CHAPTER 1

INTRODUCTION

1) Introduction

1.1) Problem definition

We are using face sensing technology along with distributed database concept which will make person identification process very fast and time efficient. Our system is embedding face sensing technologies with distributed database management concept.

We use a homogenous distributed database system which is a network of two or more SQL Server Databases that reside on one or more machines. Each database will have millions of face images of wanted or available persons. We allow replicated tables to be dispersed all over the country in a distributed database. Administrator does not have any authority to modify databases i.e. application has already prepared database. After loading image, system will perform face sensing techniques on image which includes face detection, face normalization and face recognition on respective images. Recognition includes comparing and matching faces with three databases in which available faces are distributed.

1.2) Scope

We will use a homogenous distributed database system which is a network of two or more SQL Server Databases that reside on one or more machines.

Each database will have millions of face images of wanted or available criminals. We will allow replicated tables to be dispersed all over the country in a distributed database.

After getting an image at server, it will apply face sensing technique based on Template Matching algorithm. It detects human face in different scales, various poses, different expressions, lighting conditions, and orientation. We will use parallel processing in which application will call each database at time for matching images and will receive reply from individual databases. Application displays valuable information about the person whose image has been identified or recognized with quick service.

1.3) Existing System

A Distributed Database for Bio-Molecular Images

- **What is implemented:**

The tremendous amount of information gathered from genomics will be dwarfed in the next decade by the knowledge to be gained from comprehensive, systematic studies of the properties and behaviors of all proteins and other biomolecules. High resolution imaging of molecules and cells will be critical for understanding complex systems such as the nervous system, whether it be for the localization of specific neuron types within a region of the central nervous system, the branching pattern of dendrite trees, or the localization of molecules at the subcellular level. We are developing sophisticated information technologies for collecting and interpreting the enormous volume of biological image data. A major outcome of the research will be a unique, fully operational, distributed digital library of bimolecular image data accessible to researchers around the world. Such searchable databases will make it possible to optimally understand and interpret the data, leading to a more complete and integrated understanding of cellular structure, function and regulation.

- **Idea taken in our Project:**

In our Human face sensing distributed database system we maintain distributed database of all face images with their information.

Police and Security Services Solution using Aurora's gallery Face Recognition Solution.

- **What is implemented:**

Most police forces these days have tens of thousands of digital photographs of people who appear to have committed a crime. It is not uncommon to find that a crime is committed by someone who is already known to the Force and whose photograph is on the Force's digitized database. However, searching through sometimes hundreds of thousands of photographs is so time-consuming manually as to be virtually impossible. Working with a number of police forces, Aurora have developed a system which makes a digital template of each image of the database. The 'target image' (i.e. the image of the person appearing to

commit a crime) can be swiftly digitized as well and then compared with all the other images on the database. Even with as many as 200,000 records, the search is remarkably fast, taking perhaps a second. Then, if the operator feels there is a likely match, there are a number of other facilities embedded in the software system available to the user to further enhance the likelihood of being able to re-open the case and seek a successful prosecution.
A number of UK police forces now use the Aurora system – and are achieving convictions as a result in cases that had 'gone cold'

- **Idea taken in our Project:**

  We maintain our digital images database as above, but for mining image we implement distributed system instead of centralized database.

1.4) Implemented System:

Our system is embedding face sensing techniques with distributed database management concept. We have used a distributed database system which is a network of two or more SQL Databases that reside on one or more machines. After getting an image at server, it will apply face sensing technique based on Template Matching algorithm we will use parallel processing in which application will call each database.

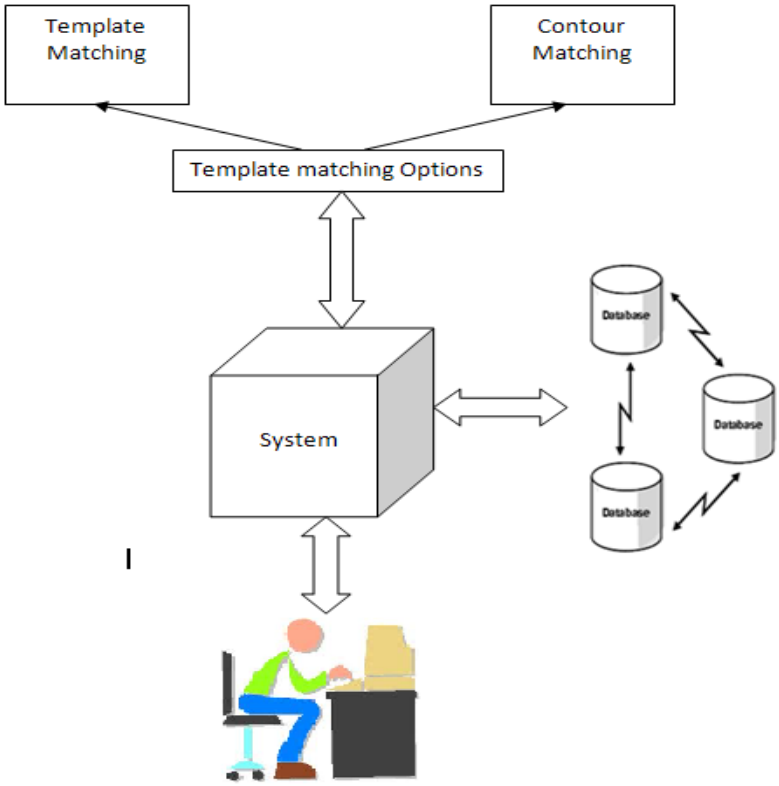
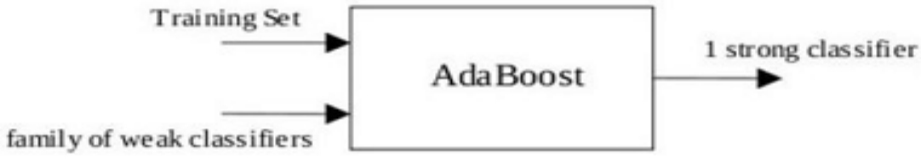


Figure 1.4: Proposed System for Human Face Sensing In DDB

Technology used:

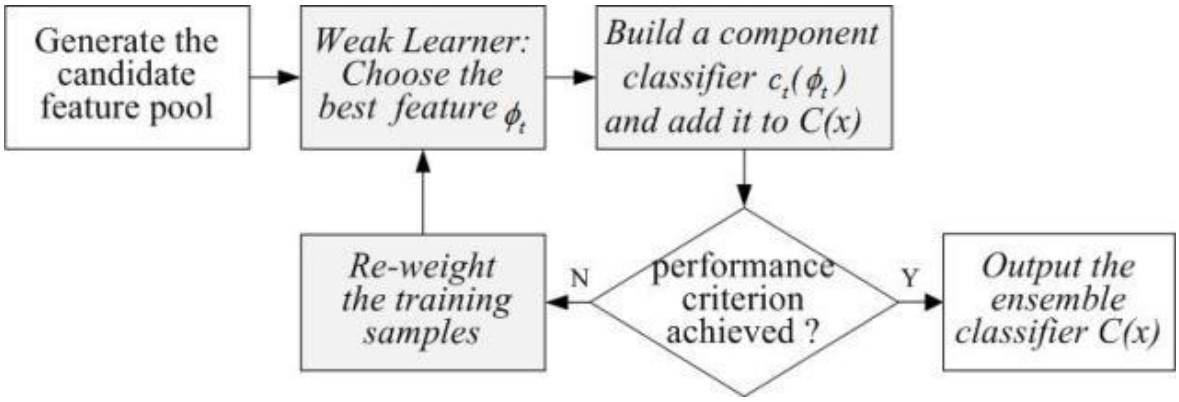
Face Detection Using Adaboost:

It is a well-known large margin learning algorithm that can select a small set of the most discriminative features (from a candidate feature pool), and combines them into an additive model



Basic scheme of AdaBoost

It involves three key modules, namely the weak learner, the component classifier and the re-weighting function.

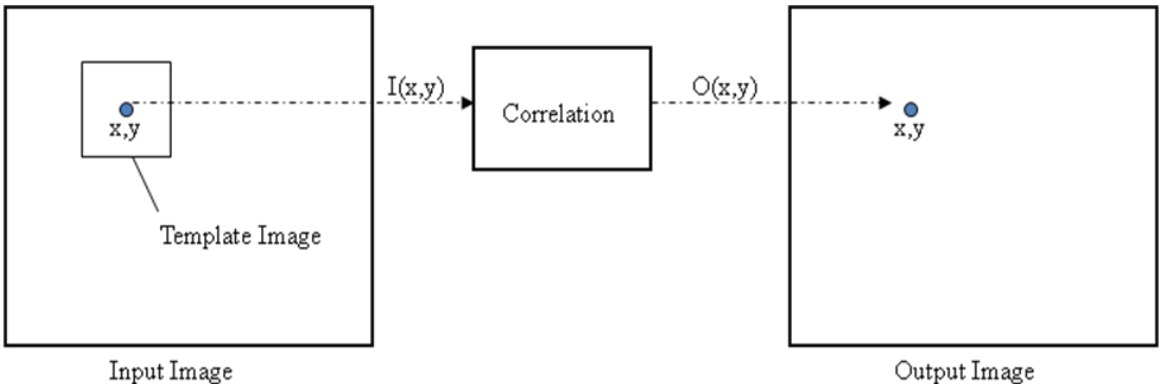


A genelized flowchart of Adaboost learning.

Template Matching:

- Template matching techniques compare portions of images against one another. Sample image may be used to recognize similar objects in source image.

- If standard deviation of the template image compared to the source image is small enough, template matching may be used. Templates are most often used to identify printed characters, numbers, and other small, simple objects.



Correlation:

- Correlation is a measure of the degree to which two variables agree, not necessary in actual value but in general behavior.
- The two variables are the corresponding pixel values in two images, template and source.

Correlation Formula

$$cor = \frac{\sum_{i=0}^{N-1}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=0}^{N-1}(x_i - \bar{x})^2 \cdot \sum_{i=0}^{N-1}(y_i - \bar{y})^2}}$$

x is the template gray level image
x is the average grey level in the template image
y is the source image section
y is the average grey level in the source image
N is the number of pixels in the section image
(N= template image size = columns * rows)
The value cor is between –1 and +1,
with larger values representing a stronger relationship between the two images.

Contour matching:

Contours are sequences of points defining a line/curve in an image. Contour matching can be used to classify image objects

Histogram Equalization: Histogram equalization is performed which enhances the contrast of images by transforming the values in an intensity image

```
cvEqualizeHist( img, out )
```

Noise Elimination: This is used to remove any noise (if present) from the image. This is done using Gaussian blurring.

```
cvSmooth(out, img1, CV_GAUSSIAN, 7, 7, 0, 0)
```

Normalization : This is used to compensate for any illumination variations or relative sizes between two sets of faces. This is done using pixel value normalization.

Formula:

Pixel value normalized image = (Blurred image pixel -Pixel Mean) / Standard deviation.

Contour generation:

```
CvSeq* contours = 0;
cvCvtColor( g_image, g_gray, CV_BGR2GRAY );
cvThreshold( g_gray, g_gray, g_thresh, 255, CV_THRESH_BINARY
);
cvFindContours( g_gray, g_storage, &contours );
cvZero( g_gray )
```

1.5) Assumptions and Constraint

- GUI is only in English.
- This system is working for Distributed database.
- Already well prepared distributed face image database.
- Maintenance problem.
- Application security.
- Environmental threats.

1.6) System Requirement

Software Interface:

- **Data Base Server:** SQL Server 2008, Operating System (windows XP).
- **Development End:** visual studio 2008, Operating System (any), openCV 2.1

Hardware Interface:

System with

- 1 GB RAM
- 80 GB HARD DISK
- AMD OR INTEL PROCESSOR

CHAPTER 2

SOFTWARE REQUIREMENT SPECIFICATION DESIGN

2) Software Requirement Specifications & Design

2.1) Functional Specification

Homogenous distributed database system which is a network of two or more SQL Server Databases that reside on one or more machines.

Each database has millions of face images of wanted or available criminals. We allow replicated tables to be dispersed all over the country in a distributed database.

After getting an image at server, it applies face sensing technique based on Template Matching algorithm. It detects human face in different scales, various poses, different expressions, lighting conditions, and orientation. We will use parallel processing in which application will call each database at time for matching images and will receive reply from individual databases. Application displays valuable information about the person whose image has been identified or recognized with quick service.

2.2) UML Diagrams

Use case diagram

Figure 2.2.1: Use-Case Diagram of Human Face Sensing In DDB

Use case specification

Actor: Admin
System
Use case:

1) Login into system:
   Precondition:
   For accessing the application, user (Admin) has to login with valid username and password
   Post Condition:
   Login into the system with valid username and password

2) Authenticate user:
   Precondition:
   Login request from user (Admin)
   Post Condition:
   System validate the user

3) Choose template matching method:
   Precondition:
   After login into the system, admin will get two options for (face sensing operation) template matching
   Post Condition:
   Required Template matching methods page will get display.

4) Preprocessing operation on image:
   Precondition:
   For preprocessing on image admin has to follow certain operation
   Post Condition:
   Expected output will be display

5) Search image in DDB
   Precondition:
   For Face sensing admin has to search input image in distributed database (DDB)
   Post condition:
   With parallel processing all three server's images is search
6) Face sensing:

Precondition:
Compare images with server's images
Post Condition:
With required template matching method compare input image with database image

7) Image found ,display information
   Precondition:
   After template matching(If face image is recognized) admin needs information regarding to the input images
   Post condition:
   System will give detail information regarding to the image

8) Image Not found ,display notification
   Precondition:
   After template matching (If face image is not recognized) admin needs notification regarding to the input images.

