm employs a sliding window idea based on the radius and neighbors parameters. The LBP computational figure describes in Figure 4.34 below

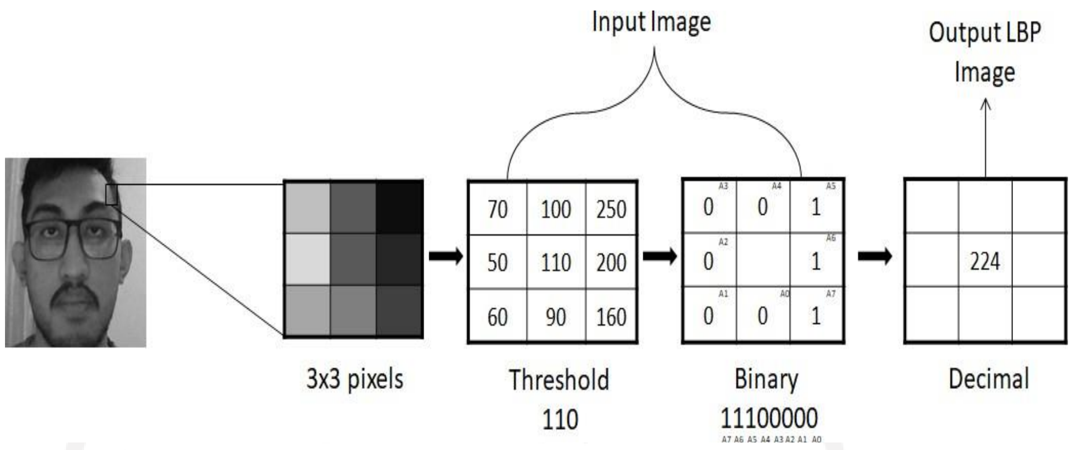


Figure 4.34 LBP Computational

So based on the figure above, for example, the researcher has an image in grayscale. In that photo, actually, there are a lot of pixels, yet the researcher takes a small part of the pixels slightly above the left eyebrow. For that small part taken, it represents 3x3 pixels. 3x3 pixels basically is a 3x3 matrix, which can represent the intensity of the color of each pixel (0-255). Take example the value of the central matrix is 110, and it is used as a threshold. This value is used for determining the new values from the 8 neighbors. For every 8 neighbors, to set the value, it referred to the central matrix or threshold. If the value is greater than the threshold, it equals to 1. Otherwise, if the value is less than the threshold, it equals to 0. (Rosebrock, 2015). Now, there is a new 3x3 matrix that consists of 0 and 1. If, for example, the value at middle-low or at the center-bottom point and work out way clockwise accumulating the 0 and 1 binary string as its go along with. It results in binary 8-bit equals to 11100000. That binary is converted into decimal to get the output of the LBP image. The LBP procedure represents a better characteristic of the original image. As a consideration, The LBP procedure was expanded to use a different number of radius and neighbors, called Circular LBP. Circular LBP is an extension of the LBP process that uses a different number of radius and neighbors. (Salton, 2017) shown in Figure 4.35 below

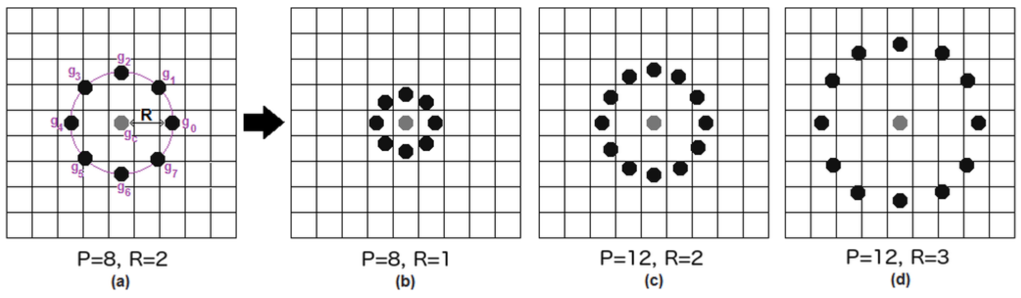


Figure 4.35 Circular LBP

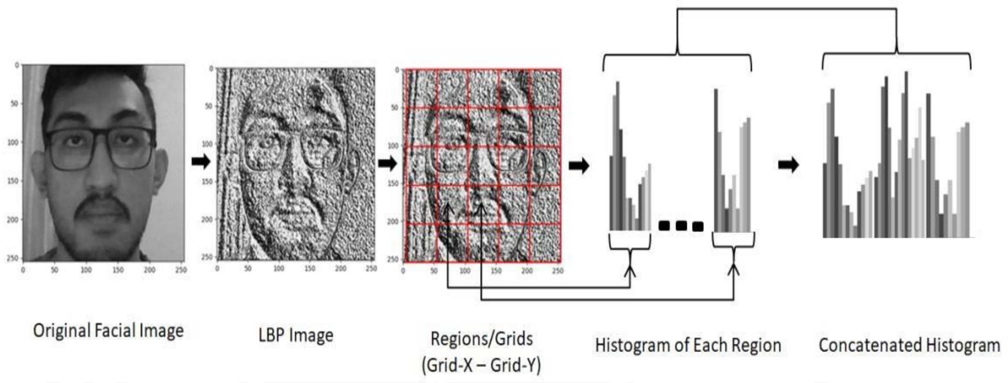


Figure 4.36 Extracting Histogram of LBP (LBPH)

Figure 4.36 shows about extracting histogram of LBP. For extracting it, grid-x and grid-y parameters are required to divide the images into multiple grids, as it can be seen in the figure above. This figure refers to a figure that has been made by (Salton, 2017). Based on the image above, a histogram is extracted of each region. This kind of action is explained in detail. From the grayscale of original facial picture, the researcher conducted the LBP computation, resulting in LBP image. By adding some grids of the X-axis and Y-axis, each grid or histograms contain only 256 positions (0-255) representing the occurrences of each pixel intensity. After that, each histogram is integrated to create a new and bigger histogram. For example, based on the figure above, the grid are 5x5, so 5x5x255 = 6.375 positions in the final integrated histogram. The characteristics of the original facial image are represented by the final integrated histogram. Finally, all four steps that have been explained pretty much describe how the LBPH Algorithm works.

4.2.2.5 The Design of Attendance Management System Interface

As explained above in the code section, there are a lot of code in terms of creating the interface of the attendance management system. Figure 4.19, and Figure 4.20 above show the code that construct the architecture of the interface. When the code is executed, the design of the system will appear. This user interface come up with functionality and simplicity. The purpose of this design interface to make a quite simple yet can execute the attendance. The function of buttons purposely to execute every code inside the system to run. The clear button is for erasing all of the filled fields. “Scan New Face” button is for face detection to collect the data. “Process New Face” button is to train the collected data to be data training. “Click to Attend” Button is designed to conduct the attendance by conducting facial recognition. Facial recognition might happen because the system will make a comparison between real-time image taken (data testing) with trained data (data training). The notification field is to notify the user for data training. The attendance field notifies the user when the attendance is recorded. The design interface of the attendance management system is shown in Figure 4.37 below

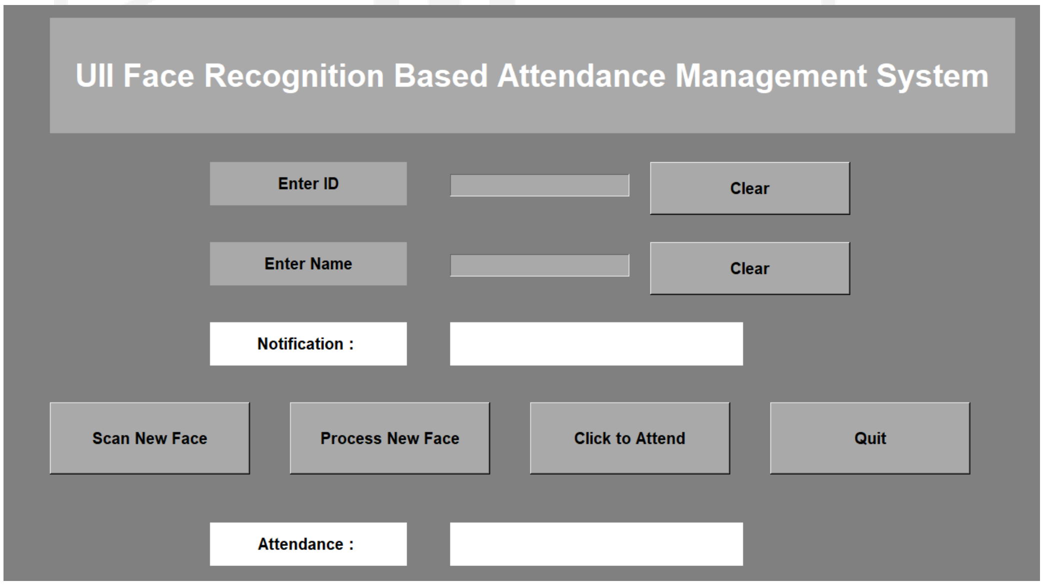


Figure 4.37 Interface Design

CHAPTER V

RESULT AND DISCUSSION

In this chapter, after data has been collected and processed, the researcher reads the result and discusses the result from the data processing. Here, the researcher will explain and report the meaning of the information without any speculation. The researcher will interpret the facts and describe the significance of the finding's result. The discussion is also about investigating, clarifying, and also understanding the result’s insight that emerges as an answer for this research. Thus, researcher will be explained result and discussion in whole detailed down below.

5.1 System Testing and System Testing Result

After the code has been written, and then the code needs to be executed. After the code has been executed, the program will start working. Since the module of liveness detection was code was written in the early sequence. Hence, the liveness detection will start working at first. Then after the liveness detection completely runs well, the attendance system starts running. Last but not least, the picture capture timer based will start working after the attendance system is completely working.

