

FACE RECOGNITION

A Project Report

Submitted by

Alfa Zareena Hisham

Amol Joseph Arickathil

Manu James

*In partial fulfillment of the requirements for award of the degree of **Bachelor of Technology** in Computer Science and Engineering*



**FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)
ANGAMALY-683577, ERNAKULAM (DIST)**

Affiliated to

MAHATMA GANDHI UNIVERSITY

Kottayam-686560

MAY 2010

FACE RECOGNITION

A Project Report

*In partial fulfillment of the requirements for award of the degree of **Bachelor of Technology** in Computer Science & Engineering*



Submitted by

ALFA ZAREENA HISHAM

AMOL JOSEPH ARICKATHIL

MANU JAMES

**FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)
ANGAMALY-683577, ERNAKULAM (DIST)**

Affiliated to

MAHATMA GANDHI UNIVERSITY

Kottayam-686560

MAY 2010

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)

Angamaly-683577



CERTIFICATE

This is to certify that the project report titled **Face Recognition** submitted by Alfa Zareena Hisham(Reg No: 239503), Amol Joseph Arickathil(Reg no 239505), Manu James(Reg no: 239557)towards partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in Computer Science and Engineering is a record of bonafide work carried out by them during the academic year 2009 –2010.

Internal Guide

Head of the Department

Place: Mookkannoor

Date:

ACKNOWLEDGEMENT

If the words were considered as symbols of Approval and token of Acknowledgement, then let the words pay the heralding role of expressing our gratitude. First and foremost, we praise the God Almighty for the grace he showered on us during our studies as well as our day-to-day life activities.

Dreams never run into reality unless a lot of effort and hard work is put into it. And no efforts bear fruit in the absence of support and guidance. It takes a lot of effort to work your way through this.

We would like to take this chance to thank the Chairman of FISAT, Mr.P.V.Mathew and Principal, Dr.P.S.Sreejith for providing us with such an environment, where students can explore their creative ideas. Equally eligible is the Head of the Department of Computer Science, Mr.Prasad.J.C for encouraging the students to make these nations true.

We are extremely grateful to the Project-in-charge, Mr.Mahesh for the valuable suggestions for the project. We also sincerely thank the Computer science and Engineering faculty for providing us with invaluable help.

We are also extremely thankful to Mr. Amar Jyothi, Project Guide (Lakhotia computer centre) for supporting us and providing all the valuable information for the successful completion of the project.

Overall, nothing comprises the support and hard work equally put in by our teammates. Last, but not the least, we thank all our families and friends for giving us the help, strength and courage fro accomplishing the task.

ABSTRACT

The project Face Recognition Using Neural Networks is used to identify a person from his face image.

The flexibility of neural network is that it can identify a person by no just comparing the pictures but first extracting a pattern and when this pattern is provided as an input to the neural network it generates an output which is compared. So the data stored in a database is just a binary array than a whole image.

The front end is done using Java(in jsdk) . XML is used as the database. This authentication can be further used to provide login to a lab, office or any other applications where user identification is needed.

The system “Face Recognition System” has been designed & developed flexibly according to the current requirements .The Administrator can add new profiles, edit profiles, compare the images and most of the facilities are also provided for the staff. So depending upon the requirements of the user in future the Face Recognition System can be updated in a much reliable manner.

CONTENTS

Chapter 1	INTRODUCTION	1
Chapter 2	SYSTEM ANALYSIS	2
2.1	LITERATURE SURVEY	2
2.1.1	STUDY OF EXISTING SYSTEM	2
2.1.2	STUDY OF PROPOSED SYSTEM	3
2.2	REQUIREMENT SPECIFICATION	4
2.3	SOFTWARE SPECIFICATION	5
2.4	HARDWARE AND SOFTWARE REQUIREMENTS	6
2.4.1	HARDWARE SPECIFICATION	6
2.4.2	SOFTWARE SPECIFICATION	6
2.5	FEATURES OF THE LANGUAGE USED	7
Chapter 3	DESIGN	12
3.1	DESIGN OF DATA STRUCTURES	11
3.2	DESIGN OF MODULES	14
Chapter 4	IMPLEMENTATION	17
4.1	TESTING	17
4.2	IMPLEMENTATION	22
Chapter 5	CONCLUSION	35
5.1	FUTURE SCOPE	35
	REFERENCES	
	APPENDIX A	i
	APPENDIX B	xxx

1. INTRODUCTION

Face recognition technology is the least intrusive and fastest biometric technology. It works with the most obvious individual identifier the human face.

Instead of requiring people to place their hand on a reader face recognition system unobtrusively take pictures of peoples face as they enter a defined area .This system captures Image from webcam or from any location then face recognition system does necessary process and finally output is displayed. They have long been used for identification because of their immutability and individuality. Immutability refers to the permanent and unchanging character of the pattern on each face. Individuality refers to the uniqueness of ridge details across individuals; the probability that two face patterns are alike is about 1 in 1.9×10^{15} . However, manual face verification is so tedious, time consuming and expensive that is incapable of meeting today's increasing performance requirements. An automatic face recognition system is widely adopted in many applications such as building or area security and Passport verification. The application should be able to identify a person from the database.

Users of the system

The various users of the Automatic face recognition are

1. System Administrator
2. User

The system administrator is a wide base role and is a super user of the system. The responsibility of a system administrator include

- >Creation of the various actors (users) of the system etc.
- >Editing the databases

2. SYSTEM ANALYSIS

The system analysis involves the identification of the objectives and requirements, evaluation of alternative solutions and recommendation for a more feasible solution. In other words, system analysis is a systematic process of gathering, recording and interpreting facts. It also includes studying the problems encountered in the present system and introducing a new computer system into an organization.

System analysis itself breaks into two stages: Preliminary and Detailed. During the preliminary analysis, the analyst lists the objectives of the proposed system. These findings come together in the preliminary report.

Once the preliminary report is approved, the system analysis phase advances into the second stage, the Detailed Analysis. During detailed analysis, required data and information are collected and a detailed study is made.

2.1. Literature Survey

2.1.1. Study of existing system

Taking into consideration the existing system of “Face Identification” it is at present being done in a manual way. This is in fact a very tedious and hectic task for the analysts of Airport. In existing system all the details are stored in a traditional manner. So retrieval and comparison are very tedious.

2.1.1.1. Disadvantages

- **Since all the process is done manually, a lot of Mistakes can occur.**

- **Since all records are stored manually, it is highly Susceptible to loss in or damage**
- **It requires a lot of human power and time.**

2.1.2. Study of the proposed system

The proposed system is called “Face recognition using neural networks”. This proposed system has both facilities used in the old system as well as some additional features.

In this project, a pixel values from certain area is extracted from the image, converted to binary and stored in a file. This file is given as input to the neural network. After processing the neural network obtain a binary output then it compared with the database and the corresponding matching result is displayed.

This system provide the facilities such as

- **Add new user profile**
- **Add new user**
- **Comparison of images(Identification)**
- **Training.**

2.2. REQUIREMENT SPECIFICATION

General

- **Administrator id and password-** Administrator will be given a username and password with which he/she can log in to the software. Only the administrator has the privilege of modifying the database, modifying the images etc.
- **User id and password-** The user will also be assigned a username and password to log in to his/her account. The user can then give the images for comparison. He/she cannot make modifications.

Functional

- Image from webcam.
- Image is converted to black and white. Binary information is extracted from this image and stored. This is the input for neural network.
- Training of neural network.
- Database- A set of records is stored in the database. The Administrator can add, edit, and delete the records.
- Identification-(matching process). After the neural network is trained, the patterns are used to compare the input image and the trained image to identify a person.

2.3. SOFTWARE SPECIFICATION

The administrator is given a user name and a password with which he/she can log in to the account. Password and user name verified with the one stored in the database. Once it is verified the administrator is given the following privileges.

- Training
- User management
- Record management
- User Management
- Change password
- Identification

The user is also given a user name and password. He/she is given only certain privileges like identification. A user cannot change his/her details or train images.

The database used here is XML. The user details like name, age, gender, email and phone number is stored in the database.

2.4. HARDWARE AND SOFTWARE REQUIREMENTS

2.4.1. Hardware Requirements

Processor	: Pentium Processor (80486)
RAM	: 128 MB
Power Supply	: 300v
Monitor	: SVGA color
Hard Disc	: 20GB (Seagate)
Mouse	: Logitech Mouse
Keyboard	: Logitech (104) Keys
Webcam	: Creative or normal

2.4.2. Software Requirements

Front end	: Java (JDK 1.5 or 1.6)
Back end	: XML
Operating System	: Window

2.5. FEATURES OF THE LANGUAGES USED

2.5.1. Java Overview

Java is object oriented programming at the core. The fundamental forces that necessitated to the invention of java are portability and security. The key that allows Java to solve both the security and the portability problems is that the output of a Java compiler is not an executable code. Rather, it is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for Byte code.

Java was designed to be easy for the professional programmers to learn and use effectively. The object model in java is simple and easy to extend. The ability to create robust programs was given a high priority in the design of java. Java supports multithreaded programming, which allows writing programs that do many things simultaneously. The important goal of java is 'write once, run anywhere, any time, and forever'. Java is designed for distributed environment because it handles TCP/IP protocol. Java programs carry substantial amounts of run-time information that is used to verify and resolve access to objects at run-time. This makes it possible to dynamically link code in a safe and expedient manner.

