Chapter 1

INTRODUCTION

We are introducing here the project overview, motivation of the project, project outline of the topic. Also providing the problems statement

1.1PROJECT OVERVIEW:

Humans have been using physical characteristics such as face, voice, gait, etc. to recognize each other for thousands of years. With new advances in technology, biometrics has become an emerging technology for recognizing individuals using their biological traits. Now, biometrics is becoming part of day to day life, where in a person is recognized by his/her personal biological characteristics. Examples of different Biometric systems include Fingerprint recognition, Face recognition, Iris recognition, Retina recognition, Hand geometry, recognition, Signature recognition, among others. Face recognition, in particular has received a considerable attention in recent years both from the industry and the research community. The objective of our project is to create a C# code that can be used to identify people using their face images.

Using the face recognition technique we use in the attendance management system. With the help of the face recognition we make the attendance of the student.

1.2 MOTIVATION OF THE PROJECT

Attendance taken by manually:-

a. This is old version method for the attendance system where the teacher/staff/representative take the attendance by call the person name and tick in the attendance register. If it's present then tick as present or absent.

Computerized attendance system:-

b. This is same method that proposed in above session. Instead of tick in the attendance register the teacher/staff/representative click in front of the student name in the computer record.

Drawback of the existing system:-

1. Since this process is manually human error can occurs.
2. The existing system is time consuming.
3. Less security.

1.3 PROBLEM DEFINITION

Every time a lecture, section starts the lecturer or teaching .This is a lengthy process and takes a lot of time and effort, especially if it is a lecture with a huge number of students. It also causes a lot of disturbance and interruption when an exam is held. Moreover the attendance sheet is subjected to damage and loss while being passed on between different students or teaching staff. And when the number of students enrolled in a certain course is huge, the doctors tend to call a couple of student names at random which is not a fair student evaluation process either. Finally, these attendance records are used by the staff to monitor the students' 3 attendance rates. This process could be easy and effective with a small number of students but on the other hand, dealing with the records of a large number of students often leads to human errors.

1.4 OUTLINE OF THE REPORT

Chapter 1 provides as introduction to Partial face recognition using core feature of face. It also proposes an enhanced method for Data Processing.

Chapter 2 provides an overview of previously used implementation and systems for Partial face recognition using core feature of face which gives low performance. As well as comparison with existing system is elaborated.

Chapter 3 specifies various requirement analysis aspects such as feasibility, functional, non-functional and system analysis regarding our system.

Chapter 4 provides of the overall system architectural design, various levels of Data Flow Diagrams (DFD).

Chapter 5 describes the algorithms used to build the system and presents few screenshots of the system while functioning.

Chapter 6 summarizes detailed results of the tests performed in Testing phase.

Chapter 7 provides overview of the planned time lines to develop and test the system. And gives details about the task distributed among group while achieving the goals specified in time line.

Chapter 8 summarizes the system and proposes the future work, modifications to improve and enhance our system.

Chapter 2

LITERATURE REVIEW

Biometrics is the automated recognition of individuals based on their behavioural or physiological characteristics .The physiological characteristics are related to the shape of the body. The most common example is fingerprint. Other examples include face recognition, hand geometry and iris recognition. The behavioural characteristics are related to the behaviour of a person. Signature is one example of these characteristics which is still widely used today. Modern approaches are the study of keystroke dynamics and voice.

With the rapid development in the field of pattern recognition and its uses in different areas e.g. (signature recognition, facial recognition), arises the importance of the utilization of this technology in different areas in large organizations. This is mainly because these applications help the top-management take decisions that improve the performance and effectiveness of the organization. On the other hand, for an organization to be effective, it needs accurate and fast means of recording the performance of the people inside this organization. Biometric recognition has the potential to become an irreplaceable part of many identification systems used for evaluating the performance of those people working within the organization. Although biometric technologies are being applied in many fields it has not yet delivered its promise of guaranteeing automatic human recognition. This research is the first of its kind to attempt to provide an automated attendance system that recognizes students using face recognition technology through an image/video stream to record their attendance in lectures or sections and evaluating their performance accordingly.

Face recognition is a biometric which uses computer software to determine the identity of the individual. Face recognition falls into the category of biometrics which is “the automatic recognition of a person using distinguishing traits” Other types of biometrics include fingerprinting, retina scans, and iris scan.

face recognition techniques:-The available face recognition techniques can be classified into four categories based on the way they represent face.

- Appearance based which uses holistic texture features.
- Model based which employ shape and texture of the face, along with 3D depth information.
- Template based face recognition.
- Techniques using Neural Networks.

The Proposed system overcomes the problem of the existing system. This project uses the face recognition using the method of Neural Network Technique.

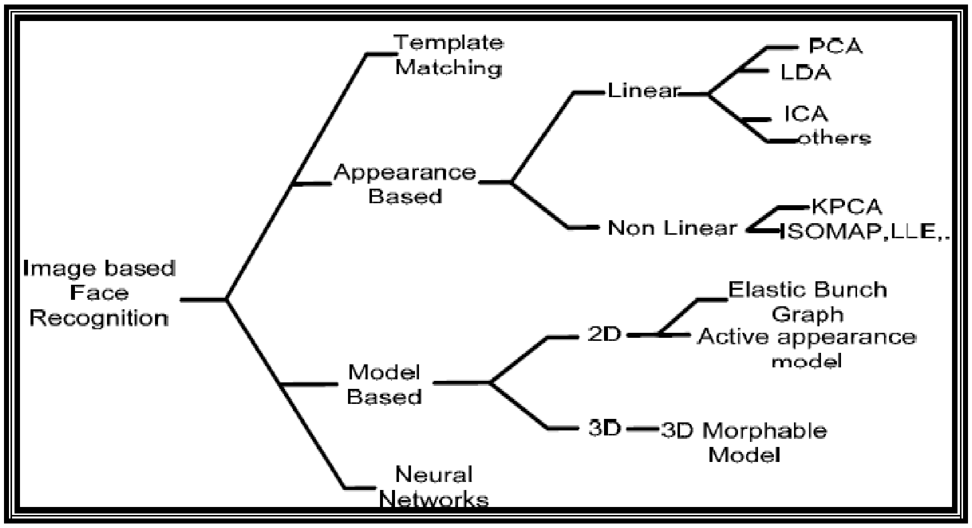


Fig2.1: face recognition techniques

Templet Matching:- Template matching is a technique in digital image processing for finding small parts of an image which match a template image. It can be used in manufacturing as a part of quality control a way to navigate a mobile robot or as a way to detect edges in images. templates without strong features, or for when the bulk of the template image constitutes the matching image, a template-based approach may be effective. As aforementioned, since template-based template matching may potentially require sampling of a large number of points.

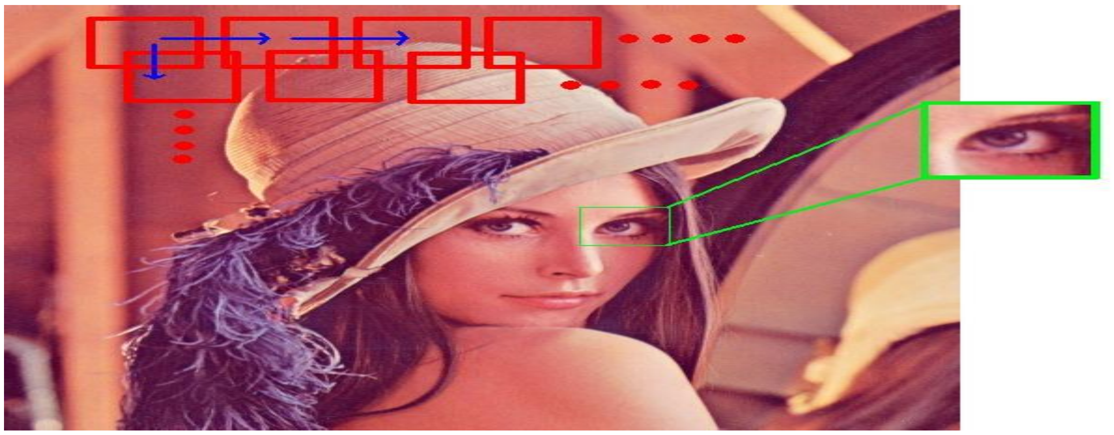


Fig2.2: Templet Matching

Appearance Based:- Appearance based methods that identify places on the basis of sensory similarity are a promising possible solution - this is presumably how humans tackle the problem. However, identifying places purely on the basis of sensory similarity is too simplistic - different places may look very similar, even to a rich sensor such as a camera (e.g. the images below). Nonetheless, people looking at these pictures are unlikely to mistake them as coming from the same place because they understand that the views in these images are not distinctive enough to make a decision

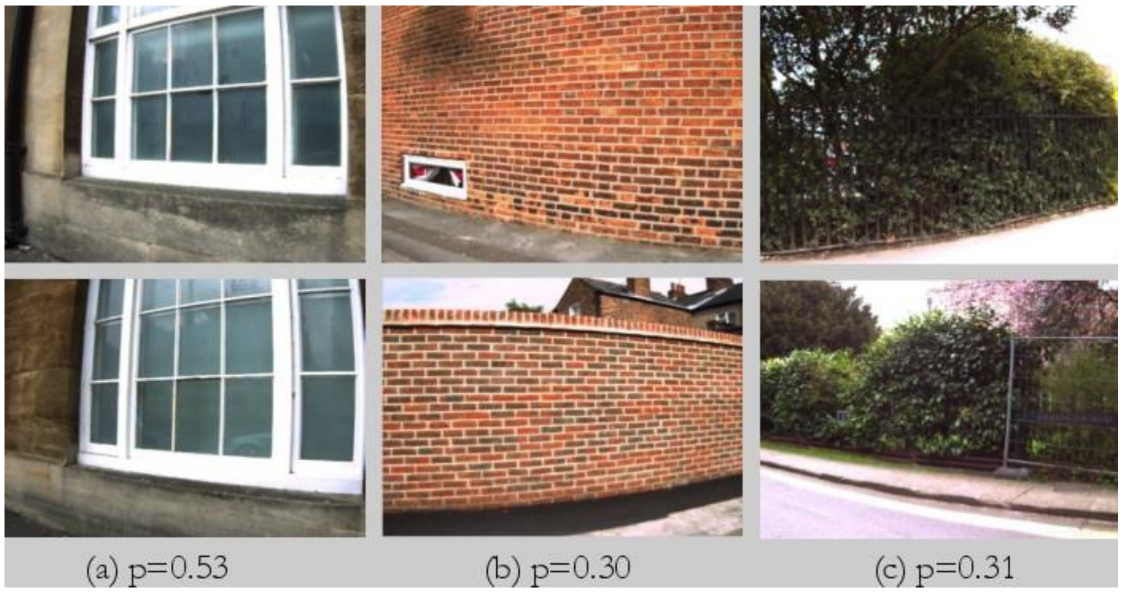


Fig2.3: Appearance Based

PRINCIPLE COMPONENT ANALYSIS (PCA):- PCA is a way of identifying patterns in data and expressing the data in such a way to highlight their similarities and differences. The purpose of PCA is to reduce the large dimensionality of the data space (observed variables) to smaller intrinsic dimensionality of feature space (independent variables) which are needed to describe the data economically.

Model-Based Design :- (MBD) is a mathematical and visual method of addressing problems associated with designing complex control signal processing and communication systems. It is used in many motion control, industrial equipment, aerospace, and automotive applications. Model-based design is a methodology applied in designing embedded software.

Model-based design provides an efficient approach for establishing a common framework for communication throughout the design process while supporting the development cycle ("V" diagram). In model-based design of control systems, development is manifested in these four steps:

1. modeling a plant
2. analyzing and synthesizing a controller for the plant
3. simulating the plant and controller

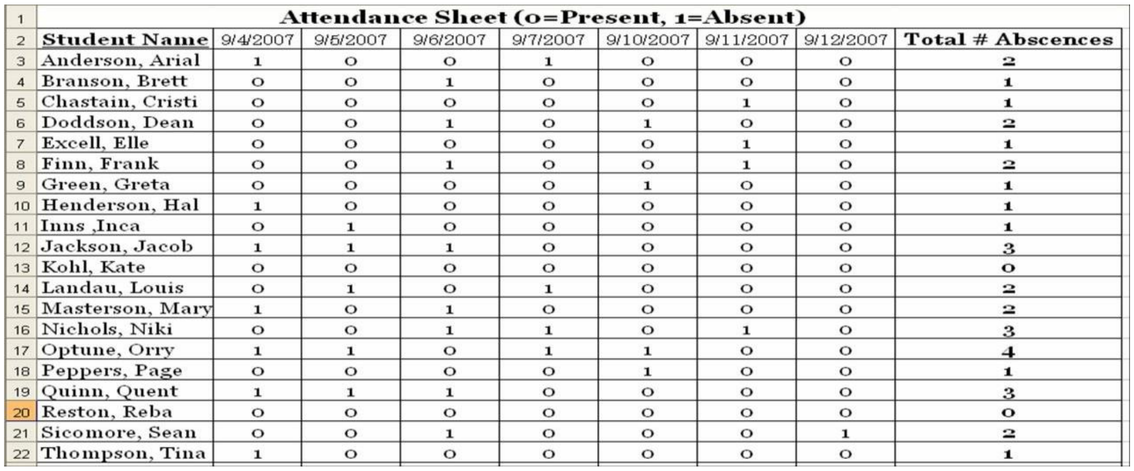
4. integrating all these phases by deploying the controller

Artificial Neural Network:-An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons.

