Attendance Management With Facial Recognition System

Submitted To

Controller of Examination
National University
Gazipur - 1704

Supervised By

Md. Mobinul Islam
Lecturer
Dept. of Computer Science & Engineering.
Northern College Bangladesh

Submitted By

Naimul haque sagar
Registration Number: 15502001366
Department of Computer Science and Engineering
Northern College Bangladesh

Session: 2015-2016

Group Members

Name	Registration Number:
Proshenjit kumar	15502001345
Md. Hafizur rahman	15502001375

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NATIONAL UNIVERSITY, BANGLADESH

DECLARATION

I am Naimul haque sagar students of Computer Science & Engineering (CSE) program Northern College Bangladesh, Dhaka. I certify that "**Attendance Management With Facial Recognition System**" project does not incorporate, without acknowledgment, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my project, if accepted, to be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

I hereby warrant that the work I have Presented do not breach any existing copyright acts.

Submitted by:
Signature of candidate: ….....….
Date: ……../……/…….
Name of the candidate: Naimul haque sagar

Countersigned by:
Signature of supervisor: ……
Date: ……../……/…….
Md. Mobinul Islam
Lecturer, Northern College Bangladesh

Signature of Examiner: ……

Signature of Examiner: ……

ACKNOWLEDGEMENTS

Apart from the efforts of mine, the success of any project depends largely on the encouragement and guidance of many others. I would like to take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

My sincere thanks to **Md. Rezaul Karim**, Principal, Northern College Bangladesh, who has allowed me to do this project and encouragement given to me.

I own deep sense of gratitude to **M.A. Walid**, Coordinator in CSE & BBA, Northern College Bangladesh, for appreciating my goal. I express our sincere thanks to him for his constant encouragement.

I would like to show my greatest appreciation to my project supervisor **Md. Mobinul Islam**, lecturer, Dept. of (CSE) for his patronage and giving me an opportunity to undertake this Project and for guiding me and providing me.

Then I would like to appreciate all the help and support given by my parents and siblings. I would like to acknowledge the lecturers of Northern College Bangladesh.

Finally, I would be grateful to National University, Bangladesh and coordinators of the Bachelor of Science in Computer Science and Engineering degree program to giving me this opportunity to apply the knowledge which I have gained through the study of CSE program.

ABSTRACT

Facial recognition is a technology capable of identifying or verifying a human from a digital camera or video. In general facial recognition work by comparing selected faces from the given faces within the database. We can call this facial recognition system a biometric based application that can identify a human by analyzing, based on the human face. Attendance management is one of the applications of facial recognition systems. Traditional attendance management system :paper based attendance ,card punching , manual process takes a lot time to complete this process. To ignore this hectic system of traditional attendance management , our Facial Recognition System is a better solution for any kind of organization . Without physical interaction of human beings our facial recognition system takes care of the attendance management . By analyzing detected faces of humans entering the area it compares the detected face with stored faces in the database. Without any human interaction the system automatically and without investing any manpower for attendance management it manages this process.

Table Of Contents

Chapter One

Introduction

1.1 Motivations

Attendance management is the way we keep track of our attendance hours. This is the system that we use to document the time of our work and the time we take off. Attendance management is all about managing attendance or presence in a work setting to minimize loss due to downtime . Facial recognition is a great way to track and keep logs and maintain security purposes. Facial recognition can be very efficient and very helpful for those organizations who have a mass population of human resources . We realized that face recognition and tracking can be really beneficial if it can be used for the mass people. They don't need to waste their organizations valuable time in such attendance management. Facial recognition is the perfect solution for them. It gives an accurate result. By comparing the traditional attendance management system our facial recognition system is better and more efficient ,low costing system.

1.2 Background

For facial recognition there are a lot of algorithms ,they have different models ,different approaches to work . And also they have different accuracy to detect the human face . We have used OpenCv as an open source machine learning library for real time computer vision . This is a library of programming functions mainly aimed at computer vision.

Cascading is a particular case of ensemble learning based on the concatenation of several classifiers, using all information collected from the output from a given classifier as additional information for the next classifier in the cascade. Unlike voting or stacking ensembles, which are multi expert systems, cascading is a multistage one.

Cascading classifiers are trained with several hundred positive sample views of a particular object and arbitrary negative images of the same size . After the classifier is trained it can be applied to a region of an image and detect the object in question . To search for the object in the entire frame ,the search window can be moved across the image and check every location for the classifier. This process is most commonly used in image processing for object detection and tracking ,primarily facial detection and recognition.

The Viola–Jones object detection framework is the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by paul viola and michael jones. Although it can be trained to detect a variety of object classes, it was moti-

vated primarily by the problem of face detection .Viola Jones face detection algorithm is a widely used method for real-time object detection. Although it can be trained to detect a variety of object classes, it was motivated primarily by the problem of face detection.

While using this facial recognition system facial spoofing is a process in which a fraudulent user can subvert or attack our face recognition system by masquerading the user and thereby gaining illegitimate access and advantages . A face spoofing attack is an attempt to acquire someone else's privileges or access rights by using a photo, video or a different substitute for an authorized person's face. Listed below are some face spoofing attacks. There are many types of attack that attackers can try : print attack ,eye cut photo attack,warped photo attack ,reply /video attack and many others .

For preventing this spoofing attack we implemented a liveness detection process which the user has to pass for authentication . For preventing this spoofing attack we will first detect each frame generated by the camera ,then we will detect faces ,eyes ,for each eye we will check if the eyes are opened or closed. If at some point it was detected that the eyes were open then closed then open, we conclude the person has blinked then it will declare the face as real face

At some point the system should be assured that the object should be human . For assuring that the object is a human we have used human face landmark detection technology to assure the system that whoever is using the system to take verification is a human .For this system checks the facial landmark on a human face .If the system founds all facial landmark in detected faces then the system mark the face as a human face.

1.3 Objectives

The objective of this study is to give a solution to the problem in an efficient way that will fill the gap of existing or traditional systems. Objective is to create a different system against the traditional system that will help the organizations to make attendance monitoring more accurate and to prove that the system developed is efficient and helpful.

This study is to enhance organizations attendance performance and security. This will develop a system to help the staff and make their attendance monitoring easy and their work with less effort. With the system, provides a user-friendly environment and effective perfor-

mance system, this will provide an easier and faster access to the data of every staff in their Attendance Report.

This system will allow organizations to monitor staff who are not actively attending the organization. The system will secure the attendance data . The automatic attendance will help the management to choose the right person for them in future.

1.4 System Security

The protection of computer based resources that include hardware, software, data, procedures and people against unauthorized use or natural disaster is known as System Security. Internal threats are proving to be a nightmare for IT administrators and computer users. Although technology helps to counter these threats, a more holistic approach is needed, one that includes strict and enforceable policies as well as a proper awareness program.

Facial recognition systems are more secure than the traditional attendance management systems. There is no way of data entry after the scheduled time. No way for illegal access to the private part of the system. Without an authorized person nobody can see or change the private data .This system gives confidentiality status to sensitive information in a database to minimize the possible invasion of privacy. It is an attribute of information that characterizes its need for protection. Keeping user data safe from breaches, be they inadvertent or malicious, is an ever evolving effort — particularly in a world of organizations, such as WikiLeaks and Anonymous, that expressly exist to expose previously secure data. Data security entails not only hardware and software requirements but also implementation and enforcement of organization wide governance policies.

Chapter Two

Feasibility study

2.1 Introduction

The feasibility study is an evaluation and analysis of the potential of a proposed project which is based on extensive investigation and research to support the process of decision making. Feasibility study, also known as feasibility analysis, is an analysis of the viability of an idea. It describes a preliminary study undertaken to determine and document a project's viability. A Project Feasibility Study is an exercise that involves documenting each of the potential solutions to a particular business problem or opportunity. During the Feasibility Study, a variety of 'assessment' methods are undertaken. Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running systems. The outcome of the Feasibility Study is a confirmed solution for implementation.

2.2 Features of the system

1. Maintain attendance management system
2. Maintain category based attendance
3. Secure storage of data which include employee information files, salary information etc
4. Better matching algorithm
5. Scalability
6. Privacy
7. Predictive analytics
8. Maintain transactional data
9. Maintain proper time of attendance
10. Maintain proper authorization

2.3 Key advantages

Managing attendance system for an Organization gives the following advantages:-

- It lowers the time

- Increases productivity
- Reduces workload
- Increases the proper use of time

In addition, we reap many benefits from attendance management that occurs every day, not just during a major threat.The advantages of utilizing attendance management systems in an organization are:

- ✔ Scalability: Facial recognition solutions have to be able to quickly scale flawlessly across hundreds of locations. This means that you need to find a solution that is built to handle large deployments. The solution should also have a support team in place to handle installations, including optimizing cameras for lighting conditions and angle.
- ✔ Speed: This attendance management system works faster than a manual attendance management system.
- ✔ Simplicity: This attendance management system is simpler to use . If once the system is installed then there is no manual process . It will work automatically.

- ✔ Better Matching: A lot of facial matching algorithms just use tens or hundreds of feature detection points on a face in order to establish identity. But a face recognition solution should be using thousands of points on a face in order to establish identity. A better algorithm can result in a much better accuracy, which will undoubtedly result in a better performance.

- ✔ Analytics: The right set of analytics can help this facial recognition system to prevent future faults .It will be crucial for the system to be able to view aggregate data across multiple store locations . This system would help companies easily identify which times of day and which locations are experiencing the highest number of matches, so that each location can be accurately staffed up.

2.4 Problem Statement

There are many problems that can occur without this system:

Manual systems put pressure on people to be correct in all details of their work at all times, the problem being that people aren’t perfect, however much each of us wishes we were. With manual systems the level of service is dependent on individuals and this puts a requirement

on management to run training continuously for staff to keep them motivated and to ensure they are following the correct procedures. It can be all too easy to accidentally switch details and end up with inconsistency in data entry or in hand written orders. This has the effect of not only causing problems with customer service but also making information unable to be used for reporting or finding trends with data discovery. Reporting and checking that data is robust can be timely and expensive. This is often an area where significant money can be saved by automation.

1. Inconsistency in data entry, room for errors, miskeying information.

2. Large ongoing staff cost.

3. System is dependent on good individuals.

4. Reduction in sharing information and services.

5. Time consuming and costly to produce reports.

6. Lack of security.

7. Duplication of data entry.

It takes more effort and physical space to keep track of paper documents, to find information and to keep details secure. When mistakes are made or changes or corrections are needed, often a manual transaction must be completely redone rather than just updated. With manual or partially automated systems information often has to be written down and copied or entered more than once. Systemisation can reduce the amount of duplication of data entry.

2.5 Technical instrument

Server**:** The term server can refer to either the hardware (the computer) or the software (the computer application) that helps to deliver content that can be accessed. We can use our local server to run our application.

Database Server: A database server is a computer program that provides database services to other computer programs or computers, as defined by the client–server model. The term may also refer to a computer dedicated to running such a program. Database management systems frequently provide database server functionality, and some DBMSs (e.g., MySQL) rely exclusively on the client–server model for database access.