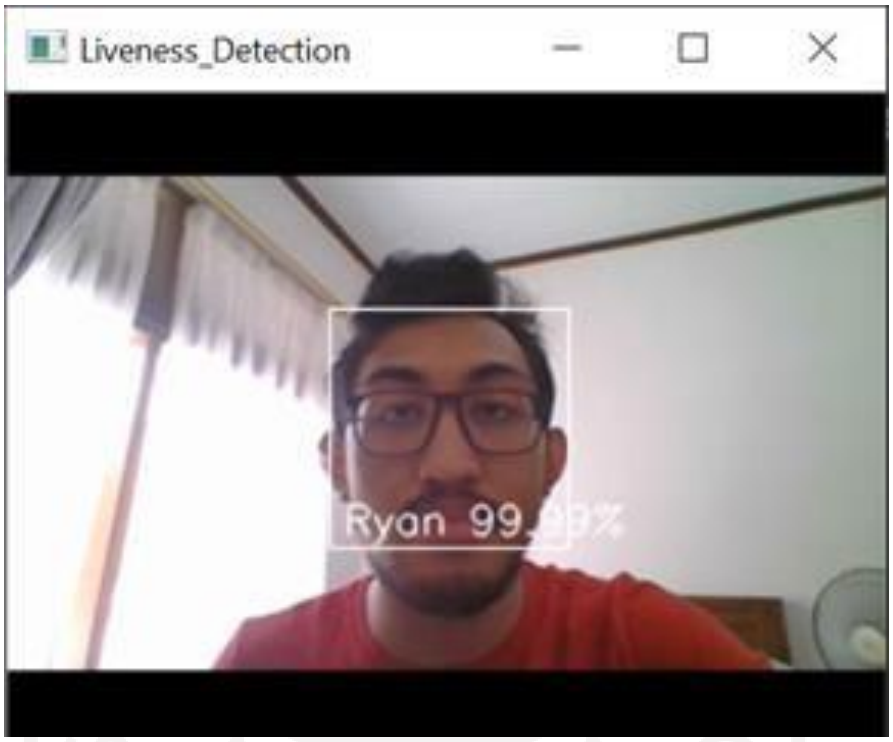
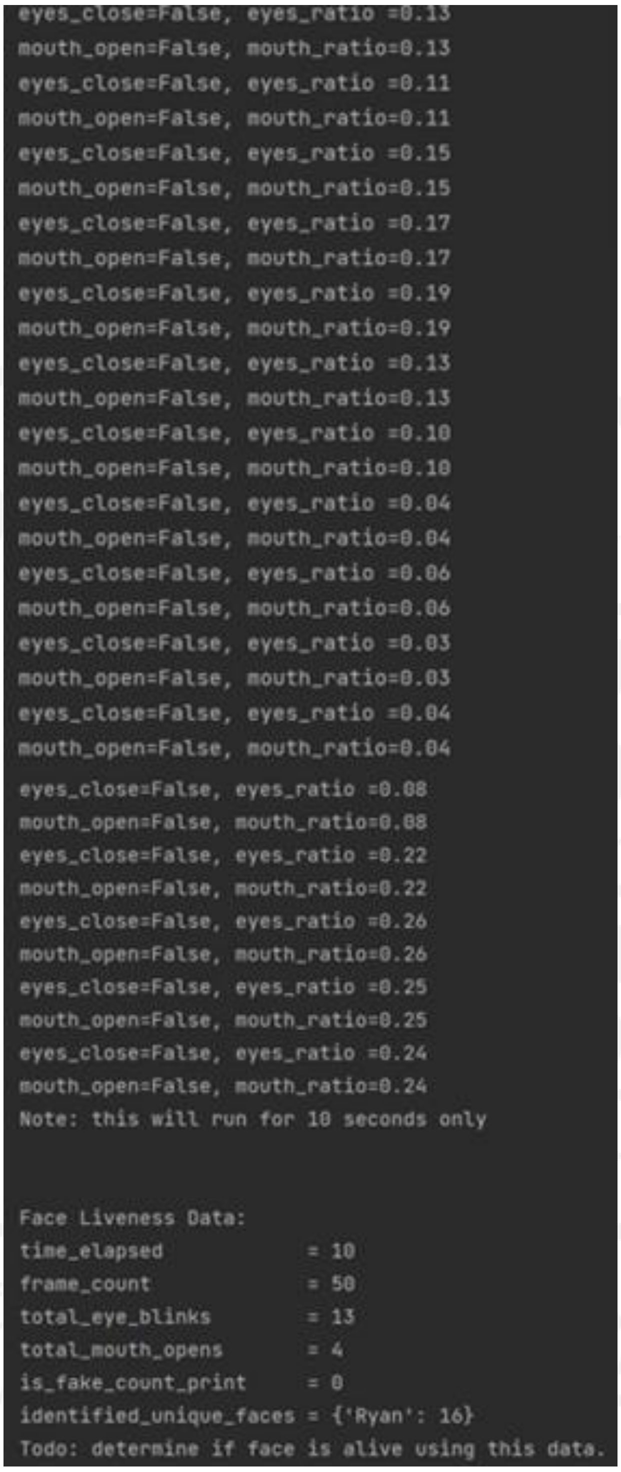


Figure 5.1 Liveness Detection

Figure 5.1 above shows how the liveness detection works. The system can determine whether the picture is real facial capture or fake. The terminal section of the python shows the mouth counter and eyes blinking counter. Those counters can detect the real facial in real time and as an indicator that the image frame taken is alive. The liveness detection stats result is shown in Figure 5.2



```
eyes_close=False, eyes_ratio =0.13
mouth_open=False, mouth_ratio=0.13
eyes_close=False, eyes_ratio =0.11
mouth_open=False, mouth_ratio=0.11
eyes_close=False, eyes_ratio =0.15
mouth_open=False, mouth_ratio=0.15
eyes_close=False, eyes_ratio =0.17
mouth_open=False, mouth_ratio=0.17
eyes_close=False, eyes_ratio =0.19
mouth_open=False, mouth_ratio=0.19
eyes_close=False, eyes_ratio =0.13
mouth_open=False, mouth_ratio=0.13
eyes_close=False, eyes_ratio =0.10
mouth_open=False, mouth_ratio=0.10
eyes_close=False, eyes_ratio =0.04
mouth_open=False, mouth_ratio=0.04
eyes_close=False, eyes_ratio =0.06
mouth_open=False, mouth_ratio=0.06
eyes_close=False, eyes_ratio =0.03
mouth_open=False, mouth_ratio=0.03
eyes_close=False, eyes_ratio =0.04
mouth_open=False, mouth_ratio=0.04
eyes_close=False, eyes_ratio =0.08
mouth_open=False, mouth_ratio=0.08
eyes_close=False, eyes_ratio =0.22
mouth_open=False, mouth_ratio=0.22
eyes_close=False, eyes_ratio =0.26
mouth_open=False, mouth_ratio=0.26
eyes_close=False, eyes_ratio =0.25
mouth_open=False, mouth_ratio=0.25
eyes_close=False, eyes_ratio =0.24
mouth_open=False, mouth_ratio=0.24
Note: this will run for 10 seconds only


Face Liveness Data:
time_elapsed           = 10
frame_count            = 50
total_eye_blinks       = 13
total_mouth_opens      = 4
is_fake_count_print    = 0
identified_unique_faces = {'Ryan': 16}
Todo: determine if face is alive using this data.
```

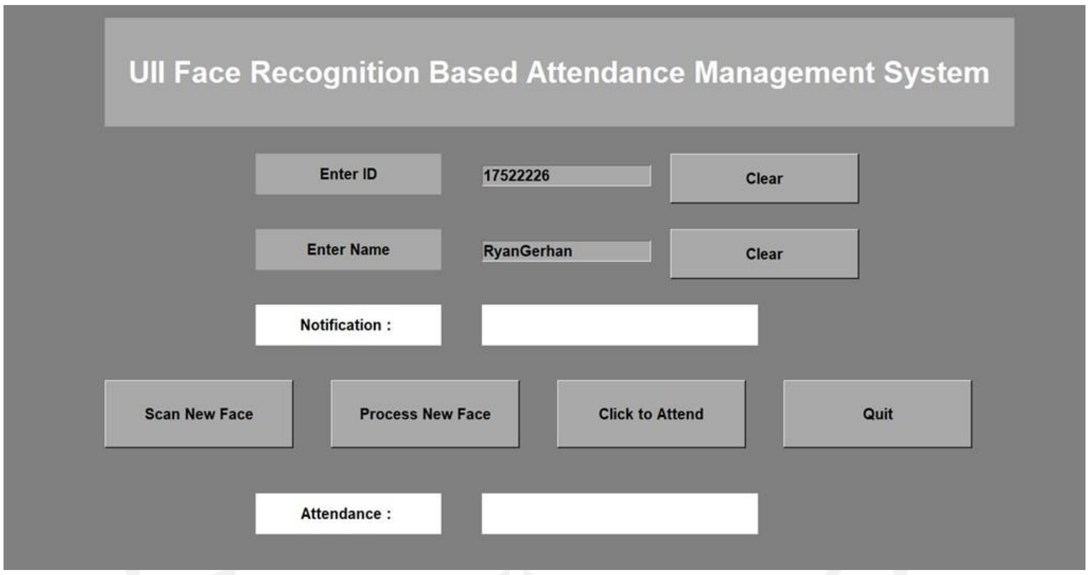
Figure 5.2 Liveness Detection Result Stats

As Figure 5.2 above shows, that is a result of liveness detection. Face liveness data stats shows about the time_elapsed, frame_count, total_eye_blinks, total_mouth_opens, is_fake_count_print, and identified_unique_faces. Time_elapsed means the total time spent during liveness detection, since the runtime code for the liveness detection is only 10 seconds, and then the time_elapsed showed 10 as 10 seconds. Frame_count showed

50 as the frame that has been captured from the video clip of 10 seconds record is 50 frames. Total_eye_blinks in those 50 frames captured were 13. The system can detect whether the eyes are blinking is by the eye aspect ratio (EAR). When the value of the eyes_ratio is 0.2 something (0.2xx), it means the eyes were opened. And then, if the value of eyes_ratio dropped as near as zero (0.0xx), it means the eyes were closed. Moreover, if the value of EAR shows those two types of values in a close gap, the system detects that the eyes are blinked.

The total_mouth_opens shows how many times the mouth were opened and shut in an interval of 10 seconds and in 50 total frames. As same as the EAR, mouth_ratio is also calculated by mouth aspect ratio or MAR. When the mouth_ratio value dropped to almost zero (0.0x) means that the mouth was opened and shut. Moreover, the result of mouth_ratio was 4, which means that in 10 seconds of video capture, the researcher opened and closed the mouth 4 times. Since the video captures user's facial, displays that there was movement in the eyes and mouth,which is represented by the stats of total_eyes_blinks and total_mouth_opens, and then the system can tell that there is no fake count. The stats of is_fake_count_print indicate 0, which means the face is the actual face. The system also tells that there were unique faces identified. Those unique faces belong to researchers who conduct liveness detection. Figure 5.2 above showed that 98.98% of the live recorded video and the data training were matched. Thus {'Ryan'=16} stats pointed out that unique faces belong to Ryan.

After liveness detection has been run, the attendance management interface will appear. The design of the attendance system interface is shown in Figure 4.35 above. There are fields that need to be filled by id and name of the user. Those fields can be cleared with a button on the right side of the interface. That button will completely erase the description box. Figure 5.3 shows the input identity information.


UII Face Recognition Based Attendance Management System

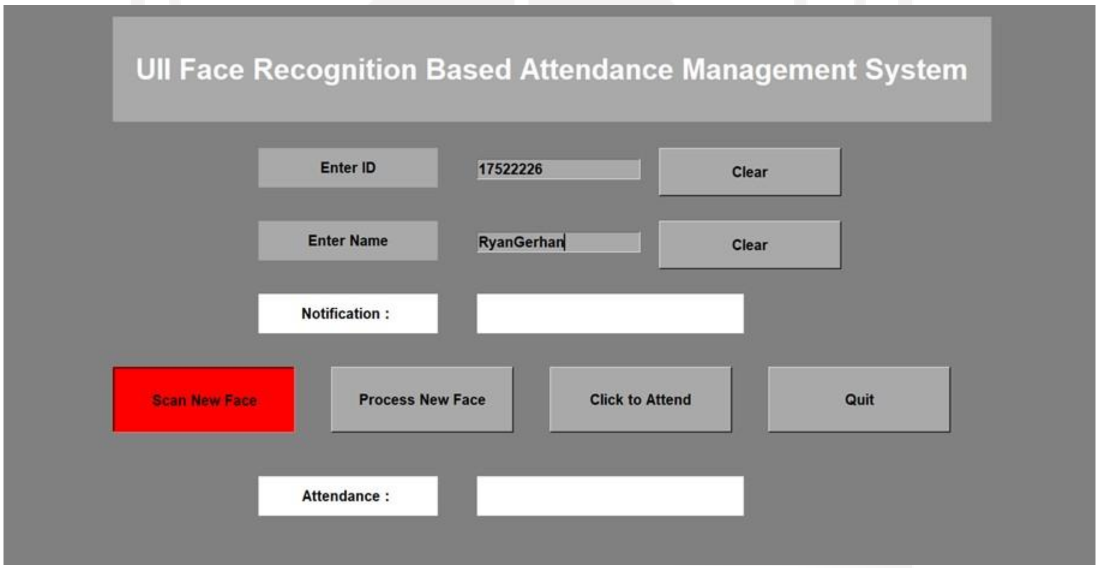
Enter ID: 17522226

Enter Name: RyanGerhan

Notification :

Attendance :

Figure 5.3 Input Identity Information


UII Face Recognition Based Attendance Management System

Enter ID: 17522226

Enter Name: RyanGerhan

Notification :