Sequence diagram

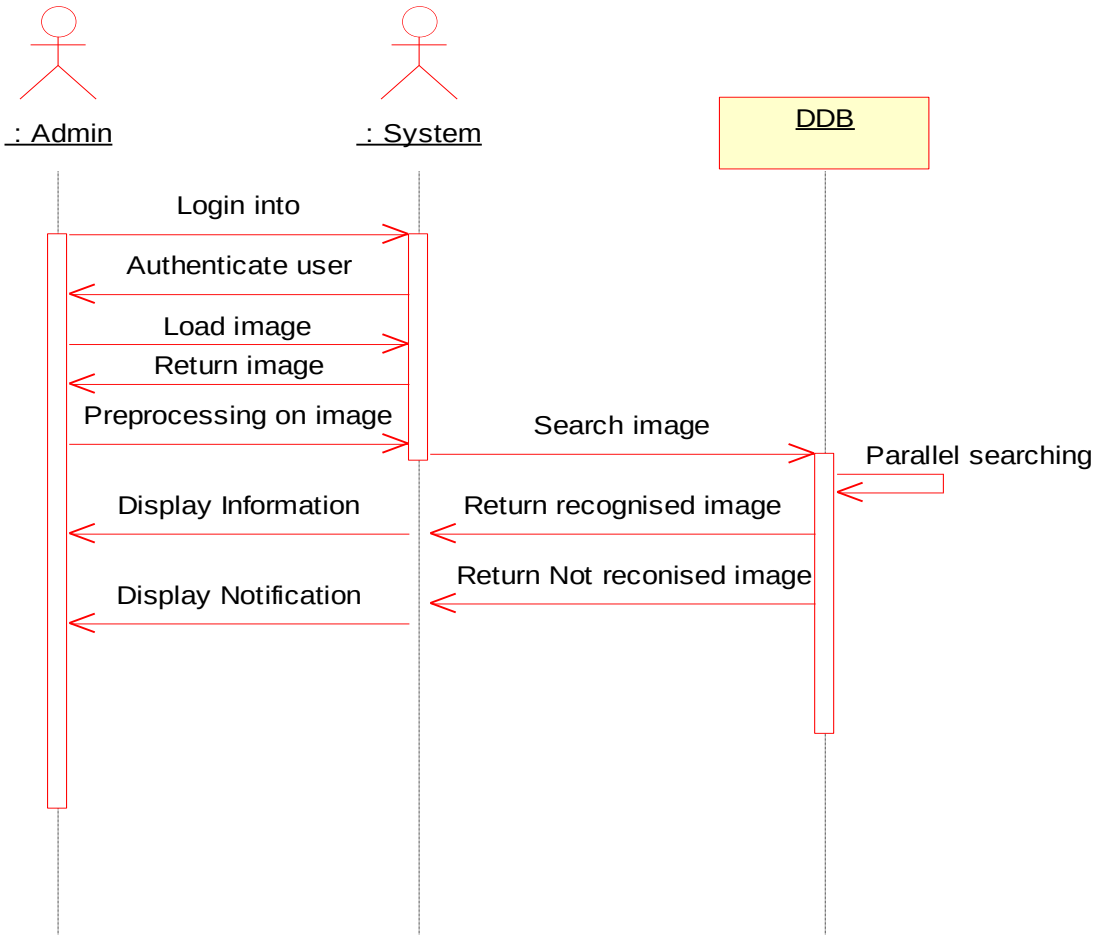


Figure 2.2.2: Sequence Diagram of Human Face Sensing In DDB

Sequence of Events:

1) Admin login into system for access the application.
2) If username and password is match system authenticate user
3) After log into system, user choose template matching method and load image for system
4) System returns required image back to the admin
5) Admin perform required preprocessing operation on face image.
6) System check preprocessed image in Distributed database

7) In DDB, with parallel Processing all images are compared.
8) Recognized image is sent back to the system.
9) System returns recognized image with detail information about the image.
10) Unrecognized image is sent back to the system.
11) System gives notification to the admin about unrecognized image.

Activity diagram:

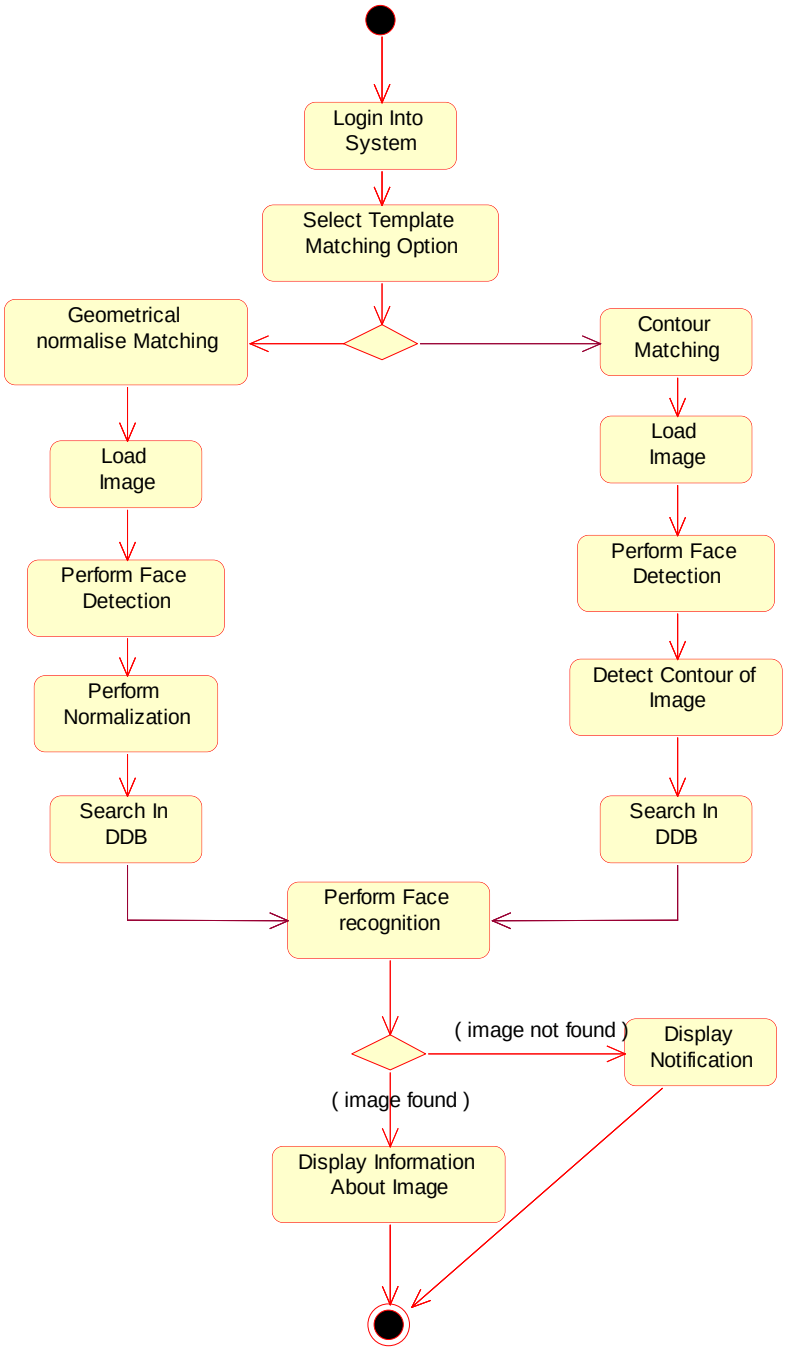


Figure 2.2.3: Activity Diagram of Human Face Sensing In DDB

Component Diagram:

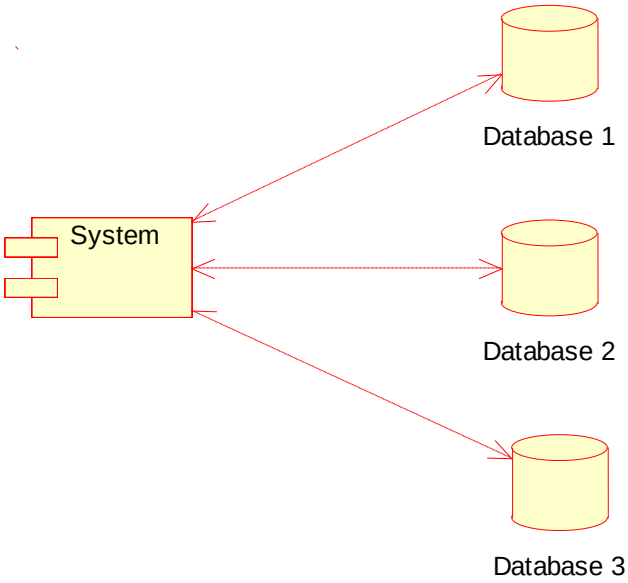


Figure 2.2.4: Component Diagram of Human Face Sensing In DDB

Deployment Diagram:

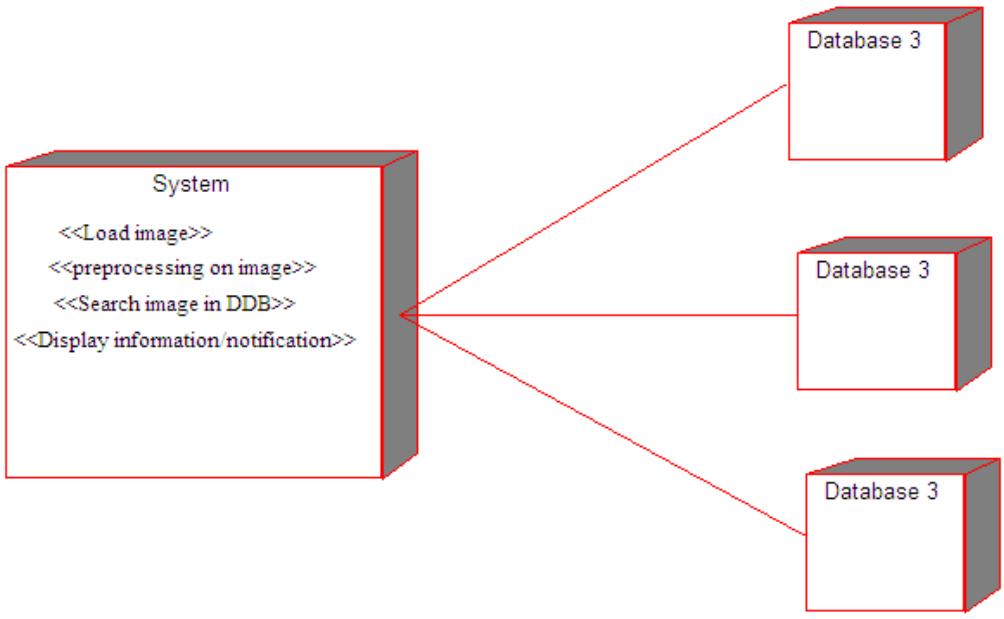


Figure 2.2.5: Deployment Diagram of Human Face Sensing In DDB

2.3) Architecture Diagram

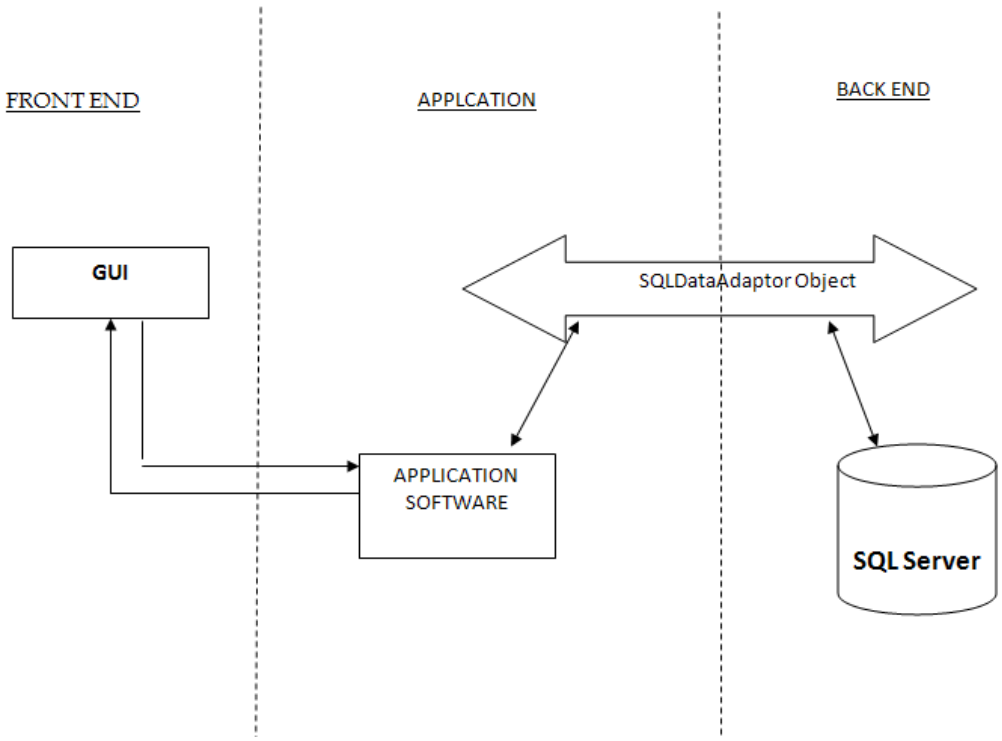


Figure 2.3: Architecture Diagram of Human Face Sensing In DDB

2.4) Data design\Schema

Fieldname	Data type
ImageID	Int(20)
ImageName	Varchar(50)
ImagePath	Varchar(50)
Imagepath2	Varchar (50)

Table 2.4.1 Data Design

2.5) Interface Design:

User Interfaces

The MFC application is the primary user interface. It consists of a page where the user can login for a service and his preferences in terms of availability response time and performance rating of the service. After submitting, the list of services that match the criteria are listed in order of preference. The most recently used services are also listed as options for the user.

Hardware Interfaces

Not applicable

Software Interfaces

The GUI will be developed using VC++ and MFC Application.

Communication Interfaces

The system's application server is accessed via ConnectionString and SqlDataAdapter drivers.