Java Environment

Java environment includes a large number of development tools and hundreds of classes and methods. The development tool is a part of the Java Development

Kit (JDK) system and the classes and methods are part of the Java Standard Library (JSL), also known as the Application Programming Interface (API).

Java Development Kit

The Java Development Kit comes with a collection of tools that are used for developing and running Java programs. They include:

- Applet viewer (for viewing Java applets)
- Javac (Java Compiler)
- Javap (Java De-assembler)
- Javah (for C header files)
- Javadoc (for creating HTML documents)
- Jdb (Java debugger)

Thread

A thread is a program's path of execution and multithreading enhance performance and functionality by allowing a program to efficiently perform multiple tasks simultaneously. One way to handle requests from more than one client is to make the server program multi-threaded. A multi-threaded server creates a thread for each communication it accepts from a client. Using threads, a multi-threaded server program can accept a connection from a client, start a thread for that communication, and continue listening for requests from other clients.

Sockets

A socket is a software endpoint that establishes bi-directional communication between a server program and one or more client programs. The socket associates the server program with a specific hardware port on the machine

where it runs. So any client program, anywhere in the network, with a socket associated with that same port could communication with the server program. A server program typically provides resources to a network of client programs. Client programs send requests to the server program, and the server program responds to the request.

Remote Method Invocation

RMI lets Java objects on different hosts communicate with each other by calling methods in objects. A remote object lives on a server. The remote object implement a remote interface that specifies which of its methods can be invoked by a client. Clients invoke the methods of the remote object almost exactly as they invoke local methods. A remote object is an object with methods that may be invoke from a different Java virtual machine.

JDBC API

The JDBC API comes as a single Java package (java. sql) containing a series of classes. These classes provide just the type of middleware needed to handle a relational database. Basically, they allow you to establish a connection with the database and send queries. You can process the results of these queries, retrieve meta-information on your database, and handle the exceptions that might occur in the midst of all this.

2.5.2. J2SE Overview

Java Platform, Standard Edition (Java SE) software is the premier platform for rapidly developing and deploying secure, portable applications that run on server and desktop systems spanning most operating systems. J2SE 5.0 is a significant release including many new features and updates while preserving compatibility and stability.

The development of J2SE 5.0 was led by Sun and progressed following the Java Community Process (JCP) to include input from a variety of constituents. Some highlights of J2SE 5.0:

- New language updates: Metadata, Generics, Enumerated types, Autoboxing of primitive types
- New JVM Monitoring and Management API
- Improved out-of-box performance
- New (but compatible) default Java look and feel

2.5.3. NetBeans IDE 6.7

The NetBeans IDE is an award-winning integrated development environment available for Windows, Mac, Linux, and Solaris. The NetBeans project consists of an opensource IDE and an application platform that enable developers to rapidly create web, enterprise, desktop, and mobile applications using the Java platform, as well as JavaFX, PHP, JavaScript and Ajax, Ruby and Ruby on Rails, Groovy and Grails, and C/C++.

The NetBeans project is supported by a vibrant developer community and offersextensive documentation and training resources resources as well as a diverse selection of third-party plugins.

2.5.4. XML

XML is the acronym for eXtensible Markup Language, the universal format for structured documents and data on the Web. XML is an industry-standard protocol administered by the World Wide Web Consortium (W3C).

3. DESIGN

Design is the first step in the development phase for any engineered product or system. It may be defined as: "the process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization ". Computer software design like engineering design approaches in other disciplines changes continually as new methods, better analysis, and broader understanding evolve.

Using one of a number of design methods the design step produces a data design, an architectural design and a procedural design. Preliminary design is concerned with transformation requirements to data and software architectures. Detail design Focus on refinements to architectural representation that lead to detailed data structure and algorithmic representation for software. The data design transforms the information domain model created during analysis in to the data structures that will be required to implement the software. The architectural design defines the relationship among major structural components in to a procedural description of the software.

3.1. DESIGN OF DATA STRUCTURES

The most important aspect of building an application is the design of tables or database schema. The overall objective in the process of table design has been to treat data as an organization of data in a database. It aims to achieve the three major objectives, are given below:

- Data Integration
- Data Abstraction
- Data Independence

Several degrees of normalization have to be applied during the process of table design. The major aim of the process of normalization is to reduce data redundancy and prevents losing data integrity. Redundancy refers to unwanted and unnecessary

repetition of data. Data Integrity has to be maintained at all levels. Poor normalization can cause problems related to the storage and retrieval of data. During the process of normalization, dependencies can be identified which cause serious problems during the updation.

The theme behind the database is to handle information as an integrated whole, thus making access to information easy, quick, inexpensive and flexible for users. The entire package depends on how the data are maintained in the system. Each table has designed with perfect vision. Minor tables have been created which though takes much space facilitates the process of querying fast and accurate.

In a database environment, common data are available in which several users can use. The concept behind a database is an integrated collection of data and provides a centralized access to the data from the program. It makes possible to treat data as a separate resource. While designing database, several objectives must be considered:

- Controlled Redundancy
- Data Independence
- More information at low cost
- Accuracy and Integrity
- Recovery from failure
- Privacy and Security.
- Performance

TABLE 1.1: Login

NO	FIELDNAME	DATA TYPE	DESCRIPTION
1	User name	Text	User ID
2	Password	Text	Password

TABLE 1.2: ADDNEWRECORD

NO	FIELD NAME	DATA TYPE	DESCRIPTION
1	Id	Text	User id
2	Name	Text	User Name
3	Age	Text	Age
4	Gender	Text	Gender
5	Phone	Text	Phone number
6	Email	Text	Email

3.2. DESIGN OF MODULES

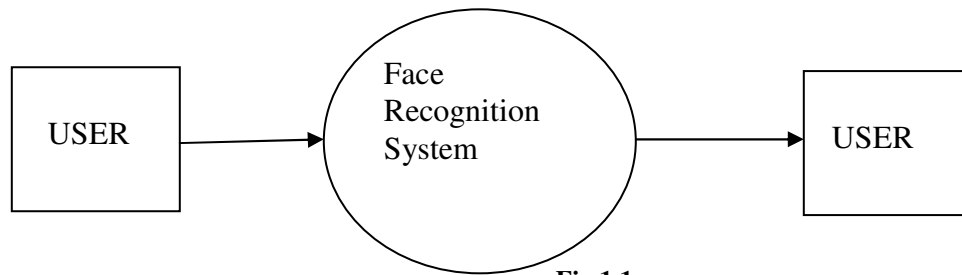
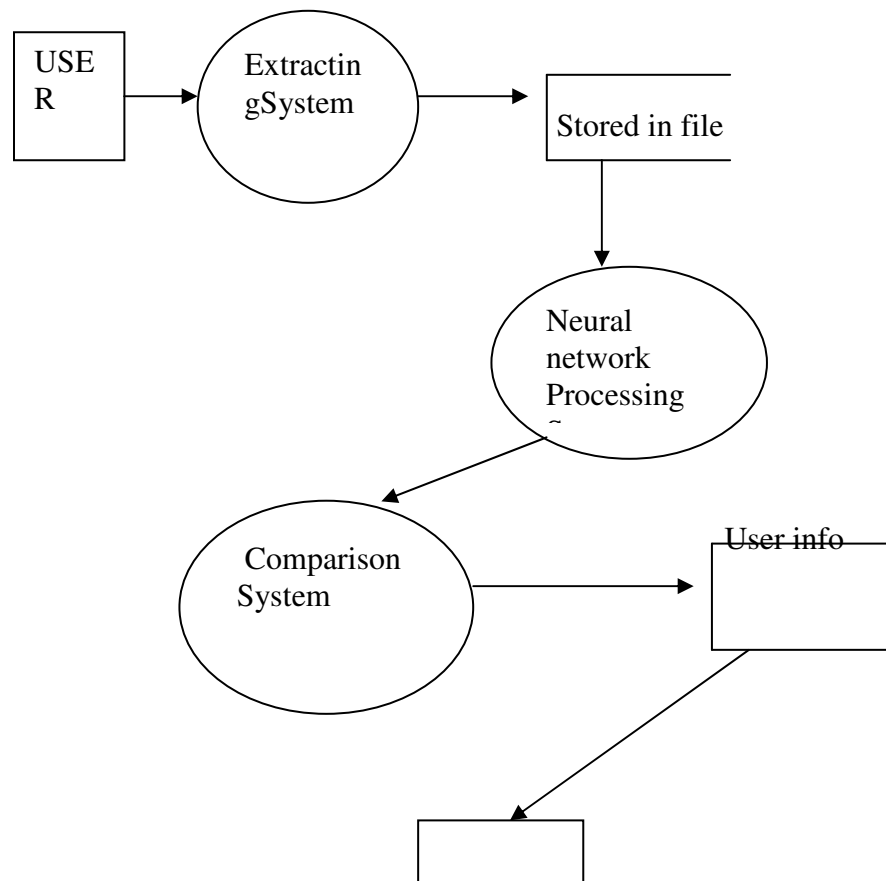