Supporting Language Java: JAVA was developed by Sun Microsystems_Inc in the year 1991, later acquired by Oracle Corporation. It was developed by James Gosling and Patrick Naughton. It is a simple programming language. Java makes writing, compiling, and debugging programming easy. It helps to create reusable code and modular programs.

Java is a class-based, object-oriented programming language and is designed to have as few implementation dependencies as possible. A general-purpose programming language made for developers to *write once run anywhere* that is compiled Java code can run on all platforms that support Java. Java applications are compiled to bytecode that can run on any Java Virtual Machine. The syntax of Java is similar to c/c++.

Supporting frameworks: Java Swing is a part of Java Foundation Classes (JFC) which was designed for enabling large-scale enterprise development of Java applications. Java Swing is a set of APIs that provides graphical user interface (GUI) for Java programs. Java Swing is also known as Java GUI widget toolkit.

Java Swing or Swing was developed based on earlier APIs called Abstract Windows Toolkit (AWT). Swing provides richer and more sophisticated GUI components than AWT. The GUI components are ranging from a simple label to complex tree and table. Besides emulating the look and feel of various platforms, Swing also provides *the pluggable look and feel* to allow the look and feel of Java programs independent from the underlying platform.

Chapter Three

Analysis

3.1 Introduction

Analysis is the process of breaking a complex topic or substance into smaller parts to gain a better understanding of it. After analyzing the requirements of the task to be performed, the next step is to analyze the problem and understand its context. The first activity in the phase is studying the existing system and another is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of the existing running system is also difficult, improper understanding of the present system can lead to diversion from solution.

Facial recognition is a technology capable of identifying or verifying a human from a digital camera or video. In general facial recognition works by comparing selected faces from the given faces within the database. We can call this facial recognition system a biometric artificial intelligence based application that can identify a human by analyzing, based on the human face. Attendance management is one of the applications of facial recognition systems. Traditional attendance management system :paper based attendance ,card punching , manual process takes a lot time to complete this process. To ignore this hectic system of traditional attendance management , our Facial Recognition System is a better solution for any kind of organization . Without physical interaction of human beings our facial recognition system takes care of the attendance management . By analyzing detected faces of humans entering the area it compares the detected face with stored faces in the database. Without any human interaction the system automatically and without investing any manpower for attendance management it manages this process.

3.2 Analysis Model

SDLC Methodologies: This document plays a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It is meant for use by developers and will be the basics during the testing phase. Any changes made to the requirements in the future will have to go through a formal change approval process.

Waterfall Model: The waterfall model is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation, and Maintenance. The waterfall is a widely accepted SDLC model. In this approach, the whole process of the software development is divided into various phases of SDLC. In this SDLC model, the outcome of one phase acts as the input for the next phase. This SDLC model is documentation-intensive, with earlier phases documenting what needs be performed in the subsequent phases.

3.3 System Analysis:

System analysis is a problem solving technique that decomposes a system into its component pieces for the purpose of studying how well those component parts work and interact to accomplish their purpose. We analyze the system in terms of data, process and interface.

3.3.1 Problem Analysis:

During the system analysis we have done the following task for the problem analysis:

- **i.** Study the problem domain
- **ii.** Analyze problems and opportunities
- **iii.** Establish system improvement objectives

Analysis from existing systems found more simple things to identify valid data which are stored as not performed security. Our proposed system keeps data more securely and identifies the valid user.

3.3.2 Requirement Analysis:

System requirements are expressed in a software requirement document. The Software requirement specification (SRS) is the official statement of what is required of the system developers. This requirement document includes the requirements definition and the requirement specification. The software requirement document is not a design document. It should set out what the system should do without specifying how it should be done. The requirement set out in this document is complete and consistent.

The software specification document satisfies the following:-

1. It specifies the external system behaviors.
2. It specifies constraints on the implementation.
3. It is easy to change.

Requirements analysis is a team effort that demands a combination of hardware, software and human factors engineering expertise as well as skills in dealing with people. It is critical to the success of a system.

The purpose of the subsequent analysis is to fully understand the problem and its implications. There is no reason to expect that a problem statement prepared without a full analysis will be correct.

3.4 System Requirement Specifications

3.4.1Hardware Requirements:

PIV 2.9 GHz Processor and Above

RAM 1GB and Above

HDD 80 GB Hard Disk Space and Above

3.4.2 Software Requirements:

Operating System: Windows XP/vista/7 or later version, Linux OS which supports networking.

Java, Java swing development tool kit

DB Server

3.5 Cost and Benefit Analysis

System cost can be defined in earlier stages of system development. So this general system, a cost plan is identified. The cost plan is divided in two phases:

Cost of Development

Cost of Using the System

3.5.1 Cost of Development

Database Maintenance Cost**:** Our system stores all important contents of users. For this purpose a MySql database should be needed. Server content management cost should be needed. System development cost is just one time cost, which will not recur after the project has been developed.

Human Resource Cost: In cost of development, programmers and analysts pay other people's salary that is related to development. They are also training to use this software.

3.5.2 Cost of Using the System

Fixed Cost:

Software purchase and license cost

System maintenance cost

Variable Cost:

Cost of hardware (Printer, Ink, External memory etc)

3.6 Risk Analysis

Risk Analysis has become an important component of project management. A Project Risk Analysis identifies the risks and potential obstacles to the future exploitation of a project's results. Our proposed system is not different in this case. We tried to find out these are given below:

Conflict of interest – Lack of interest should be a risky point for our system.

Lack of support– Any sign that a partner organization may not be fully behind a project, if we don't have sufficient support that would be a risk.

Risk Log/Register--During the project, keep an eye on the risks. Look for early warning signs that indicate a risk is about to occur. A Risk Log or Risk Register provides a means of recording the identified risks. It should be reviewed at regular points during the project as some risks might disappear and others might occur.

Making Right Decisions- Formalized decision analysis process helps IT project managers to make the right decision. Learn how to use decision analysis techniques to mitigate negative impact to psychological biases and select the most effective project decision.

Management Challenges- Many management problems are due to a lack of funds that would enable the manager to have the right people in the right places at the right time doing the right things. Money, however, is not the only issue. Equally challenging staffing issues for managers involve post planning, use of leave, staff turnover, and in-service training.

Chapter Four

Design

4.1 Overview

Software design is a process by which an agent creates a specification of a software, transforming user requirements into suitable form, which help programmers in coding and implementation. Software design is the first step of the SDLC method. This phase deals with transforming the customer requirements as described in the SRS documents into a form of implementable using a computer language. It is basically building the product from scratch. It actually deals with client's requirements .

Software design should be accurate as per as the requirement. The software design should have all components in it. The main purposes of software design is to make the product as efficient as it could be. The design should be implemented simply so that future work or development don't find any mistakes.

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirements have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