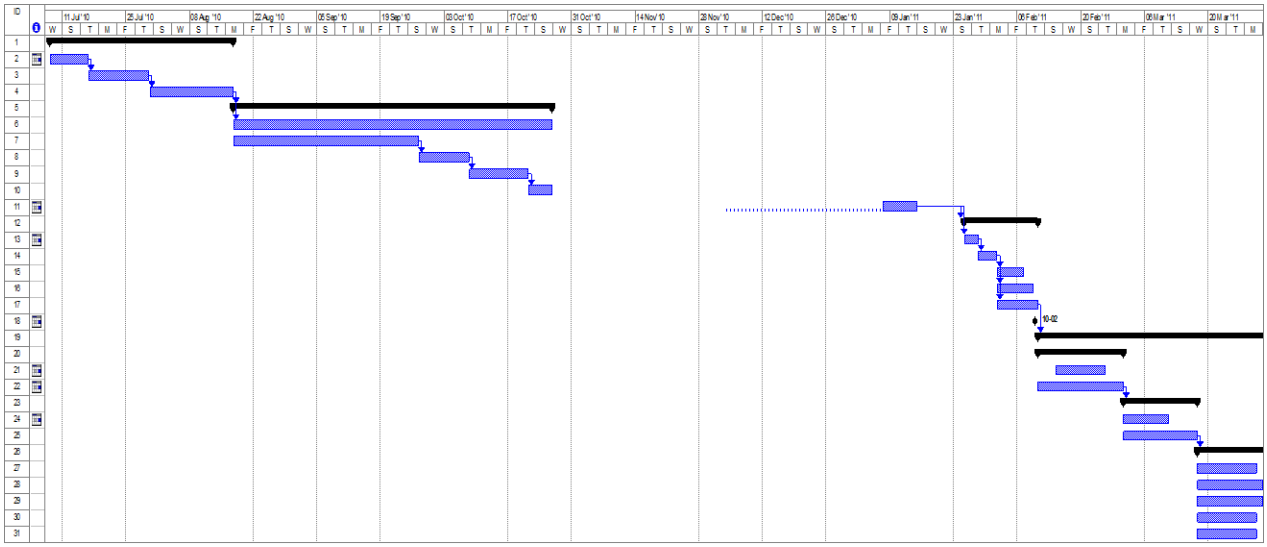
CHAPTER 3

PROJECT IMPLEMENTATION

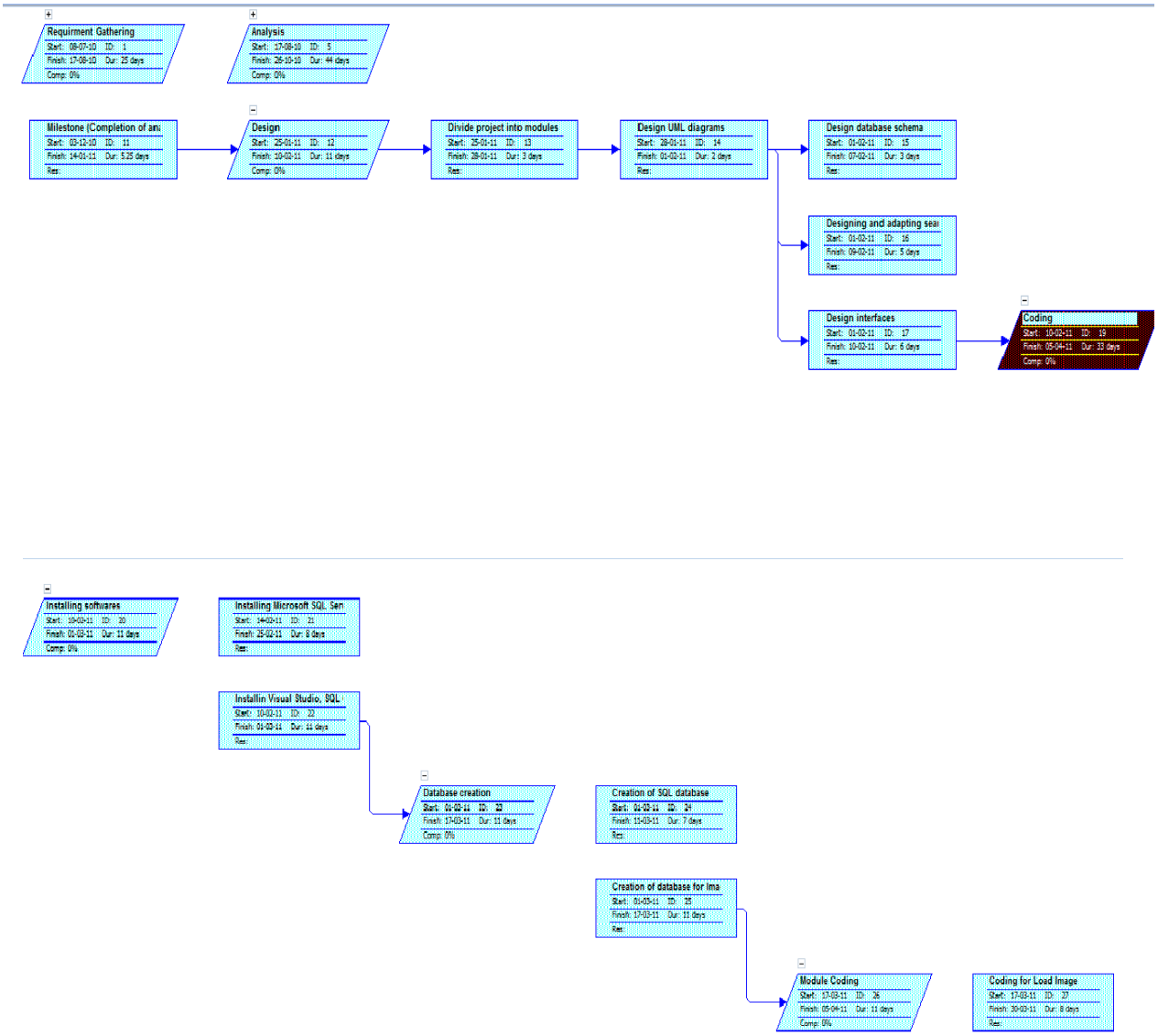
3) Implementation

3.1) Gantt chart:

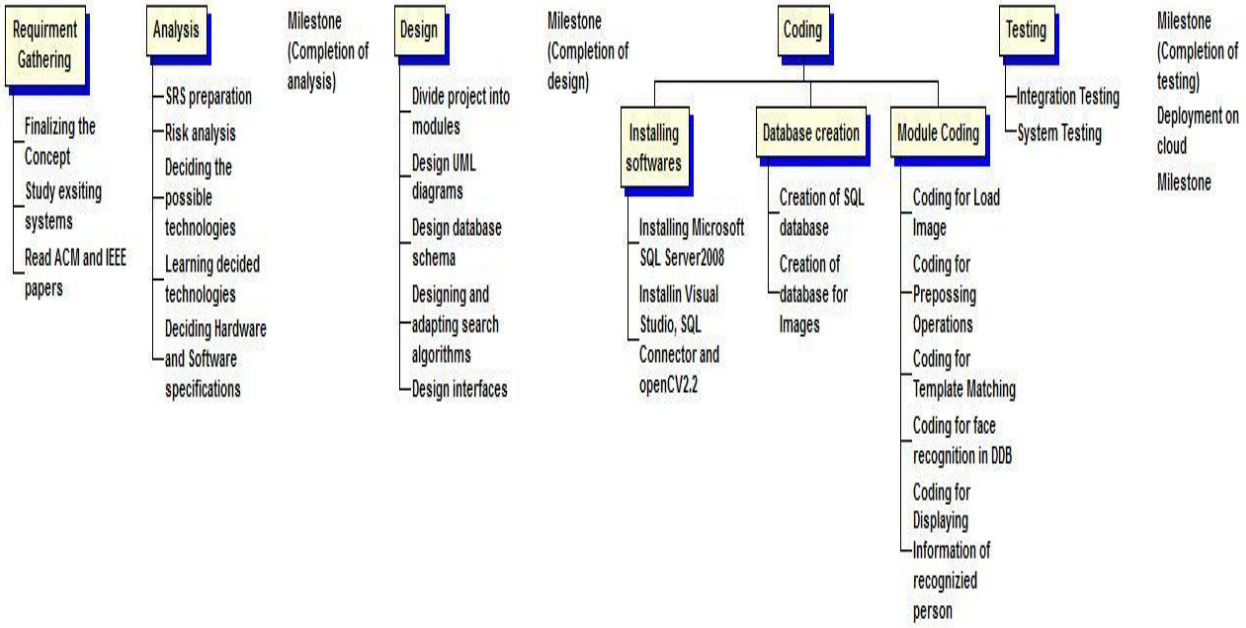
ID	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1	**Requirment Gathering**	**25 days**	**Thu 08-07-10**	**Tue 17-08-10**		
2	Finalizing the Concept	6 days	Thu 08-07-10	Fri 16-07-10		
3	Study exsiting systems	8 days	Fri 16-07-10	Thu 29-07-10	2	
4	Read ACM and IEEE papers	11 days	Fri 30-07-10	Tue 17-08-10	3	
5	**Analysis**	**44 days**	**Tue 17-08-10**	**Tue 26-10-10**	**4**	
6	SRS preparation	44 days	Tue 17-08-10	Tue 26-10-10	4	
7	Risk analysis	25 days	Tue 17-08-10	Mon 27-09-10		
8	Deciding the possible technologies	8 days	Mon 27-09-10	Fri 08-10-10	7	
9	Learning decided technologies	8 days	Fri 08-10-10	Thu 21-10-10	8	
10	Deciding Hardware and Software specifications	3 days	Thu 21-10-10	Tue 26-10-10	9	
11	Milestone (Completion of analysis)	5.25 days	Fri 03-12-10	Fri 14-01-11		
12	**Design**	**11 days**	**Tue 25-01-11**	**Thu 10-02-11**	**11**	
13	Divide project into modules	3 days	Tue 25-01-11	Fri 28-01-11	11	
14	Design UML diagrams	2 days	Fri 28-01-11	Tue 01-02-11	13	
15	Design database schema	3 days	Tue 01-02-11	Mon 07-02-11	14	
16	Designing and adapting search algorithms	5 days	Tue 01-02-11	Wed 09-02-11	14	
17	Design interfaces	6 days	Tue 01-02-11	Thu 10-02-11	14	
18	Milestone (Completion of design)	0 days	Thu 10-02-11	Thu 10-02-11		
19	**Coding**	**33 days**	**Thu 10-02-11**	**Tue 05-04-11**	**17**	
20	**Installing softwares**	**11 days**	**Thu 10-02-11**	**Tue 01-03-11**		
21	Installing Microsoft SQL Server2008	8 days	Mon 14-02-11	Fri 25-02-11		
22	Install in Visual Studio, SQL Connector and openCV2.2	11 days	Thu 10-02-11	Tue 01-03-11		
23	**Database creation**	**11 days**	**Tue 01-03-11**	**Thu 17-03-11**	**22**	
24	Creation of SQL database	7 days	Tue 01-03-11	Fri 11-03-11		
25	Creation of database for Images	11 days	Tue 01-03-11	Thu 17-03-11		
26	**Module Coding**	**11 days**	**Thu 17-03-11**	**Tue 05-04-11**	**25**	
27	Coding for Load Image	8 days	Thu 17-03-11	Wed 30-03-11		
28	Coding for Prepossing Operations	9 days	Thu 17-03-11	Thu 31-03-11		
29	Coding for Template Matching	11 days	Thu 17-03-11	Tue 05-04-11		
30	Coding for face recognition in DDB	8 days	Thu 17-03-11	Wed 30-03-11		
31	Coding for Displaying Information of recognizied person	8 days	Thu 17-03-11	Wed 30-03-11		



Network Diagram:



3.2) Work Breakdown Structure



3.3) code with reference to design –

Face recognition using template matching:

validationnewDlg.cpp:

```
#include "stdafx.h"
#include "validationnew.h"
#include "validationnewDlg.h"
#include "NewdialogDlg.h"
#include "SecondDlg.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif

class CAboutDlg : public CDialog
{
public:
        CAboutDlg();
enum { IDD = IDD_ABOUTBOX };
protected:
        virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
protected:
        DECLARE_MESSAGE_MAP()};
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{ }
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{ CDialog::DoDataExchange(pDX); }
BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
END_MESSAGE_MAP()
// CvalidationnewDlg dialog
CvalidationnewDlg::CvalidationnewDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CvalidationnewDlg::IDD, pParent)
        , m_username2(_T(""))
        , m_password2(_T(""))
{  m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME); }
void CvalidationnewDlg::DoDataExchange(CDataExchange* pDX)
{       CDialog::DoDataExchange(pDX);
        DDX_Text(pDX, IDC_EDIT4, m_username2);
        DDX_Text(pDX, IDC_EDIT5, m_password2);
}
BEGIN_MESSAGE_MAP(CvalidationnewDlg, CDialog)
        ON_WM_SYSCOMMAND()
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
        ON_BN_CLICKED(IDOK, &CvalidationnewDlg::OnBnClickedOk)
        ON_EN_CHANGE(IDC_EDIT5, &CvalidationnewDlg::OnEnChangeEdit5)
```

```
        ON_EN_CHANGE(IDC_EDIT4, &CvalidationnewDlg::OnEnChangeEdit4)
END_MESSAGE_MAP()

BOOL CvalidationnewDlg::OnInitDialog()
{   CDialog::OnInitDialog();
        ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
        ASSERT(IDM_ABOUTBOX < 0xF000);

        CMenu* pSysMenu = GetSystemMenu(FALSE);
        if (pSysMenu != NULL)
        {
                CString strAboutMenu;
                strAboutMenu.LoadString(IDS_ABOUTBOX);
                if (!strAboutMenu.IsEmpty())
                {
                        pSysMenu->AppendMenu(MF_SEPARATOR);
                        pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX,
strAboutMenu);  } }
        SetIcon(m_hIcon, TRUE);                 // Set big icon
        SetIcon(m_hIcon, FALSE);            // Set small icon

        return TRUE;  // return TRUE  unless you set the focus to a control
}
void CvalidationnewDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
        }
        else
        {
                CDialog::OnSysCommand(nID, lParam);
        } }
void CvalidationnewDlg::OnPaint()
{
        if (IsIconic())
        {
                CPaintDC dc(this); // device context for painting
                SendMessage(WM_ICONERASEBKGND,
                reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;
```

```
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        } }

        HCURSOR CvalidationnewDlg::OnQueryDragIcon()
        {
        return static_cast<HCURSOR>(m_hIcon);
        }
        void CvalidationnewDlg::OnEnChangeEdit1()
        { }

void CvalidationnewDlg::OnBnClickedOk()
{       OnOK();
        if(m_username2=="abc" && m_password2=="abc")
        {
    MessageBox("Validation sucessed");
         CNewdialogDlg Dlg;
         Dlg.DoModal();
        }
        else
        {
    MessageBox("Checked your Username and password");
         CvalidationnewDlg Dlg1;
         Dlg1.DoModal(); }}

void CvalidationnewDlg::OnEnChangeEdit5()
{ }
void CvalidationnewDlg::OnEnChangeEdit4()
{ }
```