2.1. Existing System

Following are the Existing system for the attendance system:-

2.1.1. Attendance take by manually :- This is old version method for the attendance system where the teacher/staff/representative take the attendance by call the person name and tick in the attendance register. If it's present then tick as present or absent.



1 Attendance Sheet (0=Present, 1=Absent)

2	Student Name	9/4/2007	9/5/2007	9/6/2007	9/7/2007	9/10/2007	9/11/2007	9/12/2007	Total # Abscences
3	Anderson, Arial	1	0	0	1	0	0	0	2
4	Branson, Brett	0	0	1	0	0	0	0	1
5	Chastain, Cristi	0	0	0	0	0	1	0	1
6	Doddson, Dean	0	0	1	0	1	0	0	2
7	Excell, Elle	0	0	0	0	0	1	0	1
8	Finn, Frank	0	0	1	0	0	1	0	2
9	Green, Greta	0	0	0	0	1	0	0	1
10	Henderson, Hal	1	0	0	0	0	0	0	1
11	Inns ,Inca	0	1	0	0	0	0	0	1
12	Jackson, Jacob	1	1	1	0	0	0	0	3
13	Kohl, Kate	0	0	0	0	0	0	0	0
14	Landau, Louis	0	1	0	1	0	0	0	2
15	Masterson, Mary	1	0	1	0	0	0	0	2
16	Nichols, Niki	0	0	1	1	0	1	0	3
17	Optune, Orry	1	1	0	1	1	0	0	4
18	Peppers, Page	0	0	0	0	1	0	0	1
19	Quinn, Quent	1	1	1	0	0	0	0	3
20	Reston, Reba	0	0	0	0	0	0	0	0
21	Sicomore, Sean	0	0	1	0	0	0	1	2
22	Thompson, Tina	1	0	0	0	0	0	0	1

Fig2.4: Attendance sheet

2.1.2. Finger print attendance system:- This is a system in which a persons finger print is detected by a laser and the attendance is marked.

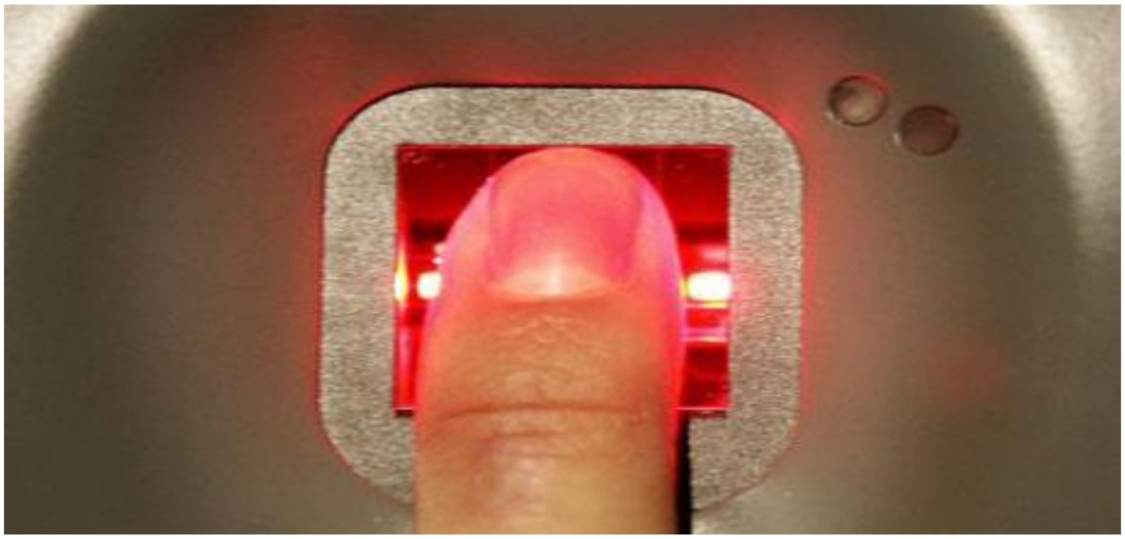


Fig2.5: Finger print

2.1.3 Retina scan:- Retina of a eye is scaned individualy by a scanner and attendance is marked.

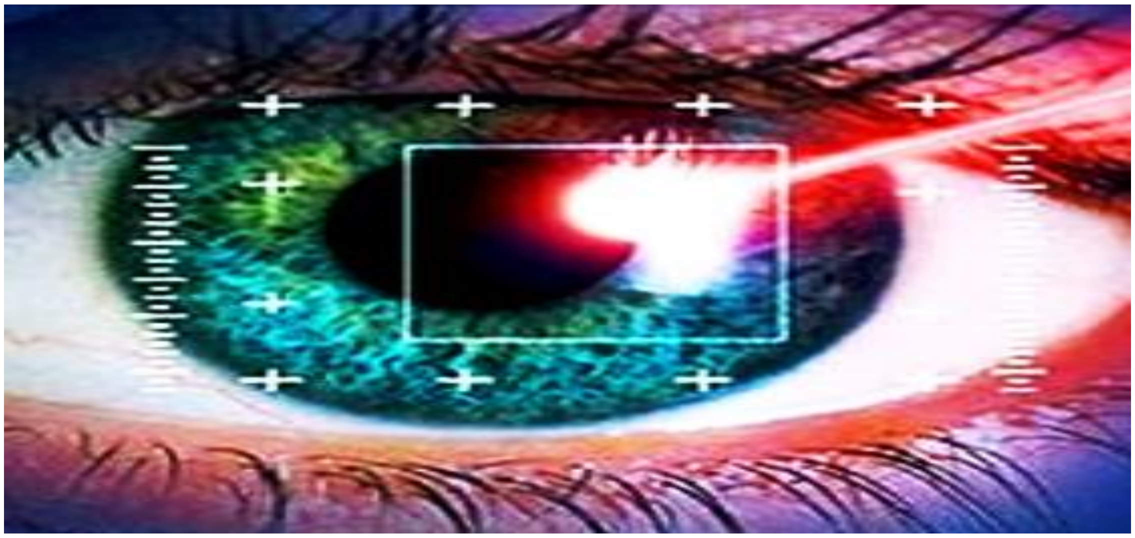


Fig2.6: Retina scan

2.1.4 ID card scanning system:- The magnetic strip on the id card is scanned by a computer and information is collected and at the same time attendance is marked.

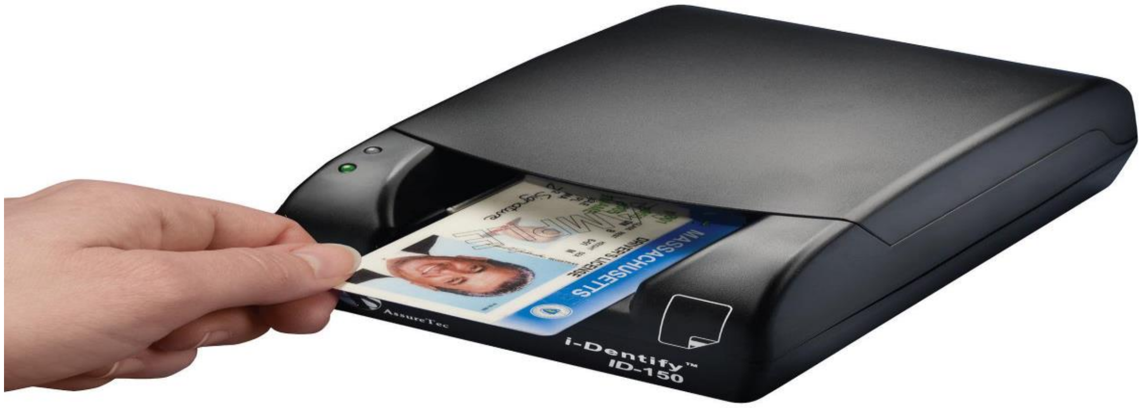


Fig2.7: ID card scanning system

Drawback of the existing system:-

1. Since this process is manually human error can occurs.
2. The existing system is time consuming.
3. Less security.

2.2 Proposed System

In the proposed system when student come to the class or lecture system application is start. It works only is standing in front of the system (Computer application) the application capture the image and send the processing side. The processing side the applicatio recognize the face of the student.

Finally the application mark as student present. If the face is not recognizing the application make as absent.

2.3 Face Recognition Process:-

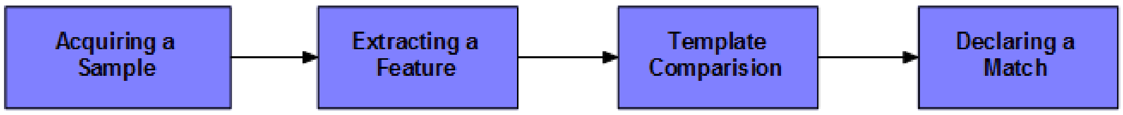


Fig2.8: Face Recognition Process

2.3.1 Acquiring a sample: In a complete, full implemented biometric system, a sensor takes an observation. The sensor might be a camera and the observation is a snapshot picture. In our system, a sensor will be ignored, and a 2D face picture "observation" will supplied manual

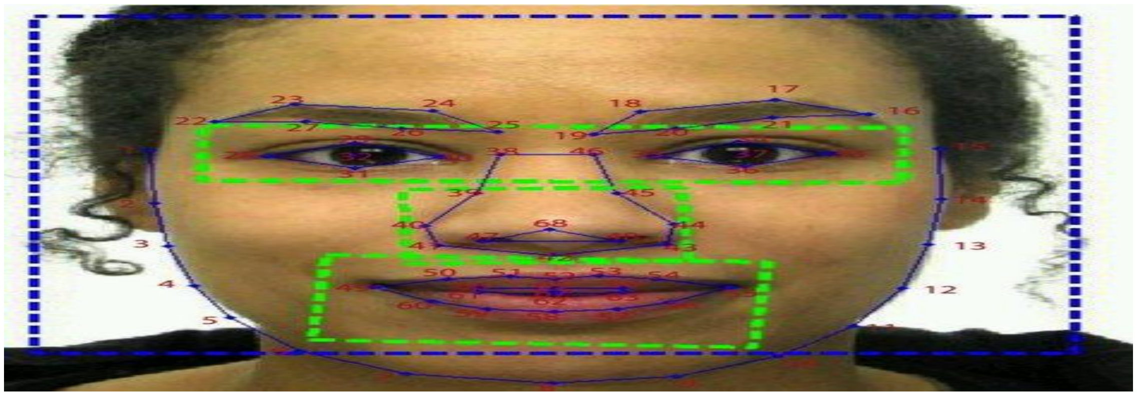


Fig2.9: Acquiring a sample

2.3.2 Extracting Features: For this step, the relevant data is extracted from the predefined captured sample. This is can be done by the use of software where many algorithms are available. The outcome of this step is a biometric template which is a reduced set of data that represents the unique features of the enrolled user's face.

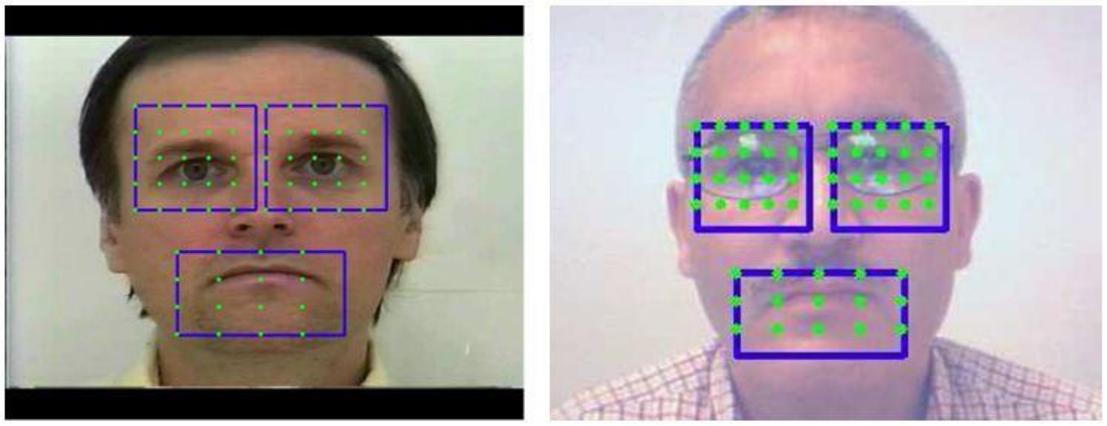


Fig2.10: Extracting Features

2.3.3 Comparison Templates: This depends on the application at hand. For identification purposes, this step will be a comparison between a given picture for the subject and all the biometric templates stored on a database. For verification, the biometric template of the claimed identity will be retrieved (either from a database or a storage medium presented by the subject) and this will be compared to a given picture.