4.2 Proposed System Diagram

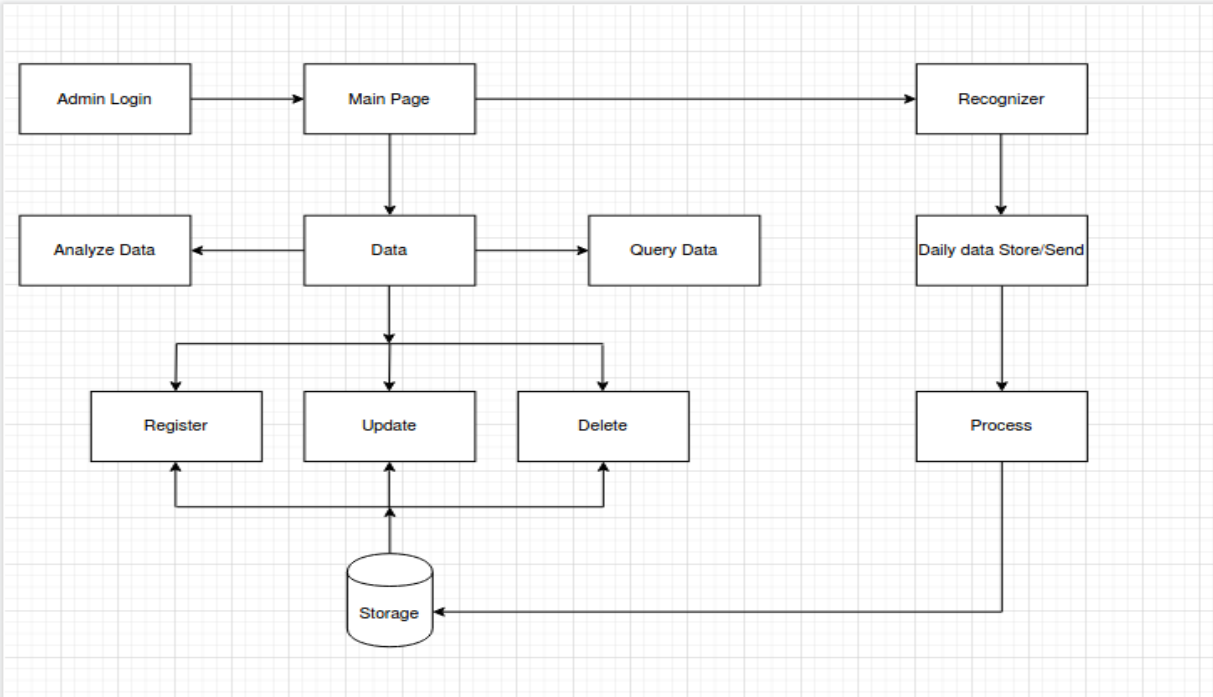


Fig : Proposed system diagram

4.3 State Transition Diagram

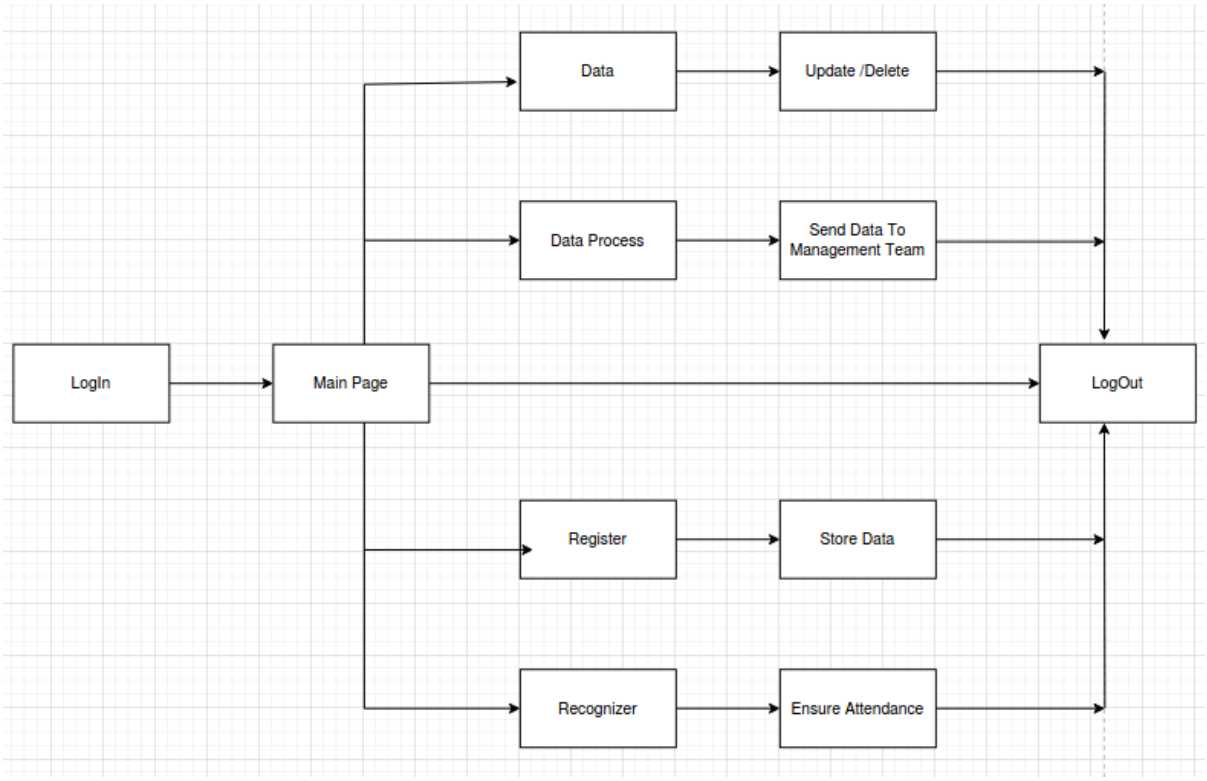


Fig : State transition diagram

4.4 E-R Diagram

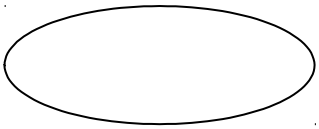
An entity-relationship (ER) diagram is a specialized graphic that illustrates the interrelationships between entities in a database. ER diagrams often use symbols to represent three different types of information. Boxes are commonly used to represent entities. Diamonds are normally used to represent relationships and ovals are used to represent attributes

An **entity-relationship model** (ERM) in software engineering is an abstract and conceptual representation of data. Entity-relationship modeling is a relational schema database modeling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion.

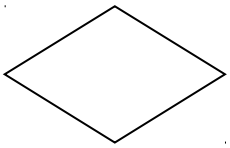
Symbols used in this E-R Diagram:

Entity: Entity is a "thing" in the real world with an independent existence. An entity may be an object with a physical existence such as person, car or employee. Entity symbol is as follows

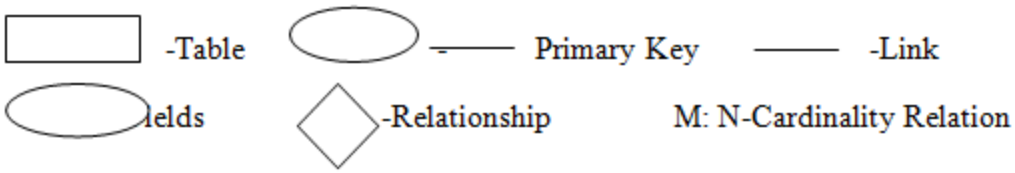
Attribute: Attribute is a particular property that describes the entity. Attribute symbol is



Relationship: Relationship will be several implicit relationships among various entity types whenever an attribute of one entity refers to another entity type some relationship exits. Relationship symbol is:



Key Attributes: An entity type usually has an attribute whose values are distinct for each individual entity in the collection. Such an attribute is called a key attribute. Key attribute symbol is as follows



Entity Relationship

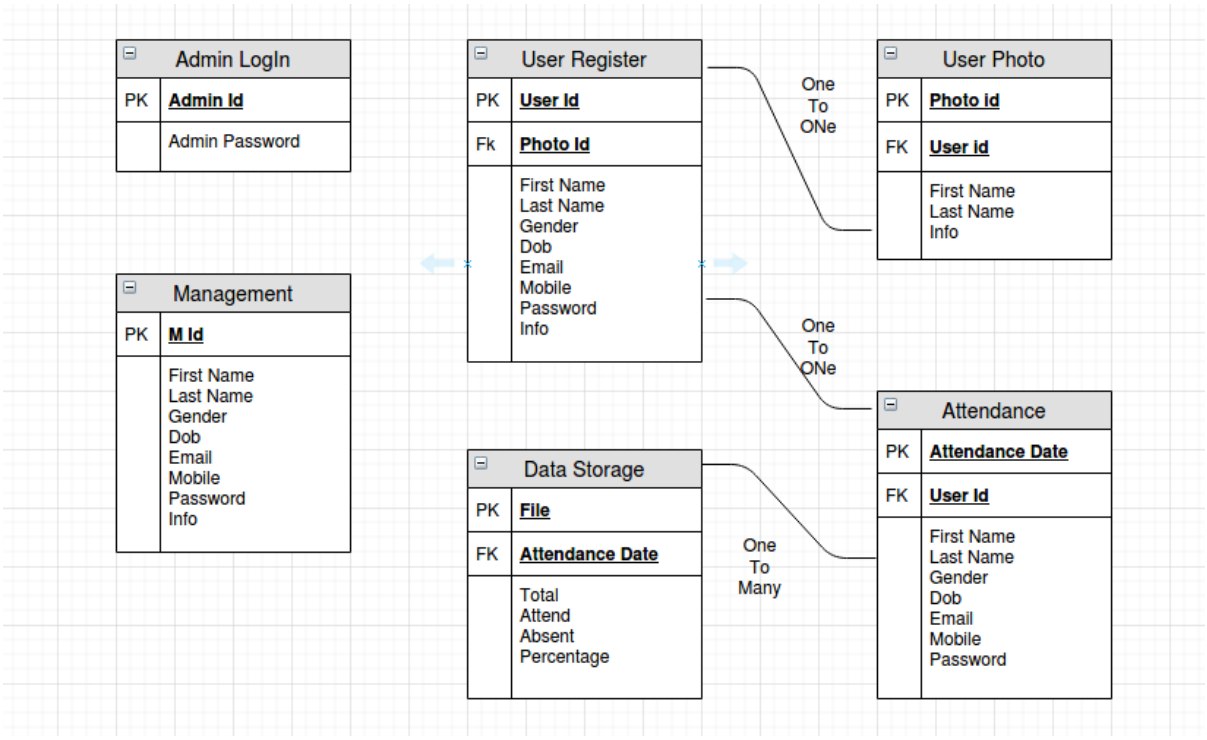
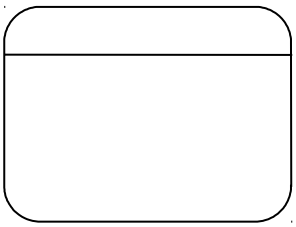


Fig : E-R Diagram

4.5 Data Flow Diagram

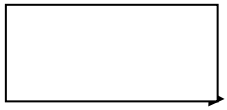
A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an Information System. A data flow diagram can also be used for the visualization of Data Processing. It is common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. This context-level DFD is then "exploded" to show more detail of the system being modeled.

An open rectangle is a data store, data at rest or a temporary repository of data



Process that transforms data flow.

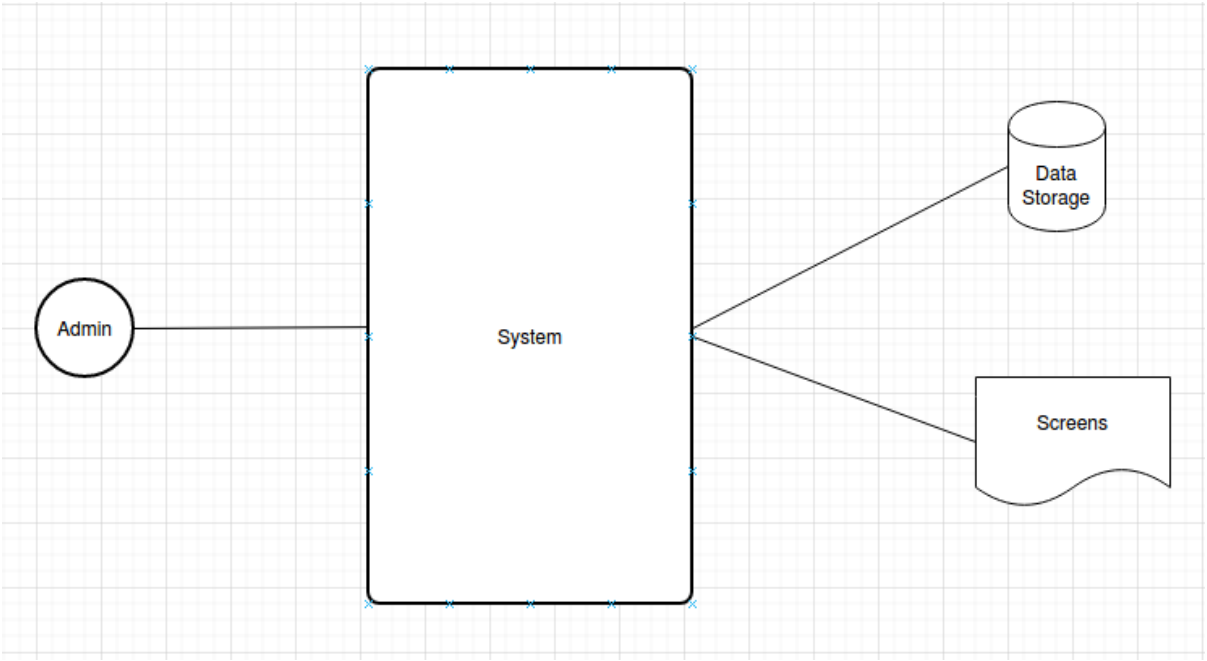
Source or Destination of data



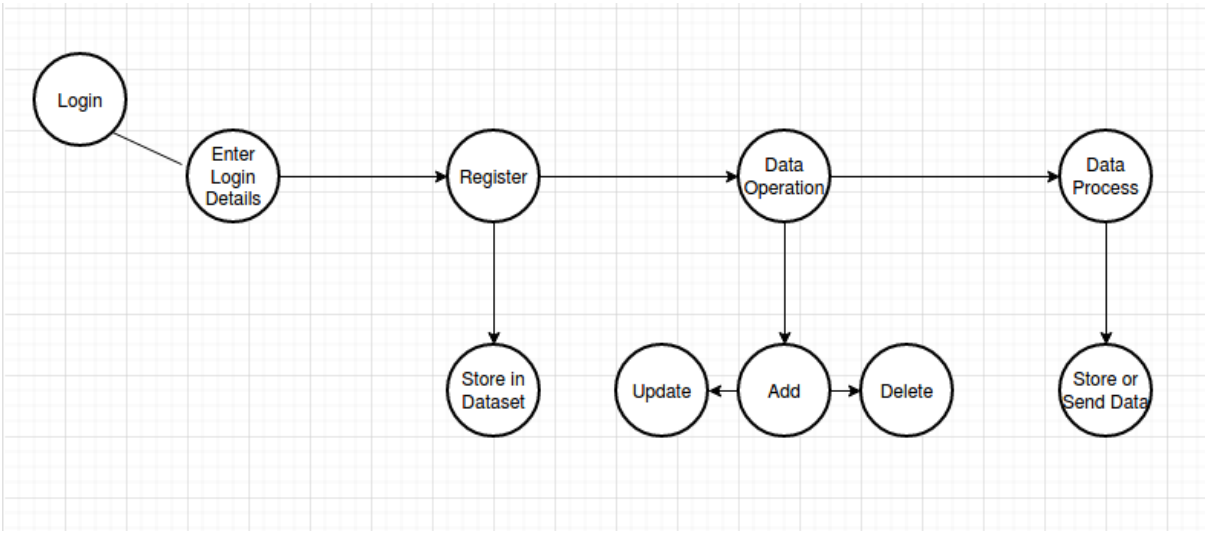
Data flow

Data Store

Data Flow Diagram



Admin DFD



4.6 Features of the system

As large chains invest in facial recognition technology, it’s critical to keep in mind that not all face recognition systems are environment friendly. Without a high match of accuracy and the right feature set, a facial recognition deployment likely won’t deliver the that organization is looking for. We have ensured that the system have following features :

Better matching algorithm : A lot of facial matching algorithms just use tens or hundreds of feature detection points on a face in order to establish identity. But a face recognition solution should be using thousands of points on a face in order to establish identity. A better algorithm can result in a much better accuracy, which will undoubtedly result in a better performance.