DialogDlg.cpp:

```
#include "stdafx.h"
#include "validationnew.h"
#include "DialogDlg.h"
#include "cv.h"
#include "cvaux.h"
#include "highgui.h"
#include <cstdlib>
#include <stdio.h>
#include <iostream>
#include <String>
#include <string>
#include <cmath>
```

```
#include<sstream>
#include<fstream>
#using <mscorlib.dll>
#using <System.dll>
#using <System.Data.dll>
#using <System.Xml.dll>
using namespace std;
using namespace System::IO;
string intstr (int b)
{
        stringstream s ;
        s << b ;
        string g ;
        s >> g ;
        return g ;
};
string double2str (double a) {
        stringstream ss ;
        ss << a ;
        string f ;
        ss >> f ;
        return f ;

}
using namespace System;
using namespace System::Data;
using namespace System::Data::SqlClient;
typedef unsigned char uchar;
IMPLEMENT_DYNAMIC(CDialogDlg, CDialog)
CDialogDlg::CDialogDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CDialogDlg::IDD, pParent)
{ }
CDialogDlg::~CDialogDlg()
{ }
void CDialogDlg::DoDataExchange(CDataExchange* pDX)
{  CDialog::DoDataExchange(pDX);  }


BEGIN_MESSAGE_MAP(CDialogDlg, CDialog)
        ON_BN_CLICKED(IDC_BUTTON1, &CDialogDlg::OnBnClickedButton1)
        ON_BN_CLICKED(IDC_BUTTON2, &CDialogDlg::OnBnClickedButton2)
        ON_BN_CLICKED(IDC_BUTTON3, &CDialogDlg::OnBnClickedButton3)
        ON_STN_CLICKED(IDC_STATIC1, &CDialogDlg::OnStnClickedStatic1)
END_MESSAGE_MAP()
void CDialogDlg::OnBnClickedButton2()
{   CFileDialog dlg(TRUE, _T("*.bmp"), NULL,
   OFN_FILEMUSTEXIST|OFN_PATHMUSTEXIST|OFN_HIDEREADONLY,
```

```
    _T("image files (*.bmp; *.jpg) |*.bmp;*.jpg|All Files (*.*)|*.*||"),NULL);
dlg.m_ofn.lpstrTitle= _T("Open Image");
if (dlg.DoModal() == IDOK) {
CString path= dlg.GetPathName();  // contain the selected filename
IplImage *image;                  // This is image pointer
image= cvLoadImage(path);         // load the image
cvNamedWindow("Original Image1",CV_WINDOW_NORMAL);
cvShowImage("Original Image1",image); // display it
cvSaveImage("C:/pixel/images/lena1.jpg",image);  } }
void CDialogDlg::OnBnClickedButton3()
{
        IplImage* img,*img2;
        img = cvLoadImage("C:/pixel/images/lena1.jpg",1 );
        CvMemStorage* storage = cvCreateMemStorage(0);
        CvHaarClassifierCascade* cascade = (CvHaarClassifierCascade*)cvLoad(
"C:/Documents and Settings/Siddhant/My Documents/Visual Studio
2008/Projects/validationnew/haarcascade_frontalface_alt2.xml" );
        double scale = 1.3;
        // Detect objects
        cvClearMemStorage( storage );
        CvSeq* objects = cvHaarDetectObjects( img, cascade, storage, 1.1, 2, 0, cvSize( 40,
        50 ));
        CvRect* r;
        for( int i = 0; i < (objects ? objects->total : 0 ); i++ ){
                r = ( CvRect* )cvGetSeqElem( objects, i );
                CvPoint pt1 = { r->x, r->y };
        CvPoint pt2 = { r->x + r->width, r->y + r->height };
        cvRectangle( img, pt1, pt2, CV_RGB(255,0,0), 2,2, 0 );
        int a = (r->x), b=(r->y), c=((r->x + r->width)/1.1), d=((r->y + r->height)/1.3);
        string f = intstr(i);
        f +=".jpg";
        cvSetImageROI(img,cvRect(a,b,c,d));
        img2=cvCreateImage(cvGetSize(img), img->depth,img->nChannels);
        cvCopyImage(img,img2,NULL);
        char other_string2[100];

        strcpy(other_string2,"C:/pixel/images/");
        strcat(other_string2,f.c_str());
        cvSaveImage(other_string2,img2);
        cvResetImageROI(img);
       }
        cvNamedWindow("Output1",CV_WINDOW_NORMAL );
        cvShowImage("Output1", img );
        cvSaveImage("Res.jpg",img);
        cvWaitKey();
        cvReleaseImage( &img ); }
void CDialogDlg::OnBnClickedButton1()
```

```
{       IplImage *img = cvLoadImage("C:/pixel/images/0.jpg", 0);
        IplImage* out = cvCreateImage( cvGetSize(img), IPL_DEPTH_8U, 1 );
        cvEqualizeHist( img, out );
        cvNamedWindow( "histogram", CV_WINDOW_NORMAL) ;
        cvShowImage( "histogram", out );
        cvSaveImage("C:/project/seperate/hist.jpg",out);
        IplImage* img1 = cvCreateImage( cvGetSize(out), IPL_DEPTH_8U, 1 );
        cvSmooth(out, img1, CV_GAUSSIAN, 7, 7, 0, 0);
        cvNamedWindow( "blur", CV_WINDOW_NORMAL) ;
        cvShowImage( "blur", img1 );
        cvSaveImage("C:/pixel/images/blur.jpg",img1);
        IplImage *image;
        int height, width;
        uchar** ImageData;
        uchar* data;
        double mean, std_dev;
        cvNamedWindow("A1", CV_WINDOW_NORMAL);
        image = cvLoadImage("C:/pixel/images/blur.jpg",1);

        //Calculate the height and width of the image
        height = image->height;
        width = image->width;
        cout << "This image has " << width << " by " << height << " pixels.";
        cvGetRawData(image, (uchar**)&data);
        ImageData = new uchar* [height];
        for(int i = 0; i < height; i++)
        { ImageData[i] = new uchar [width];}
        //Calculate the mean of the image
        double total = 0;
        for (int i = 0; i < height; i++)
        {
        for (int j = 0; j < width; j++)
        {       total += data[height * j+i];
                //total += data[width*image->width + height];
        } }
        mean = (total/ (height * width));
        cout << "Mean: " << mean << endl;
        double var = 0;
        for (int a = 0; a < height; a++)
        {
        for (int b = 0; b < width; b++)
        {
        var += ((ImageData[a][b] - mean) * (ImageData[a][b] - mean));}}
        var /= (height * width);
        std_dev = sqrt(var);
        cout << "Standard Deviation: " << std_dev << endl;
//pixel based normalition
```

```
for (int a = 0; a <image->height;)
{
for (int b = 0; b <image->width;)
{
double sum1=data[image->height * b+a];
double sum11=sum1-(mean);
double sum111=sum11/std_dev;
ImageData[a][b] = sum111;
b++; }
a++; }
cvNamedWindow( "A1", CV_WINDOW_NORMAL) ;
cvShowImage( "A1", image );
cvSaveImage("C:/pixel/images/norm.jpg",image);
cvReleaseImage(&image);
IplImage*       g_image = NULL;
IplImage*       g_gray = NULL;
int             g_thresh = 165;
CvMemStorage*       g_storage = NULL;
g_image = cvLoadImage( "C:/pixel/images/norm.jpg" );
if( g_storage == NULL )
{
g_gray = cvCreateImage( cvGetSize( g_image ), 8, 1 );
g_storage = cvCreateMemStorage(0);
}
else
{
cvClearMemStorage( g_storage );
}
CvSeq* contours = 0;
cvCvtColor( g_image, g_gray, CV_BGR2GRAY );
cvThreshold( g_gray, g_gray, g_thresh, 255, CV_THRESH_BINARY );
cvFindContours( g_gray, g_storage, &contours );
cvZero( g_gray );
if( contours )
{  cvDrawContours(g_gray,contours,cvScalarAll(255),cvScalarAll(255),100 ); }
cvShowImage( "Contours", g_gray );
cvNamedWindow( "Contours",CV_WINDOW_NORMAL) ;
cvSaveImage("C:/pixel/contour/final.jpg",g_gray);
IplImage* in;
IplImage*image1;
IplImage*image2;
FILE* InputFILE;
int i;
CvSize refSize;
CvSize tarSize;
IplImage* demo;
in=cvLoadImage("C:/pixel/contour/final.jpg",0);
```

```
if(in->width >= 92 || in->height >= 112
{  refSize.width =92; // visualisation image is
   refSize.height =112; // 256x256 pixels }
   image1=cvCreateImage(refSize,in->depth,in->nChannels);
   cvResize(in,image1,CV_INTER_AREA);
SqlConnection* cn = new SqlConnection();
DataSet *ImageDataSet = new DataSet();
SqlDataAdapter *da;
SqlCommandBuilder *cmdBuilder;
cn->ConnectionString="Server=SIDDHANT
2529B6\\SQLEXPRESS;Database=Test;Trusted_Connection=Yes;";
SqlCommand *cmd = new SqlCommand("SELECT * FROM links ", cn);
cn->Open();
SqlDataReader *dr  = cmd->ExecuteReader();
dr->Read();
dr->Close();
string str;
da = new SqlDataAdapter("select (ImagePath2 + ImageName) As Image from links", cn);
cmdBuilder = new SqlCommandBuilder(da);
da->Fill(ImageDataSet, "links");
int count1 = ImageDataSet->Tables->Item["links"]->Rows->Count ;
/*second conn*/
SqlConnection* cn1 = new SqlConnection();
DataSet *ImageDataSet1 = new DataSet();
SqlDataAdapter *da1;
SqlCommandBuilder *cmdBuilder1;
cn1->ConnectionString="Server=SIDDHANT-
2529B6\\MSSQLEXPRESS;Database=Test1;Trusted_Connection=Yes;";
SqlCommand *cmd1 = new SqlCommand("SELECT * FROM links1 ", cn1);
cn1->Open();
SqlDataReader *dr1  = cmd1->ExecuteReader();
dr1->Read();
dr1->Close();
da1 = new SqlDataAdapter("select (ImagePath2 + ImageName) As Image from links1",
cn1);
cmdBuilder1 = new SqlCommandBuilder(da1);
da1->Fill(ImageDataSet1, "links1");
int count2 = ImageDataSet1->Tables->Item["links1"]->Rows->Count ;
//third conection
SqlConnection* cn2 = new SqlConnection();
DataSet *ImageDataSet2 = new DataSet();
SqlDataAdapter *da2;
SqlCommandBuilder *cmdBuilder2;
cn2->ConnectionString="Server=SIDDHANT-
2529B6\\MS1SQLEXPRESS;Database=Test2;Trusted_Connection=Yes;";
SqlCommand *cmd2 = new SqlCommand("SELECT * FROM links2 ", cn2);
cn2->Open();
```

```
SqlDataReader *dr2  = cmd2->ExecuteReader();
dr2->Read();
dr2->Close();
da2=new SqlDataAdapter("select (ImagePath2 + ImageName) As Image from links2",
cn2);
 cmdBuilder2 = new SqlCommandBuilder(da2);
da2->Fill(ImageDataSet, "links2");
int count3 = ImageDataSet->Tables->Item["links2"]->Rows->Count ;
int count=count1+count2+count3;
String* files = new String("");
for(int k=0;k<count3;)
{
for (int i=0; i<count2 ;)
{
for(int j=0;j< count1;)
{
DataRow *rowCust = ImageDataSet->Tables->Item["links"]->Rows->Item[j];
Console::WriteLine("Customer Name before Update : {0} ", rowCust->Item["Image"]);
files = String::Concat(files , rowCust->Item["Image"]);
files = String::Concat(files, S"\n");
j++;
DataRow *rowCust1 = ImageDataSet1->Tables->Item["links1"]->Rows->Item[i];
Console::WriteLine("Customer Name before Update : {0} ", rowCust1->Item["Image"]);
files = String::Concat(files , rowCust1->Item["Image"]);
files = String::Concat(files, S"\n");
i++;
DataRow *rowCust2 = ImageDataSet->Tables->Item["links2"]->Rows->Item[k];
//Console::WriteLine("Customer Name before Update : {0} ", rowCust->Item["Image"]);
files = String::Concat(files , rowCust2->Item["Image"]);
files = String::Concat(files, S"\n");
k++;
} } }
StreamWriter* pwriter = new StreamWriter(S"C:/k1.text");
pwriter->WriteLine(files);
pwriter->Close();
fstream file_op("c:/k1.text",ios::in);
while(!file_op.eof())
{
file_op >> str;
IplImage* demo1;
demo1=cvLoadImage(str.c_str(),0);
if(demo1->width >= 92 || demo1->height >= 112)
{
        tarSize.width =92; // visualisation image is
        tarSize.height =112; // 256x256 pixels
}
        image2=cvCreateImage(tarSize,demo1->depth,demo1->nChannels);
```

```
        cvResize(demo1,image2,CV_INTER_AREA);
        IplImage* result;
                int img_width  = image1->width;
                int img_height = image1->height;
                int tpl_width  = image2->width;
                int tpl_height = image2->height;
                int res_width  = img_width - tpl_width + 1;
                int res_height = img_height - tpl_height + 1;
result = cvCreateImage(cvSize(res_width, res_height), IPL_DEPTH_32F, 1);
cvMatchTemplate( image1, image2, result, CV_TM_SQDIFF_NORMED );

/* find best matches location */
CvPoint minloc, maxloc;
double minval=0.0, maxval=0.0;
cvMinMaxLoc( result, &minval, &maxval, &minloc, &maxloc,0);
string s = double2str(minval);
GetDlgItem(IDC_STATIC2)->CWnd::SetWindowTextA(s.c_str());


                        // minval=0 implies perfect match
if (minval<0.2) // setting the threshod value to 0.2 & speed counter used
{  IplImage* recog = cvLoadImage("C:/pixel/images/lena1.jpg", 1 );
   cvNamedWindow("RECOGNIZED IMAGE1", CV_WINDOW_NORMAL );
   cvShowImage("RECOGNIZED IMAGE1", recog );
   for(int k=1;k<=120;k++)
  {
   string f = intstr(k); f +=".jpg";
   string tp=(str.c_str());
   const char * p="";
   p = tp.c_str();
   int q= tp.length();
   while(1)
   {
   if (p[q-1]=='/')
   break;
   q--;  }
   q = tp.length() - q;
   tp.erase (tp.begin(), tp.end()-q);
   if(tp==f.c_str())
   {   char other_string4[100];
       strcpy(other_string4,"D:/info/");
       strcat(other_string4,f.c_str());
       IplImage *vaish=cvLoadImage(other_string4,1);
       cvNamedWindow("Record", CV_WINDOW_NORMAL );
       cvShowImage("Record", vaish);
       cvWaitKey( 0 );
       cvDestroyWindow("Record"); }  }
```

```
      break;}  }
if(file_op.eof()==true)
{
IplImage* notrecog = cvLoadImage("C:/project/database/not.jpg", 0 );
cvNamedWindow("IMAGE NOT RECOGNIZED1 ", CV_WINDOW_NORMAL );
cvShowImage("IMAGE NOT RECOGNIZED1 ", notrecog );
cvWaitKey( 0 );
cvDestroyWindow("IMAGE NOT RECOGNIZED1 ");}
file_op.close();
cn->Close();}
void CDialogDlg::OnStnClickedStatic1()
{  }
```