Attendance :

Figure 5.4 Scan New Face Clicked

Figure 5.4 shows the ID and Name description box filled with the identity of the user, which is the researcher himself. After filling the box, the user needs to click the "Scan New Face" button to proceed with taking the image for data training. As soon as the button is clicked, the button will change into red. It means the system has already confirmed the command and proceed to scan a new face. Figure 5.5 shows the face detection process

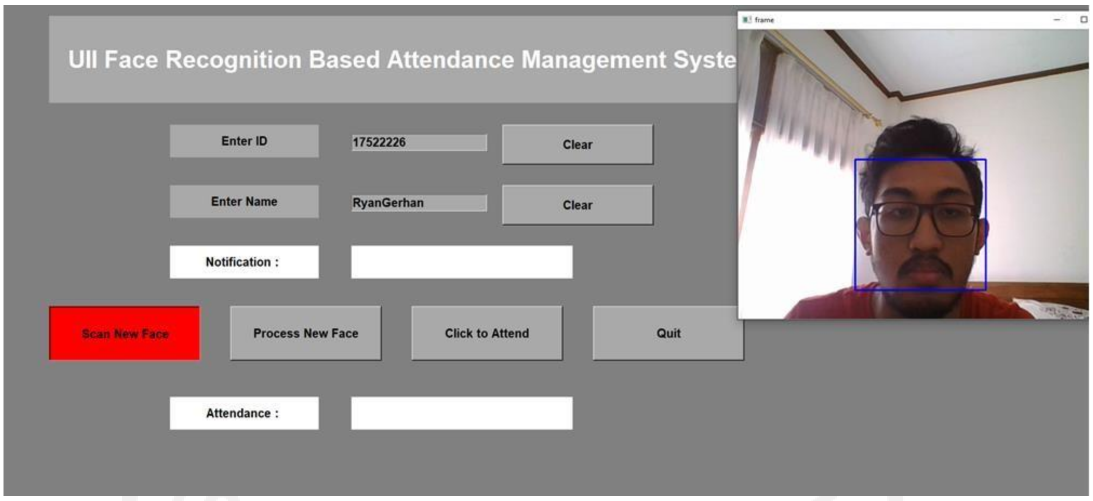
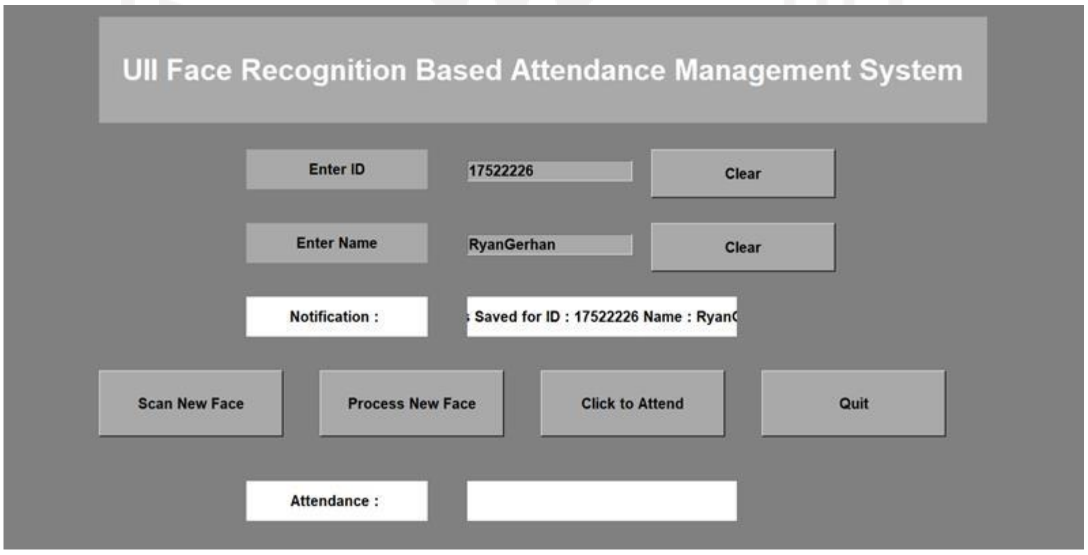


Figure 5.5 Face Detection

Figure 5.5 shows the next step after the button is clicked. The window called “frame” will appear, meaning that the system is ready to capture several images for data training. As it can be seen, the blue facial box indicates that the system has already detected the facial of the user. For every 0.5 seconds, the camera will take apicture of the user's facial area for about 60 pictures. It takes several seconds to be completed. Figure 5.6 shows the facial picture and ID saved.


UII Face Recognition Based Attendance Management System

Enter ID: 17522226

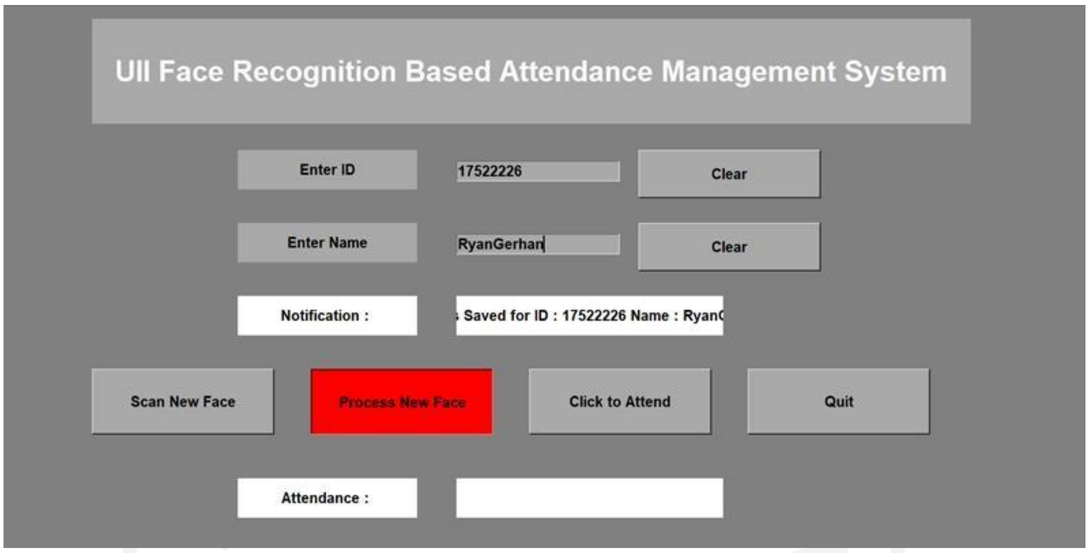
Enter Name: RyanGerhan

Notification : ; Saved for ID : 17522226 Name : Ryan(

Attendance :

Figure 5.6 Facial Picture and Identity Information Acquired

Figure 5.6 shows if the pictures have been completely taken and already stored 60 pictures of the user facial area. The notification section will notify users that their identity information is already saved. It means the system successfully stored the images, ID and user name. Figure 5.7 shows the process of a new face


UII Face Recognition Based Attendance Management System

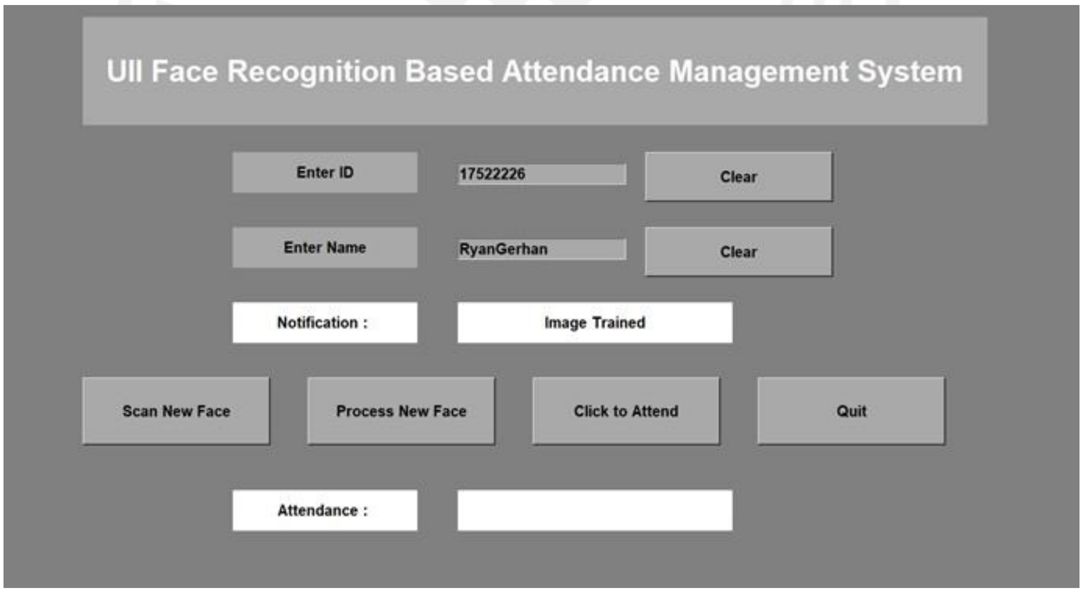
Enter ID 17522226 Clear

Enter Name RyanGerhan Clear

Notification : Saved for ID : 17522226 Name : Ryan

Figure 5.7 Process New Face Button Clicked

After the notification showed up, the next thing to do to proceed into the next step is to click the “Process New Face” button. As soon as the button has been clicked, the button will be turned into red, as it can be seen in Figure 5.7 above. It indicates that the system has already confirmed the next command is to be later processed. Figure 5.8 shows that the image has already been trained.


UII Face Recognition Based Attendance Management System

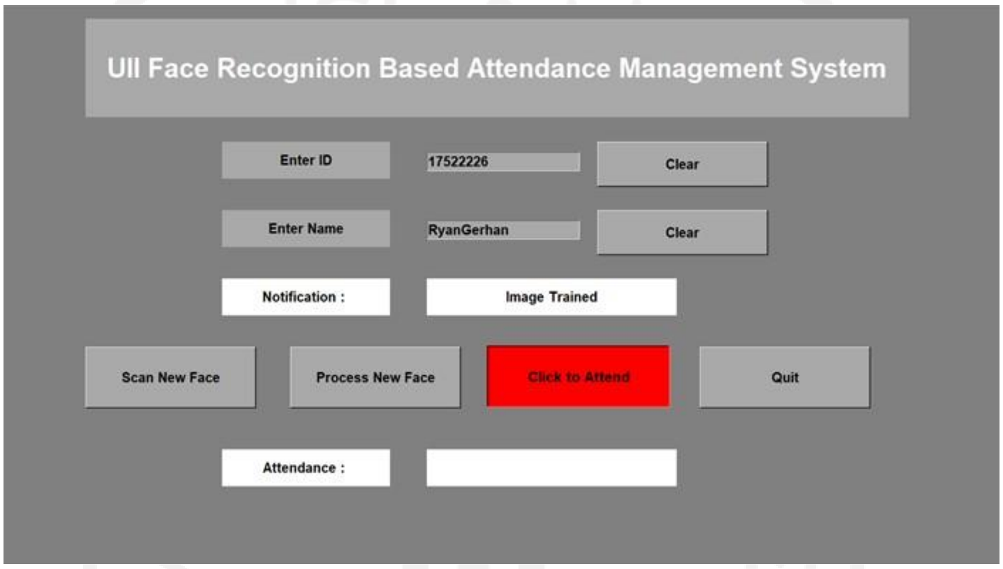
Enter ID 17522226 Clear

Enter Name RyanGerhan Clear

Notification : Image Trained

Figure 5.8 Process New Face Done and Image Trained

Moreover, if the process of creating a new face has already been done, the notification shows something. It says “Image Trained” which means the system already trained the images to be data training. This process includes labeling the images. Labeling images is the process when the system saves both pictures and identity information. In analogy, the system notices that those pictures that have been saved belong to the user who is certainly being scanned. Start from the filling up the boxes for id and name until the process new face has been completely done, pointed out by the notification said: “Image Trained”. Those steps only need to be done once. As long as the data of images and identity information is saved, users do not have to fill their name and ID and also do not have to scan new faces anymore. Figure 5.9 shows the clicked button of click to attend