Fig 1.1



The project is divided into 5 main modules

- Image capture
- Training
- Record management
- Identification

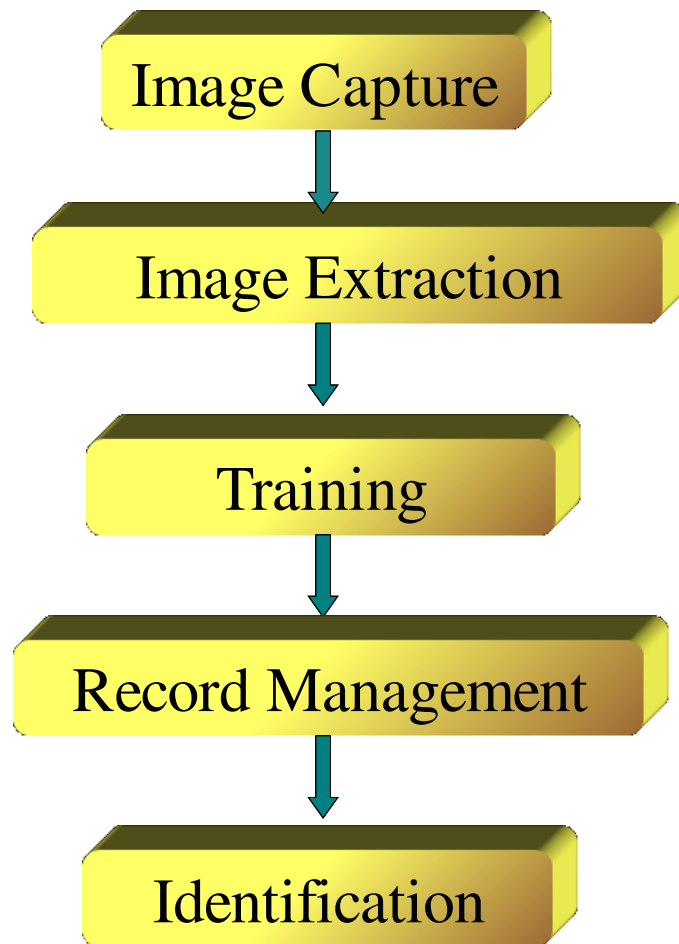


Fig 1.3

IMAGE CAPTURE

Images are captured from a web cam and stored in a folder for image extraction.

IMAGE EXTRACTION

The captured images are then converted into a black and white image. And we extract the binary information from the picture and this binary information is stored as matrix in a file. For face there is one hot spot for manipulation. This is the input for neural network.

TRAINING

Processed image is fed into the back propagation neural network as input in order to train the neural network. According to this algorithm there is an input pattern and an expected output pattern. Comparing the 2 we get an error ratio. Decrease the error ratio as far as possible in order to get the corrected output from the neural network

RECORD MANAGEMENT

A set of records are stored in the database. Administrator can add, edit and delete the records which are stored in the database.

IDENTIFICATION

After training, the neural network is ready to perform the identification operations (matching process). Neural network is used the patterns which are earlier trained for comparing the input image to trained image. if these are same, the neural network identify the person otherwise not.

4. IMPLEMENTATION AND TESTING

4.1. Testing

System testing is vital for the success of any software system. It is done to check whether the system has any bugs. In this phase, several tests and validations will be carried out on modules to check for their functionality. Testing and debugging is a very critical case in system development. The quality of the system is confirmed by the thoroughness of its testing.

Duration and cost of testing and debugging is a significant fraction of the system development cycle and hence influences overall productivity during the system development. In this phase the errors in the program or module are localized and modifications are done to eliminate them.

- Unit testing
- Integration testing
- Validation testing
- Output testing
- User acceptance testing
- White Box Testing
- Black Box Testing

Unit Testing

In this testing the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software in the module. This is also known as 'module testing'. The modules of the system are

tested separately. Smallest components of the software i.e. module are tested here. Test cases are therefore designed to test

- Program logic
- Functionality
- Boundary conditions and Interfaces
- Data structures
- All paths in the program

Integration testing

It is technique for constructing the program structure while at the same time conducting test to uncover errors associated with the interface. The objective is to take unit test module and build the program structure that has been proposed by design. All modules are combined in this testing step. Then the entire program is tested as whole. If a set of errors are encountered correction is difficult because the isolation of the error is complicated by vastness of the entire program.

Integration testing is a systematic testing that can be done with sample data. The need for the integrate test is to find the overall system performance. The process of combining multiple modules systematically for conducting tests in order to find error in the interface between modules is called integration testing. Integrating testing is done after successful completion of unit testing. Number of strategies can be followed to do integration testing. They are classified as:

- Incremental strategy
- Non-incremental strategy
- Mixed strategy

Validation Testing

Validation testing can be defined as many, but a single definition is that validation succeeds when the software functions in a manner that can be reasonable expected by the customer. After validation test have been conducted one of the two possible conditions exist. The function or performance characteristics are acceptable and confirmed to specification. A decision from specification is uncovered and defining list is created. System validations check the quality of the software in both simulated (test

data) and live environmental (live data). First the software goes through a phase (alpha testing) in which errors and failures based on simulated user requirements are verified and studied. The modified software is then objected to phase two (beta testing) in the actual use of sight or live environment.

Output Testing

After performing the validation testing the next step is output as the user about the format required testing of the proposed system since no system could be useful if does not produce the required out put in the specific format.

The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

User Acceptance Testing

After the developers complete the system testing successfully acceptance testing is done at the customer end. It is the customer or the end-user who knows designs the test cases. In this type of testing emphasis is on the usability of the product.

Acceptance testing is supported through alpha and beta testing. Alpha testing is done when the software is made operational for the first time to be tested by the users at developer's site. Hence it is possible that it will involve making lot of changes to program code. Beta testing follows alpha testing but now the testing is done at the customer's site that validates the product after using it for few days. At this stage few changes as compared to alpha testing would be made to the product.

User acceptance of a system is a key factor to the success of any system. The system under consideration was tested for user acceptance by constantly keeping in touch with

the proposed system user at the time of developing and making changes wherever required.

It is done regarding the following points.

- Input screen design
- Output screen design
- Online message to guide the user format of the reports and other output.

Black Box Testing

Knowing the specified function that a product has been designed to perform, test can be conducted that each function is fully operational. Black box test are carried out to test that input to a function is properly accepted of the system with little regard for the internal structure of the system.

Errors in the following category are observed through black box testing

- Interface errors
- Errors in database structure or external database access
- Performance errors
- Initialization
- Termination

White Box Testing

White box testing of software is predicted on a close examination of procedural details. The status of the project may be tested points to determine whether the expected or asserted status is corresponding to the actual status.

Using this, the following test cases can be derived.

- Exercise all logical condition on their true and false side
- Exercise all loops within their boundaries and their operation bounds
- Exercise internal data structure to ensure their validity.

Table 1.3 : Test Case - Administrator Login Screen

SL.N O	TEST CASE	EVENT	EXPECTED RESULT	ACTUAL RESULT
1	If User Name= “Valid”& Password= “Valid”	On click: Ok Button	Login Successful	Login Successful
2	If User Name = “Valid”& Password= “Invalid”	On click: Ok Button	Invalid Username or Password	Invalid Username or Password
3	If User Name = “Invalid”& Password= “Valid”	On click: Ok Button	Invalid Username or Password	Invalid Username or Password
4	If User Name = “Invalid”& Password= “Invalid”	On click: Ok Button	Invalid Username or Password	Invalid Username or Password

4.2. Implementation

Implementation is an activity that is contained throughout the development phase. It is a process of bringing a developed system into operational use and turning it over to the user. The new system and its components are to be tested in a structured and planned manner. A successful system should be delivered and users should have confidence that the system would work efficiently and effectively. The more complex the system being implemented the more involved will be the system analysis and design effort required for implementation. The implementation plan involves the following:

- Testing to confirm effectiveness
- Detection and correction of errors

‘Face recognition’ is implemented using artificial neural networks. A neural net is an artificial representation of the human brain that tries to simulate its learning process.

Like the human brain, a neural net also consists of neurons and connections between them. The neurons are transporting incoming information on their outgoing connections to other neurons. In neural net terms these connections are called weights. The "electrical" information is simulated with specific values stored in those weights. By simply changing these weight values the changing of the connection structure can also be simulated.

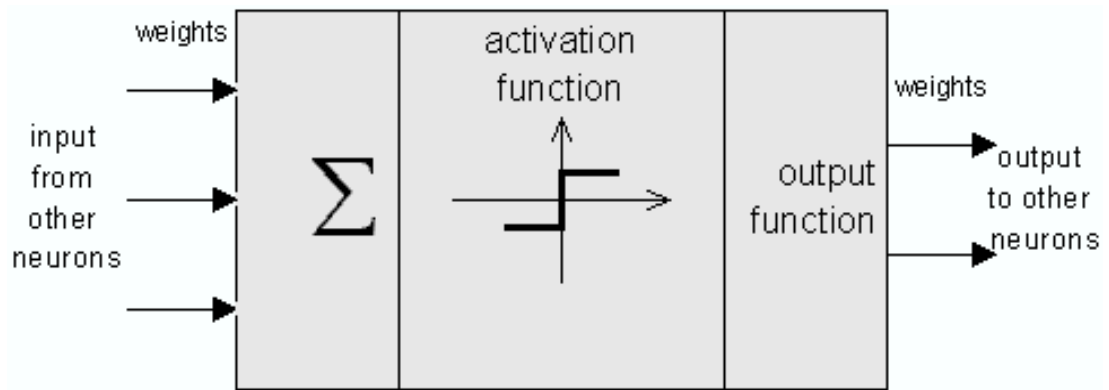


Fig 1.4

Information (called the input) is sent to the neuron on its incoming weights. This input is processed by a propagation function that adds up the values of all incoming weights.