2.3.4 Declaring a Match: The face recognition system will return a candidate match list of potential matches. In this case, the intervention of a human operator will be required in order to select the best fit from the candidate list. An illustrative analogy is that of a walk-through metal detector, where if a person causes the 7 detector to beep, a human operator steps in and checks the person manually or with a hand-held detector.

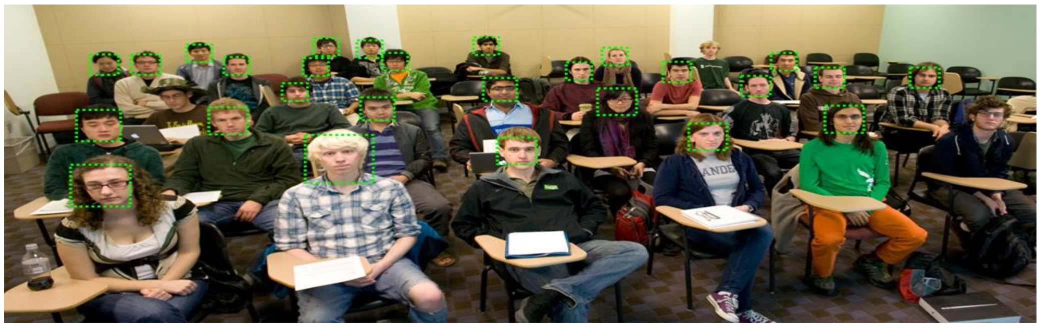


Fig2.11: Declaring a Match

- This code is supposed to grab live camera feed, display feed in a window, mark in rectangles all detected faces, get the biggest detected face (by total area), display it in separate window, convert it to grayscale and finally save as PNG to hard disk, in project directory

- It is a fast, accurate and reliable than any other existing method.
- Face recognition is easy to use and in many cases it can be performed without person even knowing.
- Face recognition is also one of the most inexpensive biometric in the market and its price should continue to go down.
- There are many benefits to face recognition system such as its convinence and social acceptability
- Security counterterrorism: Access control comparing surveillance images to know terrorist.
- Immigiration rapid progression through customs
- banking using atm the software is able to quickly verify a customer face
- Physical access control of building areas, doors or net acces

Image processing is the process of manipulating image data in order to make it suitable for computer vision applications or to make it suitable to present it to humans. For example, changing brightness or contrast is a image processing task which make the image visually pleasing for humans or suitable for further processing for a certain computer vision application.

Computer vision which go beyond image processing, helps to obtain relevant information from images and make decisions based on that information. In other words, computer vision is making the computer see as humans do. Basic steps for a typical computer vision application as follows.

1. Image acquisition
2. Image manipulation
3. Obtaining relevant information
4. Decision making

If you are new to computer vision, you may be wondering where to start. First you have to understand the basic principles of image processing and computer vision. Then you have to choose a suitable language to develop your computer vision application. Some of the most popular methods are using OpenCV with C/C++. If you don't really know why you would choose one over the other, here is my explanation.

Chapter 3

PROJECT ANALYSIS

3.1 Feasibility Study

The very first phase in any system developing life cycle is preliminary investigation. The feasibility study is a major part of this phase. A measure of how beneficial or practical the development of any information system would be to the organization is the feasibility study.

The feasibility of the development software can be studied in terms of the following aspects:

1. Operational Feasibility.
2. Technical Feasibility.
3. Economical feasibility.
4. Schedule Feasibility

3.1.1 Technical Feasibility

- At least 166 MHz Pentium Processor or Intel compatible processor.

- At least 16 MB RAM.
- 14.4 kbps or higher modem.
- A video graphics card.
- A mouse or other pointing device.
- At least 3 MB free hard disk space.
- Microsoft Internet Explorer 4.0 or higher.

3.1.2 Operational Feasibility

- The site will reduce the time consumed to maintain manual records and is not tiresome and cumbersome to maintain the records. Hence operational feasibility is assured.

3.1.3 Economical feasibility

Once the hardware and software requirements get fulfilled, there is no need for the user of our system to spend for any additional overhead.
For the user, the web site will be economically feasible in the following aspects:

- ➢ The web site will reduce a lot of paper work. Hence the cost will be reduced.
- ➢ Our web site will reduce the time that is wasted in manual processes.
- ➢ The storage and handling problems of the registers will be solved.

3.1.4 Schedule Feasibility

- ➢ Schedule Flexibility is a measure of how reasonable the project timetable is. Schedule feasibility ensures that our project can be completed before the project or technology used becomes obsolete or unnecessary. Schedule feasibility is shown using a timeline chart.

3.2 Requirement Analysis

It is an early stage in the more general activity of requirements engineering which encompasses all activities concerned with eliciting, analyzing, documenting, validating and managing software or system requirements.

3.2.1 Functional Requirement Analysis

A functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. From the functional perspective the project is Measurable, Realistic and Complete. The project covers all the functional requirements needed for a private cloud infrastructure including performance, usability and security. Eucalyptus service component exposes a well-defined language agnostic API in the form of a WSDL document containing both the operations that the service can perform and the input/output data structures. Inter-service authentication is handled via standard WS-Security & SSL security mechanisms.

3.3.2 Non-Functional Requirement

Efficiency: The Private Cloud using Eucalyptus and Xen is very user friendly. The private cloud can be accessed by using GUI provided with it or with Amazon AWS compatible Euca2ools. Any person who worked with Amazon web service will not face any difficulties while accessing this private cloud.

Adaptability: Adaptation of software systems is almost an inevitable process, due to the change in customer requirements, needs for faster development of new, or maintenance of existing, software systems, etc. Our software architecture is flexible to adapt new changes such as in our case it has very good scope in future research to consider all additional factors.

Complexity: Use of Eucalyptus and Xen reduces the complexity by great extent by providing component based architecture. Each Eucalyptus component comes as a separate module and can be modified independently. Apart from that Eucalyptus provides an API which allows it to extend easily.

Supportability: Eucalyptus offers Amazon AWS compatible interfaces written in Java that can be used with Amazon cloud service to create a hybrid cloud. Eucalyptus can run multiple versions of Linux virtual machine images. Users can build a library of Eucalyptus Machine Images (EMIs) with application metadata that are decoupled from infrastructure details to allow them to run on Eucalyptus clouds.

3.3 Hardware and Software Requirements

The hardware requirement of the project can be defined as the minimum hardware required to deploy and run the project and to develop the software normally. The system requires the following hardware:

Hardware Type	**Purpose**
1. 1.66 GHz Pentium Processor or Intel compatible processor.	To provide computational Environment.
2. 2GB RAM.	For faster reads and writes.
3. 80 GB free hard disk space.	For storage of all the data.
Web Camera	To identify the face.

Table 3.1: Hardware requirement table

The software requirement for the project can be defined as the tools required for the development of software and the software needed to deploy the software.

Software	**Purpose**
1. Visual Studio 2008(.Net framework)	Programming language
2.Asp.net	Programming language
3.Microsoft access 2007	For database connectivity

Table 3.2: Software requirement table

CHAPTER 4

HARDWAER AND SOFTWARE REQUIREMENT

4.1 Hardware:

Processor: Pentium 4

RAM: 512 MB or more

Hard disk: 16 GB or more

Camera

4.2 Software

1. Visual Studio 2008(.Net framework)
2. Asp.net
3. Microsoft access 2007

1. Visual Studio 2008(.Net framework)

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as web sites, web applications and web services. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows

Presentation Foundation, Windows Storeand Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level including adding support for source-control systems (like Subversion) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

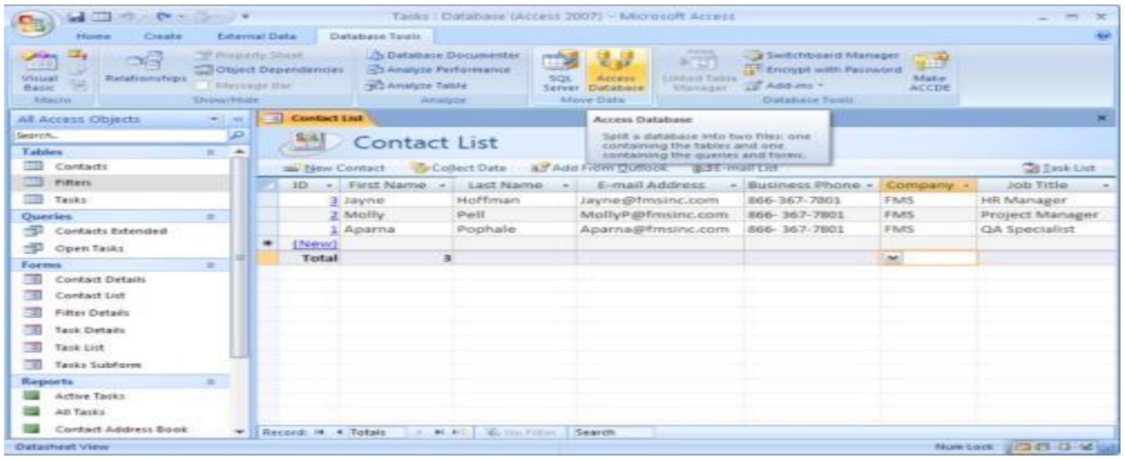
Visual Studio supports different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++and C++/CLI (via Visual C++), VB.NET (via Visual Basic .NET), C# (via Visual C#), and F# (as of Visual Studio 2010). Support for other languages such as M, Python, and Ruby among others is available via language services installed separately It also supports XML/XSLT, HTML/XHTML, JavaScript and CSS. Java (and J#) were supported in the past.

Microsoft provides "Express" editions of its Visual Studio at no cost. Commercial versions of Visual Studio along with select past versions are available for free to students via Microsoft's Dream Spark program.

2. Microsoft access 2007

Microsoft Access, also known as Microsoft Office Access, is a database management system from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software-development tools. It is a member of the Microsoft Office suite of applications, included in the Professional and higher editions or sold separately.

Microsoft Access stores data in its own format based on the Access Jet Database Engine. It can also import or link directly to data stored in other applications and databases.



ID	First Name	Last Name	E-mail Address	Business Phone	Company	Job Title
3	Jayne	Hoffman	Jayne@fmsinc.com	866-367-7801	FMS	HR Manager
2	Molly	Pell	MollyP@fmsinc.com	866- 367-7801	FMS	Project Manager
1	Aparna	Pophale	Aparna@fmsinc.com	866- 367-7801	FMS	QA Specialist
(New)						
Total	3					

Fig 4.1: Database

4.3 Diagrams

4.3.1 Sequence Diagram

A sequence diagram is an interaction diagram that shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

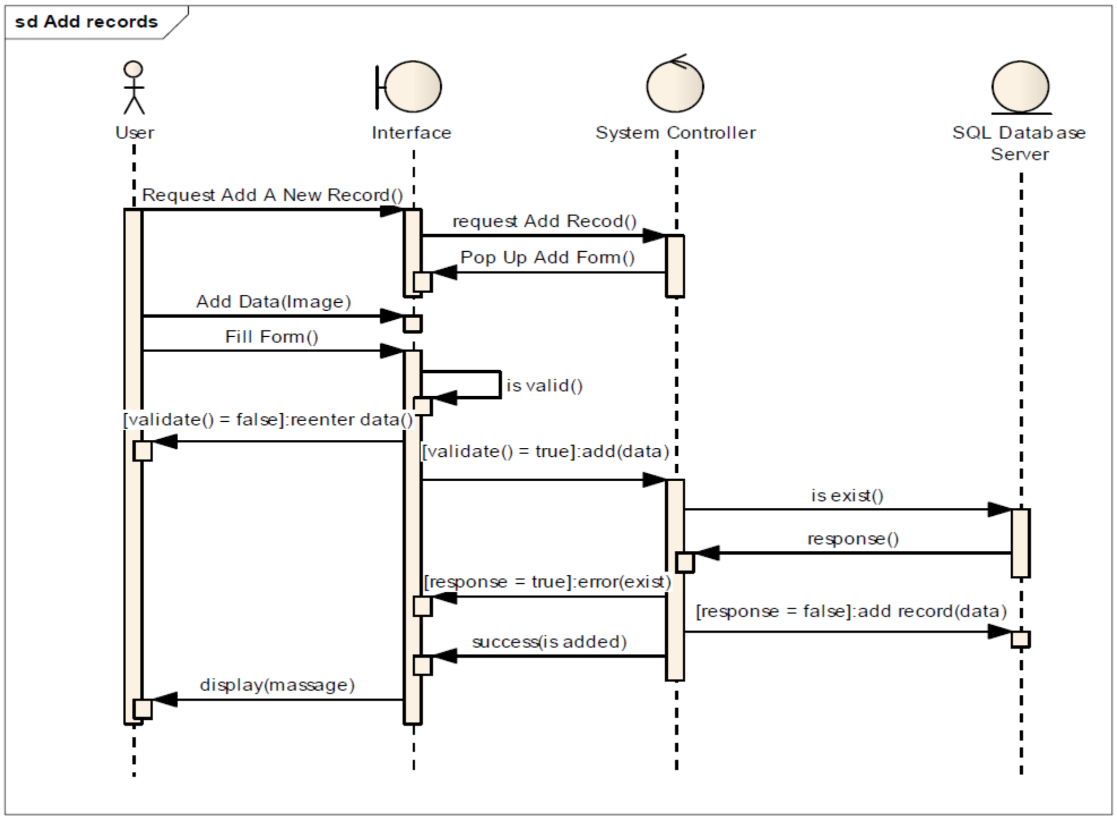


Fig 4.2: Sequence Diagram

4.3.2 Activity Diagram

Activity diagram is used to describe dynamic behavior of the system. Activity diagram is basically a flow chart to represent the message flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams are also used to construct the executable system by using forward and reverse engineering techniques.