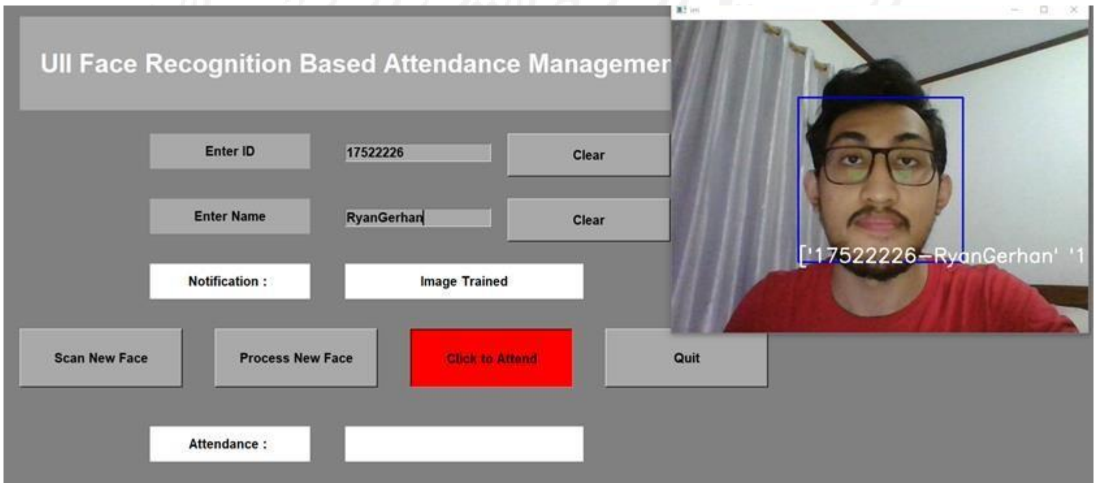
17522226

RyanGerhan

Image Trained

Figure 5.9 Conduct Attendance

After the notification box showed “Image Trained” notification, as shown by Figure 5.9 above, users need to click “Click to Attend” button to perform the attendance system. Figure 5.10 shows the facial recognition while doing the attendance



17522226

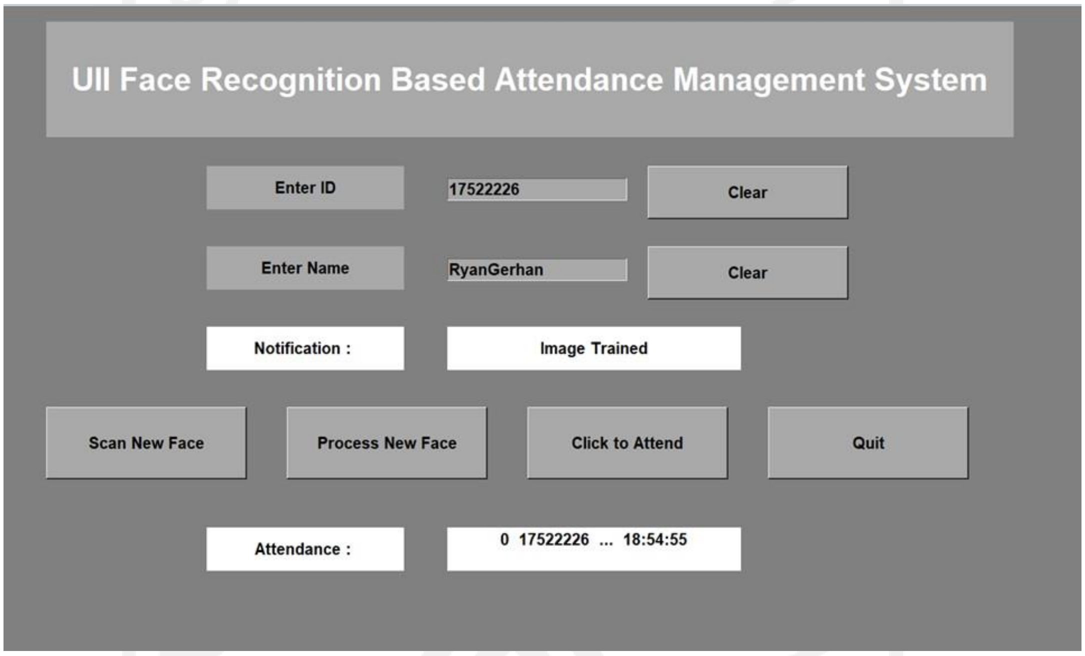
RyanGerhan

Image Trained

['17522226-RyanGerhan' '1

Figure 5.10 Facial Recognition

As soon as the button “Click to Attend” is clicked, there will be a window that pops out. The window titled “im” indicates that the system already ran the command to conduct the attendance. The blue facial box appears in the facial area of the user, it shows that the system can detect the face of the user. Basically, in this process, the system captures data testing and directly conducts match-making with the data training. Therefore, there is something different compared with Figure 5.5 above. This window shows the id and name of the user in the lower section of that facial box. This process exhibits that the system can recognize the user and can perform the attendance. Figure 5.11 shows the recorded attendance



17522226

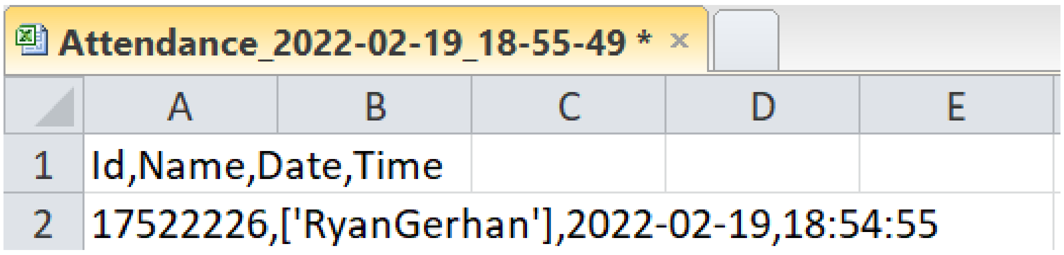
RyanGerhan

Notification : Image Trained

Attendance : 0 17522226 ... 18:54:55

Figure 5.11 Attendance Recorded

After the face has been recognized and attendance was conducted, a user needs to press (Q) from the keyboard to save the attendance. Attendance that has been saved will notify the user through the Attendance notification at the bottom of the interface. Once it shows the id and time, it means that the attendance has been completely done. Lastly, the user needs to click on the Quit button finishing the entire process of attendance. Figure 5.12 shows the recorded file


Attendance_2022-02-19_18-55-49 *

	A	B	C	D	E
1	Id,Name,Date,Time				
2	17522226,['RyanGerhan'],2022-02-19,18:54:55				

Figure 5.12 Attendance Record File

The recorded data of current attendance will be saved into csv file. That file is stored in the attendance database, which is a folder called "Attendance". Each process of attendance will show a different title of the attendance csv file. That current attendance performed by the researcher was conducted in 19th February 2022 at exactly 18:54:55 WIB. Figure 5.13 shows the captured picture by timer.

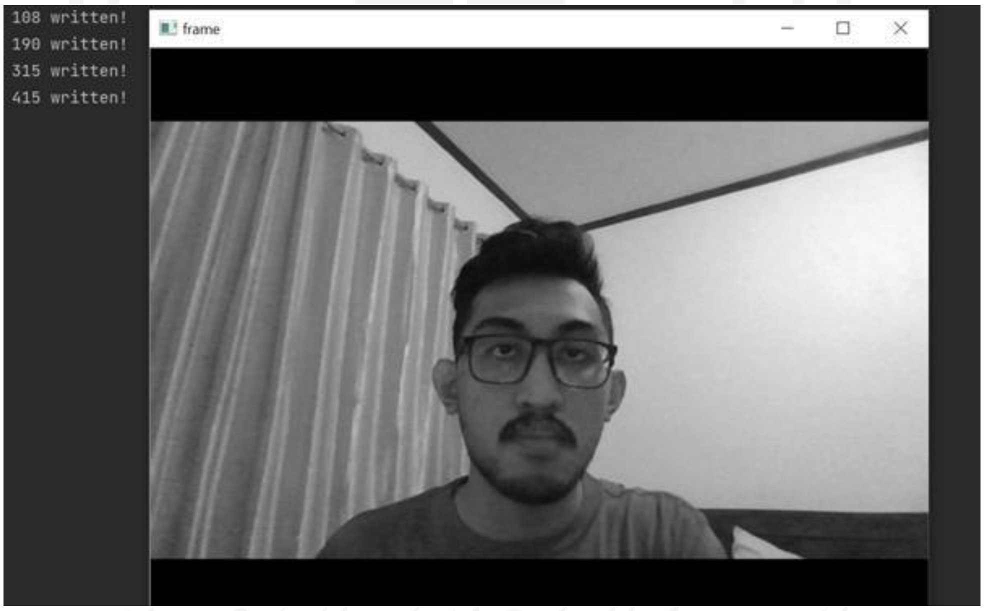

```
108 written!
190 written!
315 written!
415 written!
```

Figure 5.13 Capture Picture by Timer

After attendance management has been conducted, the system will directly generate a new "frame" window to capture a picture of the user by a timer within 5 seconds interval. As Figure 5.13 shows, there are notes written in the terminal with "108 written!; 190 written!; 315 written!; 415 written!". It means that in that frame picture has been captured. For example, "108 written" means that in frames counter 108 has been captured, and to notify the user, the system generates a notification with "108 written". After the frame is captured, and the frame has been written "{} written!" the user needs to press Q on the keyboard to end the whole system process. The captured picture file is shown in Figure 5.14

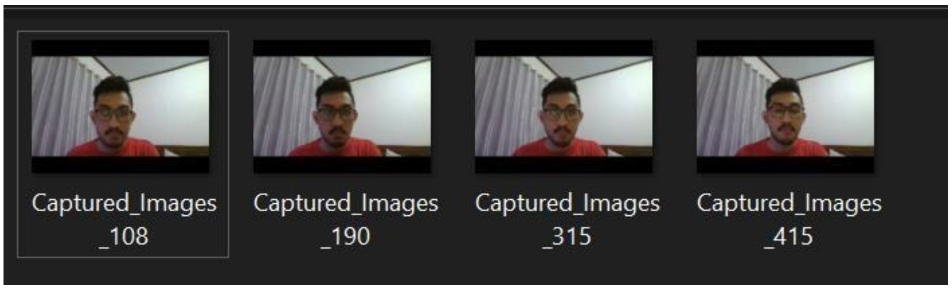


Figure 5.14 Captured Image Stored on Folder

The frame, shown in Figure 5.13 above, is in grayscale (black and white). Yet, in the database, the captured pictures are in color, as shown in Figure 5.14 above. These files are stored in "Captured Images by Timer" folder.