The resulting value is compared with a certain threshold value by the neuron's activation function. If the input exceeds the threshold value, the neuron will be activated, otherwise it will be inhibited. If activated, the neuron sends an output on its outgoing weights to all connected neurons and so on. In a neural net, the neurons are grouped in layers, called neuron layers.

Usually each neuron of one layer is connected to all neurons of the preceding and the following layer (except the input layer and the output layer of the net). The information given to a neural net is propagated layer-by-layer from input layer to output layer through none, one or more hidden layers. Depending on the learning algorithm, it is also possible that information is propagated backwards through the net.

The following figure shows a neural net with three neuron layers.

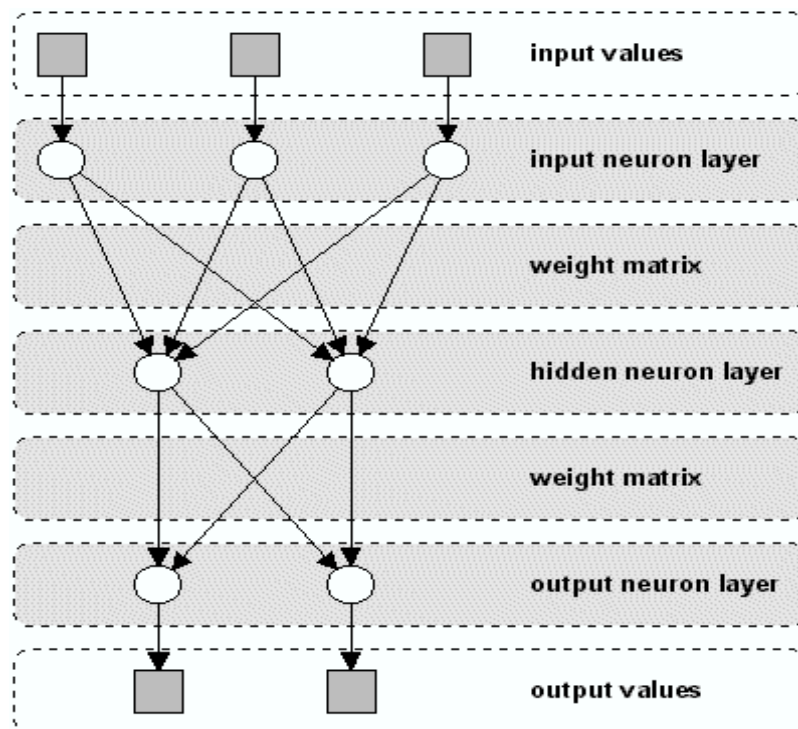


Fig 1.5

Supervised and unsupervised learning

The learning algorithm of a neural network can either be supervised or unsupervised. A neural net is said to learn *supervised*, if the desired output is already known.

Example: pattern association

Suppose, a neural net shall learn to associate the following pairs of patterns.

The input patterns are decimal numbers, each represented in a sequence of bits. The target patterns are given in form of binary values of the decimal numbers:

Input pattern	target pattern
0001	001
0010	010

0100

011

1000

100

While learning, one of the input patterns is given to the net's input layer. This pattern is propagated through the net (independent of its structure) to the net's output layer. The output layer generates an output pattern which is then compared to the target pattern.

Depending on the difference between output and target, an error value is computed. This output error indicates the net's learning effort, which can be controlled by the "imaginary supervisor". The greater the computed error value is, the more the weight values will be changed.

Two types of modes are used this system.

1. Administrator mode
2. User mode

Administrator

The administrator is given a user name and a password with which he/she can log in to the account. Password and user name verified with the one stored in the database. Once it is verified the administrator is given the following privileges.

- Training
- User management
- Record management
- User Management

Training

To train an image, we give the file name and id and call the train function in the train.java class.

Train(filename, startposition, x, y, id)

Then the pixel grabbing and conversion to 3D array takes place. The Red, Blue, Green values will be stored in the 3D array. These values will then be converted to binary and stored in an array. The binary values along with the id will be written to a file.

The control is then passed to bnp.java class. Here the following events take place

One ASCII value is converted to six binary values. Function used

readConversionFile(filename)

49 x6(multiplication factor)input neurons and 7 output neurons created. Function used

addNeuronLayer(number)

Then these neurons are connected using connectLayers() function.

We use two algorithms for training of the network

- Forward Propagation
- Backpropagation

Forward Propagation

Forward propagation is a supervised learning algorithm and describes the "flow of information" through a neural net from its input layer to its output layer.

The algorithm works as follows:

1. Set all weights to random values ranging from -1.0 to +1.0
2. Set an input pattern (binary values) to the neurons of the net's input layer
3. Activate each neuron of the following layer:
4. Multiply the weight values of the connections leading to this neuron with the output values of the preceding neurons.

5. Add up these values.
6. Pass the result to an activation function, which computes the output value of this neuron.
7. Repeat this until the output layer is reached
8. Compare the calculated output pattern to the desired target pattern and compute an error value
9. Change all weights by adding the error value to the (old) weight values
10. Go to step 2
11. The algorithm ends, if all output patterns match their target patterns

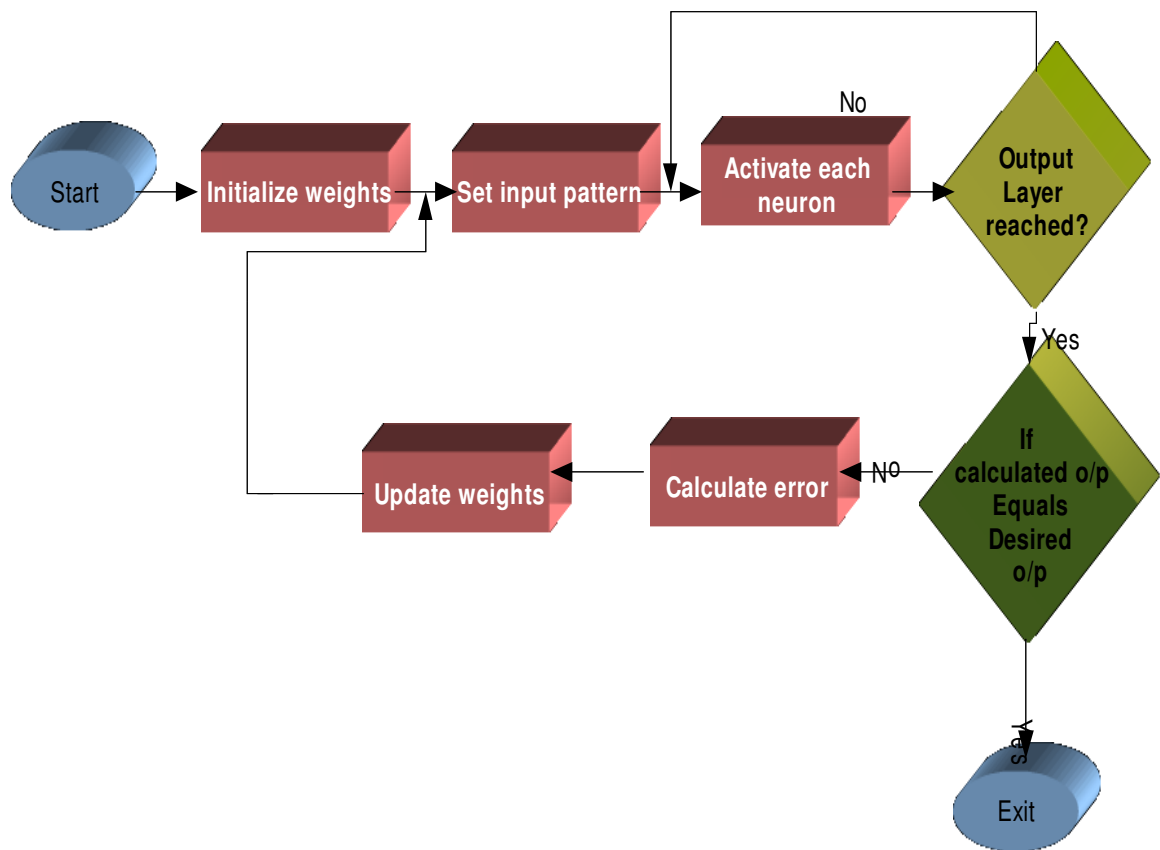


Fig 1.6

Example: Suppose you have the following 2-layered Perceptron: Patterns to be learned:

Input	Target
0 1	0
1 1	1

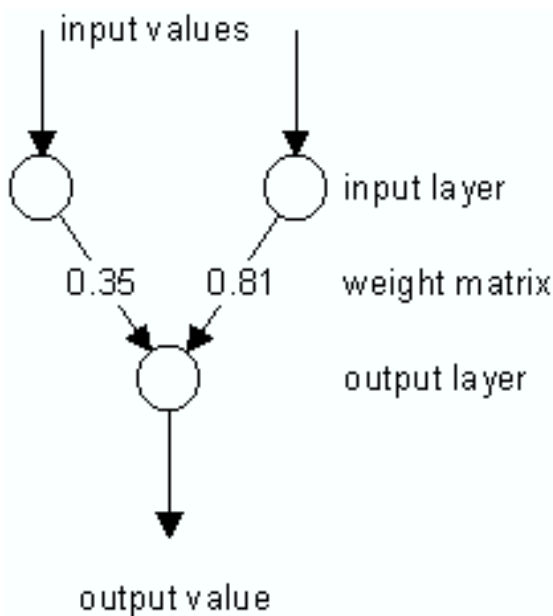


Fig 1.7

First, the weight values are set to random values (0.35 and 0.81).