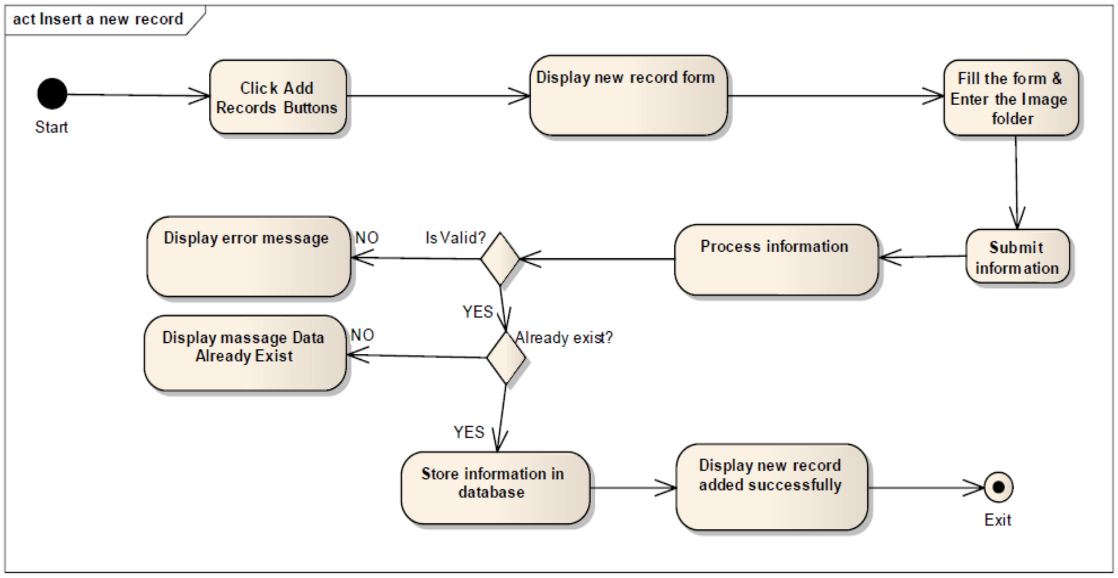

act Insert a new record
Start
Click Add Records Buttons
Display new record form
Fill the form & Enter the Image folder
Submit information
Process information
Is Valid?
NO
Display error message
YES
Already exist?
NO
Display massage Data Already Exist
YES
Store information in database
Display new record added successfully
Exit

Fig4.3: Activity Diagram

4.3.3 Class Diagram

Class diagrams are arguably the most used UML diagram type. It is the main building block of any object oriented solution. It shows the classes in a system, attributes and operations of each class and the relationship between each class. In most modelling tools a class has three parts, name at the top, attributes in the middle and operations or methods at the bottom. In large systems with many related classes, classes are grouped together to create class diagrams. Different relationships between classes are shown by different types of arrows.

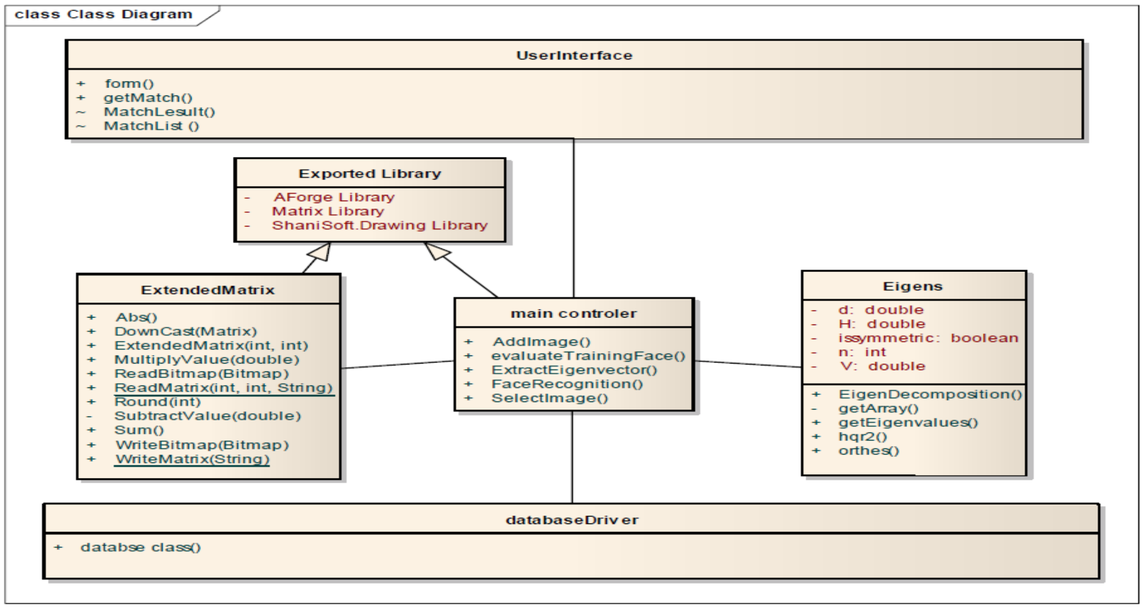


Fig4.4: Class Diagram

4.3.4 Use Case Diagram

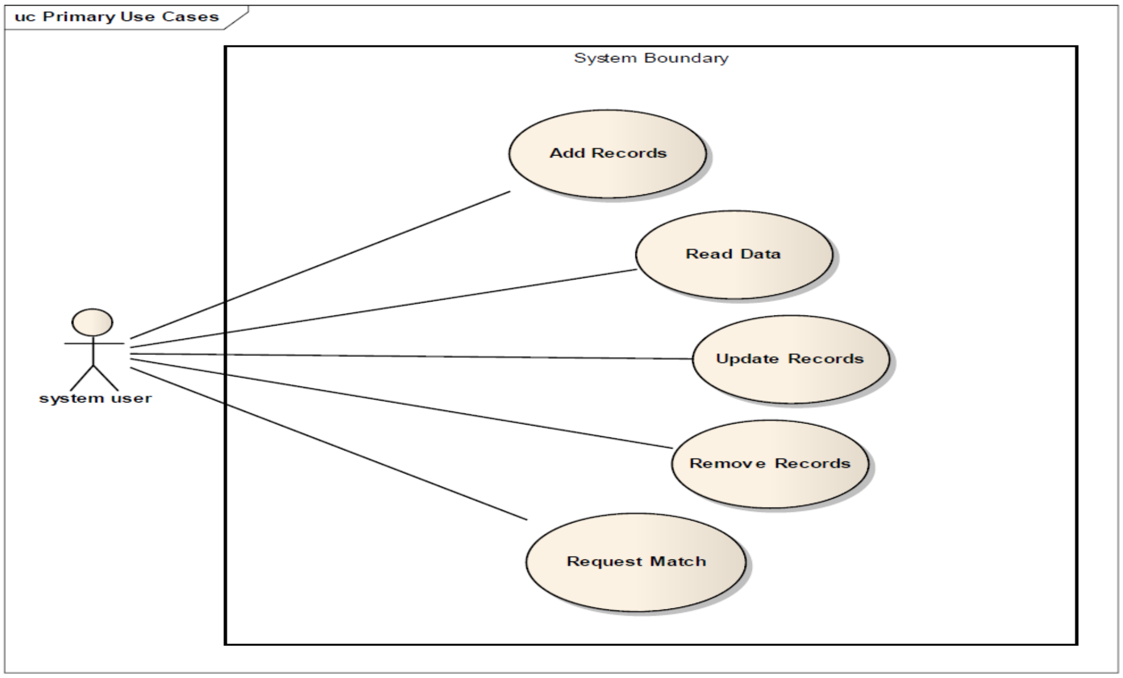


Fig4.5: Use Case Diagram

4.3.5 Deployment Diagram

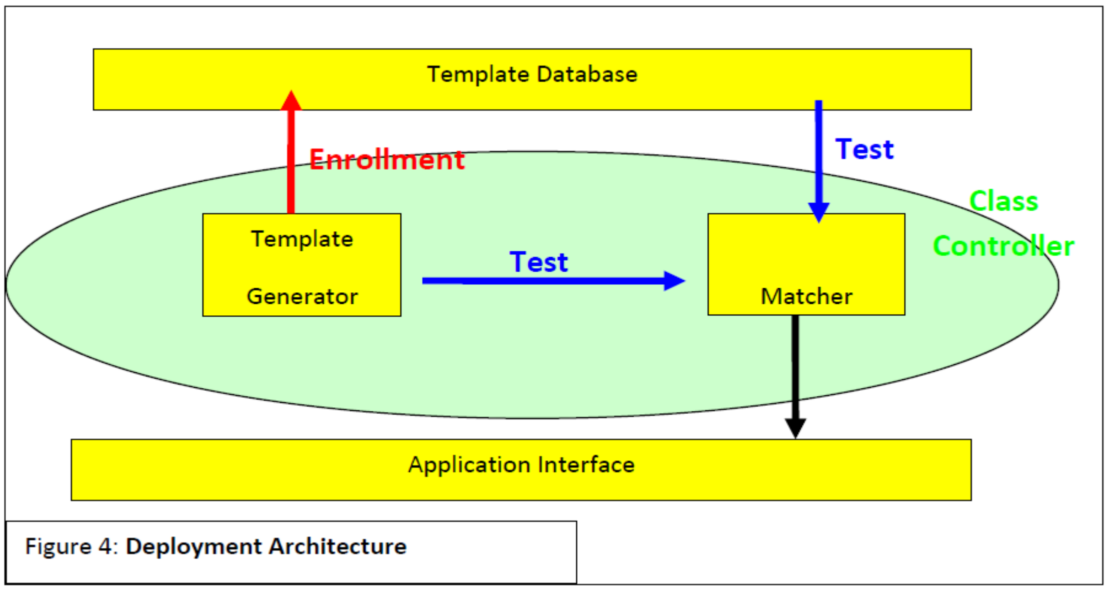


Fig4.6: Deployment Diagram

4.4 DATA FLOW DIAGRAM (DFD):

It is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

It is common practice to draw the context-level data flow diagram first, which shows the interaction between the system and external agents which act as data sources and data sinks. On the context diagram the system's interactions with the outside world are modelled purely in terms of data flows across the system boundary.

4.4.1 0 LEVEL DFD

This is the 0 level DFD for Data Processing. Here We are selecting all the databases and processing them using system to get the highest profit.