SecondDlg.cpp:

```
#include "stdafx.h"
#include "validationnew.h"
#include "SecondDlg.h"
#include "DialogDlg.h"
#include"cv.h"
#include"highgui.h"
#include"cxcore.h"
#include<string>
#include<sstream>
#include<fstream>
#include <iostream>
#include <windows.h>
using namespace std;
using namespace System::IO;
#using <mscorlib.dll>
#using <System.dll>
#using <System.Data.dll>
#using <System.Xml.dll>

using namespace System;
using namespace System::Data;
using namespace System::Data::SqlClient;
string int2str (int a) {
        stringstream ss ;
        ss << a ;
        string f ;
        ss >> f ;
        return f ;
}
string float2str (float a) {
        stringstream ss ;
        ss << a ;
```

```
        string f ;
        ss >> f ;
        return f ;

}

// CSecondDlg dialog

IMPLEMENT_DYNAMIC(CSecondDlg, CDialog)
CSecondDlg::CSecondDlg(CWnd* pParent /*=NULL*/)
: CDialog(CSecondDlg::IDD, pParent)
{ }
CSecondDlg::~CSecondDlg()
{ }
void CSecondDlg::DoDataExchange(CDataExchange* pDX)
{
CDialog::DoDataExchange(pDX);
}
BEGIN_MESSAGE_MAP(CSecondDlg, CDialog)
ON_BN_CLICKED(IDC_BUTTON1, &CSecondDlg::OnBnClickedButton1)
ON_BN_CLICKED(IDC_BUTTON2, &CSecondDlg::OnBnClickedButton2)
ON_BN_CLICKED(IDC_BUTTON3, &CSecondDlg::OnBnClickedButton3)
ON_BN_CLICKED(IDC_BUTTON4, &CSecondDlg::OnBnClickedButton4)
ON_BN_CLICKED(IDOK, &CSecondDlg::OnBnClickedOk)
ON_STN_CLICKED(IDC_STATIC2, &CSecondDlg::OnStnClickedStatic2)
ON_STN_CLICKED(IDC_STATIC1, &CSecondDlg::OnStnClickedStatic1)
END_MESSAGE_MAP()
void CSecondDlg::OnBnClickedButton1()
{   CFileDialog dlg(TRUE, _T("*.bmp"), NULL,
    OFN_FILEMUSTEXIST|OFN_PATHMUSTEXIST|OFN_HIDEREADONLY,
    _T("image files (*.bmp; *.jpg) |*.bmp;*.jpg|All Files (*.*)|*.*||"),NULL);
  dlg.m_ofn.lpstrTitle= _T("Open Image");
if (dlg.DoModal() == IDOK) {
CString path= dlg.GetPathName();  // contain the selected filename
IplImage *image;                  // This is image pointer
image= cvLoadImage(path);           // load the image
cvNamedWindow("Original Image",CV_WINDOW_NORMAL);
cvShowImage("Original Image",image); // display it
cvSaveImage("C:/Documents and Settings/Siddhant/My Documents/Visual Studio
2008/Projects/validationnew/lena1.jpg",image);}}
void CSecondDlg::OnBnClickedButton2()
{
IplImage* img,*img2;
img = cvLoadImage("C:/Documents and Settings/Siddhant/My Documents/Visual
Studio 2008/Projects/validationnew/lena1.jpg",1 );
CvMemStorage* storage = cvCreateMemStorage(0);
```

```
CvHaarClassifierCascade* cascade = (CvHaarClassifierCascade*)cvLoad(
"C:/Documents and Settings/Siddhant/My Documents/Visual Studio
2008/Projects/validationnew/haarcascade_frontalface_alt2.xml" );
double scale = 1.3;
cvClearMemStorage( storage );
CvSeq* objects = cvHaarDetectObjects( img, cascade, storage, 1.1, 2, 0, cvSize( 40, 50
));CvRect* r;
for( int i = 0; i < (objects ? objects->total : 0 ); i++ ){
r = ( CvRect* )cvGetSeqElem( objects, i );
CvPoint pt1 = { r->x, r->y };
CvPoint pt2 = { r->x + r->width, r->y + r->height };
cvRectangle( img, pt1, pt2, CV_RGB(255,0,0), 2,2, 0 );
int a = (r->x), b=(r->y), c=((r->x + r->width)/1.1), d=((r->y + r->height)/1.3);
string f = int2str(i);
f +=".jpg";
cvSetImageROI(img,cvRect(a,b,c,d));
img2=cvCreateImage(cvGetSize(img), img->depth,img->nChannels);
cvCopyImage(img,img2,NULL);
char other_string2[100];

 strcpy(other_string2,"C:/project/seperate/");
strcat(other_string2,f.c_str());
cvSaveImage(other_string2,img2);
cvResetImageROI(img); }
cvNamedWindow("Output",CV_WINDOW_NORMAL );
cvShowImage("Output", img );
cvSaveImage("Res.jpg",img);
cvWaitKey();
cvReleaseImage( &img );}
void CSecondDlg::OnBnClickedButton3()
{
CvHaarClassifierCascade* cascade; // Create a new Haar classifier
IplImage *image;  // Load the image from that filename
CvMemStorage* storage; // Create memory for calculations
CvSeq* faces;
const char* file_name1;
const char* file_name2;
int i=0;
int j=0;
float m;
double angle=0,angle1=0,angle2=0,angle3=0,angle11=0,angle22=0,angle33=0;
double angledeg=0,angledeg1=0,angledeg2=0,angledeg3=0;
CvPoint center,center1,center2,center3,center4,center5;
int rows,cols,type;
CvPoint point1,point2,point3, point4,point5,point6;
center1=cvPoint(0,0);
center2=cvPoint(0,0);
```

```
center3=cvPoint(0,0);
 center=cvPoint(0,0);
 file_name1 = "C:/Documents and Settings/Siddhant/My Documents/Visual Studio
2008/Projects/validationnew/haarcascade_frontalface_alt.xml";
cascade = (CvHaarClassifierCascade*)cvLoad(file_name1,NULL, NULL, NULL);
file_name2 = "C:/Documents and Settings/Siddhant/My Documents/Visual Studio
2008/Projects/validationnew/haarcascade_eye.xml";
cascade = (CvHaarClassifierCascade*)cvLoad(file_name2,NULL, NULL, NULL);
CFileDialog dlg(TRUE, _T("*.bmp"), NULL,

OFN_FILEMUSTEXIST|OFN_PATHMUSTEXIST|OFN_HIDEREADONLY,_T("image
files (*.bmp; *.jpg) |*.bmp;*.jpg|All Files (*.*)|*.*||"),NULL);
 dlg.m_ofn.lpstrTitle= _T("Open Image");
 IplImage *eyeimage;                    // This is image pointer
if (dlg.DoModal() == IDOK)
  {
CString path= dlg.GetPathName();  // contain the selected filename
eyeimage= cvLoadImage(path);           // load the image
cvNamedWindow("Seperate Image",CV_WINDOW_NORMAL);
cvShowImage("Seperate Image", eyeimage); // display it
cvSaveImage("lena1.jpg", eyeimage);}
image = cvLoadImage("lena1.jpg", 1 );
storage = cvCreateMemStorage(0);
faces = cvHaarDetectObjects( image, cascade, storage, 1.2, 2,
CV_HAAR_DO_CANNY_PRUNING,cvSize(20, 20));
for(i=0;i<(faces ? faces->total:0); i++ )
{
CvRect* e = (CvRect*)cvGetSeqElem( faces, i );
CvPoint pt1 = { e->x, e->y };
CvPoint pt2 = { e->x + e->width, e->y + e->height };
}
CvSeq *eyes = cvHaarDetectObjects(
image,          /* the source image, with the estimated location defined */
cascade,     /* the eye classifier */
storage,       /* memory buffer */
1.15, 3, 0,     /* tune for your app */
cvSize(0, 0)  /* minimum detection scale */
);
int xval;
int yval;
int width1;
int height1;
CvSize imgSize;
imgSize.width =256; // visualisation image is
imgSize.height =256; // 256x256 pixels
IplImage *dst8,*dst7,*dst9;
dst7 = cvCreateImage( cvSize(image->width,image->height), 8, 3 );
```

```
width1=dst7->width-10;
height1=dst7->height-10;
dst8=cvCreateImage(cvSize(width1,height1),IPL_DEPTH_8U,3);
dst9=cvCreateImage(imgSize,dst8->depth,dst8->nChannels);
const CvArr* mask;
double scale=1.2;
CvMat *dst1,*dst2;
dst1= cvCreateMat(dst9->height,dst9->width,CV_8UC3);
dst2= cvCreateMat(dst9->height,dst9->width,CV_8UC3);
IplImage* gray = cvCreateImage( cvSize(image->width,image->height), 8, 1 );
IplImage* small_img = cvCreateImage( cvSize( cvRound (image->width/scale),cvRound
(image->height/scale)), 8, 1 );
cvCopy( image, dst7, NULL );
xval=5;
yval=5;

                    CvRect rect={xval,yval,width1,height1};
                    cvSetImageROI(dst7,rect);
                    cvCopy(dst7,dst8,NULL);
                    cvResetImageROI(dst7);
                    imgSize.width =256; // visualisation image is
                    imgSize.height =256; // 256x256 pixels
                    cvResize(dst8,dst9,CV_INTER_AREA);
                    dst1= cvCreateMat(dst9->height,dst9->width,CV_8UC3);
                    dst2= cvCreateMat(dst9->height,dst9->width,CV_8UC3);
                    cvCopy( dst9, dst1, NULL );
                    cvCopy( dst9, dst2, NULL );
                    cvCvtColor( image, gray, CV_BGR2GRAY );
                    cvResize( gray, small_img, CV_INTER_LINEAR );
                    cvEqualizeHist( small_img, small_img );
                    cvClearMemStorage( storage );
if(cascade)
 {
for(i= 0; i < (eyes ? eyes->total : 0); i++ )
{
CvRect *r = (CvRect*)cvGetSeqElem(eyes, i);
                        int radius;
                        double scale=1.2;
                        center.x = cvRound((r->x + r->width*0.3)*scale);
                        center.y = cvRound((r->y + r->height*0.3)*scale);
                        radius = cvRound((r->width + r->height)*0.25*scale);
                        cvCircle( image, center, radius, CV_RGB(255,0,0), 2, CV_AA, 0 );
                        point1.x=(center.x)+5;
                        point2.x=(center.x)-5;
                        point3.y=(center.y)+5;
                        point4.y=(center.y)-5;
                        point1.y=(center.y);
```

```
                point2.y=(center.y);
                point3.x=(center.x);
                point4.x=(center.x);
                cvLine(image, point1, point2,CV_RGB(255,0,0), 1, CV_AA, 0);
                cvLine(image, point3, point4, CV_RGB(255,0,0), 1, CV_AA, 0);
    if(j<1)
    {
            center1=center;
            j++;
    }
    else if(j==1)
    {
            center2=center;
            j++;
    }
    else if(j==2)
    {
            center3=center;
            j++;
    }  }  //for loop
point1.y=(center1.y+center2.y)/2;
            point2.y=(center1.y+center2.y)/2;
            point1.x=0;
            point2.x=256;
            cvLine(image, point1, point2,  CV_RGB(255,0,0),1, CV_AA, 0);
            point1.y=(center1.y+center2.y)/2;
            point2.y=center2.y;
            point1.x=(center1.x+center2.x)/2;
            point2.x=center2.x;
            m=((float)(point2.y-point1.y)/(point2.x-point1.x));
            angle1=atan(m);
            angledeg1=(angle1*180)/3.14159265;
            point1.y=(center1.y+center3.y)/2;
            point2.y=(center1.y+center3.y)/2;
            point1.x=0;
            point2.x=256;
            point1.y=(center1.y+center3.y)/2;
            point2.y=center1.y;
            point1.x=(center1.x+center3.x)/2;
            point2.x=center1.x;
            m=((float)(point2.y-point1.y)/(point2.x-point1.x));
            angle2=atan(m);
            angledeg2=(angle2*180)/3.14159265;
            point1.y=(center2.y+center3.y)/2;
            point2.y=(center2.y+center3.y)/2;
            point1.x=0;
            point2.x=256;
```

```
                point1.y=(center2.y+center3.y)/2;
                point2.y=center3.y;
                point1.x=(center2.x+center3.x)/2;
                point2.x=center3.x;
                m=((float)(point2.y-point1.y)/(point2.x-point1.x));
                angle3=atan(m);
                angledeg3=(angle3*180)/3.14159265;

                angle11=abs(angledeg1);
                angle22=abs(angledeg2);
                angle33=abs(angledeg3);
                }//for loop