The learning rate of the net is set to 0.25. Next, the values of the first input pattern (0 1) are set to the neurons of the input layer (the output of the input layer is the same as its input). The neurons in the following layer (only one neuron in the output layer) are activated:

Input 1 of output neuron : $0 * 0.35 = 0$

Input 2 of output neuron : $1 * 0.81 = 0.81$

Add the inputs: $0 + 0.81 = 0.81 (= \text{output})$

Compute an error value by subtracting output from target: $0 - 0.81 = \mathbf{-0.81}$

Value for changing weight 1 : $0.25 * 0 * (-0.81) = 0$ (0.25 = learning rate)

Value for changing weight 2: $0.25 * 1 * (-0.81) = -0.2025$

Change weight 1: $0.35 + 0 = 0.35$ (not changed)

Change weight 2: $0.81 + (-0.2025) = 0.6075$

Now that the weights are changed, the second input pattern (1 1) is set to the input layer's neurons and the activation of the output neuron is performed again, now with the new weight values:

Input 1 of output neuron: $1 * 0.35 = 0.35$

Input 2 of output neuron: $1 * 0.6075 = 0.6075$
 Add the inputs: $0.35 + 0.6075 = 0.9575$ (= output)
 Compute an error value by subtracting output from target: $1 - 0.9575 = \mathbf{0.0425}$
 Value for changing weight 1: $0.25 * 1 * 0.0425 = 0.010625$
 Value for changing weight 2: $0.25 * 1 * 0.0425 = 0.010625$
 Change weight 1: $0.35 + 0.010625 = 0.360625$

Change weight 2: $0.6075 + 0.010625 = 0.618125$

That was one learning step. Each input pattern had been propagated through the net and the weight values were changed.

The error of the net can now be calculated by adding up the squared values of the output errors of each pattern:

Compute the net error: $(-0.81) + (0.0425) = \mathbf{0.65790625}$

By performing this procedure repeatedly, this error value gets smaller and smaller.

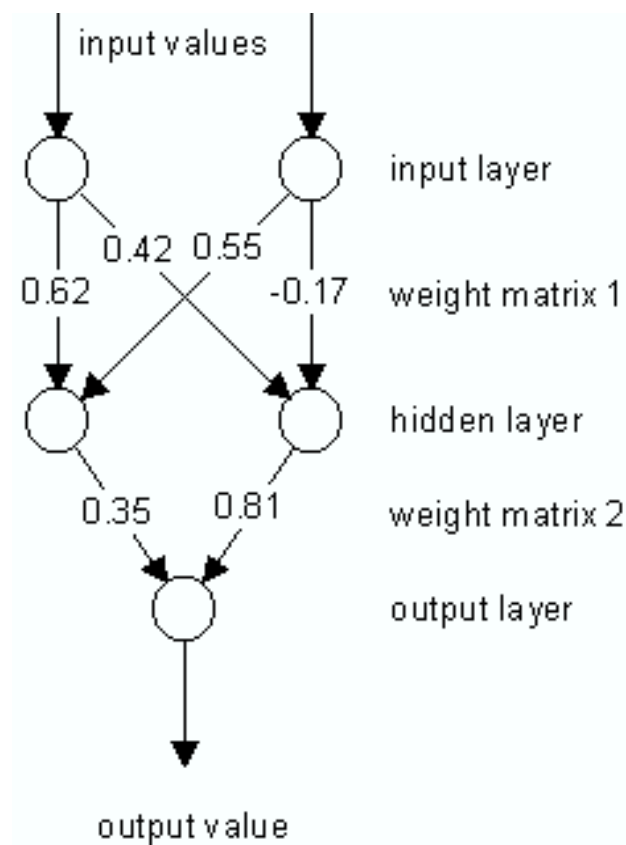


Fig 1.8

Backpropagation

Backpropagation is a supervised learning algorithm and is mainly used by Multi-Layer-Perceptrons to change the weights connected to the net's hidden neuron layer(s).

The backpropagation algorithm uses a computed output error to change the weight values in backward direction.

To get this net error, a forward propagation phase must have been done before. While propagating in forward direction, the neurons are being activated using the *sigmoid activation function*

The formula of **sigmoid activation** is:

$$f(x) = 1/(1+\exp(-x))$$

The algorithm works as follows:

1. Perform the forward propagation phase for an input pattern and calculate the output error
2. Change all weight values of each weight matrix using the formula

$$\text{weight}(\text{old}) + \text{learning rate} * \text{output error} * \text{output (neurons } i) * \text{output}(\text{neurons } i+1) \\ * (1 - \text{output}(\text{neurons } i+1))$$

3. Go to step 1
4. The algorithm ends, if all output patterns match their target patterns

Example:

Suppose you have the following 3-layered Multi-Layer-Perceptron:

Patterns to be learned:

Input	Target
-------	--------

0 1	0
-----	---

1 1	1
-----	---

First, the weight values are set to random values: 0.62, 0.42, 0.55, -0.17 for weight matrix 1 and 0.35, 0.81 for weight matrix 2.

The learning rate of the net is set to 0.25. Next, the values of the first input pattern (0 1) are set to the neurons of the input layer (the output of the input layer is the same as its input).

The neurons in the hidden layer are activated:

$$\text{Input of hidden neuron 1: } 0 * 0.62 + 1 * 0.55 = 0.55$$

$$\text{Input of hidden neuron 2: } 0 * 0.42 + 1 * (-0.17) = -0.17$$

$$\text{Output of hidden neuron 1: } 1 / (1 + \exp(-0.55)) = 0.634135591$$

$$\text{Output of hidden neuron 2: } 1 / (1 + \exp(+0.17)) = 0.457602059$$

The neurons in the output layer are activated:

$$\text{Input of output neuron: } 0.634135591 * 0.35 + 0.457602059 * 0.81 = 0.592605124$$

$$\text{Output of output neuron: } 1 / (1 + \exp(-0.592605124)) = 0.643962658$$

Compute an error value by subtracting output from target: $0 - 0.643962658 =$
0.643962658

Now that we got the output error, let's do the backpropagation.

We start with changing the weights in weight matrix 2:

$$\text{Value for changing weight 1: } 0.25 * (-0.643962658) * 0.634135591 * 0.643962658 * (1 - 0.643962658) = -0.023406638$$

$$\text{Value for changing weight 2: } 0.25 * (-0.643962658) * 0.457602059 * 0.643962658 * (1 - 0.643962658) = -0.016890593$$

$$\text{Change weight 1: } 0.35 + (-0.023406638) = 0.326593362$$

$$\text{Change weight 2: } 0.81 + (-0.016890593) = 0.793109407$$

Now we will change the weights in weight matrix 1:

$$\text{Value for changing weight 1 : } 0.25 * (-0.643962658) * 0 * 0.634135591 * (1 - 0.634135591) = 0$$

$$\text{Value for changing weight 2 : } 0.25 * (-0.643962658) * 0 * 0.457602059 * (1 - 0.457602059) = 0$$

$$\text{Value for changing weight 3: } 0.25 * (-0.643962658) * 1 * 0.634135591 * (1 - 0.634135591) = -0.037351064$$

Value for changing weight 4: $0.25 * (-0.643962658) * 1 * 0.457602059 * (1 - 0.457602059) = -0.039958271$

Change weight 1: $0.62 + 0 = 0.62$ (not changed)

Change weight 2: $0.42 + 0 = 0.42$ (not changed)

Change weight 3: $0.55 + (-0.037351064) = 0.512648936$

Change weight 4: $-0.17 + (-0.039958271) = -0.209958271$

The first input pattern had been propagated through the net.

The same procedure is used for the next input pattern, but then with the changed weight values.

After the forward and backward propagation of the second pattern, one learning step is complete and the net error can be calculated by adding up the squared output errors of each pattern.

By performing this procedure repeatedly, this error value gets smaller and smaller. The algorithm is successfully finished, if the net error is zero (perfect) or approximately zero.