Scalability : Facial recognition solutions have to be able to quickly scale flawlessly across hundreds of locations. This means that you need to find a solution that is built to handle large deployments. The solution should also have a support team in place to handle installations, including optimizing cameras for lighting conditions and angle.

Privacy : In this system image data is encrypted both at rest and during transmission. Biometric templates stored within this system would never be converted back into a face image in the case of a data breach. This system is built to prevent profiling by race, age, gender or national origin.

Predictive analytics : The right set of analytics can help this facial recognition system to prevent future faults .It will be crucial for the system to be able to view aggregate data across multiple store locations . This system would help companies easily identify which times of day and which locations are experiencing the highest number of matches, so that each location can be accurately staffed up.

Chapter Five

Proposed Methodologies

5.1 Introduction

Face recognition has become more significant and relevant in recent years owing to its potential applications. Since the faces are highly dynamic and pose more issues and challenges to solve, researchers in the domain of pattern recognition, computer vision and artificial intelligence have proposed many solutions to reduce such difficulties so as to improve the robustness and recognition accuracy. As many approaches have been proposed, efforts are also put in to provide an extensive survey of the methods developed over the years. The objective of this paper is to provide a survey of face recognition papers that appeared in the literature over the past decade under all severe conditions that were not discussed in the previous survey and to categorize them into meaningful approaches, viz. appearance based, feature based and soft computing based. A comparative study of merits and demerits of these approaches have been presented. In recent years, facial recognition draws much attention due to its wide potential applications. As a unique technology in Biometric Identification, facial recognition represents a significant improvement since it could be operated without cooperation of people under detection. Hence, facial recognition will be taken into defense systems, medical detection, human behavior understanding, etc.

5.2 Recognition

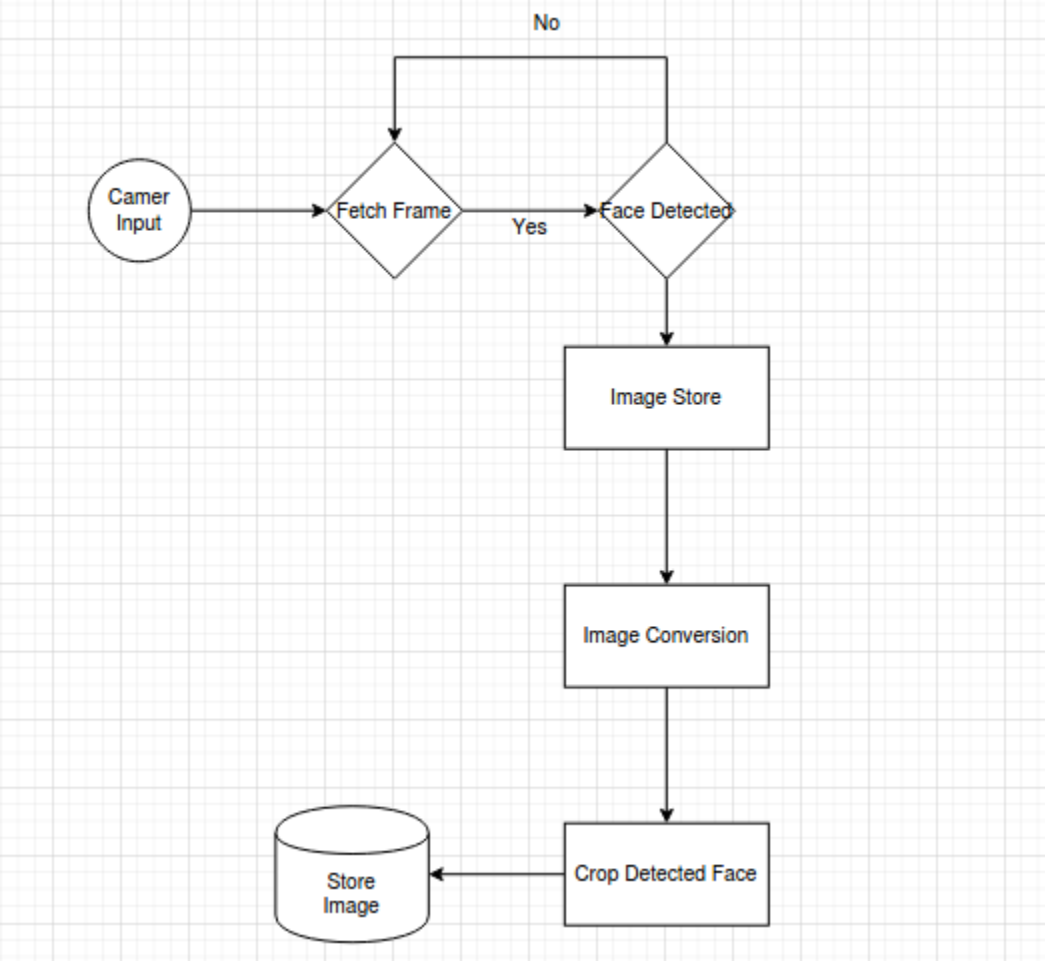
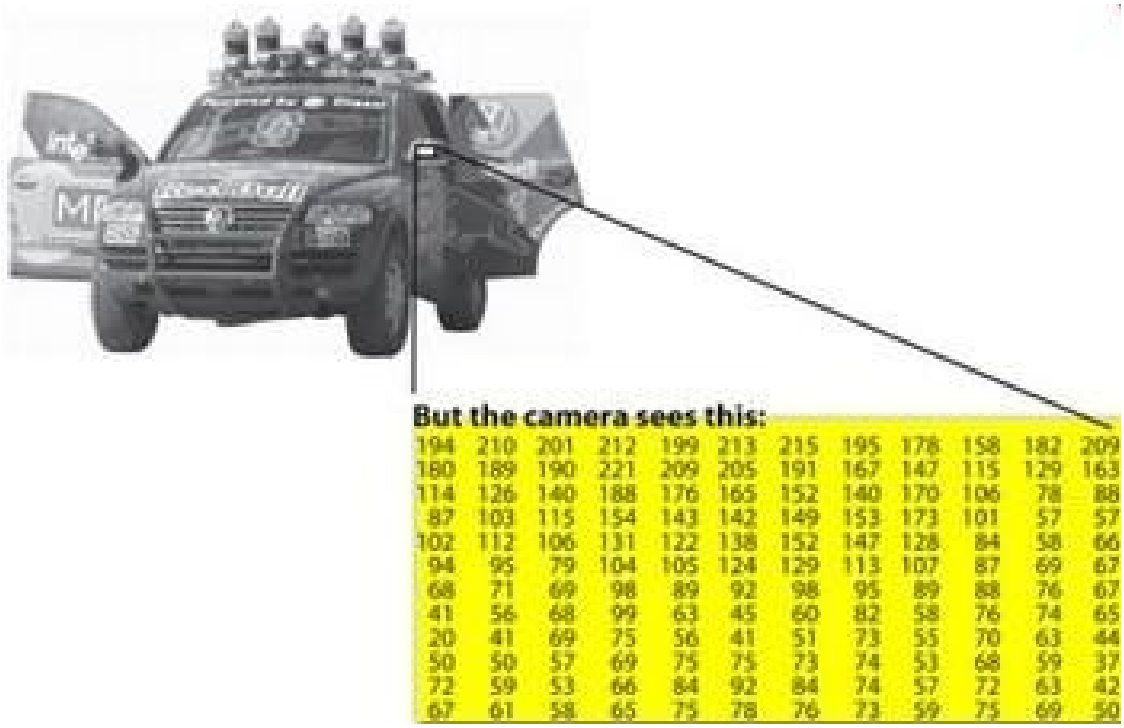


Fig : Face recognizer diagram

Face recognizer diagram shows us that firstly we need input from the cameras . Camera is the starting point of the system. After getting the input from the camera it starts it's work.

After it gets the input secondly it starts processing the data for further work. It processes the detected image and in the last stage it compares detected images with stored images.