5.2 System Result Comparison

After discussing the system testing and system testing result, the researcher will discuss the system result comparison. Here, the comparison will show if the system completely works and can differentiate between whole processes. For example, the liveness detection, in this comparison system result, the researcher wanted to know if the liveness detection is completely working and can differentiate between the live stream actual face by camera and photo image. For the attendance system, here researcher wanted to explain that the attendance management system can conduct attendance directly without even inputting the identity information and images for data training anymore if the user has already saved and stored their identity and data training images. Figure 5.15 shows an attack or spoofing of liveness detection

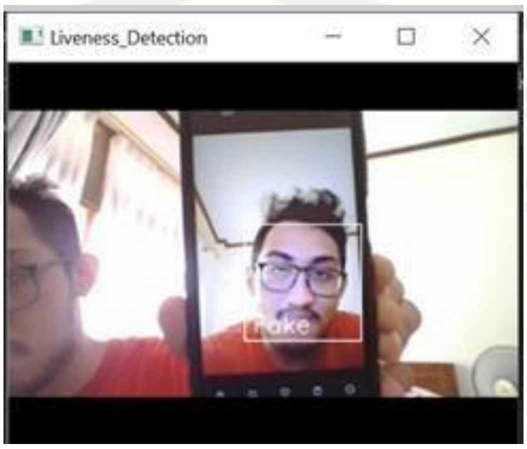


Figure 5.15 An Attack on Liveness Detection

So Figure 5.16 is the result of liveness detection. In Figure 5.15, the user tries to conduct the liveness detection using the researcher's picture from a smartphone. The system can detect that the image from the smartphone is not an actual face from the user. Therefore the system determined this face resemblance was not an actual face, or in other words, the system considered it a fake face. The system noticed that this kind of action was actually an attack or spoofing for liveness detection. For the comparison from Figure 5.15 above, the liveness detection is actually working and can distinguish the attack or spoofing. Figure 5.16 shows the stats of the attack on liveness detection

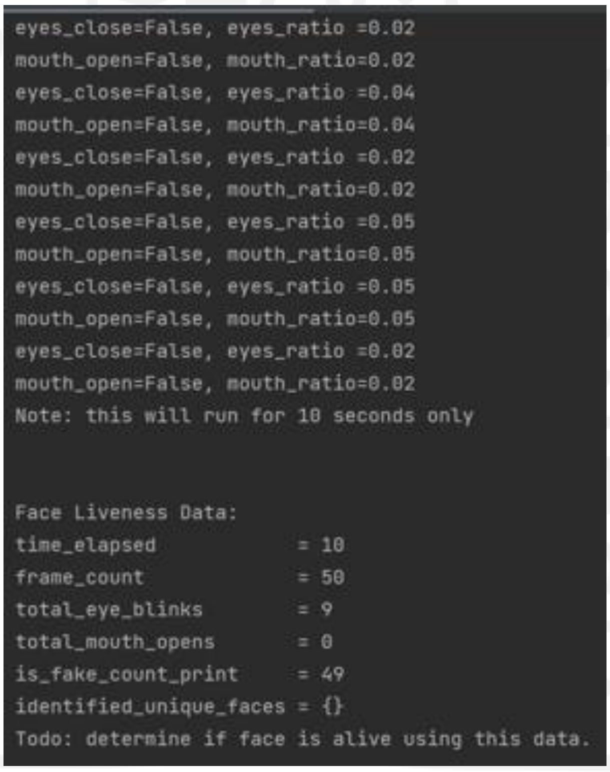

```
eyes_close=False, eyes_ratio =0.02
mouth_open=False, mouth_ratio=0.02
eyes_close=False, eyes_ratio =0.04
mouth_open=False, mouth_ratio=0.04
eyes_close=False, eyes_ratio =0.02
mouth_open=False, mouth_ratio=0.02
eyes_close=False, eyes_ratio =0.05
mouth_open=False, mouth_ratio=0.05
eyes_close=False, eyes_ratio =0.05
mouth_open=False, mouth_ratio=0.05
eyes_close=False, eyes_ratio =0.02
mouth_open=False, mouth_ratio=0.02
Note: this will run for 10 seconds only


Face Liveness Data:
time_elapsed          = 10
frame_count           = 50
total_eye_blinks      = 9
total_mouth_opens     = 0
is_fake_count_print   = 49
identified_unique_faces = {}
Todo: determine if face is alive using this data.
```

Figure 5.16 Liveness Detection on Spoofing Result

Figure 5.16 above shows the stats of Figure 5.15 above that image is considered fake by the system. As it can be seen that the identified_unique_faces = {}, it means that the system cannot detect the authenticity of the researcher's face in that picture. Is_fake_count_print result is 49, which means that in 10 seconds of video has been recorded, there are 50 frames, and 49 in between were fake. However, this may be due to movement and motion when the researcher is holding the researcher's phone. The system detects that the total_eyes_blinks is equal to 9. This may happen because when there are movement and motion, the central point of eye aspect ratio calculation has been changed. With that, a miscalculation by the system may occur. Yet, the system can still detect the authenticity of the image, and still can differentiate if there is spoofing or no. Figure 5.17 shows that the user tried directly clicking the “Click to Attend” button without inputting the ID and Name

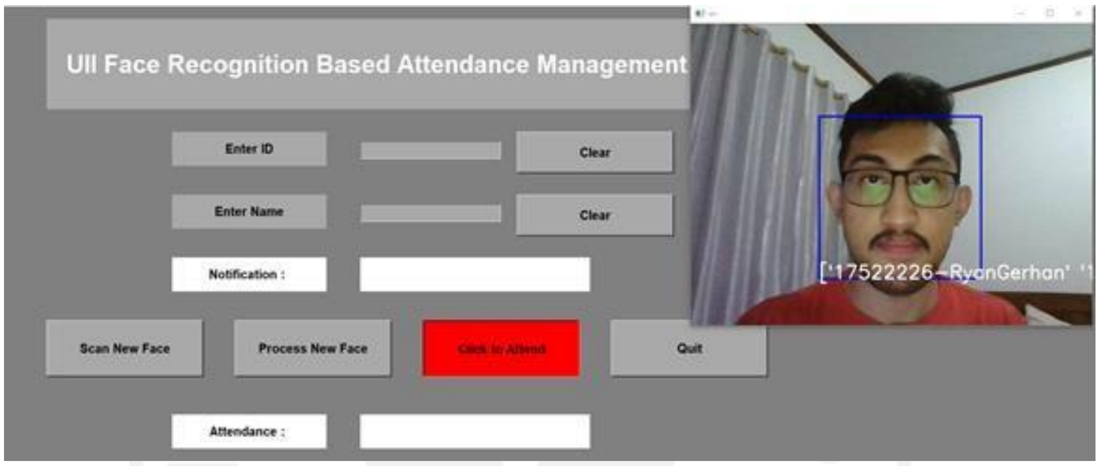
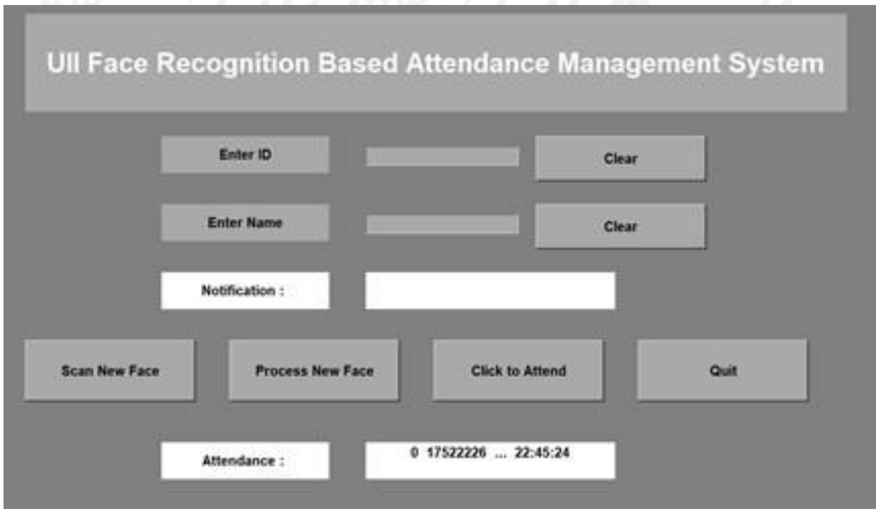


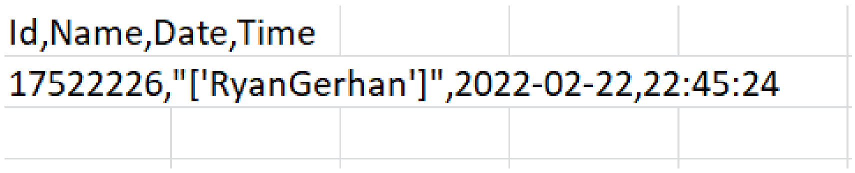
Figure 5.17 Conduct Attendance Directly

So here, Figure 5.17 shows if the user has already submitted the identity information submission and if the user’s faces have already been captured to be data training. Attendance can be conducted directly by clicking the “Click to Attend” button until it turns red. As soon as the button is clicked, the “im” window will appear, and facial recognition will be conducted. As the facial can be recognized, the system will notify the person by showing a rectangular or facial box with ID and name at the bottom of that box. The user needs to press “Q” on the keyboard to record the attendance. Figure 5.18 shows the recorded attendance by directly clicking the “Click to Attend” button.



0 17522226 ... 22:45:24

Figure 5.18 Recorded Attendance by Directly Attendance

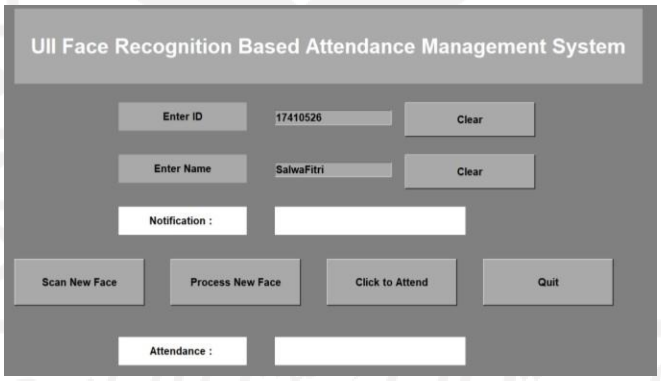


```
Id,Name,Date,Time
17522226,"['RyanGerhan']",2022-02-22,22:45:24
```

Figure 5.19 Recorded Attendance (Direct Attendance)

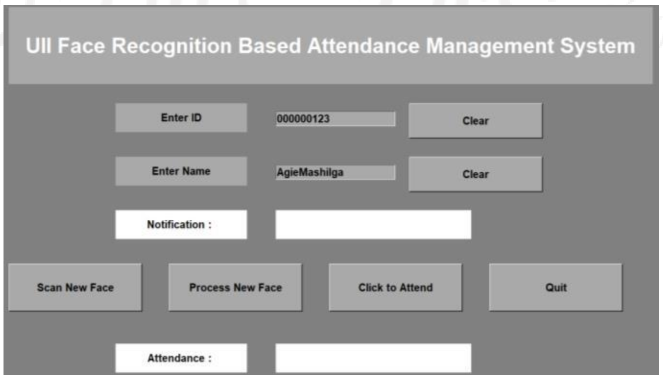
Once the “Q” on the keyboard has been pressed, the “Attendance” description box will show the id and time. As Figure 5.18 shows that the attendance has been saved and recorded. Figure 5.19 shows that the attendance recorded file will be in csv format and saved into the “Attendance” folder.

Furthermore, the system has been tested on two (2) other respondents. The testing processes for those respondents are the same as the steps shown in Figure 5.3 until Figure
5.11. The first step of testing the system on other respondents is by entering the ID and Name as shown in Figure 5.20 and Figure 5.21



Enter ID: 17410526
Enter Name: SalwaFitri

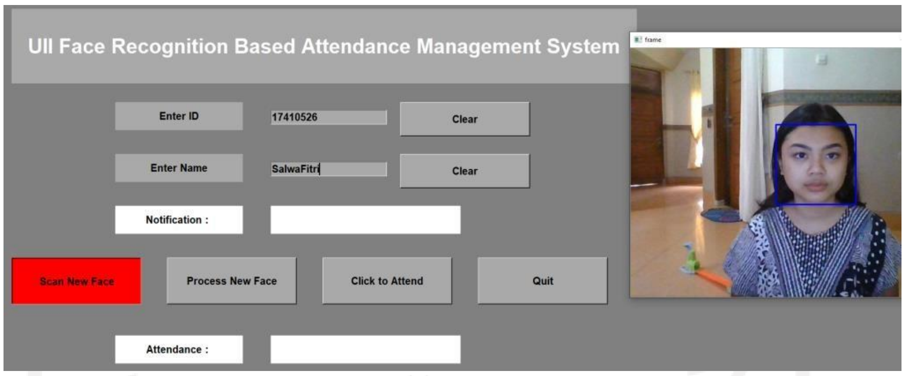
Figure 5.20 Adding New Respondent (Respondent1)



Enter ID: 000000123
Enter Name: AgieMashilga

Figure 5.21 Adding New Respondent (Respondent2)

After the field of ID and Name has been filled in, respondents click “Scan New Face” to save the ID and Name of respondents and also to initiate the camera to take a picture for data training. This process is shown in Figure 5.22 and Figure 5.23.