0 Level DFD

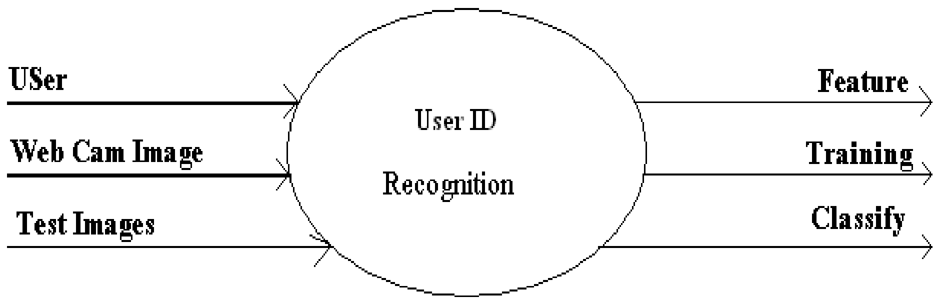
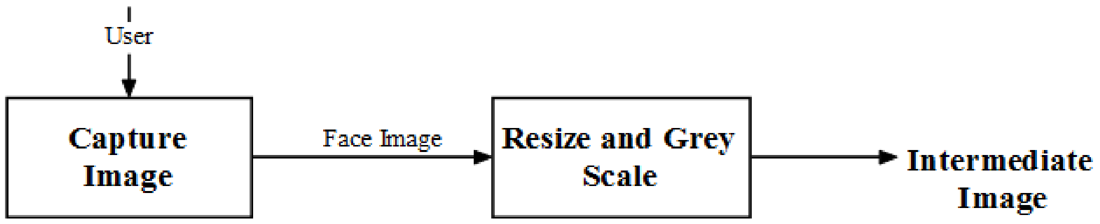
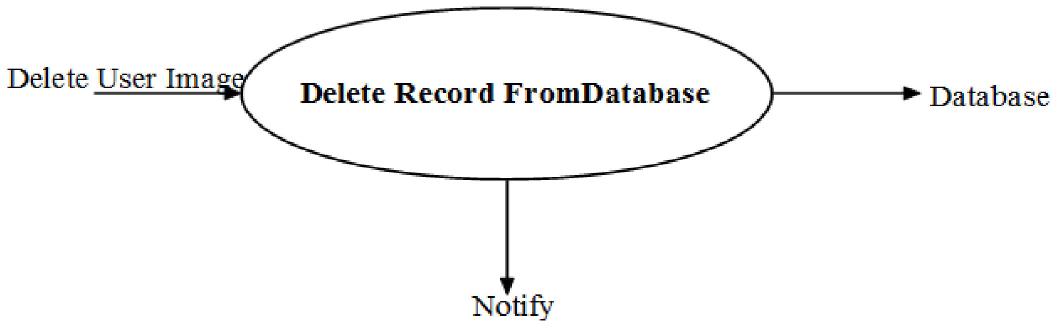


Fig4.7:DFD Level 0

1 Level DFD

This is the level 1DFD in which the different databases such as sales, discount and expenditure details are processed using system and used that result for getting highest profit by applying them to the current year information.





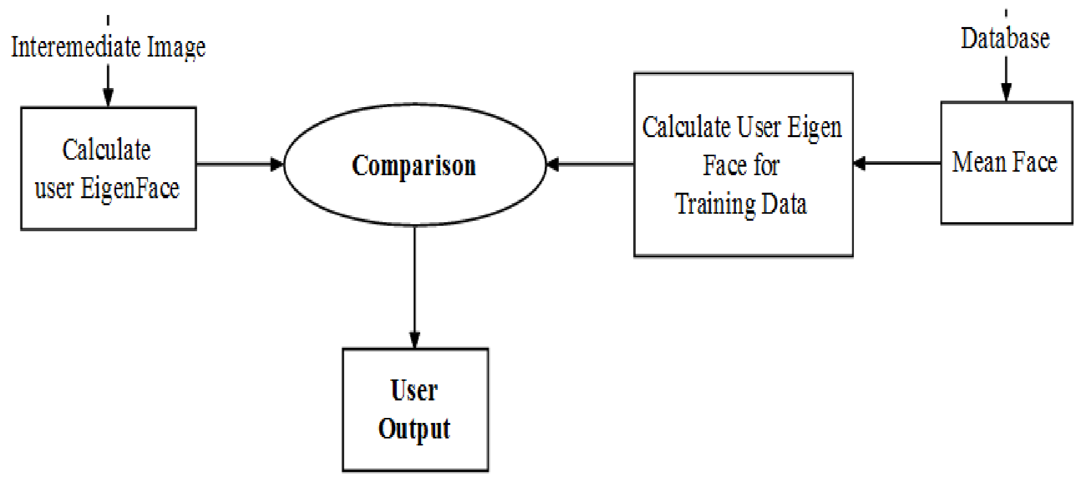


Fig 4.8: DFD Level 1

Level 2 DFD

This level shows a little more detail as compared to level 1. It shows the entire connection which is required for the flow of data from the user to the motors which are responsible for movement of the robot. The input taken from user is sent to the microcontroller. The microcontroller processes this data and makes a decision about the robot movement. This is sent to the motors and causes robot to move towards the destination.

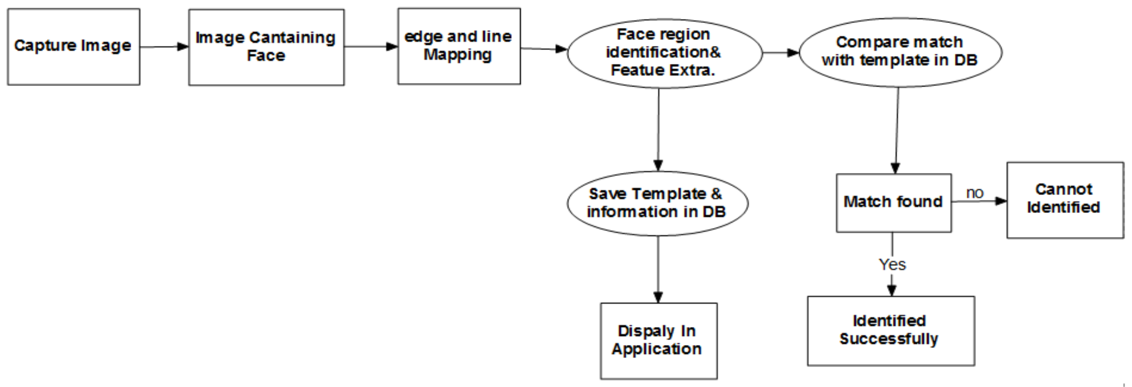


Fig 4.9: DFD Level 2

Chapter 5

IMPLEMENTATION DETAILS

The chapter provides the brief idea about the Implementation Details

Block Diagram

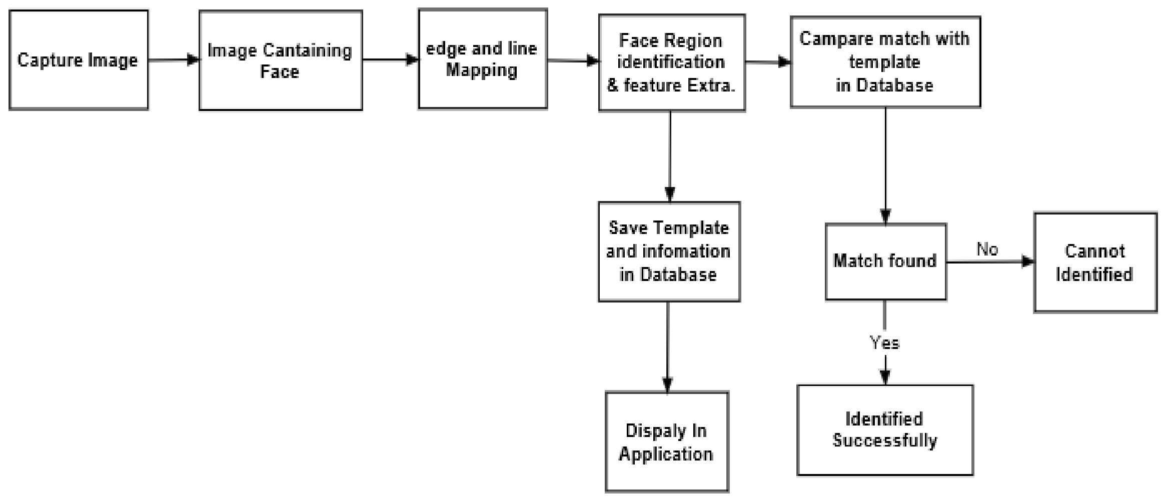


Fig5.1: Block Diagram

Edge Detection

Edge detection is the name for a set of mathematical methods which aim at identifying points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. The points at which image brightness changes sharply are typically organized into a set of curved line segments termed edges. The same problem of finding discontinuities in 1D signals is known as step detection and the problem of finding signal discontinuities over time is known as change detection. Edge detection is a fundamental tool in image processing, machine vision and computer vision, particularly in the areas of feature detection and feature extraction

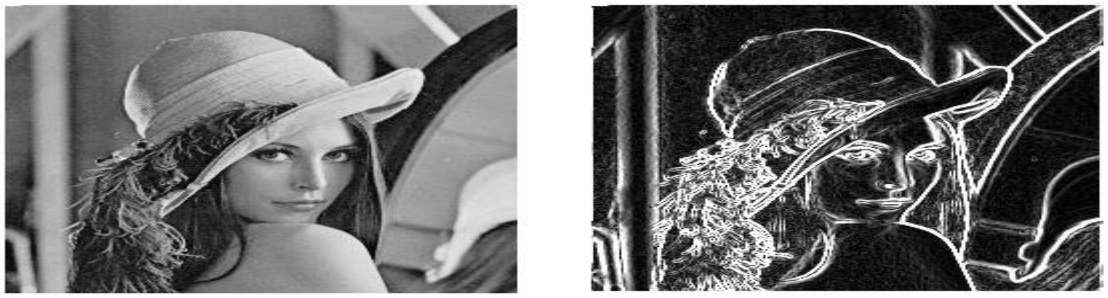


Fig 5.2: Edge Detection

Motivations

Canny edge detection applied to a photograph

The purpose of detecting sharp changes in image brightness is to capture important events and changes in properties of the world. It can be shown that under rather general assumptions for an image formation model, discontinuities in image brightness are likely to correspond to discontinuities in depth discontinuities in surface orientation changes in material properties and variations in scene illumination.

In the ideal case, the result of applying an edge detector to an image may lead to a set of connected curves that indicate the boundaries of objects, the boundaries of surface markings as well as curves that correspond to discontinuities in surface orientation. Thus, applying an edge detection algorithm to an image may significantly reduce the amount of data to be processed and may therefore filter out information that may be regarded as less relevant, while preserving the important structural properties of an image. If the edge detection step is successful, the subsequent task of interpreting the information contents in the original image may therefore be substantially simplified. However, it is not always possible to obtain such ideal edges from real life images of moderate complexity.

Edges extracted from non-trivial images are often hampered by fragmentation, meaning that the edge curves are not connected, missing edge segments as well as false edges not corresponding to interesting phenomena in the image – thus complicating the subsequent task of interpreting the image data.

Edge detection is one of the fundamental steps in image processing, image analysis, image pattern recognition, and computer vision techniques.

Algorithm used for capturing the Image

Harr Cascade

A simple rectangular Haar-like feature can be defined as the difference of the sum of pixels of areas inside the rectangle, which can be at any position and scale within the original image. This modified feature set is called *2-rectangle feature*. Viola and Jones also defined 3-rectangle features and 4-rectangle features. The values indicate certain characteristics of a particular area of the image. Each feature type can indicate the existence (or absence) of certain characteristics in the image, such as edges or changes in texture. For example, a 2-rectangle feature can indicate where the border lies between a dark region and a light region.

Algorithm For detection of face

Step1: The integral image is an array containing the sums of the pixels' intensity values located directly to the left of a pixel and directly above the pixel at location (x, y) inclusive

$$AI[x, y] = \sum_{x' \leq x,\, y' \leq y} A(x', y')$$

Step2: The features rotated by forty-five degrees, like the line feature shown in require another intermediate representation called the rotated integral image or rotated sum auxiliary image

$$AR[x, y] = \sum_{x' \leq x,\, x' \leq x - \left| y - y' \right|} A(x', y')$$

Step3: The cascading of the classifiers allows only the sub-images with the highest probability to be analyzed for all Haar-features that distinguish an object. It also allows one to vary the accuracy of a classifier

ALGORITHM

Step 1: Let $X = \{ x_{1}, x_{2}, \ldots, x_{n} \}$ be a random vector with observations $x_i \in R^{d}$.

Step 2: Compute the mean μ

$\mu = \frac{1}{n} \sum_{i=1}^{n} x_{i}$

Step3: Compute the the Covariance Matrix S

$S = \frac{1}{n} \sum_{i=1}^{n} (x_{i} - \mu) (x_{i} - \mu)^{T}$`

Step 4: Compute the eigenvalues λ_{i} and eigenvectors v_{i} of S

$S v_{i} = \lambda_{i} v_{i}, i=1,2,\ldots,n$

Step 5: Order the eigenvectors descending by their eigenvalue. The k principal components are the eigenvectors corresponding to the k largest eigenvalues.The k principal components of the observed vector x are then given by:

$y = W^{T} (x - \mu)$

where $W = (v_{1}, v_{2}, \ldots, v_{k})$.