5.3 Computer Recognition



Source : Opencv.org

Fig : Example of how computers recognize images

5.4 Facial landmark

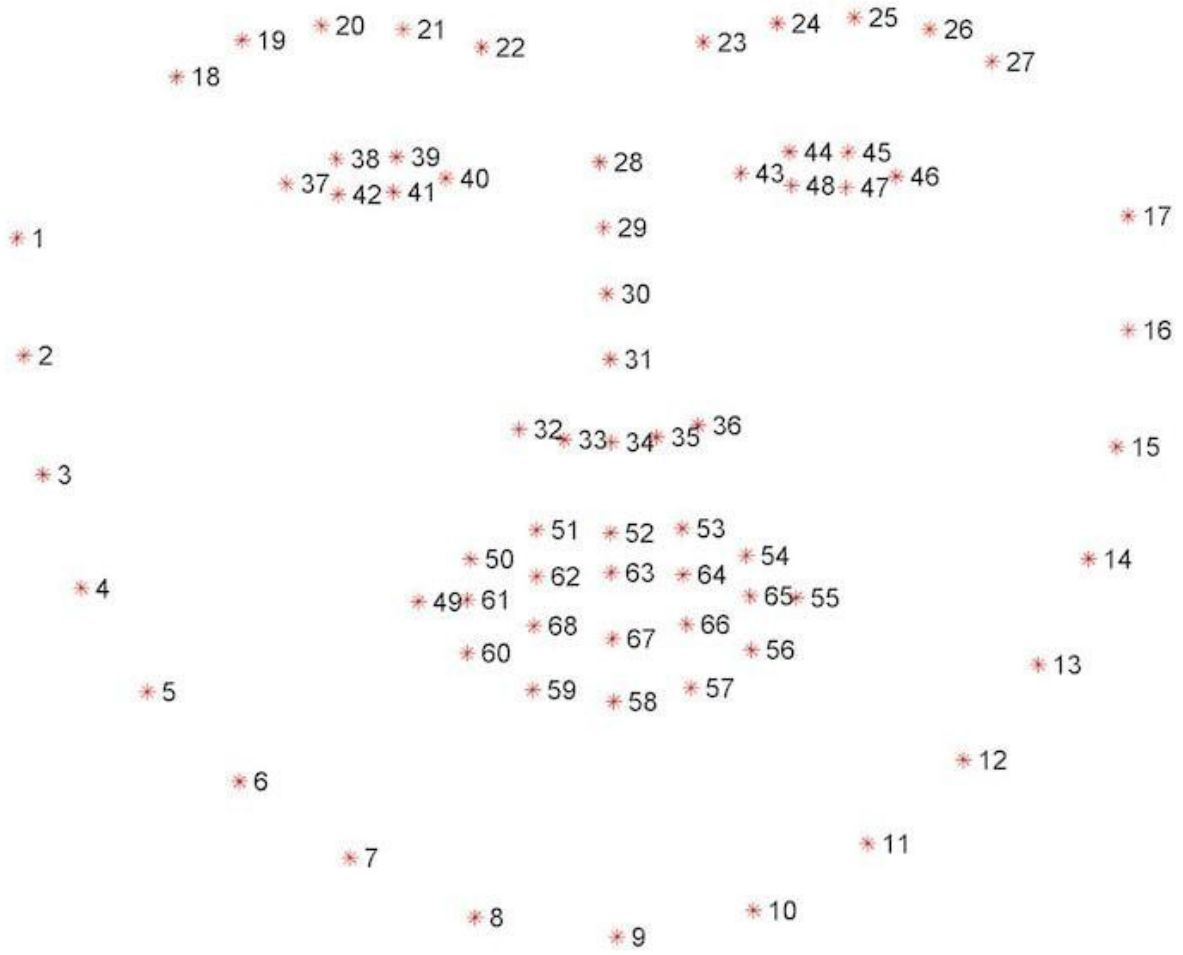


Fig : Facial landmark

For protecting the system from image spoofing attackers we have used the facial landmark model. Facial landmarks are those landmarks that are universal to human beings . By using those landmarks we have ensured that no image spoofing attackers can not get any benefits from the system. In image spoofing attacks attackers use face images to get the entry of the system .

For protecting our system from those kinds of attackers we use eye- blink . By using eye blink we can identify that if the detected face is real .

Chapter Six

Coding

6.1 Introduction

Computer programming (often shortened to programming) is a process that leads from an original formulation of a computing problem to executable programs. It involves activities such as analysis, understanding, and generically solving such problems resulting in an algorithm, verification of requirements of the algorithm including its correctness and its resource consumption, implementation of the algorithm in a target programming language, testing, debugging, and maintaining the source code, implementation of the build system and management of derived artifacts such as machine code of computer programs. The algorithm is often only represented in human-parsable form and reasoned about using logic. Source code is written in one or more programming languages (such as C, C++, C#, Java, Python, Smalltalk, JavaScript, etc.). The purpose of programming is to find a sequence of instructions that will automate performing a specific task or solve a given problem. The process of programming thus often requires expertise in many different subjects, including knowledge of the application domain, specialized algorithms and formal logic.

6.2 Key Components

Reliability: how often the results of a program are correct. This depends on conceptual correctness of algorithms, and minimization of programming mistakes, such as mistakes in resource management (e.g., buffer overflows and race conditions) and logic errors (such as division by zero or off-by-one errors).

Robustness: how well a program anticipates problems due to errors (not bugs). This includes situations such as incorrect, inappropriate or corrupt data, unavailability of needed resources such as memory, operating system services and network connections, user error, and unexpected power outages.

Usability: the ergonomics of a program: the ease with which a person can use the program for its intended purpose or in some cases even unanticipated purposes. Such issues can make or break its success even regardless of other issues. This involves a wide range of textual, graphical and sometimes hardware elements that improve the clarity, intuitiveness, cohesiveness and completeness of a program's user interface.

Portability: the range of computer hardware and operating system platforms on which the source code of a program can be compiled/interpreted and run. This depends on differences in the programming facilities provided by the different platforms, including hardware and operating system resources, expected behavior of the hardware and operating system, and avail-

ability of platform specific compilers (and sometimes libraries) for the language of the source code.

Maintainability: the ease with which a program can be modified by its present or future developers in order to make improvements or customizations, fix bugs and security holes, or adapt it to new environments. Good practices during initial development make the difference in this regard. This quality may not be directly apparent to the end user but it can significantly affect the fate of a program over the long term.

Efficiency/performance: the amount of system resources a program consumes (processor time, memory space, slow devices such as disks, network bandwidth and to some extent even user interaction): the less, the better. This also includes careful management of resources, for example cleaning up temporary files and eliminating memory leaks.

Login box to verify authentication:

```
package auth;
import landing_page.Menu;
import javax.swing.JOptionPane;
public class Login extends javax.swing.JFrame {
    public Login() {
        initComponents();
    }
    private void initComponents() {
        jPanel2 = new javax.swing.JPanel();
        txt_user = new javax.swing.JTextField();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        txt_pass = new javax.swing.JPasswordField();
        jButton1 = new javax.swing.JButton();
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("System Login");
        getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());
```

```
        jPanel2.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());
        txt_user.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txt_userActionPerformed(evt);
            }
        });
        jPanel2.add(txt_user, new org.netbeans.lib.awtextra.AbsoluteConstraints(60, 110, 260,
-1));
        jLabel1.setText("User");
        jPanel2.add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(60, 70, -1, -1));
        jLabel2.setText("Password");
        jPanel2.add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(60, 150, -1,
-1));
        jLabel3.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel3.setText("Login");
        jPanel2.add(jLabel3, new org.netbeans.lib.awtextra.AbsoluteConstraints(170, 30, -1,
-1));
        txt_pass.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txt_passActionPerformed(evt);
            }
        });
        jPanel2.add(txt_pass, new org.netbeans.lib.awtextra.AbsoluteConstraints(60, 180, 260,
-1));
        jButton1.setText("Enter");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });
```

```
        jPanel2.add(jButton1, new org.netbeans.lib.awtextra.AbsoluteConstraints(160, 240, -1, -1));

        getContentPane().add(jPanel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(160, 40, 390, 320));

        setSize(new java.awt.Dimension(738, 440));

        setLocationRelativeTo(null);

    }// </editor-fold>

    private void txt_userActionPerformed(java.awt.event.ActionEvent evt) {

 }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        String user=txt_user.getText();

        String password=txt_pass.getText();

        if( password.equals("admin")){

            Menu menu=new Menu(user);

            menu.setVisible(true);

            dispose();

        }else JOptionPane.showMessageDialog(null, "Try again....");

    }

    private void txt_passActionPerformed(java.awt.event.ActionEvent evt) {

}

    public static void main(String args[]) {

        try {

            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

                if ("Windows".equals(info.getName())) {

                    javax.swing.UIManager.setLookAndFeel(info.getClassName());

                    break;

                }

            }

        } catch (ClassNotFoundException ex) {
```

```
        java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.-
SEVERE, null, ex);

        } catch (InstantiationException ex) {          java.util.logging.Logger.getLogger(Login.-
class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (IllegalAccessException ex) {          java.util.logging.Logger.getLogger(Login.-
class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (javax.swing.UnsupportedLookAndFeelException ex) {             java.util.log-
ging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);

        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {

                new Login().setVisible(true);

            }

        });

    }

    private javax.swing.JButton jButton1;

    private javax.swing.JLabel jLabel1;

    private javax.swing.JLabel jLabel2;

    private javax.swing.JLabel jLabel3;

    private javax.swing.JPanel jPanel2;

    private javax.swing.JPasswordField txt_pass;

    private javax.swing.JTextField txt_user;
```

Menu page for navigation:

```
package landing_page;

import register.RegisterStudent;

import attendance.AttendanceCheck;

public class Menu extends javax.swing.JFrame {

    public Menu(String user) {
```

```
        initComponents();
        txt_title_menu.setText("Welcome "+user);
    }
    private Menu() {
        throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
    }
    private void initComponents() {
        jPanel1 = new javax.swing.JPanel();
        jLabel2 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        txt_title_menu = new javax.swing.JLabel();
        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("Menu");
        getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());
        jPanel1.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());
        jLabel2.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel2.setText("Choose an option");
        jPanel1.add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(402, 96, -1-1));
        jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel1.setText("What you want??");
        jPanel1.add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(246, 96, -1-1));
        jButton1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
```

```
        jButton1.setText("Capture");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }       });
        jPanel1.add(jButton1, new org.netbeans.lib.awtextra.AbsoluteConstraints(40, 200,130));
        jButton2.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jButton2.setText("Data");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });
        jPanel1.add(jButton2, new org.netbeans.lib.awtextra.AbsoluteConstraints(560, 200, 180,
130));
        jButton3.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jButton3.setText("Recognize");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });
        jPanel1.add(jButton3, new org.netbeans.lib.awtextra.AbsoluteConstraints(310, 200, 170,
130));
        txt_title_menu.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        txt_title_menu.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
```

```
        txt_title_menu.setText("Welcome User");

        jPanel1.add(txt_title_menu, new org.netbeans.lib.awtextra.AbsoluteConstraints(330, 50,
-1, -1));

        getContentPane().add(jPanel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(10,
10, 790, 470));

        setSize(new java.awt.Dimension(826, 526));

        setLocationRelativeTo(null);

    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        new RegisterStudent().setVisible(true);

    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

        new AttendanceCheck().setVisible(true);

    }

    public static void main(String args[]) {

        try {

            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManag-
er.getInstalledLookAndFeels()) {

                if ("Windows".equals(info.getName())) {

                    javax.swing.UIManager.setLookAndFeel(info.getClassName());

                    break;

                }

            }

        } catch (ClassNotFoundException ex) {
```

```
            java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Lev-
el.SEVERE, null, ex);

        } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(Menu.-
class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (IllegalAccessException ex) {          java.util.logging.Logger.getLogger(Menu.-
class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (javax.swing.UnsupportedLookAndFeelException ex) {          java.util.log-
ging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);

        }

        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {

                new Menu().setVisible(true);

            }

        });

    }

    private javax.swing.JButton jButton1;

    private javax.swing.JButton jButton2;

    private javax.swing.JButton jButton3;

    private javax.swing.JLabel jLabel1;

    private javax.swing.JLabel jLabel2;

    private javax.swing.JPanel jPanel1;

    private javax.swing.JLabel txt_title_menu;

}
```