UII Face Recognition Based Attendance Management System

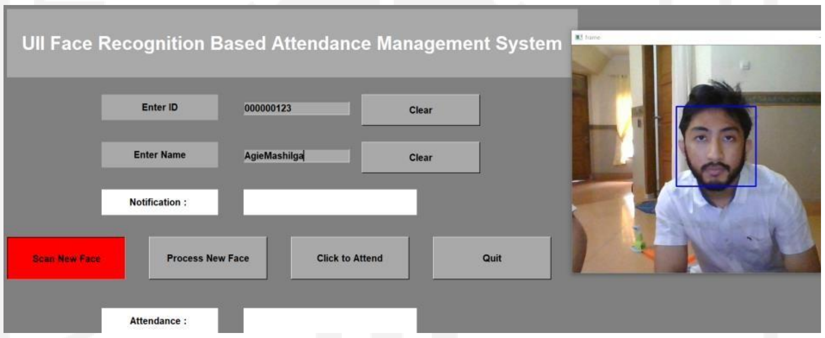
Enter ID 17410526

Enter Name SalwaFitri

Notification :

Attendance :

Figure 5.22 Scan New Face Button Clicked (Respondent1)


UII Face Recognition Based Attendance Management System

Enter ID 000000123

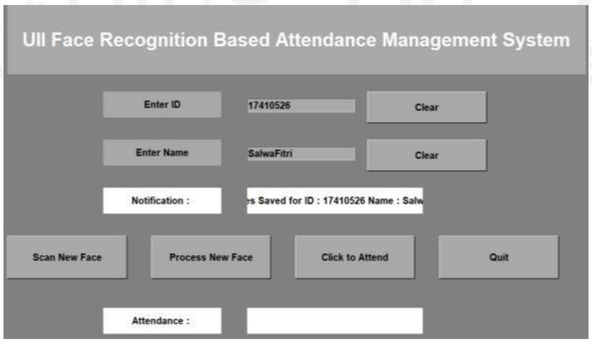
Enter Name AgieMashilga

Notification :

Attendance :

Figure 5.23 Scan New Face Button Clicked (Respondent2)

Once “Scan New Face” button is clicked, the facial box will appear to capture the faces of respondents. After faces have been captured, the system will notify the user with a notification if the name and Id have been saved. Figure 5.24 and Figure 5.25 show the system notification.


UII Face Recognition Based Attendance Management System

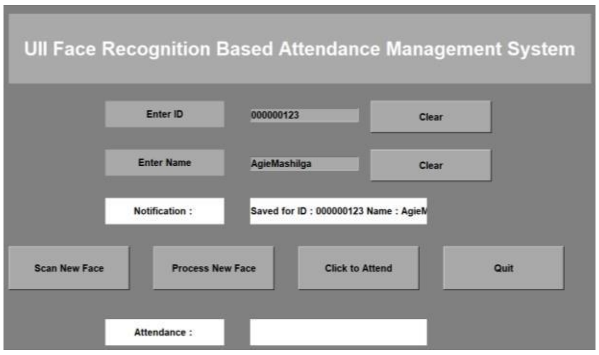
Enter ID 17410526

Enter Name SalwaFitri

Notification : es Saved for ID : 17410526 Name : Salw

Attendance :

Figure 5.24 System Notify Respondent1


UII Face Recognition Based Attendance Management System

Enter ID: 000000123

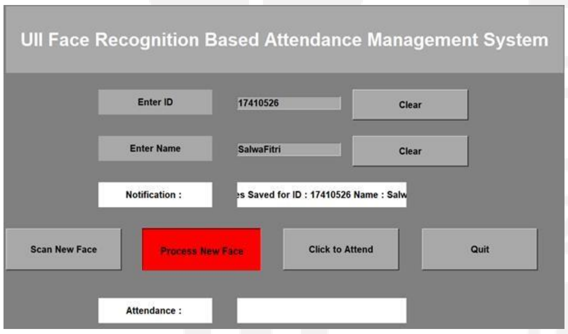
Enter Name: AgieMashilga

Notification : Saved for ID : 000000123 Name : AgieM

Attendance :

Figure 5.25 System Notify Respondent2

After ID and Name have been saved, respondents need to click “Process New Face” button to train the images of respondent faces. Figure 5.26 and Figure 5.27 below show when “Process New Face” button clicked


UII Face Recognition Based Attendance Management System

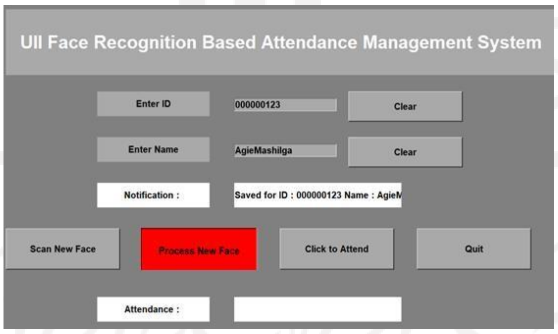
Enter ID: 17410526

Enter Name: SalwaFitri

Notification : es Saved for ID : 17410526 Name : Salw

Attendance :

Figure 5.26 “Process New Face” clicked by Respondent1


UII Face Recognition Based Attendance Management System

Enter ID: 000000123

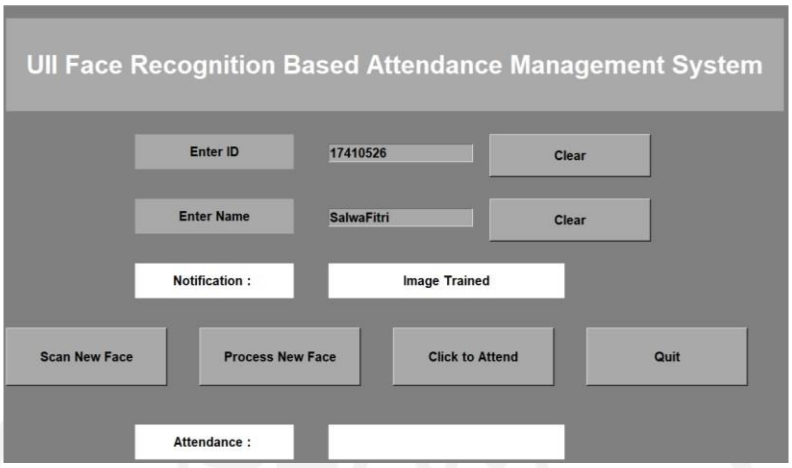
Enter Name: AgieMashilga

Notification : Saved for ID : 000000123 Name : AgieM

Attendance :

Figure 5.27 “Process New Face” clicked by Respondent1

When “Process New Face” is clicked, the system will initiate data training immediately. Once the data has been trained, the system will notify respondents again. Figure 5.28 and Figure 5.29 shows the system notification if data has been trained.


UII Face Recognition Based Attendance Management System

Enter ID 17410526 Clear

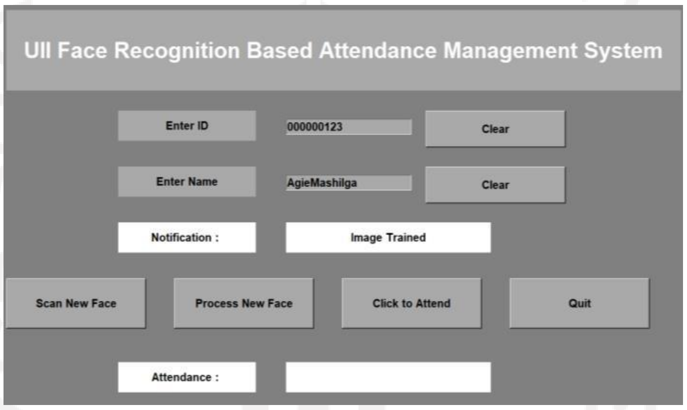
Enter Name SalwaFitri Clear

Notification : Image Trained

Scan New Face | Process New Face | Click to Attend | Quit

Attendance :

Figure 5.28 Respondent1's Data Trained


UII Face Recognition Based Attendance Management System

Enter ID 000000123 Clear

Enter Name AgieMashilga Clear

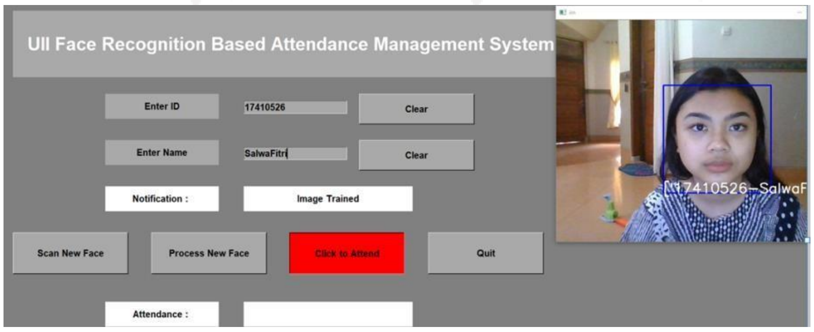
Notification : Image Trained

Scan New Face | Process New Face | Click to Attend | Quit

Attendance :

Figure 5.29 Respondent2's Data Trained

Saving the respondent's faces and the identity information only need to be conducted once (one time). Once the data of identities information has been saved, and the image of respondent's faces has been trained. Respondents can conduct attendance by clicking "Click to Attend" button. Attendance proceeds by recognizing the trained data and the actual face of respondents in real-time. Figure 5.30 and Figure 5.31 show the attendance process.


UII Face Recognition Based Attendance Management System

Enter ID 17410526 Clear

Enter Name SalwaFitri Clear

Notification : Image Trained

Scan New Face | Process New Face | Click to Attend | Quit

Attendance :

Figure 5.30 Respondent 1 Conduct Attendances.