if((angle11<=angle22) && (angle11<=angle33))
                {
                angle=angle1;
                }
                if((angle22<angle11) && (angle22<angle33))
                {  angle=angle2; }
                if((angle33<angle11) && (angle33<angle22))
                {  angle=angle3;}
                angledeg=(angle*180)/3.14159265;

                //Rotation matix
                double T[]= { 1, 0, 128, 0, 1, 128, 0, 0, 1 };
                double R[]= { cos(angle), -sin(angle), 0, sin(angle), cos(angle), 0, 0, 0, 1 };
                double T1[]={ 1, 0, -128, 0, 1, -128, 0, 0, 1 };
                double c[9],Tm[9];
                int scale,x,y,m,n;
                CvMat MT,MR,MT1,Mc;
                CvMat *MTm;
                //Initializes matrix header
                cvInitMatHeader( &MT, 3, 3, CV_64FC1, T, CV_AUTOSTEP);
                cvInitMatHeader( &MR, 3, 3, CV_64FC1, R, CV_AUTOSTEP);
                cvInitMatHeader( &Mc, 3, 3, CV_64FC1, c, CV_AUTOSTEP);
                cvInitMatHeader( &MT1, 3, 3, CV_64FC1, T1, CV_AUTOSTEP);
                MTm=cvCreateMat(3,3,CV_64FC1);
                cvMatMulAdd( &MT, &MR, 0, &Mc);
                cvMatMulAdd( &Mc, &MT1, 0,MTm);
                double t1,t2,t3;
                CvScalar scal;
                CvMat *MP1;
                MP1=cvCreateMat(3, 1, CV_64FC1);
                for(x=8;x<dst1->rows-5;x++)
                {
                        for(y=8;y<dst1->cols-5;y++)
                        {
```

```
                        double P[] = { x, y, 1 };
                        CvMat MP;
                        cvInitMatHeader( &MP, 3, 1, CV_64FC1, P, CV_AUTOSTEP);
                        scal=cvGet2D(dst1,x,y);
                        cvMatMulAdd( MTm, &MP, 0, MP1);
                        m=0;n=0;
                        t1=cvmGet(MP1,m,n);//to access the (i,j) cell of a 2D float matrix.
                        m=1;n=0;
                        t2=cvmGet(MP1,m,n);
                        m=2;n=0;
                        t3=cvmGet(MP1,m,n);
                        if(t1<256 && t1>=0 && t2<256 && t2>=0)
                        {
                        cvSet2D(dst2,t1,t2,scal);
                        } } } }
cvResetImageROI(image);
/* end of detectEyes() */
string f = int2str(i);
f +=".jpg";
char other_string[100];
strcpy(other_string,"C:/project/facedetect/");
cvNamedWindow( " EYE LOCALIZATION ", CV_WINDOW_NORMAL );
 cvShowImage( " EYE LOCALIZATION ", image );
strcat(other_string,f.c_str());
cvSaveImage(other_string,image);//Detectd faces are saved);
char other_string1[100];
strcpy(other_string1,"C:/project/normalised/");
 cvNamedWindow( "NORMALIZED IMAGE", CV_WINDOW_NORMAL );
 cvShowImage( "NORMALIZED IMAGE", dst2 );
cvSaveImage("C:/project/normalised/norm.jpg",dst2);}
void CSecondDlg::OnBnClickedButton4()
{
        IplImage* in;
        IplImage*image1;
        IplImage*image2;
        float coreff=0.0;
        FILE* InputFILE;
        char ch[30],tmp[30],rec[100];
        int i;
        CvSize refSize;
        CvSize tarSize;
        IplImage* demo;
        in=cvLoadImage("C:/project/normalised/norm.jpg",0);
        IplImage* ref=cvCreateImage( cvGetSize(in), IPL_DEPTH_8U, 1 );
        cvEqualizeHist( in, ref );
        if(ref->width >= 92 || ref->height >= 112)
         {
```

```
            refSize.width =92; // visualisation image is
            refSize.height =112; // 256x256 pixels
}
image1=cvCreateImage(refSize,ref->depth,ref->nChannels);
cvResize(ref,image1,CV_INTER_AREA);
uchar *ref_data = ( uchar*) image1->imageData;//pointer to first image
int width1= image1->width;
int height1 = image1->height;
int step1= image1->widthStep;
int fft_size1 = width1* height1; //first image size
float sum=0;
for(i=0;i<fft_size1;i++)
sum=sum+(ref_data[i]/256.0);
float avg=sum/fft_size1;
float std1[10304];
for(i=0;i<fft_size1;i++)
std1[i]=(ref_data[i]/256.0);
for(i=0;i<fft_size1;i++)
std1[i]=std1[i]-avg;

for(i=0;i<fft_size1;i++)
std1[i]=(std1[i]*std1[i]);
float  sum1=0;
for(i=0;i<fft_size1;i++)
sum1=sum1+std1[i];
float  sx=sqrt(sum1/fft_size1);//standard deviation of image 1
SqlConnection* cn = new SqlConnection();
DataSet *ImageDataSet = new DataSet();
SqlDataAdapter *da;
SqlCommandBuilder *cmdBuilder;
cn->ConnectionString =  "Server=SIDDHANT-
2529B6\\SQLEXPRESS;Database=Test;Trusted_Connection=Yes;";
SqlCommand *cmd = new SqlCommand("SELECT * FROM links ", cn);
cn->Open();
SqlDataReader *dr  = cmd->ExecuteReader();
dr->Read();
dr->Close();
string str;
da = new SqlDataAdapter("select (ImagePath + ImageName) As Image from
links", cn);
cmdBuilder = new SqlCommandBuilder(da);
da->Fill(ImageDataSet, "links");
int count1 = ImageDataSet->Tables->Item["links"]->Rows->Count ;
/*second conn*/
SqlConnection* cn1 = new SqlConnection();
DataSet *ImageDataSet1 = new DataSet();
SqlDataAdapter *da1;
```

```
SqlCommandBuilder *cmdBuilder1;
cn1->ConnectionString =  "Server=SIDDHANT-
2529B6\\MSSQLEXPRESS;Database=Test1;Trusted_Connection=Yes;";
SqlCommand *cmd1 = new SqlCommand("SELECT * FROM links1 ", cn1);
cn1->Open();
SqlDataReader *dr1  = cmd1->ExecuteReader();
dr1->Read();
dr1->Close();
da1 = new SqlDataAdapter("select (ImagePath + ImageName) As Image from
links1", cn1);
cmdBuilder1 = new SqlCommandBuilder(da1);
da1->Fill(ImageDataSet1, "links1");
int count2 = ImageDataSet1->Tables->Item["links1"]->Rows->Count ;
//THIRD CONNECTION
SqlConnection* cn2 = new SqlConnection();
DataSet *ImageDataSet2 = new DataSet();
SqlDataAdapter *da2;
SqlCommandBuilder *cmdBuilder2;
cn2->ConnectionString =  "Server=SIDDHANT-
2529B6\\MS1SQLEXPRESS;Database=Test2;Trusted_Connection=Yes;";
SqlCommand *cmd2 = new SqlCommand("SELECT * FROM links2 ", cn2);
cn2->Open();
SqlDataReader *dr2  = cmd2->ExecuteReader();
dr2->Read();
dr2->Close();
//string str;
da2 = new SqlDataAdapter("select (ImagePath + ImageName) As Image from
links2", cn2);
cmdBuilder2 = new SqlCommandBuilder(da2);
da2->Fill(ImageDataSet, "links2");
int count3 = ImageDataSet->Tables->Item["links2"]->Rows->Count ;
int count=count1+count2+count3;
String* files = new String("");
for (int k=0; k<count3 ;)
{
for (int i=0; i<count2 ;)
{
for(int j=0;j< count1;)
{
DataRow *rowCust = ImageDataSet->Tables->Item["links"]->Rows->Item[j];
Console::WriteLine("Customer Name before Update : {0} ",
rowCust>Item["Image"]);
files = String::Concat(files , rowCust->Item["Image"]);
files = String::Concat(files, S"\n");
j++;
//i+6;
DataRow *rowCust1 = ImageDataSet1->Tables->Item["links1"]->Rows->Item[i];
```

```
Console::WriteLine("Customer Name before Update : {0} ",
rowCust1>Item["Image"]);
files = String::Concat(files , rowCust1->Item["Image"]);
files = String::Concat(files, S"\n");
i++;
//for k
DataRow *rowCust2 = ImageDataSet->Tables->Item["links2"]->Rows->Item[k];
Console::WriteLine("Customer Name before Update : {0} ",
rowCust2>Item["Image"]);
files = String::Concat(files , rowCust2->Item["Image"]);
files = String::Concat(files, S"\n");
k++; } } }
StreamWriter* pwriter = new StreamWriter(S"C:/k.text");
pwriter->WriteLine(files);
pwriter->Close();
fstream file_op("c:/k.text",ios::in);

while(!file_op.eof())
{
file_op >> str;
IplImage* demo;
demo=cvLoadImage(str.c_str(),0);
IplImage* target=cvCreateImage( cvGetSize(demo), IPL_DEPTH_8U, 1 );
cvEqualizeHist(demo,target);
//uchar *ref_data3 = ( uchar* ) target->imageData;
cout<< target->width;
cout << target->height;
if(target->width >= 92 || target->height >= 112)
 {
tarSize.width =92; // visualisation image is
tarSize.height =112; // 256x256 pixels
 }
image2=cvCreateImage(tarSize,target->depth,target->nChannels);
cvResize(target,image2,CV_INTER_AREA);
uchar *ref_data1 =(uchar*) image2->imageData;//pointer to second image
int width2 = image2->width;
int height2 = image2->height;
int step2 = image2->widthStep;
int fft_size2= width2 * height2;//second image size
float  sum4=0;
for(i=0;i<fft_size2;i++)
sum4=sum4+(ref_data1[i]/256.0);
float  avg2=sum4/fft_size2;
float std2[10304];
for(i=0;i<fft_size2;i++)
std2[i]=(ref_data1[i]/256.0);
for(i=0;i<fft_size2;i++)
```

```
std2[i]=std2[i]-avg2;
for(i=0;i<fft_size2;i++)
std2[i]=std2[i]*std2[i];
float sum5=0;
for(i=0;i<fft_size2;i++)
sum5=sum5+std2[i];
float sy=sqrt(sum5/fft_size2);
float  add=0;
float rx1[10304];
for(i=0;i<fft_size2;i++)
rx1[i]=((ref_data[i]/256.0)-avg)*((ref_data1[i]/256.0)-avg2);
float  sum6=0;
for(i=0;i<fft_size2;i++)
sum6=sum6+rx1[i];
float rxy = sum6/(((fft_size1-1)*sx)*sy);
string s = float2str(rxy);
GetDlgItem(IDC_STATIC1)->CWnd::SetWindowTextA(s.c_str());
if(rxy>coreff)
{  coreff=rxy;  }
if(coreff>0.99)
coreff=1.0;
float threshold=0.9;
if(coreff>=threshold)
{

        IplImage* recog = cvLoadImage(str.c_str(), 1);
        cvNamedWindow("RECOGNIZED IMAGE", CV_WINDOW_NORMAL);
        cvShowImage("RECOGNIZED IMAGE", recog );
        for(int k=1;k<=120;k++)
        {
        string f = int2str(k);
        f +=".jpg";
                string tp=(str.c_str());
                const char * p="";
                p = tp.c_str();
                int q= tp.length();
                while(1)
                {
                if (p[q-1]=='/')
                break;
                q--;
                }
                q = tp.length() - q;
                tp.erase (tp.begin(), tp.end()-q);
                if(tp==f.c_str())
                {
                 char other_string4[100];
```

```
                                strcpy(other_string4,"D:/info/");
                                strcat(other_string4,f.c_str());
                                IplImage *vaish=cvLoadImage(other_string4,1);
                                cvNamedWindow("Information", CV_WINDOW_NORMAL );
                               cvShowImage("Information", vaish);
                               }
                               } break;
                               }

                               cn->Close(); }
if(file_op.eof()==true)
{
IplImage* notrecog = cvLoadImage("C:/project/database/not.jpg", 0 );
cvNamedWindow("IMAGE NOT RECOGNIZED ", CV_WINDOW_NORMAL );
cvShowImage("IMAGE NOT RECOGNIZED ", notrecog );
cvWaitKey( 0 );
cvDestroyWindow("IMAGE NOT RECOGNIZED ");
}
file_op.close();
}
void CSecondDlg::OnBnClickedOk()
{
OnOK();
}
void CSecondDlg::OnStnClickedStatic2()
{ }

void CSecondDlg::OnStnClickedStatic1()
{ }
```