The reconstruction from the PCA basis is given by:

$x = W y + \mu$

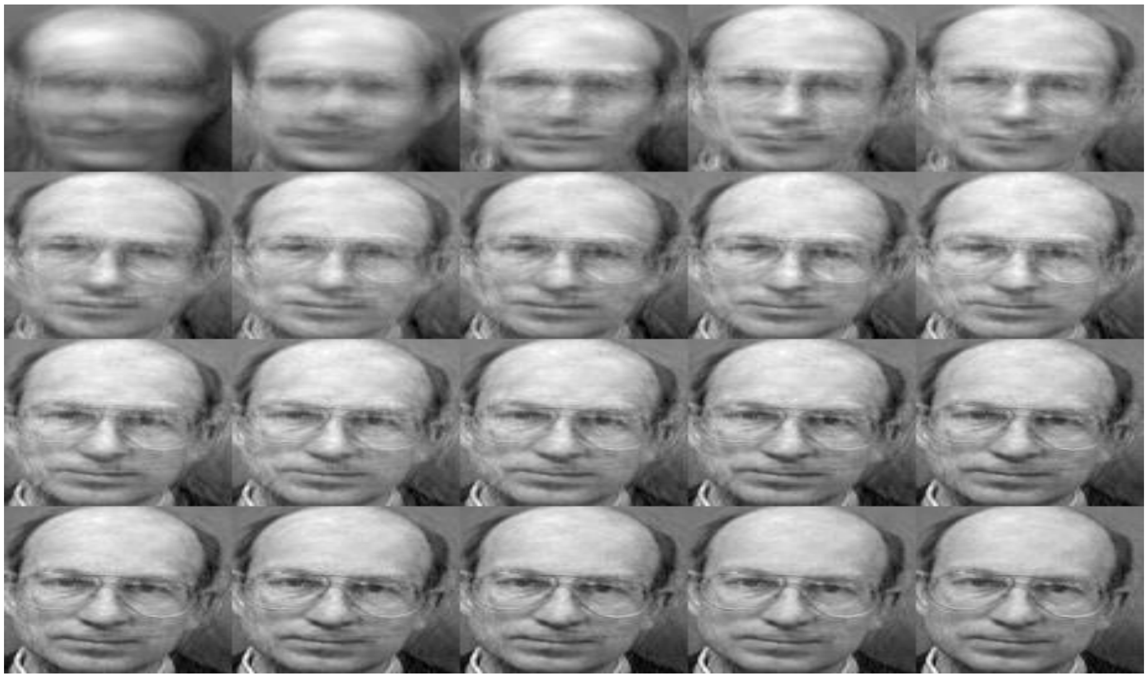
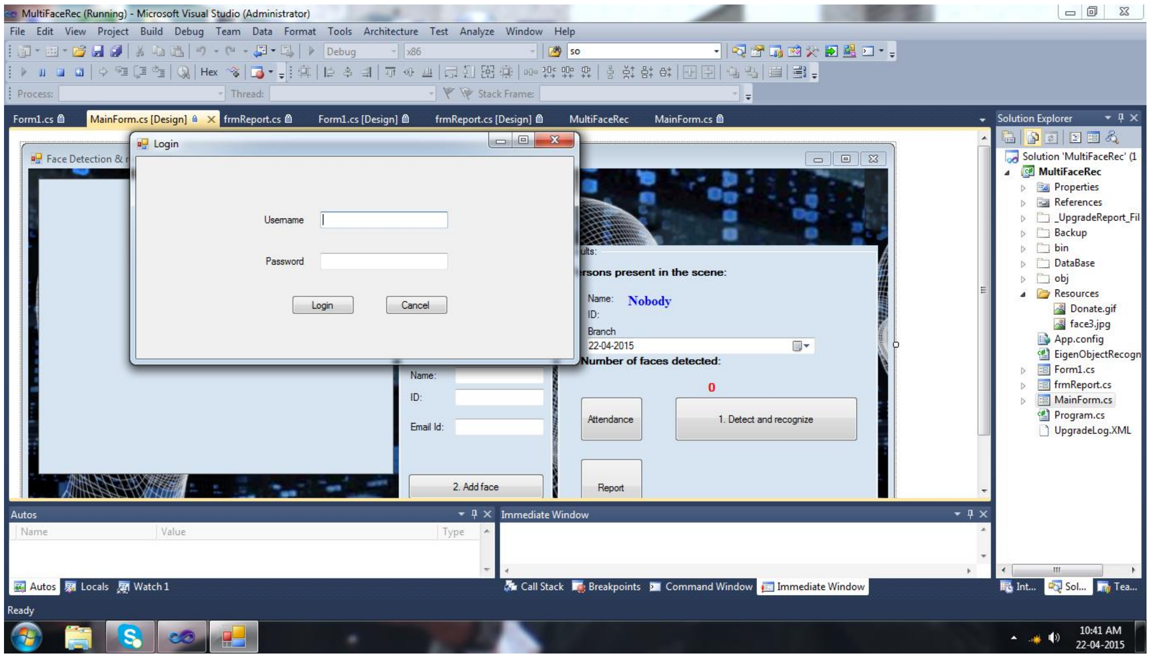
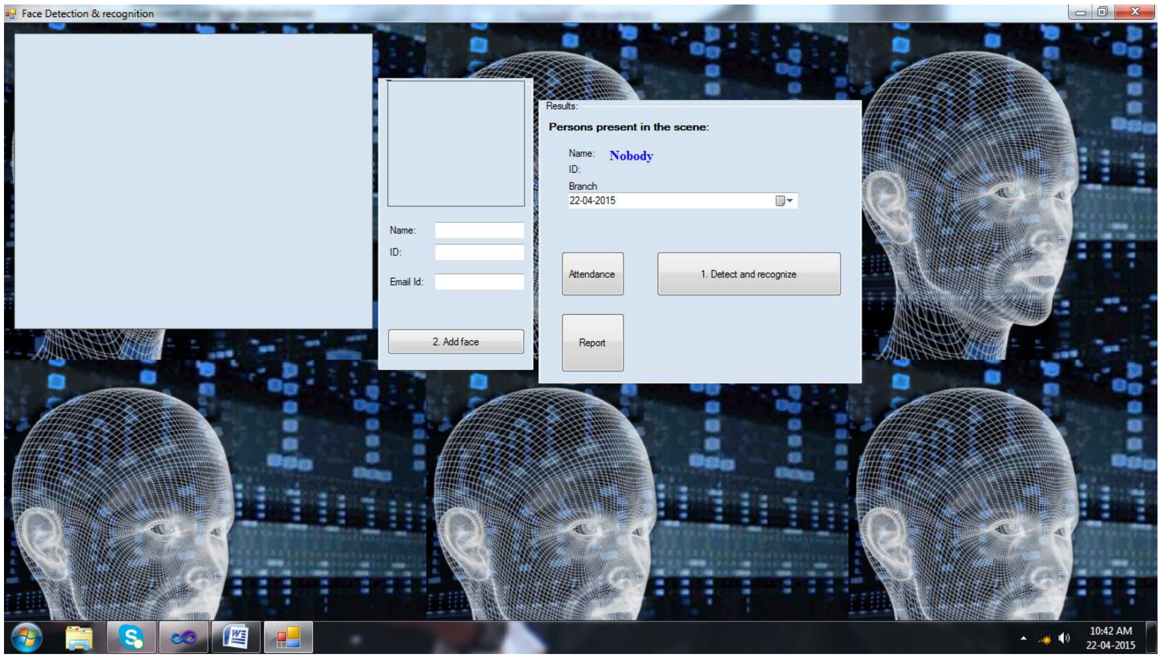
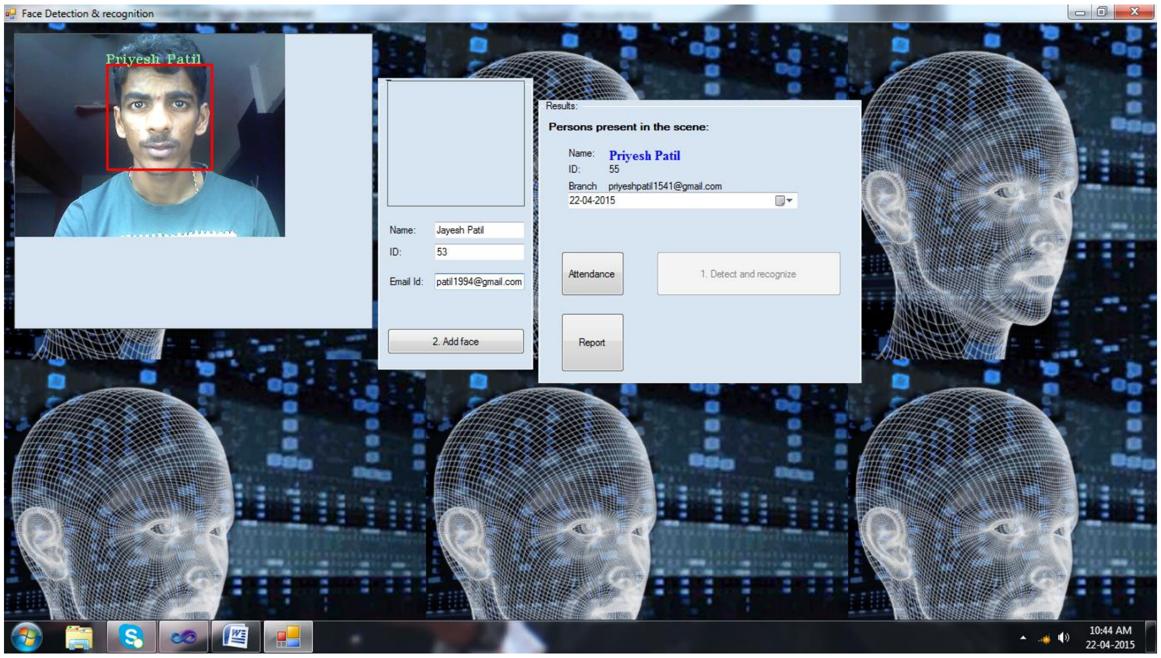


Fig 5.1: Capturing the image

Output Screen:



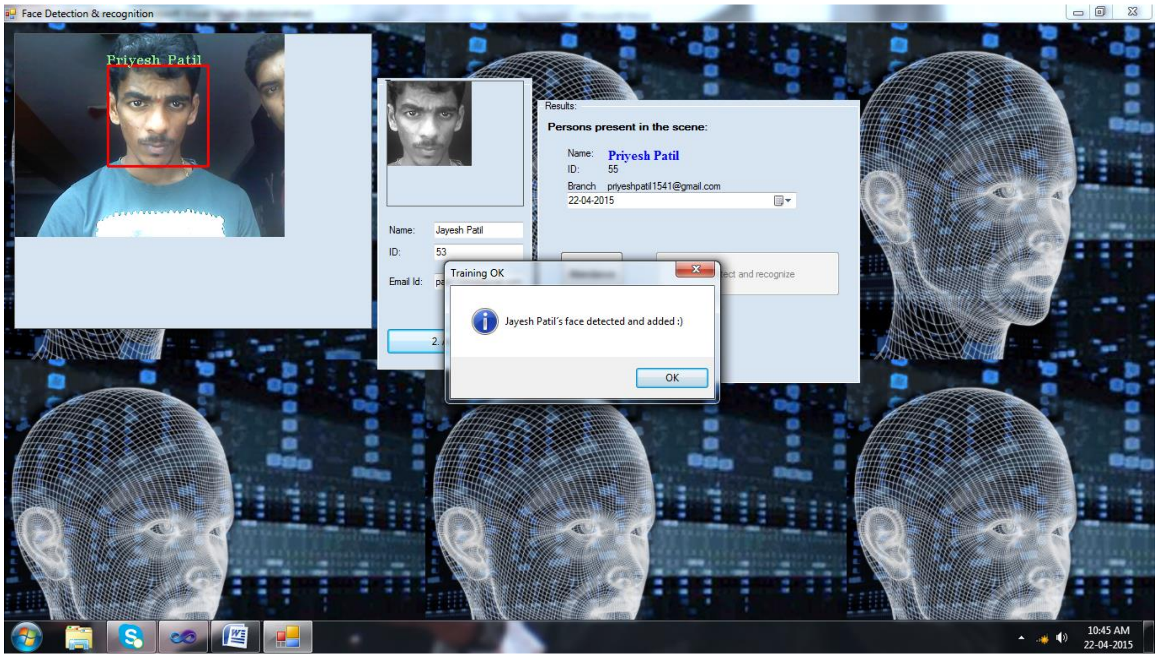




Name: Jayesh Patil
ID: 53
Email Id: patil1994@gmail.com

Persons present in the scene:

Name: Priyesh Patil
ID: 55
Branch priyeshpatil1541@gmail.com
22-04-2015



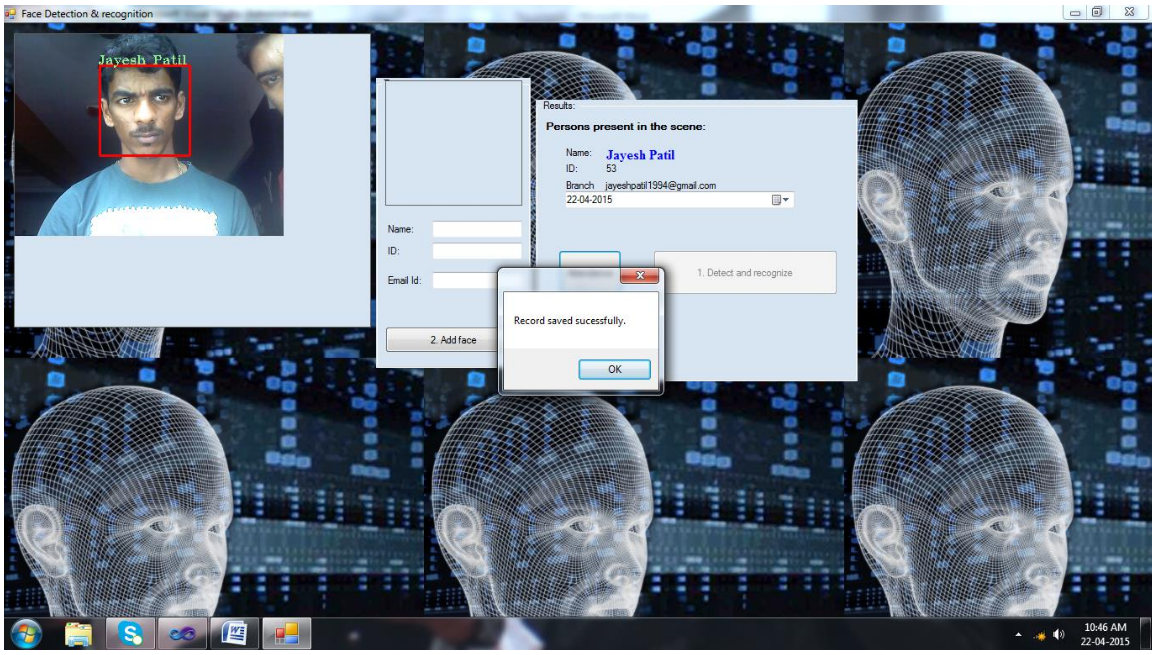
Name: Jayesh Patil
ID: 53

Persons present in the scene:

Name: Priyesh Patil
ID: 55
Branch priyeshpatil1541@gmail.com
22-04-2015

Training OK

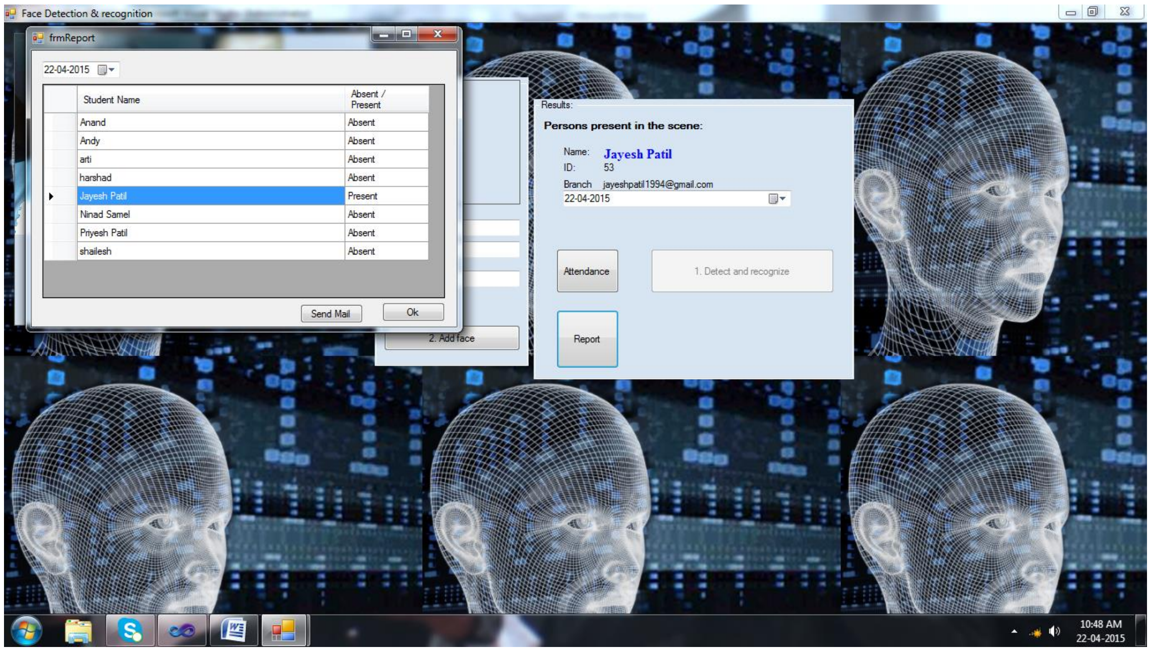
Jayesh Patil's face detected and added :)



Persons present in the scene:

Name: Jayesh Patil
ID: 53
Branch jayeshpatil1994@gmail.com
22-04-2015

Record saved sucessfully.



22-04-2015

Student Name	Absent / Present
Anand	Absent
Andy	Absent
arti	Absent
harshad	Absent
Jayesh Patil	Present
Ninad Samel	Absent
Priyesh Patil	Absent
shailesh	Absent

Persons present in the scene:

Name: Jayesh Patil
ID: 53
Branch jayeshpatil1994@gmail.com
22-04-2015

Chapter 6

TESTING TECHNOLOGY

System testing is a critical phase implementation. Testing of the system involves hardware devise and debugging of the computer programs and testing information processing procedures. Testing can be done with text data, which attempts to stimulate all possible conditions that may arise during processing. If structured programming Methodologies have been adopted during coding the testing proceeds from higher level to lower level of program module until the entire program is tested as unit. The testing methods adopted during the testing of the system were unit testing and integrated testing.

6.1 Unit Testing

Unit testing focuses first on the modules, independently of one another, to locate errors. This enables the tester to detect errors in coding and logical errors that is contained within that module alone. Those resulting from the interaction between modules are initially avoided.

6.2 Integration Testing

Integration testing is a systematic technique for constructing the program structure while at the same time to uncover the errors associated with interfacing. The objective is to take unit-tested module and build a program structure that has been detected by designing. It also tests to find the discrepancies between the system and its original objectives. Subordinate stubs are replaced one at time actual module. Tests were conducted at each module was integrated. On completion of each set another stub was replaced with the real module.

6.3 Functional Testing

Functional testing is a technique in which all the functionalities of the program are tested to check whether all the functions that where proposed during the planning phase are full filled. This is also to check that if all the functions proposed are working properly.

6.4 Performance Testing

Expected Result

- The client should be able to connect to the server properly without any problems.
- The connection establishment between the mobile device and the server should take minimal time.
- The mobile device should be able receive data from the server uninterruptedly.
- Information provided by the application should be correct and as per the user's need.

Observation

- Connection can be established easily provided that the server is on.
- The connection with the server takes time as it uses Internet connection.
- Receiving data from the server takes time.
- Information coming from the database is correct.

6.5 Load / Stress Testing

Expected Result

- Response time should be unaffected irrespective of the no of users.
- The introduction of the newer clients should not make the server to work hap hazardously.
- Continuous use of the server by different clients should not result into the server getting slowed down.
- Response time should not be degraded if there is congestion in network.

Observation

The speed of transmission was fine even when the newer clients were getting added. The response of the server was satisfying even with the introduction of newer client.

Chapter 7

PROJECT PLANNING AND SCHEDULING

7.1 Timeline Chart

Months	January				February			
Weeks	W1	W2	W3	W4	W1	W2	W3	W4
Deployment &Coding								
Installing and Configuring data processing model				🟦				
Installing net beans libraries						🟦		
Installing& Configuring user interface screens							🟦	🟦

Months	March				April			
Weeks	W1	W2	W3	W4	W1	W2	W3	W
Integration & Setting up the system	🟦							

Coding UI for using the data processing		🟦						
Deployment & Coding [Milestone]								
Testing								
Prepare Test Cases			🟦					
Execute Test Cases				🟦				
Analysis of Test Results				🟦				
Testing [Milestone]								
Final Project Report								
Prepare Final Report					🟦			
Final Report Approved by Project Guide						🟦		
Submission of Final Project Report							🟦	
Project Presentation							🟦	

Months	August				September			
Weeks	W1	W2	W3	W4	W1	W2	W2	W3
Requirement Analysis								
Literature survey	🟦	🟦						
Analysis of existing systems			🟦					
Study of existing system				🟦				
Hardware & Software Analysis				🟦				
Feasibility Study					🟦			
Preparing Problem Definition					🟦			
Consulting Faculty Members						🟦		
Making necessary changes						🟦		
Approved [Milestone]								
Planning								
Identification of Project Goals							🟦	
Identification of Project Deliverables							🟦	
Scheduling								🟦

Planning [Milestone]								

Months	October				November			
	W1	W2	W3	W4	W1	W2	W3	W4
Design								
Infrastructure Design	🟦							
Design User Interface	🟦							
Design & Integration of Controller & User nodes		🟦						
Design[Milestone]								
Project Report								
Aggregate necessary information			🟦					
Prepare Report			🟦					
Project Report Approved by Guide				🟦				
Report Submission					🟦			
Project Presentation					🟦			
Project Report Complete [Milestone]								

Fig7.1: TimeLine Chart

7.2 Task Distribution

TASKSHEET

WORK TASK	**ASSIGNED PERSON**
1. To Design the Interface	
1A. To develop Front End	Karthik Pillai Jayesh Patil
1B. To develop Back End	Kartik Pillai Priyesh Patil
2. To Develop Project Design	
2A. To develop DFD diagrams	Jayesh Patil,
2B.To provide timeline and task distribution	Jayesh Patil

3. To implement individual modules	
3A. To implement Data Merging	Priyesh Patil, Kartik Pillai
3B. To implement DB Connection	Kartik Pillai,
3C. To To implement Main GUI	Priyesh Patil Karthik pillai
4. Final Debugging and testing	Priyesh Patil,Kartik Pillai
6. Report Making	Priyesh Patil , Jayesh Patil, Kartik Pillai
7. Presentation	Priyesh Patil Jayesh Patil Karthik

Table no7.1: Tasksheet

Chapter 8

Conclusion and Future Work

8.1 Conclusion

Face recognition technology has come a long way in the last twenty years. Today, machines are able to automatically verify identity information for secure transactions, for surveillance and security tasks, and for access control to buildings etc. These applications usually work in controlled environments and recognition algorithms can take advantage of the environmental constraints to obtain high recognition accuracy. However, next generation face recognition systems are going to have widespread application in smart environments -- where computers and machines are more like helpful assistants.

To achieve this goal computers must be able to reliably identify nearby people in a manner that fits naturally within the pattern of normal human interactions. They must not require special interactions and must conform to human intuitions about when recognition is likely. This implies that future smart environments should use the same modalities as humans, and have approximately the same limitations. These goals now appear in reach -- however, substantial research remains to be done in making person recognition technology work reliably, in widely varying conditions using information from single or multiple modalities.

8.2 Future Work

Face recognition systems used today work very well under constrained conditions, although all systems work much better with frontal mug-shot images and constant lighting. All current face recognition algorithms fail under the vastly varying conditions under which humans need to and are able to identify other people. Next generation person recognition systems will need to recognize people in real-time and in much less constrained situations.

We believe that identification systems that are robust in natural environments, in the presence of noise and illumination changes, cannot rely on a single modality, so that fusion with other modalities is essential Technology used in smart environments has to be unobtrusive and allow users to act freely. Wearable systems in particular require their sensing technology to be small, low powered and easily integrable with the user's clothing. Considering all the requirements, identification systems that use face recognition and speaker identification seem to us to have the most potential for wide-spread application.