BACK PROPAGATION NETWORKS

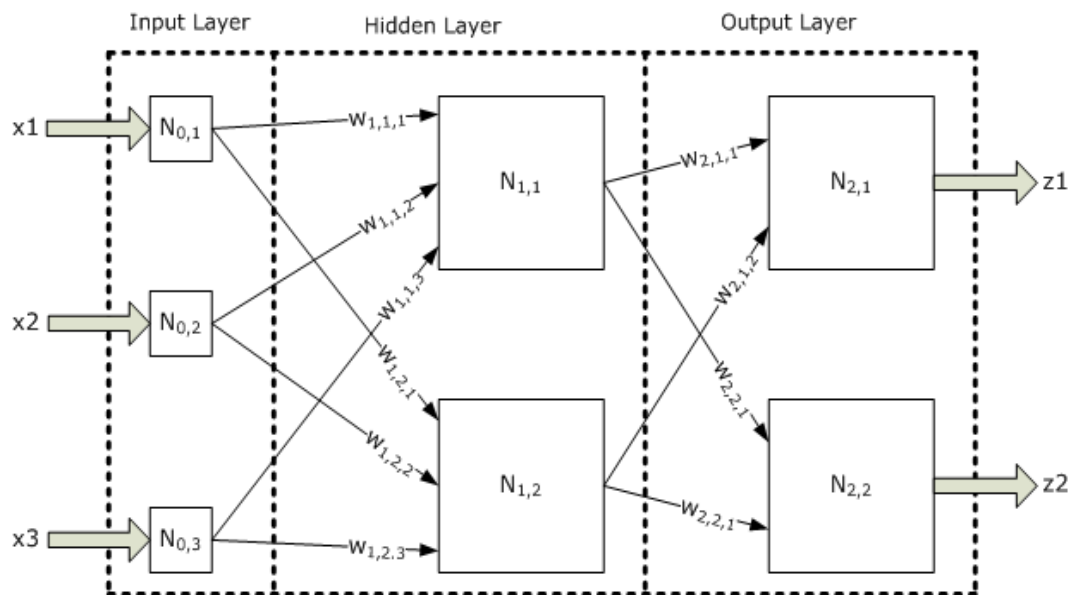


Fig 1.9

5. CONCLUSION

The project Face Recognition Using Neural Networks is used to identify a person from his face image. The flexibility of neural network is that it can identify a person by not just comparing the pictures but first extracting a pattern and when this pattern is provided as an input to the neural network it generates an output which is compared. So the data stored in a database is just a binary array than a whole image.

An automatic face recognition system is widely adopted in many applications such as building area security and Passport verification.

Neural Networks are being used in many applications. Their ability to learn by example is very attractive in environments where the business rules are either not well defined or are hard to enumerate and define.

5.1. Future Scope

The system “Face Recognition System” has been designed & developed flexibly according to the current requirements. The Administrator can add new profiles, edit profiles, compare the facial patterns and most of the facilities are also provided for the staff. So depending upon the requirements of the user in future the Face Recognition System can be updated in a much reliable manner.

This software has been developed on the basis of face images. So in future we can add the facilities like comparing palm prints. The scope future development is very large for this project.

The most widely used biometric technique is face recognition. Since face images have been used in forensic (criminal science) applications for several years, there is a wealth of information concerning the uniqueness of facial patterns. In fact, face

recognition can be used to replace the PIN in most security aspect e.g using face images instead of PIN in the smart card applications. An automatic face recognition

system is widely adopted in many applications such as building or area security and ATM machines.

Identification pattern systems play a critical role in modern society, in both criminal and civil applications. For example, criminal identification in public safety sectors is an integral part of any present day investigation. Similarly in civil applications such as credit card or personal identity fraud, face identification has become an essential part of the security process.

Another recent use of face recognition in a day-to-day setting has been the increasing reliance on biometrics in schools where face recognition and, to a lesser extent, iris scans are used to validate electronic registration, cashless catering, and library access. This practice is particularly widespread in U.K, where more than 3500 schools currently use such technology, though it is also starting to be adopted in some states in the US.

The science of face recognition can assert its standing amongst forensic sciences for many reasons, including the following:

- Has served all governments worldwide during the past years to provide accurate identification of criminals. No two faces have ever been found identical in many billions of human and automated computer comparisons. Face images are the very basis for criminal history foundation at every police agency.
- Remains the most commonly used forensic evidence worldwide—in most jurisdictions
- Face image examination cases match or outnumber all other forensic examination casework combined.
- Continues to expand as the premier method for identifying persons, with tens of thousands of persons added to image repositories daily in America alone—far outdistancing similar databases in growth.

REFERENCES

- [1] Philip D.Wasserman, *Neural Computing Theory and Practice*, Van Nostrand Reinhold,1989
- [2] Herbert Schildt.Java,*The Complete Reference*,TATA McGRAW HILL,2008
- [3] www.javareference.com
- [4] www.java.sun.com/docs/books/jls

APPENDIX A

Class: training.java

```
import java.io.*;

import java.util.*;

import javax.swing.*;

import java.awt.event.*;

import java.awt.*;

public class training extends JFrame implements ActionListener
{

    static String train_path;

    JLabel selectimg,imgid,ll,mess;

    JTextField selecttxt;

    JComboBox id;

    JButton train,back,browse;

    Font ft,ft1;

    Color clr ;

    JFrame f;

    public training()
    {

        super("Training");

        ft=new Font("Monotype Corsiva",10,26);

        ft1=new Font("Impact",10,15);

        selectimg=new JLabel("Select image :");

        selectimg.setForeground(Color.magenta);

        mess=new JLabel("Training in progress...");

        mess.setBounds(100,10,300,30);

        mess.setFont(ft1);

        imgid=new JLabel(" ID :");

        id=new JComboBox();

        imgid.setForeground(Color.magenta);
```

```
selecttxt=new JTextField(15);
browse=new JButton("Browse");
train=new JButton("Train");
back=new JButton("Back");
back.setForeground(Color.magenta);
selectimg.setBounds(20,50,100,30);
selecttxt.setBounds(125,50,200,30);
imgid.setBounds(20,125,100,30);
id.setBounds(125,125,200,30);
browse.setBounds(350,50,100,30);
back.setBounds(350,200,100,40);

train.setBounds(225,200,100,40);
selectimg.setFont(ft1);
imgid.setFont(ft1);
selecttxt.setFont(ft1);
JPanel x=new JPanel();
ll=new JLabel(new ImageIcon("image/iris_image_capture.PNG"));
x.setLayout(null);
train.setForeground(Color.magenta);
browse.setForeground(Color.magenta);
ll.add(selectimg);
ll.add(mess);
ll.add(selecttxt);
ll.add(browse);
ll.add(imgid);
ll.add(id);
ll.add(train);
ll.add(back);

x.setBackground(clr.green);
f=new JFrame("TRAINING");
f.add(ll);
```



```

        f.setBounds(250,200,500,300);
        f.setVisible(true);
        train.addActionListener(this);
        back.addActionListener(this);
        browse.addActionListener(this);
        Vector v=new RecordDB().getAllIDs();
        try{
            for(int i=0;i<v.size();i++)
            { id.addItem(v.get(i));
              }
        }catch(Exception e)
        { System.out.println("error "+e)      ;
          }
        id.addActionListener(this);
    }
    public static void main(String args[])
    {
        new training();
    }

```



```

public void actionPerformed(ActionEvent e)
{
    try{
        if (e.getSource()==back)
        {f.dispose();
        }
        if (e.getSource()==browse)
        {JFileChooser jf=new JFileChooser("./");
        jf.showOpenDialog(null);
        selecttxt.setText(jf.getSelectedFile().getAbsolutePath());
        System.out.println(selecttxt);
        }

        if (e.getSource()==train)
        {if (selecttxt.getText().length()>0)
        {
            FileInputStream fis=new
FileInputStream(selecttxt.getText());

            byte b[]=new byte[fis.available()];
            fis.read(b);

            FileOutputStream fos=new FileOutputStream(new
File("1.jpg"));

            fos.write(b);
            fos.close();

            train_path=id.getSelectedItem().toString()+"\\";
            for(int i =1;i<=10;i++)
            {File f=new File(train_path+"TrainningResult"+i+".txt");
            if(f.exists())
            f.delete();
            }

            File countfile=new File(train_path+"count.txt");

```

```

if(countfile.exists())
{ countfile.delete();
}
new
Train(train_path+"TrainningResult1.txt",230,190,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("1st cycle completed...");
new
Train(train_path+"TrainningResult2.txt",300,350,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("2nd cycle completed...");
new
Train(train_path+"TrainningResult3.txt",230,275,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("3rd cycle completed...");
new
Train(train_path+"TrainningResult4.txt",229,314,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("4th cycle completed...");
new
Train(train_path+"TrainningResult5.txt",295,190,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("5th cycle completed...");
new
Train(train_path+"TrainningResult6.txt",227,107,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("6th cycle completed...");
new
Train(train_path+"TrainningResult7.txt",362,274,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("7th cycle completed...");

```

```

new
Train(train_path+"TrainningResult8.txt",362,325,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("8th cycle completed...");
new
Train(train_path+"TrainningResult9.txt",387,303,Integer.parseInt
(id.getSelectedItem().toString()));
mess.setText("9th cycle completed...");
new
Train(train_path+"TrainningResult10.txt",216,303,Integer.parseInt
(id.getSelectedItem().toString()));

```

```

Retrieval.count=0;
identify temp=new identify(1);

```

```

identify.id2retrieve=Train.format(Integer.parseInt
(id.getSelectedItem().toString()));
identify.idn_path=train_path;
temp.identifyMe();
System.out.print( "Total count===>" +Retrieval.count);
File countf = new File(train_path+"count.txt");
FileOutputStream count= new FileOutputStream(countf);
String countstr=Integer.toString(Retrieval.count);
count.write(countstr.getBytes());
count.close();
System.out.print("\n*****
\n \n");
System.out.println("Trainning completed");
mess.setVisible(false);
JOptionPane.showMessageDialog(this,"Training successfully
completed");
}

```

```

        else{
            JOptionPane.showMessageDialog (null,"Please choose the
            image. ");
        }
    }

    }catch(Exception ex)
    {System.out.println("error "+ex)  ;
    ex.printStackTrace();
    }
}
}