3.4) Snapshots of UI and Report

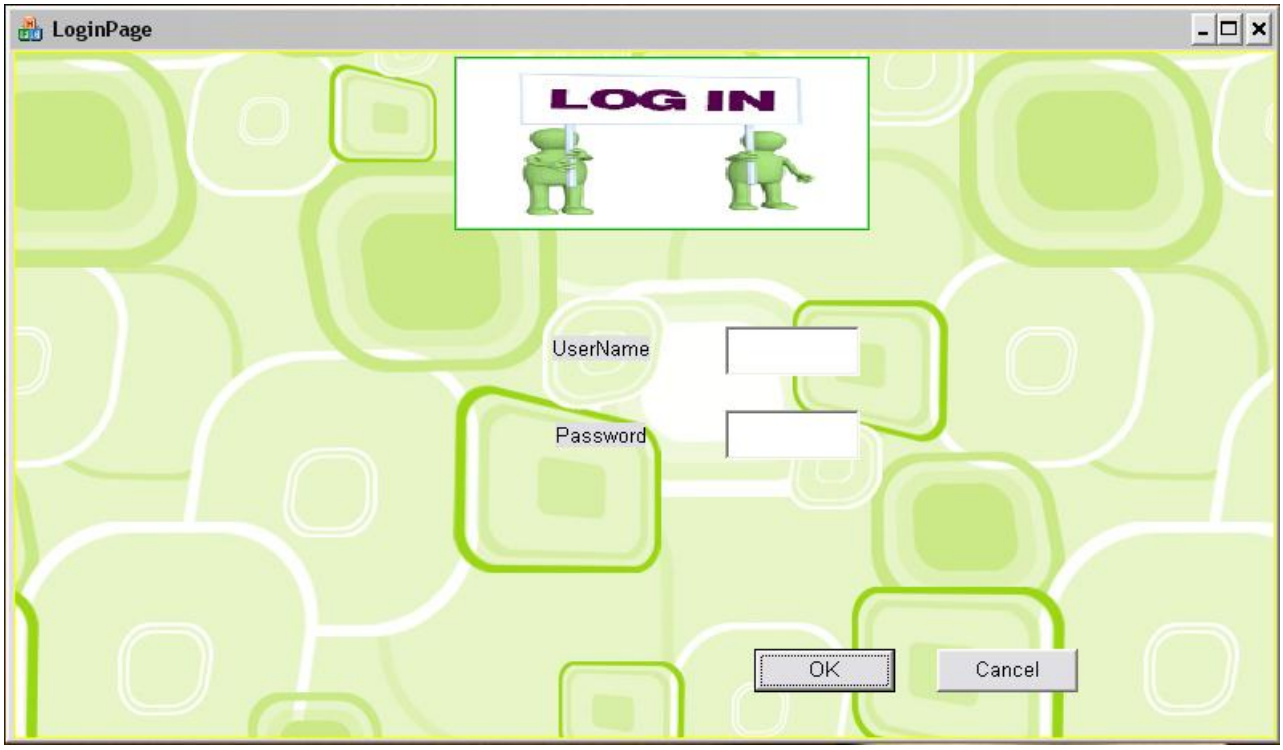


Figure:3.4.1:Login Page

Figure:3.4.2:Main Page

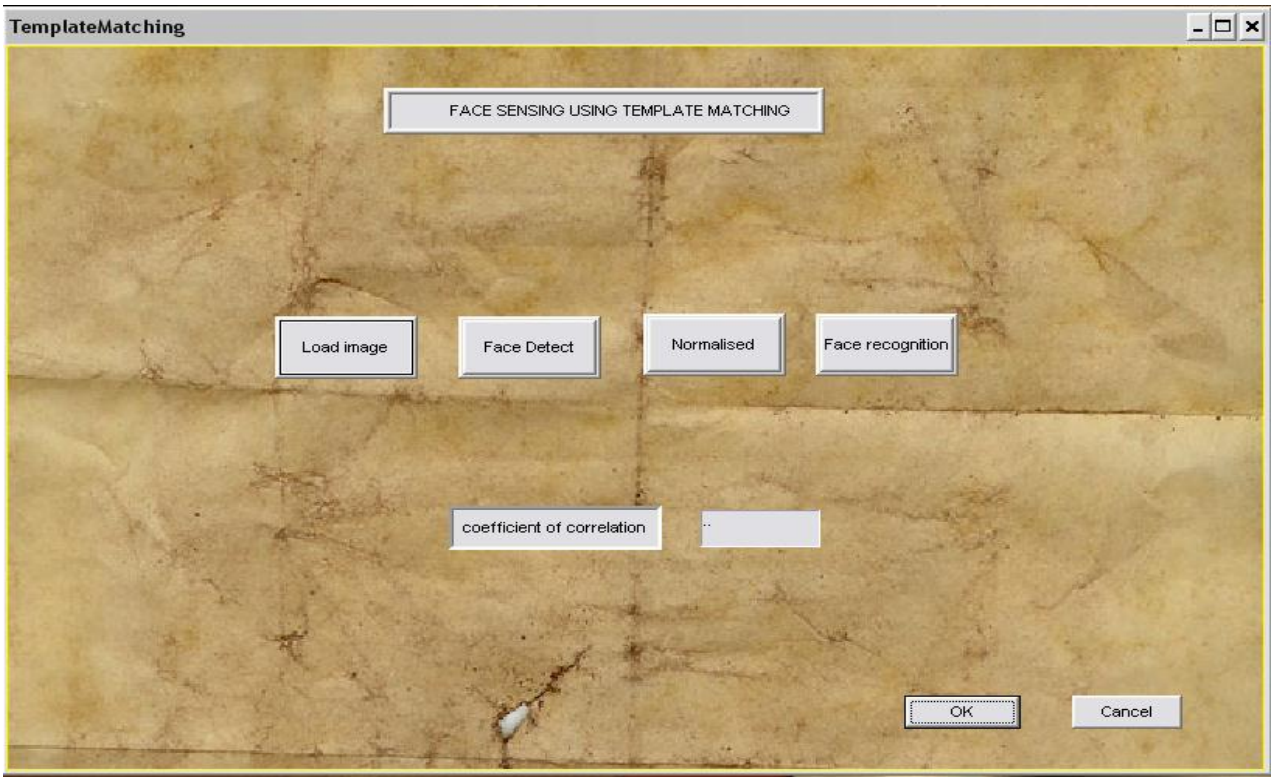


Figure:3.4.3: Template Matching

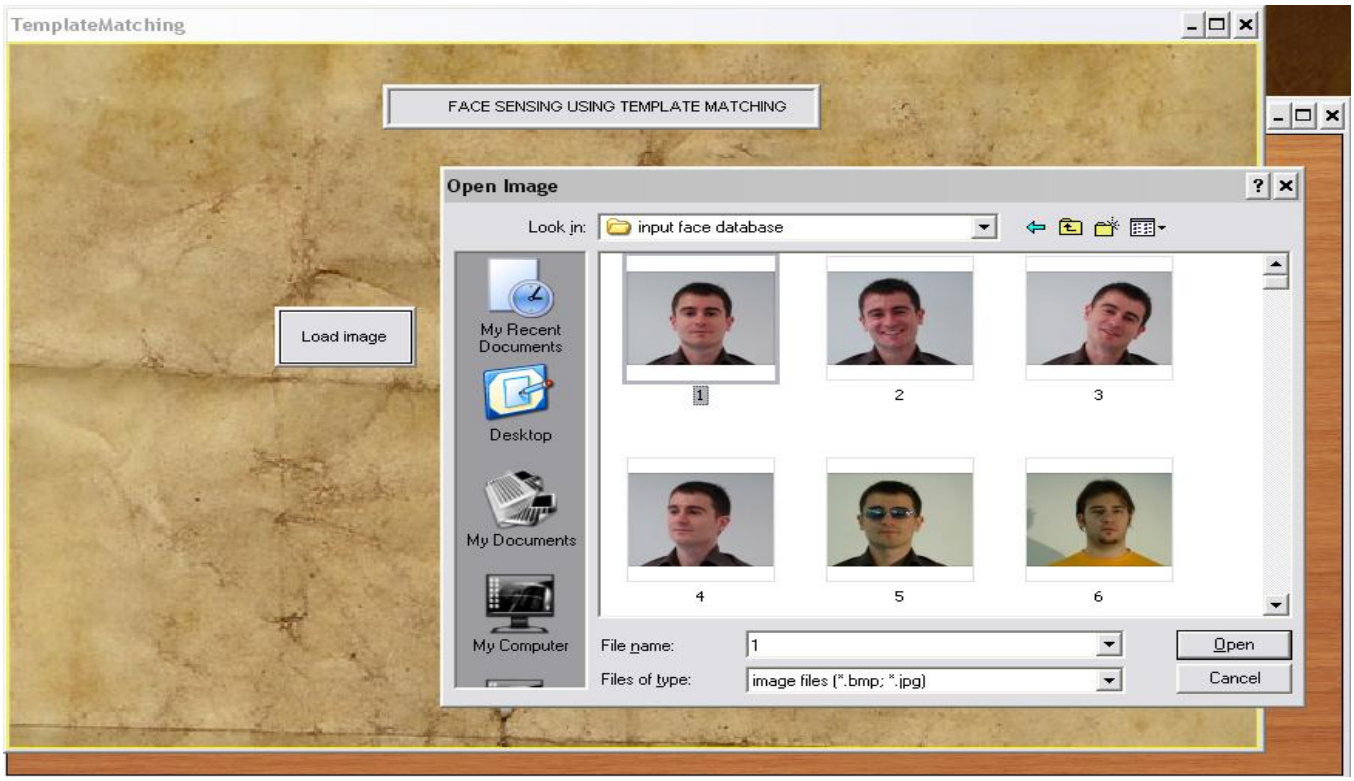


Figure:3.4.4: Template Matching With Load Image Option

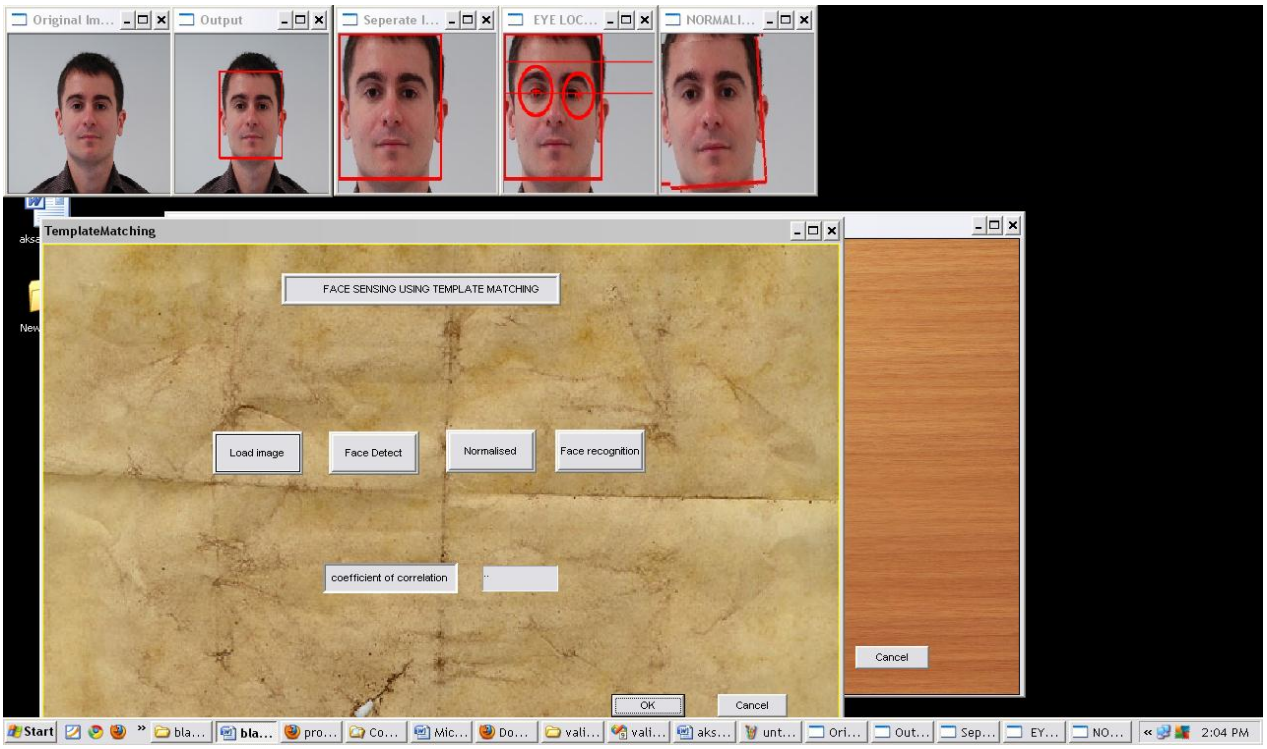


Figure 3.4.5: Template matching with preprocessing option

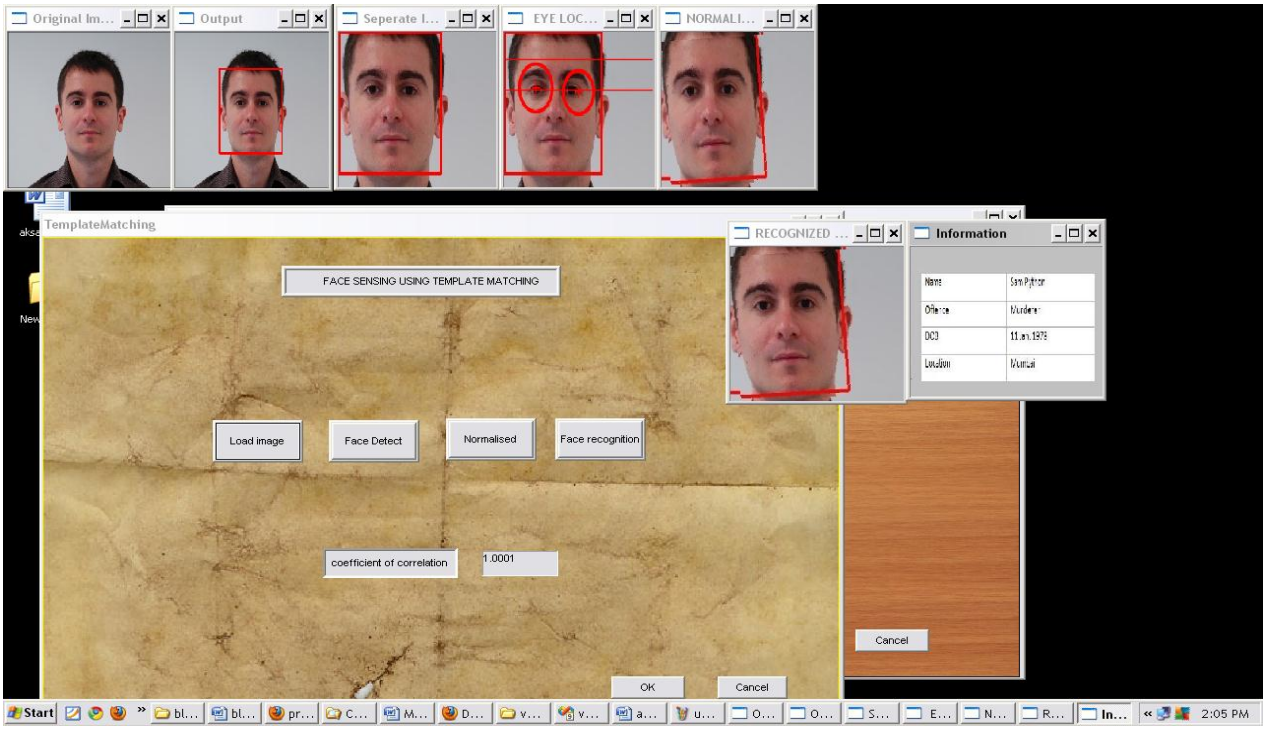


Figure3.4.6: Template matching final output of face reorganization

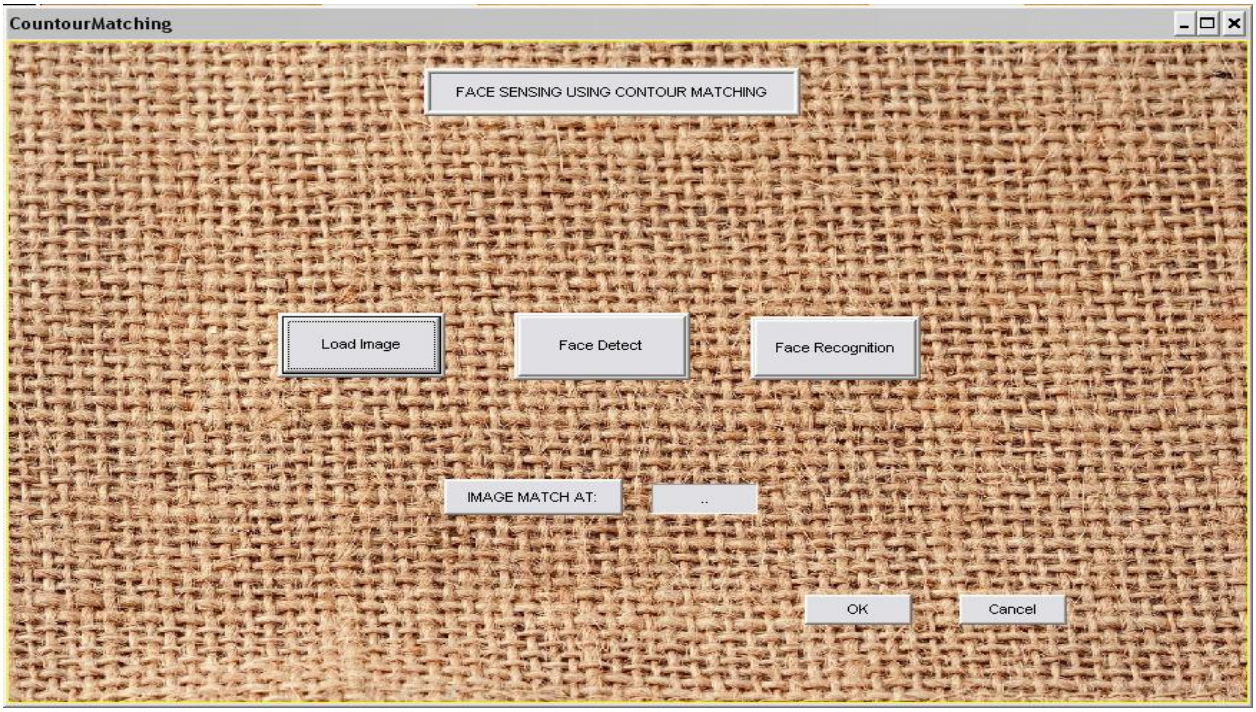


Figure 3.4.7: Contour matching page

Figure 3.4.8: Contour matching with load image option:

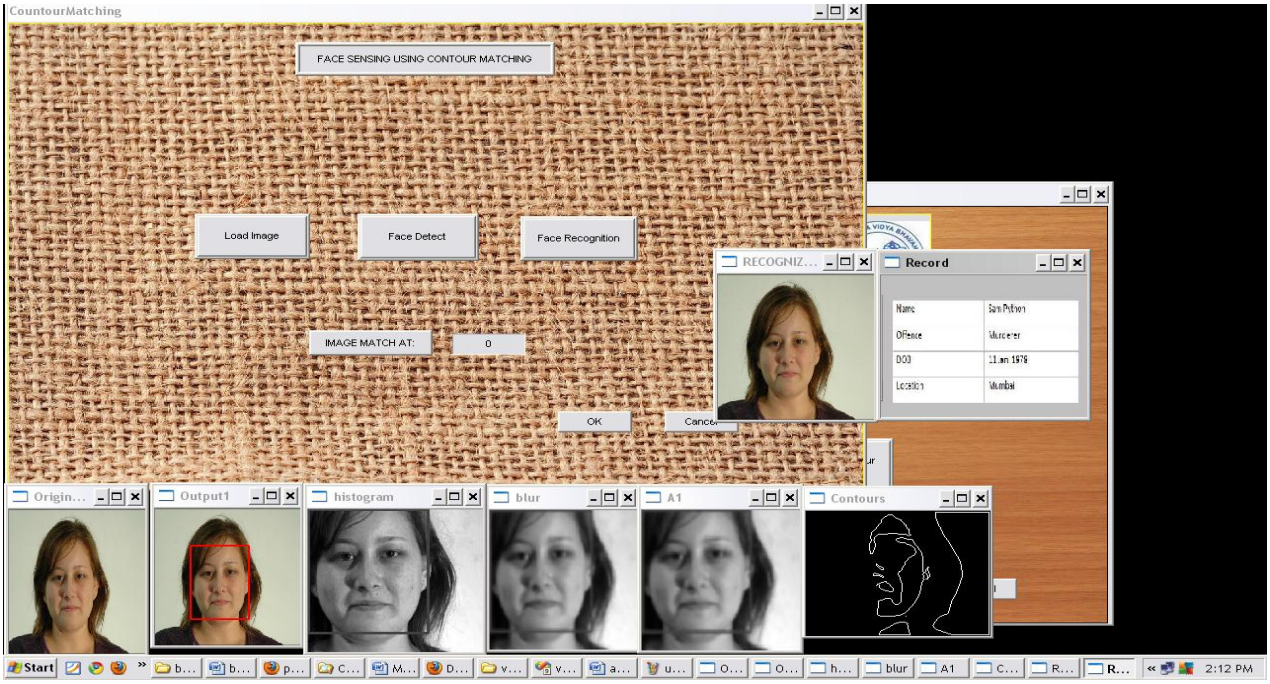


Figure 3.4.9: Contour matching with preprocessing option

3.5) Test cases & reports:

TEST CASE: 1

Module Name		**Project path**	
Form name	**validationnewDlg.cpp**	**Form path**	
Sl No.	**Field name/Test Case Name**	**Test Case Description**	**Expected Output**
1.	**Username**	**Leave the field blank**	**It will display error message, "check your username and password"**
2.	**Password**	**Leave the field blank**	**It will display error message, "check your username and password"**

Table 3.5.1:Test Case1

TEST CASE: 2

Form name	**validationnewDlg.cpp**		
Serial no	**Field name/Test Case Name**	**Test Case Description**	**Expected Output**
1.	**username password**	**Username and password do not match**	**It will display error message, "Username and password do not match"**

Table 3.5.2:Test Case2

TEST CASE: 3

Form name	validationnewDlg.cpp		
Serial no	Field name/Test Case Name	Test Case Description	Expected Output
1.	Template matching based face recognition	Button click	Display page for template matching
2.	Contour Matching Based face recognition	Button click	Display page for contour matching

Table 3.5.3:Test Case3

TEST CASE: 4

Form name	secondDlg.cpp		
Serial no	Field name/Test Case Name	Test Case Description	Expected Output
1.	Load Image	Button Click	Selected image will be loaded and displayed
2.	Face Detection	Button Click	Face of selected image will be detected and displayed
3.	Normalization	Button Click	Normalization operations will be performed on detected face and results will be displayed
4.	Face recognition	Button Click	Recognized face will be displayed along with its details or face not recognized message will be displayed.

Table 3.5.4: Test Case4

TEST CASE: 5

Form name	DialogDlg.cpp		
Serial no	Field name/Test Case Name	Test Case Description	Expected Output
1.	Load Image	Button Click	Selected image will be loaded and displayed
2.	Face Detection	Button Click	Face of selected image will be detected and displayed
3.	Face recognition	Button Click	Recognized face will be displayed along with its details or face not recognized message will be displayed.

Table 3.5.5: Test Case 5

CHAPTER 4
DOCUMENTATION AND CD

4) Documentation &CD

4.1) DEMO:

- Insert DVD which has required software.
- Read installation guide and install software.
- Insert Application DVD into laptop or desktop.
- Click on application icon and start application.

4.2) System Manual:

A. Microsoft SQL Server 2008

1. Insert the **Microsoft SQL Server 2008** installation CD-Rom into the CD-Rom drive. The installation will start once you click on the setup.exe file.
2. Click on the Installation hyperlink on the left hand side of the screen
3. Click on the **"New Server stand-alone installation"** link on the right side of the screen
4. If any checks have failed, click on the Show details button or "View detailed report link" to find out the cause, correct it, then click on the Re-run button to perform the checks again.
5. If all checks have passed, click on the OK button. After a few moments, the option to select the edition and to enter the license key (or "product key") will appear. Note that the product key box may already be populated, depending on which edition you have. Don't enter the product key we've shown here, it won't work on your system!
6. Enter the product key into the box, or choose the free edition if you're evaluating SQL Server 2008, and click on the Next button
7. Click in the **"I accept the license terms"** check box, and then click on the **Next** button again.
8. The setup support file screen appears; click on the Install button.
9. Select the features you want to install.
   At a minimum, the following are useful (I'd argue essential), but what you need will depend on your needs.
10. **Instance Configuration:** For most installations, keep the default settings. Click on the **Next** button.
11. **Server Configuration:** This step allows you to set up the service accounts that will be used to run SQL Server. If you have created Windows NT or Active Directory accounts for use with services, use these.

12. **Database Engine Configuration – Account Provision:** This allows you to set up database engine security.