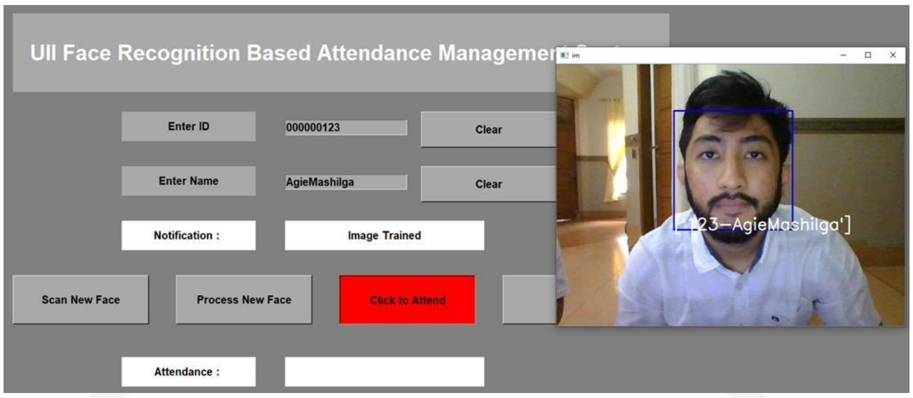
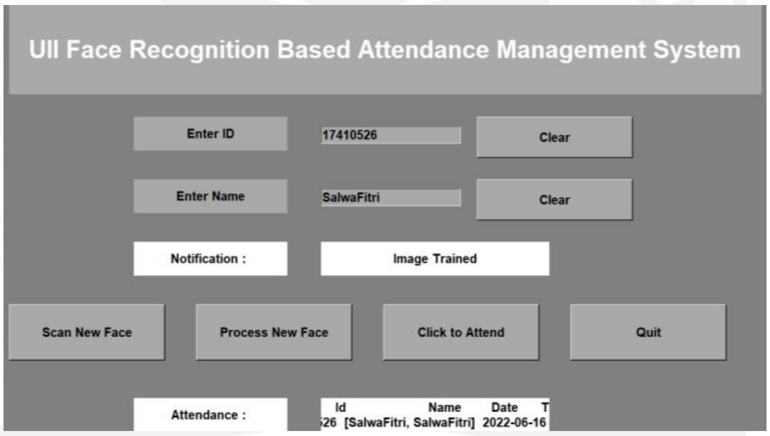


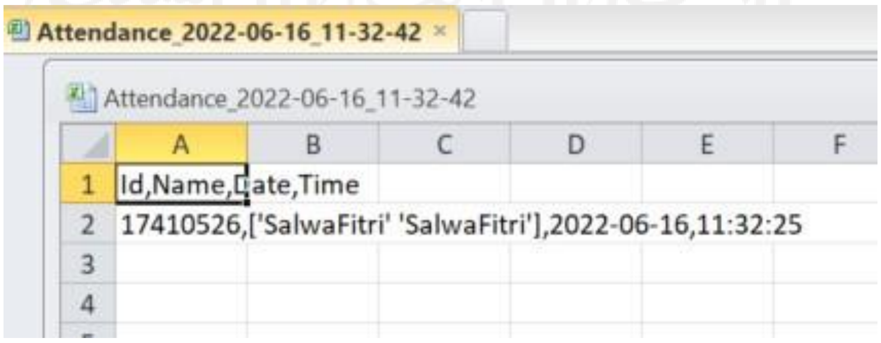
Figure 5.31 Respondent 2 Conduct Attendance.

Once attendance has been completed, the system will notify the respondent that the attendance has been recorded. The exact date and time of recorded attendance can be shown in the "Attendance" field. Figure 5.32 and Figure 5.33 show the recorded attendance of respondent1. Figure 5.34 and Figure 35 showed the recorded attendance of respondent2



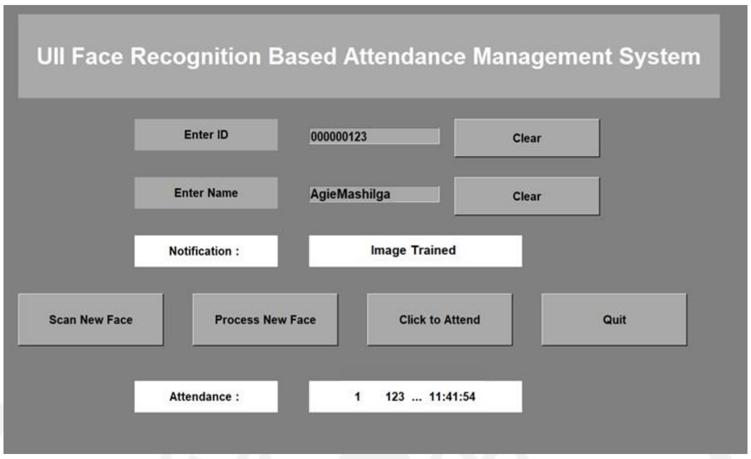
```
 Id          Name          Date      T
26  [SalwaFitri, SalwaFitri]  2022-06-16
```

Figure 5.32 Respondent 1 Recorded Attendance System Notification.


Attendance_2022-06-16_11-32-42
Attendance_2022-06-16_11-32-42

	A	B	C	D	E	F
1	Id,Name,Date,Time					
2	17410526,['SalwaFitri' 'SalwaFitri'],2022-06-16,11:32:25					
3						
4						

Figure 5.33 Respondent 1 Recorded Attendance saved in excel (csv format).



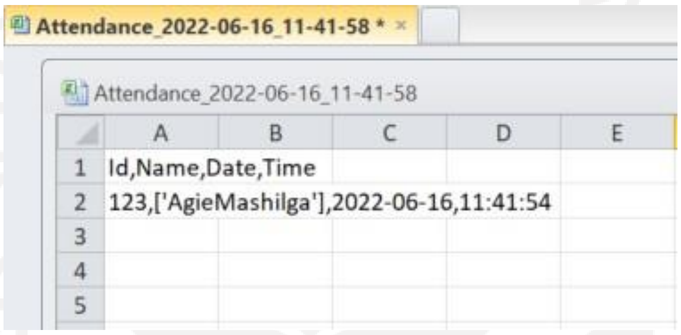
Enter ID: 000000123

Enter Name: AgieMashilga

Notification : Image Trained

Attendance : 1 123 ... 11:41:54

Figure 5.34 Respondent 2 Recorded Attendance System Notification.



Attendance_2022-06-16_11-41-58 *

Attendance_2022-06-16_11-41-58

	A	B	C	D	E
1	Id,Name,Date,Time				
2	123,['AgieMashilga'],2022-06-16,11:41:54				
3					
4					
5					

Figure 5.35 Respondent 2 Recorded Attendance saved in excel (csv format).

Respondent1 and the researcher also perform the attendance process directly without filling in the ID and Name. Figure 5.36 shows the attendance process when respondent1 and the researcher conduct the attendance directly by clicking the “Click to Attend” button from the beginning. Figure 5.37 shows the recorded attendance by the system. Figure 5.38 shows the recorded attendance in excel, csv format.

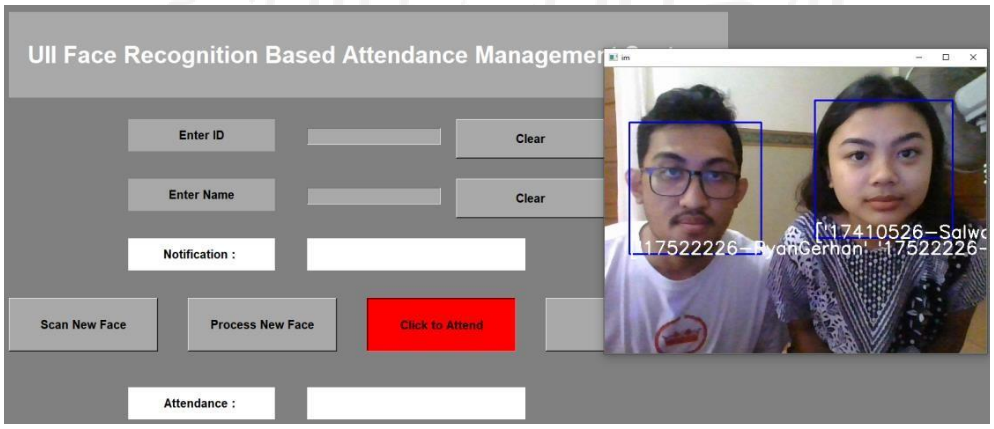
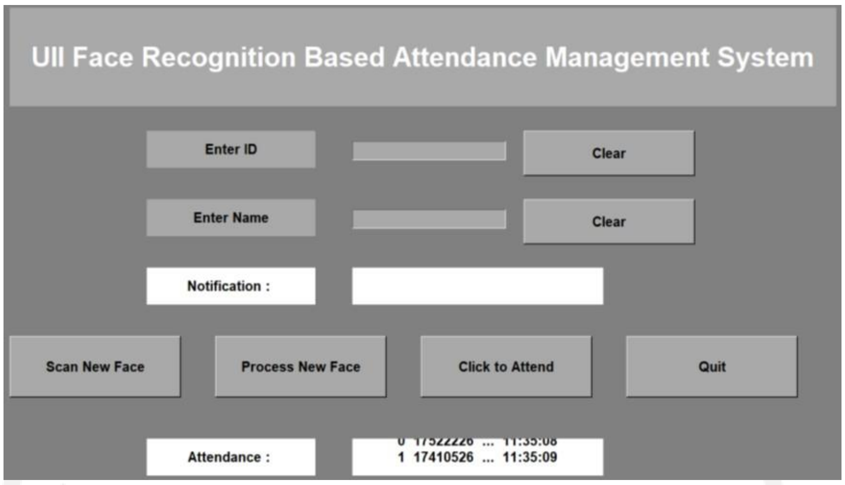
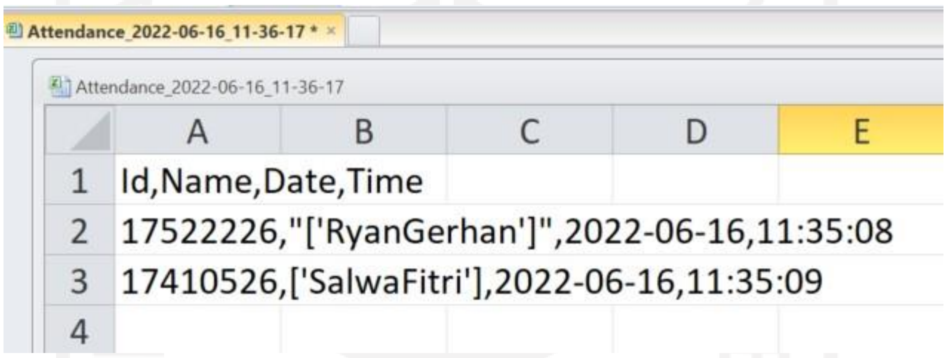


Figure 5.36 Respondent 1 and Researcher conduct Attendance together.



```
0  17522226  ...  11:35:08
1  17410526  ...  11:35:09
```

Figure 5.37 Respondent 1 and Researcher Recorded Attendance System Notification


Attendance_2022-06-16_11-36-17 *
Attendance_2022-06-16_11-36-17

	A	B	C	D	E
1	Id,Name,Date,Time				
2	17522226,"['RyanGerhan']",2022-06-16,11:35:08				
3	17410526,['SalwaFitri'],2022-06-16,11:35:09				
4					

Figure 5.38 Respondent 1 and Researcher Recorded Attendance Saved in Excel (csv format).

CHAPTER VI

CONCLUSION AND RECOMMENDATION

After a discussion of the result has been conducted, this chapter delivers the conclusion and recommendations about the whole research. Conclusion wrapped the whole research to answer the problem. Conclusions also give the reader a strong final impression of the whole research. Conclusions synthesize key points of this research and bring the objectives of this research to be concluded together. The recommendation aims to provide suggestions in terms to give a better point of view related to this research. The recommendation also intended to add some scientific advices and logical guidelines based on this research to perform future studies relatedly.

6.1 Conclusion

Based on the discussion above, the researcher took the key point of the whole research and synthesized it to get the conclusion of this research. Therefore, based on the previous chapters, it can be concluded that how to design the attendance system based on facial recognition is by adding some features to it. The advanced real-time attendance system based on facial recognition has the feature of preventing misconduct while doing attendance. Those features related to one another to maximize security to prevent misconduct in terms of attendance. Liveness detection, a real-time facial recognition attendance system, and capturing picture by timer are the features.

Based on the discussion above, the researcher wants to conclude how the advanced facial recognition attendance system developed. The development of advanced real-time attendance system based on facial recognition uses python programming language. The researcher created several modules to improve the attendance management system. Those modules are liveness detection, real-time facial recognition attendance system, and picture capture by timer modules. The liveness detection has a reliable ability to avoid an attack or to spoof when attendance is in process. Liveness detection is developed by contouring the expression, such as calculating mouth open and eyes blinking; liveness detection can detect the real face and the fake one. If the impostor is doing an attack or spoofing while conducting the attendance, the system will notify that the images of that person were fake. It works quite well to reduce the chance of cheating in attendance.