```

Class: train.java

```

import javax.swing.*;
import java.awt.image.*;
import java.awt.*;
import java.awt.Toolkit;
import java.io.*;

public class Train extends JFrame
{
    int rsum,gsum,bsum;
    int i=0;

```

```

int j=0;
static String binaryID="";
public Train(String fname,int x,int y,int id) throws Exception
{
//Pixel Grabbing
Image img;
File f=new File("1.jpg");
System.out.println(f.getAbsolutePath());
img=Toolkit.getDefaultToolkit().getImage(f.getAbsolutePath());
int oneDim[]=new int[49*49];
int oneDim2[] = new int[10000];
PixelGrabber pg=new PixelGrabber(img,x,y,49,49,oneDim,0,49);
pg.grabPixels();

//Convert to 3D array

int rgb[][][] = new int[49][49][4];
for(int i=0;i<49;i++)
{
for (int j=0;j<49;j++)
{
rgb[i][j][0]=(oneDim[i*49+j] >> 24) & 0xFF; //Hex to Integer conversion
rgb[i][j][1]=(oneDim[i*49+j] >> 16) & 0xFF;
rgb[i][j][2]=(oneDim[i*49+j] >> 8) & 0xFF;
rgb[i][j][3]=(oneDim[i*49+j]) & 0xFF;
}
}
char bin[][] = new char [49][49];
for(i=0;i<49;i++)
{
for (j=0;j<49;j++)
{
rgb[i][j][1]=rgb[i][j][1]-80;

```

```

        if(((rgb[i][j][1]>47)&&(rgb[i][j][1]<58)) || ((rgb[i][j][1]>64)&&(rgb[i][j][1]<91))
        &&(rgb[i][j][1]>31&&bin[i][j]!=127))
        {
            bin[i][j]=(char)rgb[i][j][1];
        }
        else
        {
            bin[i][j]=48;
        }
        System.out.print(" "+bin[i][j]);
    }
    System.out.println();
}
binaryID=format(id);
String str="49\n49\n7\n";

for(i=0;i<49;i++)
{
    for(j=0;j<49;j++)
    {
        str=str+bin[i][j];
    }
    str=str+" "+binaryID+"\n";
}
System.out.println(str);
FileOutputStream fout=null;
File fxx=new File(fname);
System.out.println(fxx.getAbsolutePath());
if (!fxx.exists())
{
    fout=new FileOutputStream(fname);
    fout.write(str.getBytes());
}
fout.close();
new BPN(fname);
}

```

```

public static String format(int num)
{
String binary_num=Integer.toBinaryString(num);
String formatted_data="";
for(int i=0;i<7;i++)
{
if(i<binary_num.length())
    formatted_data+=binary_num.charAt(i);
else
    formatted_data="0"+formatted_data;
}
System.out.println(formatted_data);
return formatted_data;
}

```

```

private static String formatnumber(int num)
{
String s =Integer.toString(num);
String formatted_data="";
for(int i=0;i<2;i++)
{
if(i<s.length())
    formatted_data+=s.charAt(i);
else
    formatted_data="0"+formatted_data;
}
System.out.println(formatted_data);
return formatted_data;    }    }

```

Class: BackpropogationNet

```

import java.io.*;
import java.util.Enumeration;
import java.util.Vector;

```

```
public class BackpropagationNet extends NeuralNet
{
    Vector neuronLayerVector;
    NeuronLayer neuronLayerArray[];
    WeightMatrix weightMatrixArray[];
    Pattern inputPatternArray[];
    Pattern targetPatternArray[];
    String outputPatternArray[];
    double minimumError;
    double error;
    double accuracy;
    float layerOutputError[][];
    String conversionTable[][];
    int lastLayer;

    int lastMatrix;
    int lastPattern;
    int multiplier;

    public BackpropagationNet()
    {
        super.learningCycle = 0;
        super.maxLearningCycles = -1;
        minimumError = 0.00050000000000000000001D;
        super.learningRate = 0.25D;
        error = 1000D;
        multiplier = 0;
        accuracy = 0.20000000000000000001D;
        neuronLayerVector = new Vector();
        super.stopLearning = false;
        resetTime();
    }
    void addNeuronLayer(int i)
```



```
{ neuronLayerVector.addElement(new NeuronLayer(i * multiplier));  
}
```

```
void connectLayers()  
{  
    weightMatrixArray = new WeightMatrix[neuronLayerVector.size() - 1];  
    System.out.println("neuronLayerVector.size()"+neuronLayerVector.size());  
    neuronLayerArray = new NeuronLayer[neuronLayerVector.size()];  
    int i = 0;  
    for(Enumeration enumeration = neuronLayerVector.elements();  
        enumeration.hasMoreElements();)  
        { neuronLayerArray[i++] = (NeuronLayer)enumeration.nextElement();  
        }  
    neuronLayerVector = null;  
    for(int j = 0; j < weightMatrixArray.length; j++)  
        { weightMatrixArray[j] = new WeightMatrix(neuronLayerArray[j].size(),  
            neuronLayerArray[j + 1].size(), true);  
            weightMatrixArray[j].init();  
        }  
    lastLayer = neuronLayerArray.length - 1;  
    lastMatrix = weightMatrixArray.length - 1; } }
```

```
void connectLayersToRetrieve()  
{  
    weightMatrixArray = new WeightMatrix[neuronLayerVector.size() - 1];  
    neuronLayerArray = new NeuronLayer[neuronLayerVector.size()];  
    int i = 0;  
    for(Enumeration enumeration = neuronLayerVector.elements();  
        enumeration.hasMoreElements();)  
        { neuronLayerArray[i++] = (NeuronLayer)enumeration.nextElement();  
        }  
    neuronLayerVector = null;  
    for(int j = 0; j < weightMatrixArray.length; j++)
```

```

{
weightMatrixArray[j] = new WeightMatrix(neuronLayerArray[j].size(),
neuronLayerArray[j + 1].size(), true);
weightMatrixArray[j].init1();
}
lastLayer = neuronLayerArray.length - 1;
lastMatrix = weightMatrixArray.length - 1;
}

```

```

void setMinimumError(double d)
{ minimumError = d; }
double getMinimumError()
{ return minimumError; }

```

```

void setAccuracy(double d)
{ accuracy = d; }
double getAccuracy()
{ return accuracy; }

```

```

float[][] getWeightValues(int i)
{ return weightMatrixArray[i].getWeights(); }

```

```

float[] getNeuronOutputs(int i)
{ return neuronLayerArray[i].getOutput(); }

```

```

int getNumberOfLayers()
{ return neuronLayerArray.length; }

```

```

int getNumberOfNeurons(int i)
{ return neuronLayerArray[i].size(); }

```

```

int getNumberOfWeights()
{

```

```

int i = 0;
for(int j = 0; j <= lastMatrix; j++)
{ i += weightMatrixArray[j].size();          }
return i;
}

int getNumberOfWeights(int i)
{return weightMatrixArray[i].size();          }

int getNumberOfPatterns()
{return inputPatternArray.length;             }

String getInputPattern(int i)
{return inputPatternArray[i].getPatternString();    }

String getTargetPattern(int i)
{return targetPatternArray[i].getPatternString();    }

String getOutputPattern(int i)
{
String s = new String();
String s1 = new String();
float f = 0.0F;
for(int j = 0; j < layerOutputError[0].length; j++)
{
float f1 = targetPatternArray[i].getValue(j) - layerOutputError[i][j];
s1 += (double)f1 >= accuracy / 10D ? "1" : "0";
}
s = "";
for(int k = 0; k < s1.length(); k += multiplier)
{s += getAsciiValue(s1.substring(k, k + multiplier));          }
return s;
}

float getPatternError(int i)

```

```

{
float f = 0.0F;
for(int j = 0; j < layerOutputError[0].length; j++)
{ f += Math.abs(layerOutputError[i][j]);          }
return f;
}

double getError()
{ return error;          }

void learn()
{
if(error > minimumError && (super.learningCycle < super.maxLearningCycles ||
super.maxLearningCycles == -1))
{ super.learningCycle++;
for(int i = 0; i <= lastPattern; i++)
{
neuronLayerArray[0].setInput(inputPatternArray[i]);
for(int j = 1; j <= lastLayer; j++)
{
neuronLayerArray[j].computeInput(neuronLayerArray[j - 1], weightMatrixArray[j - 1]);
neuronLayerArray[j].computeOutput();
}
neuronLayerArray[lastLayer].computeLayerError(targetPatternArray[i]);
layerOutputError[i] = neuronLayerArray[lastLayer].getLayerError();
for(int k = lastMatrix; k >= 0; k--)
{
weightMatrixArray[k].changeWeights(neuronLayerArray[k].getOutput(),
neuronLayerArray[k + 1].getLayerError(), super.learningRate);
if(k > 0)
{
neuronLayerArray[k].computeLayerError(neuronLayerArray[k + 1],
weightMatrixArray[k]);
}
}
}
}

```

```

double d = 0.0D;
for(int l = 0; l < layerOutputError.length; l++)
{
    for(int i1 = 0; i1 < layerOutputError[0].length; i1++)
        { d += square(layerOutputError[l][i1]); }
}
error = Math.abs(d * 0.5D);
return;
} else
{
    super.stopLearning = true;
    return;
}
}

void retrieve()
{
    if(error > minimumError && (super.learningCycle < super.maxLearningCycles ||
    super.maxLearningCycles == -1))
    {
        super.learningCycle++;
        for(int i = 0; i <= lastPattern; i++)
        {
            neuronLayerArray[0].setInput(inputPatternArray[i]);
            for(int j = 1; j <= lastLayer; j++)
            {
                neuronLayerArray[j].computeInput(neuronLayerArray[j - 1], weightMatrixArray[j
- 1]);
                neuronLayerArray[j].computeOutput();
            }
        }
    } else
    {
        super.stopLearning = true;
        return;
    }
}

String recall(String s)