Model file:

```
package models;

import lombok.AllArgsConstructor;
```

```
import lombok.Data;
import lombok.NoArgsConstructor;
public class Student {
    private int id;
    private String first_name;
    private String last_name;
    private int student_class;
    private String class_section;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getFirst_name() {
        return first_name;
    }
    public void setFirst_name(String first_name) {
        this.first_name = first_name;
    }
    public String getLast_name() {
        return last_name;
    }

    public void setLast_name(String last_name) {
```

```
        this.last_name = last_name;
    }
    public int getStudent_class() {
        return student_class;
    }
    public void setStudent_class(int student_class) {
        this.student_class = student_class;
    }
    public String getClass_section() {
        return class_section;
    }
    public void setClass_section(String class_section) {
        this.class_section = class_section;
    }
    @Override
    public String toString() {
        return "Student{" + "id=" + id + ", first_name=" + first_name + ", last_name=" +
last_name + ", student_class=" + student_class + ", class_section=" + class_section + '}';
    }
}


package models;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
public class Employee {
```

```
    private int id;
    private String first_name;
    private String last_name;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getFirst_name() {
        return first_name;
    }
    public void setFirst_name(String first_name) {
        this.first_name = first_name;
    }
    public String getLast_name() {
        return last_name;
    }
    public void setLast_name(String last_name) {
        this.last_name = last_name;
    }
    @Override
    public String toString() {
        return "Employee{" + "id=" + id + ", first_name=" + first_name + ", last_name=" +
last_name + '}';
    }
```

```
}
```

Database connection:

```
package db_connection;

public class DB_Connection {

   public Connection dbConnection;

   public ResultSet resultSet;

   public boolean ifTableExists(String tableName){

      boolean ans=false;

      try{

         DatabaseMetaData databaseMetaData = dbConnection.getMetaData();

         ResultSet resultSet = databaseMetaData.getTables(null, null, "student", new String[]
{"TABLE"});

         ans = resultSet.next();

      }catch(SQLException exception){

         System.out.println(exception);

      }

      return ans;

   }

   public void createTable(String tableName) {

      try {

         PreparedStatement preparedStatement = dbConnection.

               prepareStatement("CREATE TABLE student (id int, first_name varchar(255),
last_name varchar(255), student_class int, class_section varchar(22))");

         preparedStatement.executeUpdate();

      } catch (SQLException exception) {
```

```
            System.out.println(exception);
        }
    }
    public void connectDatabase() {
        try {
            System.setProperty("jdbc.Driver", "org.mysql.Driver");
            dbConnection = DriverManager.
                    getConnection("jdbc:mysql://127.0.0.1/face_recognition", "root", "");
            System.out.println("Successfuly connected to database");
        } catch (SQLException e) {
            System.out.println("Unsuccessful database connection " + e);
        }
    }
    public void disconnectDatabase() {
        try {
            dbConnection.close();
        } catch (SQLException e) {
            System.out.println("Could not disconnect database connection " + e);
        }
    }
    public void executesql(String sql) {
        try {
            Statement statement =
dbConnection.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CON-
CUR_READ_ONLY);
            resultSet = statement.executeQuery(sql);
```

```
            System.out.println(resultSet);
        } catch (Exception e) {
            System.out.println("Could not execute query " + e);
        }
    }
}
```

Register face:

```
package register;
public class CaptureFace extends javax.swing.JFrame {
    private CaptureFace.DaemonThread myThread = null;
    VideoCapture webSource = null;
    Mat cameraImage = new Mat();
    CascadeClassifier cascade = new CascadeClassifier(harcascade);
    BytePointer mem = new BytePointer();
    RectVector detectedFaces = new RectVector();
    public CaptureFace(int id,String first_name,String last_name,int student_class,String stu-
dent_section){
        initComponents();
        this.studentId=id;
        this.first_name=first_name;
        this.last_name=last_name;
        this.student_class= student_class;
        this.student_section= student_section;
        startCamera();
    }
```

```
    public CaptureFace(int id,String first_name,String last_name){
        initComponents();
        this.employeeId=id;
        this.first_name=first_name;
        this.last_name=last_name;
        startCamera();
    }
    private CaptureFace() {
        throw new UnsupportedOperationException("Not supported yet."); //To change body of
generated methods, choose Tools | Templates.
    }
    private void initComponents() {
        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        label_photo = new javax.swing.JLabel();
        jPanel2 = new javax.swing.JPanel();
        counterLabel = new javax.swing.JLabel();
        saveButton = new javax.swing.JButton();
        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("Face capture");
        getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());
        jPanel1.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());
        jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText("Capture 25 snapshot");
```

```
        jPanel1.add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(30, 10, 420,
30));

        label_photo.setBorder(javax.swing.BorderFactory.createMatteBorder(1, 1, 1, 1, new ja-
va.awt.Color(200, 200, 200)));

        jPanel1.add(label_photo, new org.netbeans.lib.awtextra.AbsoluteConstraints(30, 60,
420, 280));

        jPanel2.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

        counterLabel.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

        counterLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

        counterLabel.setText("00");

        jPanel2.add(counterLabel, new org.netbeans.lib.awtextra.AbsoluteConstraints(20, 20,
380, 30));



        saveButton.setText("Capture");

        jPanel2.add(saveButton, new org.netbeans.lib.awtextra.AbsoluteConstraints(180, 60, -1,
-1));

        jPanel1.add(jPanel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(30, 360, 420,
100));

        getContentPane().add(jPanel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(10,
10, 470, 470));

        setSize(new java.awt.Dimension(507, 531));

        setLocationRelativeTo(null);

    }

    private javax.swing.JLabel counterLabel;

    private javax.swing.JLabel jLabel1;

    private javax.swing.JPanel jPanel1;

    private javax.swing.JPanel jPanel2;

    private javax.swing.JLabel label_photo;

    private javax.swing.JButton saveButton;
```

```
class DaemonThread implements Runnable {
    protected volatile boolean runnable = false;
    @Override
    public void run(){
        synchronized (this) {
            while (runnable) {
                try {
                    if (webSource.grab()) {
                        webSource.retrieve(cameraImage);
                        Graphics g = label_photo.getGraphics();
                        Mat imageColor = new Mat();
                        imageColor = cameraImage;
                        Mat imageGray = new Mat();
                        cvtColor(imageColor, imageGray, COLOR_BGRA2GRAY);
                        RectVector detectedFaces = new RectVector();
                        cascade.detectMultiScale(imageColor, detectedFaces, 1.1, 1, 0, new
Size(150, 150), new Size(500, 500));
                        for (int i = 0; i < detectedFaces.size(); i++) {
                            Rect dadosFace = detectedFaces.get(0);
                            rectangle(imageColor, dadosFace, new Scalar(255, 255, 255, 5));
                            Mat face = new Mat(imageGray, dadosFace);
                            opencv_imgproc.resize(face, face, new Size(160, 160));
                            if (saveButton.getModel().isPressed()) {
                                if (sample <= numSamples) {
                                    imwrite(cropped, face);
                                        counterLabel.setText(String.valueOf(sample));
```

```
                    sample++;
                }
                if (sample > 5) {
                    generate();
                    System.out.println("person register done");
                    stopCamera();
                }
            }
        }
        imencode(".bmp", cameraImage, mem);
        Image im = ImageIO.read(new ByteArrayInputStream(mem.getString-
Bytes()));
        BufferedImage buff = (BufferedImage) im;
        if (g.drawImage(buff, 0, 0, getWidth(), getHeight() - 90, 0, 0, buf-
f.getWidth(), buff.getHeight(), null)) {
            if (runnable == false) {
                System.out.println("Salve a Foto");
                this.wait();
            }
        }
    }
} catch (IOException ex) {
    ex.printStackTrace();
    JOptionPane.showMessageDialog(null, "Erro ao iniciar camera (IOEx)\n" +
} catch (InterruptedException ex) {
    ex.printStackTrace();
```

```
                    JOptionPane.showMessageDialog(null, "Erro ao iniciar camera
(Interrupted)\n" + ex);

                }

            }

        }

    }

    public void generate() {

        File directory=new File("/home/sagar/Desktop/Nu/Face_Recognition_Attendance/Com-
puter_vision_attendance/Images/");

        FilenameFilter filter=new FilenameFilter(){

            @Override

            public boolean accept(File dir, String name) {

                return name.endsWith(".jpg") || name.endsWith(".png");

            }

        };

        File[] files=directory.listFiles(filter);

        MatVector photos=new MatVector(files.length);

        Mat labels=new Mat(files.length,1,CV_32SC1);

        IntBuffer labelsBuffer=labels.createBuffer();



        int counter=0;

        for(File image: files){

            Mat photo=imread(image.getAbsolutePath(),CV_32SC1);

            int idPerson=Integer.parseInt(image.getName().split("\\.")[1]);

            opencv_imgproc.resize(photo, photo, new Size(160,160));
```

```
            photos.put(counter,photo);
            labelsBuffer.put(counter,idPerson);
            counter++;
        }
        FaceRecognizer lbph=LBPHFaceRecognizer.create();
        lbph.train(photos, labels);

lbph.save("/home/sagar/Desktop/Nu/Face_Recognition_Attendance/Computer_vision_attendance/Images/classifierLBPH.yml");
    }
    public void insertDatabase() throws SQLException{
        StoreStudentInformation storeStudentInformation = new StoreStudentInformation();
        Student  student = new Student();
        student.setId(studentId);
        student.setFirst_name(first_name);
        student.setLast_name(last_name);
        student.setStudent_class(student_class);
        student.setClass_section(student_section);
        storeStudentInformation.insertStudentInformation(student);
    }

    public void stopCamera() {
        myThread.runnable = false;
        webSource.release();
        dispose();
    }
```

```
    public void startCamera() {
        webSource=new VideoCapture(0);
        myThread=new CaptureFace.DaemonThread();
        Thread t=new Thread(myThread);
        t.setDaemon(true);
        myThread.runnable=true;
        t.start();
    }
}
```

Register :

```
package register;
import register.*;
import db_connection.DB_Connection;
import java.sql.SQLException;

public class RegisterEmployee extends javax.swing.JFrame {
    int id = 1;
    DB_Connection dbConnection = new DB_Connection();

    public RegisterEmployee() {
        initComponents();
        setIdForRegister();
    }
     private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        String first_name = txt_first_name.getText();
        String last_name = txt_last_name.getText();
        int id = Integer.parseInt(txt_id_label.getText());
        new CaptureFace(id, first_name, last_name).setVisible(true);
    }

    public static void main(String args[]) {
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Windows".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(RegisterEmployee.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(RegisterEmployee.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(RegisterEmployee.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(RegisterEmployee.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
```

```
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new RegisterEmployee().setVisible(true);
        }
    });
}
private javax.swing.JLabel txt_id_label;
private javax.swing.JTextField txt_last_name;
private javax.swing.JTextField txt_office;
private void setIdForRegister() {
    dbConnection.connectDatabase();
    if (dbConnection.ifTableExists("student")) {
        dbConnection.executesql("SELECT id FROM student ORDER BY id DESC ");
    } else {
        dbConnection.createTable("student");
        dbConnection.executesql("SELECT id FROM student ORDER BY id DESC ");
    }
    try {
        if (dbConnection.resultSet != null) {
            dbConnection.resultSet.first();
            txt_id_label.setText(String.valueOf(dbConnection.resultSet.getInt("id")));
            id = Integer.parseInt(txt_id_label.getText());
            id++;
            txt_id_label.setText(String.valueOf(id));
        }else{
```

```
            txt_id_label.setText(String.valueOf(id));
        }
    } catch (NumberFormatException | SQLException e) {
        System.out.println(e);
    }
  }
}
```