The attendance system was developed based on the real-time scenario, which can determine the exact time when the attendance was conducted. After that, the system will notify the exact time and date of the attendance. Moreover, the recorded attendance will be stored in the database. This kind of action can prevent cheating regarding attendance. This feature also helps the lecturer or authority to trace and recap the attendance of the students. Last but not least, the picture capture by timer developed can capture everything that happens in the determined time interval. This feature can supervise the students every time interval, whether the students are there or are not. Captured pictures are stored in a database and can be accessed by the authority to monitor the students' appearance in class.

6.2 Recommendation

The recommendations for future research are:

1. Adding more features to the facial recognition attendance system.
2. Conducting the real experiments by involving the students as the user to measure the productivity based on the targeted subject.
3. Upgrading the system by changing the algorithm to perform a better face recognition procedure or facial analysis.
4. Deploying the system to be an independent system.

REFERENCES

Aggarwal, N. (2021, June 25). *Command Line Arguments in Python : Geeksforgeeks*. Retrieved January 31, 2022, from Geeksforgeeks: https://www.geeksforgeeks.org/command-line-arguments-in-python/

Aggarwal, N. (2021, November 17). *Python datetime module : Geeksforgeeks*. Retrieved January 31, 2022, from Geeksforgeeks: https://www.geeksforgeeks.org/python-datetime-module/

Aggarwal, N. (2021, November 17). *Python datetime module: Geeksforgeeks*. Retrieved January 31, 2022, from Geeksforgeeks: https://www.geeksforgeeks.org/python-datetime-module/

Ahmed, A., Guo, J., Ali, F., & Deeba, F. (2018). LBPH Based Improved Face Recognition at Low Resolution. *International Conference on Artificial Intelligence and Big Data (ICAIBD)*, 144-147.

Ahonen, H. A., T, & M.Pietikinen. (2006). Face Description with Local Binary Patterns. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2037-2041.

Boult, T., & Scheirer, W. (2009). *Long-Range Facial Image Acquisition and Quality*. Colorado: University of Colorado.

Clyde, G., Sagar, C., Tanmay, D., & Dipti, J. (2020). Class Attendance Management System using Facial Recognition. *International Conference on Automation, Computing and Communication* (pp. 1-6 Vol 32). Mumbai: EDP.

Dagar, B. S., Rao, M., Dash, N. K., & Joseph, K. S. (2020). *BESC-131 Education: Concept, Nature and Perspectives.* New Delhi: Indira Gandhi National Open University.

Deeba, F., Ahmed, A., Memon, H., Dharejo, F. A., & Ghaffar, A. (2019). LBPH-based Enhanced Real-Time Face Recognition. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 274-280.

Dhaw, Moammer H., Z., Amer R, A. A., Nabel K., T., & M., W. (2008). Biometric Recognition System "Fingerprint and Face Recognition". *9th International* *conference on Sciences and Techniques of Automatic control & computer engineering.* (pp. 1-11). Tunisia: Academic Publication Center of Tunis.

Dmello, R., Yerremreddy, S., Basu, S., Bhitle, T., Kokate, Y., & Gharpure, P. (2019). Automated Facial Recognition Attendance System Leveraging IoT Cameras. *International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (pp. 556-561). Noida: IEEE.

Edward, F., & Barr, A. (1979). *Handbook of Artificial Intelligence.* California: Stanford University.

Geeksforgeeks. (2022, January 22). *Time Functions in Python : Geeksforgeeks*. Retrieved January 31, 2022, from Geeksforgeeks: https://www.geeksforgeeks.org/time-functions-in-python-set-1-time-ctime-sleep/

Grossfeld, B. (2021, September 21). *Deep learning vs. machine learning: a simple way to learn the difference.* Retrieved September 24, 2021, from Zendesk: https://www.zendesk.com/blog/machine-learning-and-deep-learning/

Guoying, Z., & Pietikainen, M. (2007). Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 915-928.

Hajri, E. A., Hafeez, F., & V, A. A. (2019). Fully Automated Classroom Attendance System. *International Journal of Interactive Mobile Techonologies*, 95-106.

Ion, M. (2010). *"Face Recognition Algorithm" Proyecto Fin de Carrera.* Universidad del Pais Vasco.

javatpoint. (2022, January 26). *Python Pillow Tutorial : Pillow : Javatpoint*. Retrieved January 31, 2022, from Javatpoint: https://www.javatpoint.com/python-pillow

Jenif D, S. W., Jothi, S., & Chandrasekar, A. (2019). Automated Attendance Marking and Management System by Facial Recognition Using Histogram. *5th International Conference on Advanced Computing & Communication Systems* (pp. 66-69). Tamil Nadu: IEEE.

Kagoyal, V. (2020, February 19). *Command-Line Option and Argument Parsing using argparse in Python : Geeksforgeeks*. Retrieved January 31, 2022, from Geeksforgeeks: https://www.geeksforgeeks.org/command-line-option-and-argument-parsing-using-argparse-in-python/

Kelvin S, d. P. (2017, October 9). *Local Binary Patterns Histogram (LBPH)*. Retrieved February 23, 2022, from Github: https://github.com/kelvins/lbph

Kiran, T. A., Reddy, N. D., Ninan, A. I., Krishnan, P., Aravindhar, D. J., & Geetha, A. (2020). PCA based Facial Recognition for Attendance System. *2020 International Conference on Smart Electronics and Communication (ICOSEC)* (pp. 248-252). Trichy: IEEE.

Kumar, D. C., Ravi, C., & Dinesh, J. (2014). Human Face Recognition and Detection System with Genetic and Ant Colony Optimization Algorithm . *Journal of Computer Engineering (JCE)*, Vol. 16 Issue 4.

Kumaravnish. (2020, May 22). *Python Tkinter Tutorial: Geeksforgeeks*. Retrieved January 31, 2022, from Geeksforgeeks: https://www.geeksforgeeks.org/python-tkinter-tutorial/?ref=lbp

Li, Y., & Chan, S. (2019). Face Recognition System. *CoRR*, 5-8.

Malcolm, H., Wilson, V., Davidson, J., & Kirk, S. (2003). *Absence from School: A study of its causes and effects in seven LEAs.* Glasgow: The SCRE Centre, University of Glasgow.

Maneetsingh. (2014). On Recognizing Face Images with Weight and Age Variations. *IEEE*, 822-830.

Matthew, D. Z., & Rob, F. (2014). Visualizing and Understanding Convolutional Networks. *European Conference on Computer Vision* (pp. 818-833). Swizterland: LNCS.

Minichino, J., & Howse, J. (2015). *Learning OpenCV 3 Computer Vision with Python Second Edition.* Birmingham: Packt Publishing Ltd.

Mohamed, S. S., Mohamed, W. A., Khalil, A. T., & Mohra, a. A. (2019). Deep Learning Face Detection and Recognition. *INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS*, 2.

Mooseop, K., Deokgyu, L., & Ki-Young, K. (2015). System Architecture for Real-Time Face Detection onAnalog Video Camera. *Journal of Distributed Sensor Networks*, 1-11.

Mutiara, Q. D., & Prastyo, E. W. (2019). Perbandingan Metode Eigenface, Fisherface, dan LBPH pada Sistem Pengenalan Wajah. *Jurnal Ilmiah KOMPUTASI*, 315-322.

Nabatchian, A. (2011). *Human Face Recognition.* Ontario: University of Windsor.

Nurulhuda, I., & Idayu, M. M. (2009). Review of existing algorithms for face detection and recognition. *International Conference on Computational Intelligence, Man-machine System and Cybernetics* (pp. 30-39). Puerto De La Cruz: CIMMACS.

P.M, O. T., & Hardwood, D. (1996). A Comparative Study of Texture Measures with Classification Based on Feature Distribution. *Pattern Recognition*, 51-59.

Pandey, P. (2018, December 21). *Face Detection with Python using OpenCV: Tutorial*. Retrieved February 22, 2022, from Datacamp: https://www.datacamp.com/community/tutorials/face-detection-python-opencv

Pietikinen, M. (2010). Local Binary Patterns. *Vol.5*, 9975.

Pissarenko, D. (2002, December 1). *Eigenface-based facial recognition.* Retrieved February 26, 2022, from Sematic Scholar: https://www.semanticscholar.org/paper/Eigenface-based-facial-recognition-Pissarenko/a7bf6328bd40d4bc6db07034041d4c873ebadc60

Richmondu. (2019, August 24). *Richmondu: Libfaceid: Github*. Retrieved January 31, 2022, from Github: https://github.com/richmondu/libfaceid

Ripal, P., Nidhi, R., & Ami, S. (2012). Comparative Analysis of Face Recognirion Approaches: A Survey. *International Journal of Computer Applications* , 50-61.

Rosebrock, A. (2015, December 7). *Local Binary Patterns with Python & OpenCV*. Retrieved February 23, 2022, from pyimagesearch: https://pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/

Saha, R. (2021, December 03). *Reading CSV File in Python : Geeksforgeeks*. Retrieved January 31, 2022, from Geeksforgeeks: https://www.geeksforgeeks.org/reading-csv-files-in-python/

Saleh, Z. (2019). Artificial Intelligence Definition, Ethics and Standards. .

Salton, K. d. (2017, November 11). *Face Recognition: Understanding LBPH Algorithm*. Retrieved February 23, 2022, from Towardsdatascience: https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b

Satyam, K. (2021, October 07). *Shuntil Module in Python: Geeksforgeeks*. Retrieved January 31, 2022, from Geeksforgeeks: https://www.geeksforgeeks.org/shutil-module-in-python/

Schleicher, A. (2020). The Impact of COVID-19 on Education: Insights From Education at a Glance 2020. *OECD* (p. 4). Paris: OECD.

Shailender, K., Dhruv, K., Dipen, S., & Mandeep, V. (2020). Convolutional Neural Network based Automated Attendance System by using Facial Recognition Domain. *International Conference on Intelligent Computing and Control System* (pp. 654-659). Madurai: IEEE.

Shang-Hung, L. (2000). An Introduction to Face Recognition Technology. *Informing Science The International Journal of an Emerging Transdiscipline*, Vol.3 No.1.

Shepley, A. J. (2019, July 12). Deep Learning For Face Recognition: A Critical Analysis. Casuarina, Dawrin, Australia.

Smitha, P. S., & Afshin. (2020). Face Recognition based Attendance Management System. *International Journal of Engineering Research and Technology*, 1190-1192.

State, A. (2021, November 25). *What Is Pandas in Python? Everything You Need to Know*. Retrieved January 31, 2022, from ActiveState: https://www.activestate.com/resources/quick-reads/what-is-pandas-in-python-everything-you-need-to-know/

Suresh, D. V. (2019). Facial Recognition Attendance System Using Python and OpenCv. *Journal of Software Engineering and Simulation*, 18-29.

Taqqiyah, W. I., Lawi, A., & Galsan, A. M. (2019). *Face Image Detection Using Haar Cascade Classifier.* Makassar: UNHAS.

Ummi, S. M., Andrew, F., & Risanti, I. J. (2018). NFC Based Mobile Attendence System with Facial Authorization on Raspberry Pi and Cloud Server. *6th International Conference on Cyber and IT Service Management (CITSM)* (pp. 1-6). Parapat: IEEE.

Viola, P., & Jones, M. (2001). Rapid Object Detection Using a Boosted Cascade of Simple Features. *Computer Vision and Pattern Recognition* (pp. 1-9). Kauai, Hawaii, USA: IEEE.

Wayne, H., Maya, B., & Charles, F. (2019). *Artificial Intelligence in Education: Promises and Implications for Teaching and learning.* Boston: Center for Curriculum Redesign.

Yalamanchili, P., & Paladugu, B. (2007). *Comparative Study of Face Recognition Algorithms.* South Carolina: Clemson University.