13. **Database Engine Configuration – Data Directories:** Click on the **Data Directories** tab. Change the directories to specify which drives in your system will be used for the various types of database files.
14. **Installation Rules:** This simply checks if there are any processes or other installations running which will stop the installation of SQL Server 2008. Click on **Next** again – you're almost ready to install.
15. **Ready to Install :** This summarizes what you are about to install and gives you a last chance to cancel or change anything that's wrongly configured.
16. **Installation Progress:** SQL Server 2008 will now install. How long it takes depends on the speed of your machine, what load it's under, the installation media (CD is slower) and what you've chosen to install.
17. **Installation Complete:** It may be worth clicking on the installation log at the top of the screen to check everything's gone as expected. Not that this is MUCH smaller than the usual SQL Server installation log files of old. Finally, **click on the Close button**. **Click on OK – your server will NOT re-boot at this point.** The dialog box will disappear and you will be returned to the Installation Center.

B. Microsoft Visual Studio 2008 Professional:

1. Insert the **Microsoft Visual Studio 2008 Professional** installation CD-Rom into the CD-Rom drive. The installation will start once you click on the setup.exe file.
2. click the Install Visual Studio 2008 link to start the installation. The setup wizard will start copying needed files into a temporary folder.
3. Click the Next button to go to the next step.The setup wizard will list down all the required components need to be installed. Any already installed components will also be mentioned. Notice that VS 2008 (version 8.x) needs .NET Framework version 3.5. Key in the Product key and accept the license terms. Then click the Next button.
4. In the installation type, as usual we have three choices: **Default**, **Full** or **Custom**. In this case we select the Full installation type and accept the default installation path given.
5. The installation starts. Just wait and see the step-by-step, Visual Studio 2008 components being installed.

4.3) User manual with installation procedure:

Our project is easy to use. By following the below steps the code can be setup with full functionality.

PRE-CONDITIONS:

- For implementing our project scenario, Microsoft Visual Studio 2008 professional and Microsoft SQL Server 2008 need to be installing on computer.

- OpenCV 2.1 needs to be installing on computer.

Installation Procedure:

1 Double click on validation.exe and login page will open.

2 Enter correct username and password and click ok.

3 Main page with template matching and contour matching options will open

4 Clicking on template matching button opens template matching window, it contains separate buttons for load image, face detect, normalization and face recognition.

5 Clicking on load image opens window having input database of face images.

6 Select image and then click on face detect which gives detected face. Separated faces are stored in separate folder.

7 Then click on normalization button which will open separate folder, then select respective face and click ok. A normalized face images will display.

8 Then finally click on face recognition button. If image recognized, input face with details of persons will displayed .or not recognized message will come.

9 Click ok on template matching form and close all images and now click on contour matching button on main page.

10 Clicking on contour matching button opens template matching window, it contains separate buttons for load image, face detect and face recognition.

11 Follow the same procedures as in steps 5, 6, 7 and 8.

12 Click ok or cancel of any form to log out from application.

List of Figures and Tables:

CHAPTER 5
BIBLIOGRAPHY

5) Bibliography

- http://opencv.willowgarage.com/wiki/FaceDetection
- http://resnikb.wordpress.com/2009/10/28/using-visual-studio-2008-with-visual-c-6-0-compiler/
- http://dasl.mem.drexel.edu/~noahKuntz/openCVTut11.html
- http://www710.univ-lyon1.fr/~bouakaz/OpenCV-0.9.5/docs/ref/OpenCVRef_Experimental.htm
- http://www.site.uottawa.ca/~laganier/tutorial/opencv+directshow/cvision.htm
- http://www.youtube.com/watch?v=tTbXHQlbzlE
- http://www.functionx.com/visualc/
- http://www.softwaretrainingtutorials.com/ms-sql-server-2008.php
- http://blogs.msdn.com/b/sync/archive/2009/12/14/how-to-synchronize-multiple-geographically-distributed-sql-server-databases-using-sql-azure-data-sync.aspx
- http://www.sqlservercurry.com/2008/03/how-to-set-up-your-database-for.html
- http://www.amerusa-criminal-records.com/sample.html
- http://www.codeproject.com/KB/database/Blobfield.aspx