Register student:

```
package register;
import db_connection.DB_Connection;
import java.sql.SQLException;
public class RegisterStudent extends javax.swing.JFrame {
  int id = 1;
  DB_Connection dbConnection = new DB_Connection();
  public RegisterStudent() {
    initComponents();
    setIdForRegister();
  }
  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    String first_name = txt_first_name.getText();
    String last_name = txt_last_name.getText();
    int student_class = 4;
    String class_section = "a";
    int id = Integer.parseInt(txt_id_label.getText());
```

```
        new CaptureFace(id, first_name, last_name, student_class,
class_section).setVisible(true);

    }

    public static void main(String args[]) {

        try {

            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManag-
er.getInstalledLookAndFeels()) {

                if ("Windows".equals(info.getName())) {

                    javax.swing.UIManager.setLookAndFeel(info.getClassName());

                    break;

                }

            }

        } catch (ClassNotFoundException ex) {          java.util.logging.Logger.getLogger(Regis-
terStudent.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (InstantiationException ex) {          java.util.logging.Logger.getLogger(Regis-
terStudent.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (IllegalAccessException ex) {          java.util.logging.Logger.getLogger(Regis-
terStudent.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (javax.swing.UnsupportedLookAndFeelException ex) {          java.util.log-
ging.Logger.getLogger(RegisterStudent.class.getName()).log(java.util.logging.Level.SE-
VERE, null, ex);

        }

        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {

                new RegisterStudent().setVisible(true);

            }

        });

    }

    private javax.swing.JButton jButton1;
```

```
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JPanel jPanel3;
private javax.swing.JFormattedTextField txt_dob;
private javax.swing.JTextField txt_first_name;
private javax.swing.JLabel txt_id_label;
private javax.swing.JTextField txt_last_name;
private javax.swing.JTextField txt_office;
private void setIdForRegister() {
    dbConnection.connectDatabase();
    if (dbConnection.ifTableExists("student")) {
        dbConnection.executesql("SELECT id FROM student ORDER BY id DESC ");
    } else {
        dbConnection.createTable("student");
        dbConnection.executesql("SELECT id FROM student ORDER BY id DESC ");
    }
    try {
        if (dbConnection.resultSet != null) {
            dbConnection.resultSet.first();
            txt_id_label.setText(String.valueOf(dbConnection.resultSet.getInt("id")));
            id = Integer.parseInt(txt_id_label.getText());
```

```
            id++;
            txt_id_label.setText(String.valueOf(id));
        }else{
            txt_id_label.setText(String.valueOf(id));
        }
    } catch (NumberFormatException | SQLException e) {
        System.out.println(e);
    }
  }
}
```

Data access object:

```
package dao;
import db_connection.DB_Connection;
import models.Student;
import java.sql.PreparedStatement;
import java.sql.SQLException;
public class StoreStudentInformation {
  DB_Connection dbConnectionClass = new DB_Connection();
  public void insertStudentInformation(Student student) {
    dbConnectionClass.connectDatabase();
    if (dbConnectionClass.ifTableExists("student")) {
        System.out.println("table exists");
        insertStudent(student);
    } else {
```

```
            System.out.println("table not exists");
            dbConnectionClass.createTable("student");
            insertStudent(student);
        }
        dbConnectionClass.disconnectDatabase();
    }
    private void insertStudent(Student student) {
        try {
            PreparedStatement preparedStatement = dbConnectionClass.dbConnection.
                    prepareStatement("INSERT INTO student
(id,first_name,last_name,student_class,class_section) values (?,?,?,?,?)");
            preparedStatement.setInt(1, student.getId());
            preparedStatement.setString(2, student.getFirst_name());
            preparedStatement.setString(3, student.getLast_name());
            preparedStatement.setInt(4, student.getStudent_class());
            preparedStatement.setString(5, student.getClass_section());
            preparedStatement.executeUpdate();
        } catch (SQLException exception) {
            System.out.println(exception);
        }
    }
}
```

Attendance:

```
package attendance;

public class AttendanceCheck extends javax.swing.JFrame {

    private AttendanceCheck.DaemonThread myThread = null;

    VideoCapture webSource = null;

    Mat cameraImage = new Mat();

    CascadeClassifier cascade = new
CascadeClassifier("/home/sagar/Desktop/Nu/Face_Recognition_Attendance/Computer_vi-
sion_attendance/haarcascade_frontalface_alt.xml");

    BytePointer mem = new BytePointer();

    FaceRecognizer recognizer=LBPHFaceRecognizer.create();

    RectVector detectedFaces = new RectVector();

    String root ,firstNamePerson,lastNamePerson,officePerson,dobPerson;

    int idPerson;

    DB_Connection cd = new DB_Connection();

    public AttendanceCheck() {

        initComponents();

        recognizer.read("/home/sagar/Desktop/Nu/Face_Recognition_Attendance/Computer_vi-
sion_attendance/Images/classifierLBPH.yml");

        recognizer.setThreshold(80);

        startCamera();

    }

    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();

        jLabel1 = new javax.swing.JLabel();

        label_photo = new javax.swing.JLabel();

        jPanel2 = new javax.swing.JPanel();

        labelOffice = new javax.swing.JLabel();
```

```
        label_name = new javax.swing.JLabel();

        jButton1 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);

        setTitle("Face Recognizer");

        getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

        jPanel1.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

        jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

        jLabel1.setText("Capture 25 snapshot");

        jPanel1.add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(30, 60, 420,
30));

        label_photo.setBorder(javax.swing.BorderFactory.createMatteBorder(1, 1, 1, 1, new ja-
va.awt.Color(200, 200, 200)));

        jPanel1.add(label_photo, new org.netbeans.lib.awtextra.AbsoluteConstraints(30, 110,
420, 280));

        jPanel2.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

        labelOffice.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

        labelOffice.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

        labelOffice.setText("Office");

        jPanel2.add(labelOffice, new org.netbeans.lib.awtextra.AbsoluteConstraints(20, 60, 380,
30));

        label_name.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

        label_name.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

        label_name.setText("Name");

        jPanel2.add(label_name, new org.netbeans.lib.awtextra.AbsoluteConstraints(20, 20,
380, 30));
```

```
        jPanel1.add(jPanel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(30, 410, 420,
100));

        jButton1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

        jButton1.setText("Stop camera");

        jButton1.addActionListener(new java.awt.event.ActionListener() {

            public void actionPerformed(java.awt.event.ActionEvent evt) {

                jButton1ActionPerformed(evt);

            }

        });

        jPanel1.add(jButton1, new org.netbeans.lib.awtextra.AbsoluteConstraints(180, 10, -1,
-1));

        getContentPane().add(jPanel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(10,
10, 470, 520));

        setSize(new java.awt.Dimension(514, 577));

        setLocationRelativeTo(null);

    }// </editor-fold>

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        stopCamera();

    }

    public static void main(String args[]) {

            try {

           for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManag-
er.getInstalledLookAndFeels()) {

                if ("Windows".equals(info.getName())) {

                    javax.swing.UIManager.setLookAndFeel(info.getClassName());

                    break;

                }
```

```
        }       } catch (ClassNotFoundException ex) {

          java.util.logging.Logger.getLogger(AttendanceCheck.class.getName()).log(java.u-
til.logging.Level.SEVERE, null, ex);       } catch (InstantiationException ex) {          java.u-
til.logging.Logger.getLogger(AttendanceCheck.class.getName()).log(java.util.logging.Level.-
SEVERE, null, ex);

      } catch (IllegalAccessException ex) {          java.util.logging.Logger.getLogger(Atten-
danceCheck.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

      } catch (javax.swing.UnsupportedLookAndFeelException ex) {          java.util.log-
ging.Logger.getLogger(AttendanceCheck.class.getName()).log(java.util.logging.Level.SE-
VERE, null, ex);       }

      java.awt.EventQueue.invokeLater(new Runnable() {

          public void run() {

              new AttendanceCheck().setVisible(true);

          }

      });

  }

  private javax.swing.JButton jButton1;

  private javax.swing.JLabel jLabel1;

  private javax.swing.JPanel jPanel1;

  private javax.swing.JPanel jPanel2;

  private javax.swing.JLabel labelOffice;

  private javax.swing.JLabel label_name;

  private javax.swing.JLabel label_photo;

  // End of variables declaration

class DaemonThread implements Runnable {

      protected volatile boolean runnable = false;


      @Override
```

```
        public void run() {
            synchronized (this) {
                while (runnable) {
                    try {
                        if (webSource.grab()) {
                            webSource.retrieve(cameraImage);
                            Graphics g = label_photo.getGraphics();
                            Mat imageGray = new Mat();
                            cvtColor(cameraImage, imageGray, COLOR_BGRA2GRAY);
                            RectVector detectedFace = new RectVector();
                            cascade.detectMultiScale(imageGray, detectedFace, 1.1, 2, 0, new Size(150,
150), new Size(500, 500));
                            for (int i = 0; i < detectedFace.size(); i++) {
                                Rect dadosFace = detectedFace.get(i);
                                rectangle(cameraImage, dadosFace, new Scalar(0, 255, 0, 0));
                                Mat faceCapturada = new Mat(imageGray, dadosFace);
                                opencv_imgproc.resize(faceCapturada, faceCapturada, new Size(160,
160));
                                IntPointer rotulo = new IntPointer(1);
                                DoublePointer confidence = new DoublePointer(1);
                                recognizer.predict(faceCapturada, rotulo, confidence);
                                int prediction = rotulo.get(0);
                                String name = null;
                                if (prediction == -1) {
                                    label_name.setText("Not Recognized");
                                    labelOffice.setText("Not Found");
```

```
                    idPerson = 0;
                } else {
                    System.out.println(confidence.get(0));
                    idPerson = prediction;
                    rec();
                }
                //int x = Math.max(dadosFace.tl().x() - 10, 0);
                //int y = Math.max(dadosFace.tl().y() - 10, 0);
                //putText(imagemCamera, nome, new Point(x, y),
FONT_HERSHEY_PLAIN, 1.7, new Scalar(0, 255, 0, 2));
            }
            imencode(".bmp", cameraImage, mem);
            Image im = ImageIO.read(new ByteArrayInputStream(mem.getString-
Bytes()));
            BufferedImage buff = (BufferedImage) im
            if (g.drawImage(buff, 0, 0, getWidth(), getHeight() - 100, 0, 0, buf-
f.getWidth(), buff.getHeight(), null)) {
                if (runnable == false) {
                    System.out.println("Salve a Foto");
                    this.wait();
                }
            }
        }
    } catch (IOException | InterruptedException ex) {
    }
  }
}
```

```
    }


    private void rec() {
        SwingWorker worker=new SwingWorker() {
            @Override
            protected Object doInBackground() throws Exception {
                cd.connectDatabase();
                try {
                    String sql="SELECT * FROM person WHERE id ="+String.valueOf(idPer-
son);
                    cd.executesql(sql);
                    while(cd.resultSet.next()){
                        label_name.setText(cd.resultSet.getString("first_name")+" "+ cd.result-
Set.getString("last_name"));
                        labelOffice.setText(cd.resultSet.getString("office"));
                        System.out.println("person : "+cd.resultSet.getString("id"));
                        Array ident=cd.resultSet.getArray(2);
                        String[] person=(String[]) ident.getArray();
                        for (int i = 0; i < person.length; i++) {
                            System.out.println(person[i]);
                        }
                    }
                } catch (Exception e) {


                }
                cd.disconnectDatabase();
```

```
                return null;
            }
        };
        worker.execute();
    }
}
  public void stopCamera() {
    myThread.runnable = false;
    webSource.release();
    dispose();
}
public void startCamera() {
    webSource=new VideoCapture(0);
    myThread=new AttendanceCheck.DaemonThread();
    Thread t=new Thread(myThread);
    t.setDaemon(true);
    myThread.runnable=true;
    t.start();
}
```