```

```

{
    Pattern pattern = new Pattern(s, conversionTable);
    float af[] = new float[targetPatternArray[0].size()];
    neuronLayerArray[0].setInput(pattern);
    for(int i = 1; i <= lastLayer; i++)
    {
        neuronLayerArray[i].computeInput(neuronLayerArray[i - 1], weightMatrixArray[i -
1]);
        neuronLayerArray[i].computeOutput();
    }
    af = neuronLayerArray[lastLayer].getOutput();
    String s2;
    String s1 = s2 = "";
    for(int j = 0; j < af.length; j++)
    {   s1 += (double)af[j] >= accuracy ? "1" : "0";   }
    for(int k = 0; k < s1.length(); k += multiplier)
    {   s2 += getAsciiValue(s1.substring(k, k + multiplier));   }
    return s2;   }

```

```

String recallaft(String s,float weight[][] )
{
    Pattern pattern = new Pattern(s, conversionTable);
    float af[] = new float[targetPatternArray[0].size()];
    neuronLayerArray[0].setInput(pattern);
    for(int i = 1; i <= lastLayer; i++)
    {
        neuronLayerArray[i].computeInput(neuronLayerArray[i - 1], weightMatrixArray[i - 1]);
        neuronLayerArray[i].computeOutput();
    }
    af = neuronLayerArray[lastLayer].getOutput();
    String s2;
    String s1 = s2 = "";
    for(int j = 0; j < af.length; j++)

```

```

    { s1 += (double)af[j] >= accuracy ? "1" : "0";    }
    for(int k = 0; k < s1.length(); k += multiplier)
    { s2 += getAsciiValue(s1.substring(k, k + multiplier));    }
    return s2;    }

    synchronized void readConversionFile(String s)
    {
        Object obj = null;
        try{
            DataInputStream datainputstream = new DataInputStream(new FileInputStream(s));
            int i = Integer.parseInt(datainputstream.readLine());
            conversionTable = new String[i][2];
            for(int j = 0; j < i; j++)
            {
                String s1 = datainputstream.readLine();
                conversionTable[j][0] = String.valueOf(s1.charAt(0));
                conversionTable[j][1] = s1.substring(1);
            }
            multiplier = conversionTable[0][1].length();
            return;
        }
        catch(FileNotFoundException _ex)
        {
            error(105);
            return;    }
        catch(IOException _ex)
        { error(104);    }
    }

    String getAsciiValue(String s)
    {
        int i = 0;
        int j = conversionTable.length;
        boolean flag = false;
        boolean flag1 = false;
        while(i < j)

```

```

    {
        int k = i + j >> 1;
        int l = s.compareTo(conversionTable[k][1]);
        if(l == 0)
        {
            return conversionTable[k][0];
        }
        if(l > 0)
        {
            i = k;
        }
    }
else
{
    j = k;
}
return "*";
}

synchronized void readPatternFile(String s)
{
    Object obj = null;
    try{
        DataInputStream datainputstream = new DataInputStream(new FileInputStream(s));
        try{

            int i = Integer.parseInt(datainputstream.readLine());
            int j = Integer.parseInt(datainputstream.readLine());
            if(j * multiplier != neuronLayerArray[0].size())
            {
                error(106);
            }
            int k = Integer.parseInt(datainputstream.readLine());
            if(k * multiplier != neuronLayerArray[lastLayer].size())
            {
                error(107);
            }
            inputPatternArray = new Pattern[i];
            targetPatternArray = new Pattern[i];
            outputPatternArray = new String[i];
            lastPattern = inputPatternArray.length - 1;
            layerOutputError = new float[lastPattern + 1][neuronLayerArray[lastLayer].size()];
            for(int l = 0; l < i; l++)
            {
                String s1 = datainputstream.readLine();
                if(s1 == null)

```



```

{   error(102);           }
else if(s1.length() != j + k + 1)
{   error(100);           }
else
    {

        inputPatternArray[l] = new Pattern(s1.substring(0, j), conversionTable);
        targetPatternArray[l] = new Pattern(s1.substring(j + 1), conversionTable);
        outputPatternArray[l] = new String();
    }
}   return;               }

catch(EOFException _ex)
{   error(102);           }
return;                   }

catch(FileNotFoundException _ex)
{   error(105);
return;                     }

catch(IOException _ex)
{   error(104);           }
}                           }

```

APPENDIX B

Screen shots

 Login Form

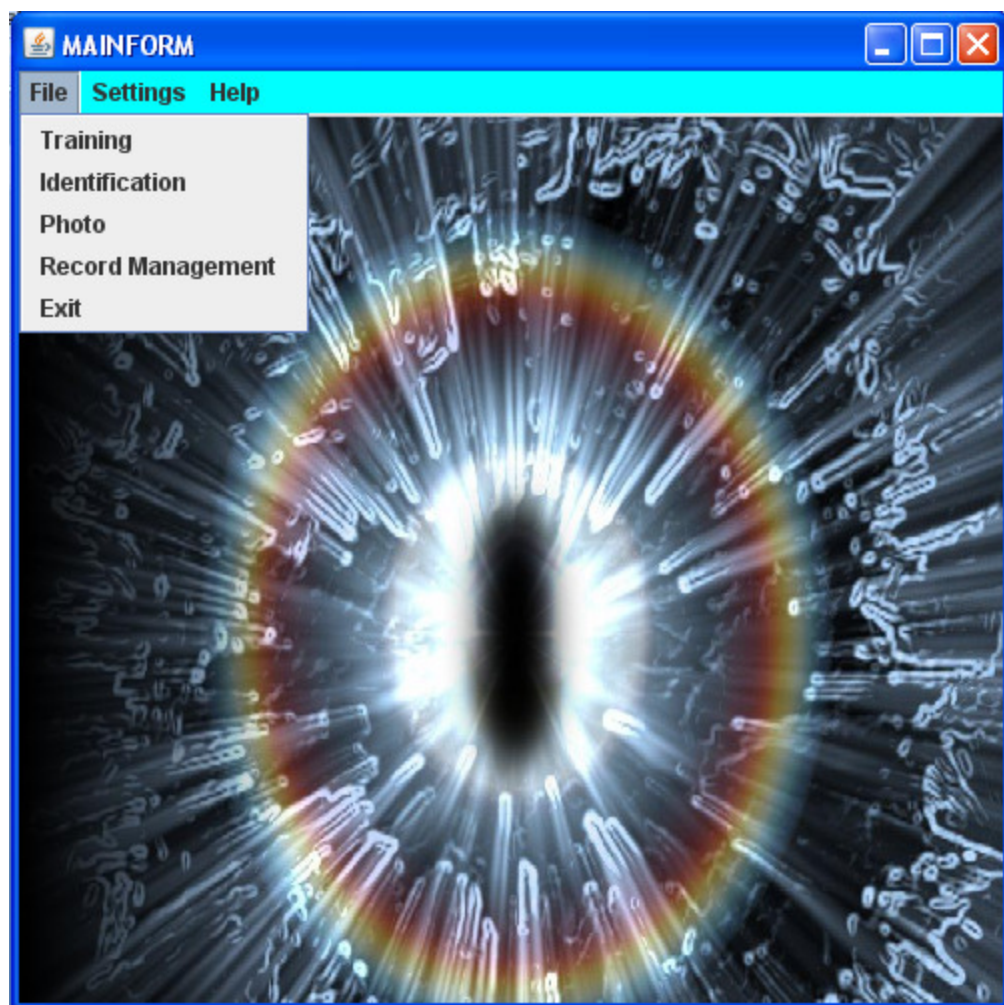
FACE RECOGNITION


UserName:




Password:

Message

 SUCCESSFUL LOGIN



 PRIVILLAGES



User Name :

SELECT

☐ Training.

☐ Identification.


☐ Record Management.

☐ Change Password.

☐ User Management.

save

back

 USER MANAGEMENT

Add

UserName :

Password :

Re-Enter :

☐ Training

☐ Identification


☐ Record Management

☐ Change Password

☐ User Management

Save

Back

 RECORD MANAGEMENT

AddEditDelete

Name :

Age :

Phone :

Email :


Sex :

☐ Male

☐ Female

Save


Back

 **TRAINING** [-] [Max] [X]

Training in progress...

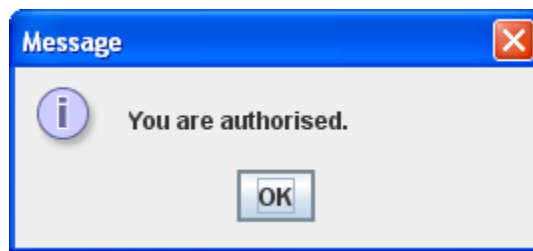
Select image:





ID: ▼

 **IDENTIFY** [-] [Max] [X]

Select image:

Enter ID



 **Personal Details Of ID No :1**   

Name :

Age :

Phone :

Email :

Sex : ☒ **Male** ☐ **Female**