Cameras and microphones today are very small, light-weight and have been successfully integrated with wearable systems. Audio and video based recognition systems have the critical advantage that they use the modalities humans use for recognition. Finally, researchers are beginning to demonstrate that unobtrusive audio-and-video based person identification systems can achieve high recognition rates without requiring the user to be in highly controlled environments

Appendix

```
using System;
using System.Configuration;
using System.Collections.Generic;
using System.Drawing;
using System.IO;
using System.Threading;
using System.Windows.Forms;
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Structure;

namespace MultiFaceRec
{
    public partial class FrmPrincipal : Form
    {
        //Declararation of all variables, vectors and haarcascades
        Image<Bgr, Byte> currentFrame; //to store image or current frame
        Capture grabber; // object for camera
        HaarCascade face;    // Haar classifier
        HaarCascade eye;
        MCvFont font = new MCvFont(FONT.CV_FONT_HERSHEY_TRIPLEX, 0.5d, 0.5d);
        Image<Gray, byte> result, TrainedFace = null;
        Image<Gray, byte> gray = null;
        List<Image<Gray, byte>> trainingImages = new List<Image<Gray, byte>>();
        List<string> labels = new List<string>();
        List<string> NamePersons = new List<string>();
        int ContTrain, NumLabels, t;
        string name, names = null;
        string dat1, dat2, dat3;
        String strConnectionString =
ConfigurationManager.ConnectionStrings["myConnectionString"].ConnectionString;

        public FrmPrincipal()
        {
            InitializeComponent();
            //Load haarcascades for face detection
            face = new HaarCascade("haarcascade_frontalface_default.xml");
            //  eye = new HaarCascade("haarcascade_eye.xml");
            try
            {
                //Load of previus trainned faces and labels for each image
                string Labelsinfo = File.ReadAllText(Application.StartupPath +
"/TrainedFaces/TrainedLabels.txt");
                string[] Labels = Labelsinfo.Split('%');
                NumLabels = Convert.ToInt16(Labels[0]);
                ContTrain = NumLabels;
                string LoadFaces;

                for (int tf = 1; tf < NumLabels + 1; tf++)
                {
                    LoadFaces = "face" + tf + ".bmp";
                    trainingImages.Add(new Image<Gray, byte>(Application.StartupPath +
"/TrainedFaces/" + LoadFaces));
                    labels.Add(Labels[tf]);
                }

            }
            catch (Exception e)
            {
                //MessageBox.Show(e.ToString());
                MessageBox.Show("Nothing in binary database, please add at least a face(Simply
train the prototype with the Add Face Button).", "Triained faces load",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }

        }



        private void button1_Click(object sender, EventArgs e)
        {
            //Initialize the capture device
            grabber = new Capture();
            grabber.QueryFrame();
            //Initialize the FrameGraber event
            Application.Idle += new EventHandler(FrameGrabber);
            button1.Enabled = false;
        }



        private void button2_Click(object sender, System.EventArgs e)
        {
            try
            {
                if ((textBox1.Text != "") || (textBox2.Text != "") || (textBox3.Text != ""))
```

```
          {
            //Trained face counter
            ContTrain = ContTrain + 1;

            //Get a gray frame from capture device
            gray = grabber.QueryGrayFrame().Resize(320, 240,
Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);

            //Face Detector
            MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
            face,
            1.2,
            10,
            Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
            new Size(20, 20));

            //Action for each element detected
            foreach (MCvAvgComp f in facesDetected[0])
            {
              TrainedFace = currentFrame.Copy(f.rect).Convert<Gray, byte>();
              break;
            }

            //resize face detected image for force to compare the same size with the
            //test image with cubic interpolation type method
            TrainedFace = result.Resize(100, 100,
Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
            trainingImages.Add(TrainedFace);
            string user_data;
            user_data = textBox1.Text + ";" + textBox2.Text + ";" + textBox3.Text;

            labels.Add(user_data);
```

```
                    //Show face added in gray scale
                    imageBox1.Image = TrainedFace;

                    //Write the number of triained faces in a file text for further load
                    File.WriteAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt",
trainingImages.ToArray().Length.ToString() + "%");

                    //Write the labels of triained faces in a file text for further load
                    for (int i = 1; i < trainingImages.ToArray().Length + 1; i++)
                    {
                        trainingImages.ToArray()[i - 1].Save(Application.StartupPath +
"/TrainedFaces/face" + i + ".bmp");
                        File.AppendAllText(Application.StartupPath +
"/TrainedFaces/TrainedLabels.txt", labels.ToArray()[i - 1] + "%");
                    }
                    String strCommandText;
                    strCommandText = "Insert Into [Student] ([StudentName], [StudentID],
[Email_id]) Values ('" + textBox1.Text + "', '" + textBox2.Text + "' ,'" + textBox3.Text + "' )";

                    System.Data.OleDb.OleDbCommand objOleDbCommand = new
System.Data.OleDb.OleDbCommand();

                    try
                    {
                        objOleDbCommand.Connection = new
System.Data.OleDb.OleDbConnection(strConnectionString);
                        if (objOleDbCommand.Connection.State ==
System.Data.ConnectionState.Closed)
                        {
                            objOleDbCommand.Connection.Open();
                        }
                        objOleDbCommand.CommandText = strCommandText;
                        objOleDbCommand.CommandType = System.Data.CommandType.Text;
                        objOleDbCommand.ExecuteNonQuery();
                        //MessageBox.Show("Record saved sucessfully.");
                    }
                    catch (Exception ex)
                    {
                        if (objOleDbCommand.Connection.State ==
System.Data.ConnectionState.Open)
                        {
                            objOleDbCommand.Connection.Close();
                        }
                        //MessageBox.Show("Error while saving the record.");
                    }

                    MessageBox.Show(textBox1.Text + "´s face detected and added :)", "Training
OK", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    textBox1.Text = "";
                    textBox2.Text = "";
                    textBox3.Text = "";
                    imageBox1.Image = null;
                }
                else
                {
                    MessageBox.Show("Failed", "Training Fail", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
                }
            }
            catch
            {
                MessageBox.Show("Enable the face detection first", "Training Fail",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
        }
        void FrameGrabber(object sender, EventArgs e)
        {
            Thread.Sleep(300);
            label3.Text = "0";
            //label4.Text = "";
            NamePersons.Add("");



            //Get the current frame form capture device
            currentFrame = grabber.QueryFrame().Resize(320, 240,
Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);

            //Convert it to Grayscale
            gray = currentFrame.Convert<Gray, Byte>();

            //Face Detector
            MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
          face,
          1.2,
          10,
          Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
          new Size(20, 20));

            int num_faces;
            num_faces = facesDetected.GetLength(0);
            if (num_faces <= 1)
            {
                //Action for each element detected
                foreach (MCvAvgComp f in facesDetected[0])
                {
                    if (num_faces <= 1)
                    {
```

```
                    t = t + 1;
                    result = currentFrame.Copy(f.rect).Convert<Gray, byte>().Resize(100, 100,
Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
                    //draw the face detected in the 0th (gray) channel with blue color
                    currentFrame.Draw(f.rect, new Bgr(Color.Red), 2);



                    if (trainingImages.ToArray().Length != 0)
                    {
                        //TermCriteria for face recognition with numbers of trained images like
maxIteration
                        MCvTermCriteria termCrit = new MCvTermCriteria(ContTrain, 0.010);
                        //MCvTermCriteria termCrit = new MCvTermCriteria(ContTrain, 0.001);

                        //Eigen face recognizer
                        EigenObjectRecognizer recognizer = new EigenObjectRecognizer(
                          trainingImages.ToArray(),
                          labels.ToArray(),
                         0,
                          ref termCrit);



                        string[] arr;
                        name = recognizer.Recognize(result);
                        arr = name.Split(';');
                        dat1 = arr[0];
                        dat2 = arr[1];
                        dat3 = arr[2];

                        //Draw the label for each face detected and recognized
                        currentFrame.Draw(dat1, ref font, new Point(f.rect.X - 2, f.rect.Y - 2), new
Bgr(Color.LightGreen));

                    }

                    NamePersons[t - 1] = name;
                    NamePersons.Add("");

                }
                //Set the number of faces detected on the scene
                label3.Text = facesDetected[0].Length.ToString();

                num_faces = num_faces + 1;
                //Set the region of interest on the faces

                //    gray.ROI = f.rect;
                //    MCvAvgComp[][] eyesDetected = gray.DetectHaarCascade(
                //       eye,
                //       1.1,
                //       10,
                //       Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
                //       new Size(20, 20));
                //    gray.ROI = Rectangle.Empty;

                //    foreach (MCvAvgComp ey in eyesDetected[0])
                //    {
                //        Rectangle eyeRect = ey.rect;
                //        eyeRect.Offset(f.rect.X, f.rect.Y);
                //        currentFrame.Draw(eyeRect, new Bgr(Color.Blue), 2);
                //    }


            }
```

```
                t = 0;

                //Names concatenation of persons recognized
                //for (int nnn = 0; nnn < facesDetected[0].Length; nnn++)
                //{
                //    names = names + NamePersons[nnn] + ", ";
                //}
                names = NamePersons[0];

                //Show the faces procesed and recognized
                imageBoxFrameGrabber.Image = currentFrame;
                label4.Text = dat1;
                label8.Text = dat2;
                label9.Text = dat3;
                names = "";
                return;
                //Clear the list(vector) of names
                // NamePersons.Clear();
            }
        }

        private void FrmPrincipal_Load(object sender, EventArgs e)
        {
            dtpDate.Value = DateTime.Today.Date.Date;
        }
        private long GenerateReport()
        {
            string strCommandText;
            strCommandText = "Select [StudentName], 'Present' as [Status] From [Attendance]
    Where [AttendanceDate] = @dtpDate Union all Select [StudentName], 'Absent' as [Status]
    From [Student] where Studentname not in(select [Studentname] from [Attendance] where
    AttendanceDate = @dtpDate) ";
```

```
            //strCommandText = "Select [StudentName] From [Attendance] Where
[AttendanceDate] = @dtpDate";
            System.Data.OleDb.OleDbDataAdapter objOleDbDataAdapter = new
System.Data.OleDb.OleDbDataAdapter();
            System.Data.DataTable objDataTable = new System.Data.DataTable();
            try
            {
                objOleDbDataAdapter.SelectCommand = new
System.Data.OleDb.OleDbCommand();
                objOleDbDataAdapter.SelectCommand.Connection = new
System.Data.OleDb.OleDbConnection(strConnectionString);
                objOleDbDataAdapter.SelectCommand.CommandText = strCommandText;
                objOleDbDataAdapter.SelectCommand.Parameters.AddWithValue("@dtpDate",
dtpDate.Value);
                objOleDbDataAdapter.SelectCommand.CommandType =
System.Data.CommandType.Text;
                objOleDbDataAdapter.Fill(objDataTable);
                return objDataTable.Rows.Count;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }
        private long CheckAttendance()
        {
            string strCommandText;
            string adate = dtpDate.Value.ToShortDateString();
            strCommandText = "Select 1 From [Attendance] Where [StudentName] = '" +
label4.Text + "'";

            System.Data.OleDb.OleDbDataAdapter objOleDbDataAdapter = new
System.Data.OleDb.OleDbDataAdapter();
```

```
            System.Data.DataTable objDataTable = new System.Data.DataTable();
            try
            {
                objOleDbDataAdapter.SelectCommand = new
System.Data.OleDb.OleDbCommand();
                objOleDbDataAdapter.SelectCommand.Connection = new
System.Data.OleDb.OleDbConnection(strConnectionString);
                objOleDbDataAdapter.SelectCommand.CommandText = strCommandText;
                objOleDbDataAdapter.SelectCommand.CommandType =
System.Data.CommandType.Text;
                objOleDbDataAdapter.Fill(objDataTable);
                return objDataTable.Rows.Count;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }

        private void btnAttendance_Click(object sender, EventArgs e)
        {
            if (CheckAttendance() > 0)
            {
                String strCommandText;
                strCommandText = "Insert Into [Attendance] ([StudentName], [AttendanceDate])
Values ('" + label4.Text + "', '" + dtpDate.Value + "' )";
                System.Data.OleDb.OleDbCommand objOleDbCommand = new
System.Data.OleDb.OleDbCommand();
                try
                {
                    objOleDbCommand.Connection = new
System.Data.OleDb.OleDbConnection(strConnectionString);
                    if (objOleDbCommand.Connection.State == System.Data.ConnectionState.Closed)
                    {
                        objOleDbCommand.Connection.Open();
                    }
                    objOleDbCommand.CommandText = strCommandText;
                    objOleDbCommand.CommandType = System.Data.CommandType.Text;
                    objOleDbCommand.ExecuteNonQuery();
                    MessageBox.Show("Record saved sucessfully.");
                }

                catch (Exception ex)
                {
                    if (objOleDbCommand.Connection.State == System.Data.ConnectionState.Open)
                    {
                        objOleDbCommand.Connection.Close();
                    }
                    MessageBox.Show("Error while saving the record.");
                }
            }
            else
            {
                String strCommandText;
                strCommandText = "Insert Into [Attendance] ([StudentName], [AttendanceDate])
Values ('" + label4.Text + "', '" + dtpDate.Value + "' )";
                System.Data.OleDb.OleDbCommand objOleDbCommand = new
System.Data.OleDb.OleDbCommand();
                try
                {
                    objOleDbCommand.Connection = new
System.Data.OleDb.OleDbConnection(strConnectionString);
                    if (objOleDbCommand.Connection.State == System.Data.ConnectionState.Closed)
                    {
                        objOleDbCommand.Connection.Open();
                    }
                    objOleDbCommand.CommandText = strCommandText;
                    objOleDbCommand.CommandType = System.Data.CommandType.Text;
                    objOleDbCommand.ExecuteNonQuery();
                    MessageBox.Show("Record saved sucessfully.");
                }

                catch (Exception ex)
                {
                    if (objOleDbCommand.Connection.State == System.Data.ConnectionState.Open)
                    {
                        objOleDbCommand.Connection.Close();
                    }
                    MessageBox.Show("Error while saving the record.");
                }
            }

        }

        private void btnReport_Click(object sender, EventArgs e)
        {
            frmReport rep = new frmReport();
            rep.Show();
            //GenerateReport();
        }
      }

}
```

REFERENCES

1) H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *ECCV*, 2006.

2)P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *PAMI*, 1997.

3)F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *PAMI*, 1989.

4) W. Chen and Y. Gao. Recognizing partially occluded faces from a single sample per class using string-based matching. In *ECCV*. 2010.

5) H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *CVIU*, 2003.

6) D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph

matching in pattern recognition. *PAMI*, 2004.

7)T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *PAMI*, 2001.

8)E. Elhamifar and R. Vidal. Robust classification using structured sparse representation. In *CVPR*, 2011.

9)A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *PAMI*, 2001.