Chapter Seven

Testing

7.1 Introduction:

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs (errors or other defects). Software testing can be stated as the process of validating and verifying that a computer program/application/product:

- Meets the requirements that guided its design and development,
- Works as expected,
- Can be implemented with the same characteristics,

In a more traditional model, most of the test execution occurs after the requirements have been defined and the coding process has been completed. There are several testing are viewed in system development life cycle, they are

- White-Box testing
- Black-box testing
- Unit testing
- Integration testing
- System testing
- Acceptance testing
- Alpha testing
- Beta testing

7.1.1 White-Box testing: White-box testing (also known as clear box testing, glass box testing, transparent box testing and structural testing) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).

While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit,paths between units during integration, and between subsystems during a system–level test. Though

this method of test design can uncover many errors or problems, it might not detect unimplemented parts of the specification or missing requirements.

7.1.2 Black-box testing: Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation. The tester is only aware of what the software is supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, state transition tables, decision table testing, fuzz testing, model-based testing, use case testing, exploratory testing and specification-based testing.

7.1.3 Unit Testing: Unit testing, also known as component testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors. These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to assure that the building blocks the software uses work independently of each other.

Unit testing is a software development process that involves synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs. It is performed by the software developer or engineer during the construction phase of the software development lifecycle. Rather than replace traditional QA focuses, it augments it. Unit testing aims to eliminate construction errors before code is promoted to QA; this strategy is intended to increase the quality of the resulting software as well as the efficiency of the overall development and QA process.

7.1.4 Integration Testing: Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be localized more quickly and fixed. Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

7.1.5 System Testing: The process of performing a variety of tests on a system to explore functionality or to identify problems. System testing is usually required before and after a system is put in place. A series of systematic procedures are referred to while testing is being

performed. These procedures tell the tester how the system should perform and where common mistakes may be found. Testers usually try to "break the system" by entering data that may cause the system to malfunction or return incorrect information. For example, a tester may put in a city in a search engine designed to only accept states, to see how the system will respond to the incorrect input.

7.1.6 Acceptance Testing: Acceptance tests are created from user stories. During iteration the user stories selected during the iteration planning meeting will be translated into acceptance tests. The customer specifies scenarios to test when a user story has been correctly implemented. A story can have one or many acceptance tests, whatever it takes to ensure the functionality works.

Acceptance tests are black box system tests. Each acceptance test represents some expected result from the system. Customers are responsible for verifying the correctness of the acceptance tests and reviewing test scores to decide which failed tests are of highest priority. Acceptance tests are also used as regression tests prior to a production release.

7.1.7 Alpha testing: Alpha testing is simulated or actual operational testing by potential users/customers or an independent test team at the developers' site. Alpha testing is often employed for off-the-shelf software as a form of internal acceptance testing, before the software goes to beta testing.

7.1.8 Beta Testing: Beta testing comes after alpha testing and can be considered a form of external user acceptance testing. Versions of the software, known as beta versions, are released to a limited audience outside of the programming team. The software is released to groups of people so that further testing can ensure the product has few faults or bugs. Sometimes, beta versions are made available to the open public to increase the feedback field to a maximal number of future users.

7.2 Login Testing

Login is successfully held by the correct username and password. System detects user type from given username.We have used static username and password for login . By providing proper username and password users can access the application.

After login: After login admin will be able to use the application. There is a menu page where the application page navigation starts.

7.3 Recognition testing

After successful login admin will be redirected to the menu page . Where is the navigation menu from where admin can navigate to another page. From the menu admin can register employees or students. After registering the system will work automatically.

Chapter Eight

Implementation

7.1 Introduction

Implementation is the carrying out, execution, or practice of a plan, a method, or any design for doing something. As such, implementation is the action that must follow any preliminary thinking in order for something to actually happen. In an information technology context, implementation encompasses all the processes involved in getting new software or hardware operating properly in its environment, including installation, configuration, running, testing, and making necessary changes.

In computer science, an implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through computer programming and deployment.

7.2 Key Components

7.2.1 Defined roles and responsibilities: I think the importance of defining roles and responsibilities in an implementation methodology goes without saying, but I also feel like you can never speak to it enough. It is something that we all know we need to do, but whenever something goes wrong, unclear roles and responsibilities tend to be a culprit (e.g. “I didn’t know I was supposed to do that. I thought he was going to do it.”) Defining roles and responsibilities is one of, if not the most, crucial elements of any aspect of a business, but is even more so in a software implementation. Roles and responsibilities must be clearly defined as it allows people who work together to understand their individual component of the implementation, what others are doing and how it all fits together.

7.2.2 Flexibility: I believe an implementation methodology should be a set of high-level activities and procedures to allow for flexibility at the task level. I don’t believe a methodology should have detailed tasks that are required to be followed. Every client and implementation varies and if a methodology has detailed tasks, it can create a situation where you end up trying to fit a square peg into a round hole; eventually wasting time doing tasks that do not benefit the client or your company. By having a methodology that includes high-level activities and procedures, it leaves room for flexibility at the task level and allows your employees to

tailor pieces of the implementation to fit the needs of your clients and the implementation. There are definitely people who argue this point with me. Many believe that detailed tasks should be included in a methodology to ensure that employees do the right thing. My argument is that you need to trust that your employees can and will do the right thing. If you don't feel like you can't trust your employees, then maybe they are in the wrong role. Ultimately having a methodology that allows room for flexibility drives better behavior and paves the way for more successful implementations.

7.2.3 Customer Communication: It is crucial to build customer communication into a software implementation methodology. One of the primary reasons why software implementations fail is because there was a lack of communication which led to missed expectations, requirements, etc. Building in specific customer interaction and touch points at critical times throughout the implementation can significantly increase the success of the implementation.

7.2.4 Change Management: The business world is a place of constant change. What may have worked well for a company six months ago may no longer work well. The same goes for an implementation methodology. Things change, we find better ways to implement software, and we get feedback from employees that drive change. It is essential that a company have a good change management structure in place for its implementation methodology. A good change management structure will help drive the success of the change and ensure that the change sticks with employees. I could write an entire article on key components to a change management methodology, but I think the more important aspect for an implementation methodology is to ensure that you have some change management structure – stakeholders identified, communication plan, training plan, etc

7.2.5 Algorithmic complexity:

The academic field and the engineering practice of computer programming are both largely concerned with discovering and implementing the most efficient algorithms for a given class of problem. For this purpose, algorithms are classified into orders using so-called Big O notation, O(n), which expresses resource use, such as execution time or memory consumption, in terms of the size of an input. Expert programmers are familiar with a variety of well-established algorithms and their respective complexities and use this knowledge to choose algorithms that are best suited to the circumstances.

7.2.6 Methodologies:

The first step in most formal software development projects is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). There exist a lot of differing approaches for each of those tasks. One approach popular for requirements analysis is Use Case analysis. Nowadays many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years. There are many approaches to the Software development process

Popular modeling techniques include Object-Oriented Analysis and Design and Model-Driven Architecture. The Unified Modeling Language is a notation used for both the OOAD and MDA.

A similar technique used for database design is Entity-Relationship Modeling.

Implementation techniques include imperative languages, functional languages, and logic languages.

7.2.7 Programming languages:

Different programming languages support different styles of programming. The choice of language used is subject to many considerations, such as company policy, suitability to task, availability of third-party packages, or individual preference. Ideally, the programming language best suited for the task at hand will be selected. Trade-offs from this ideal involve finding enough programmers who know the language to build a team, the availability of compilers for that language, and the efficiency with which programs written in a given language execute. Languages form an approximate spectrum from "low-level" to "high-level"; "low-level" languages are typically more machine-oriented and faster to execute, whereas "high-level" languages are more abstract and easier to use but execute less quickly.

Allen Downey, in his book How to think like a Computer Scientist, writes:

The details look different in different languages, but a few basic instructions appear in just about every language:

- input: Get data from the keyboard, a file, or some other device.
- output: Display data on the screen or send data to a file or other device.
- arithmetic: Perform basic arithmetical operations like addition and multiplication.
- conditional execution: Check for certain conditions and execute the appropriate sequence of statements.
- repetition: Perform some action repeatedly, usually with some variation.

Many computer languages provide a mechanism to call functions provided by libraries. Provided the functions in a library follow the appropriate run time conventions.

Full life cycle and components are applied into coding and testing section which is one of the major parts for implementation.

Future Works

Technology is changing. Today, more than ever before, technology plays an important role in society. It is changing and will continue to change every aspect of how we live. It is changing the way we communicate, the way we do business, how we learn and teach, and even it's changing the way our brains work. Because of the arrival of technology, the learning environment is changing. In the near future we would like to improve the system to make it a more efficient system. Our future plans are stated below -

1. Currently our system can only work with 2d faces ,in future we will like to add a feature that can work with 3d faces.

2. Currently there is no sensor for advanced work ,we would like to work with the libre-alsense library .

3. The system can't detect faces from a long distance .In future we will work to fill this gap.

Conclusion

Our system will give attendance to the staff who will be present in time and it will store the record in the database. The system will detect faces and will count attendance for a scheduled time. After that it will stop it's work automatically. After the schedule time database will be updated and the attendance sheet will be printed out . These will ensure that no attendance will be given if he or she fails to come within time .This makes to maintain a rule which will be followed by the authority.

Reference

[1] https://opencv.org/ --OpenCV is a library of programming functions mainly aimed at real-time computer vision.

[2] https://github.com/bytedeco/javacv --**JavaCV** is a collection of wrappers for many famous libraries used for image processing and computer vision-related applications.

[3] https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework --The **Viola–Jones object detection framework** is the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones.

[4] http://www.technovelgy.com/ct/Technology-Article.asp?ArtNum=16 --Biometric authentication: what method works best?

[5] http://www.andrewsenior.com/ papers/SeniorB02FaceChap.pdf. --Face recognition and it's application.

[6] https://www.quora.com/What-is-a-grayscale-image –Grayscale image

[7] https://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html –why we convert grayscale