BAB IV
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Sistem

Implementasi merupakan tahap dimana sistem siap untuk dioperasikan, Hal ini dilakukan setelah penulisan kode program, uji sub sistem dan penggabungan masing-masing sub sistem. Pada tahap implementasi sistem ini, diharapkan sistem yang telah dirancang siap untuk dioperasikan pada keadaan yang sebenarnya, sehingga akan diketahui apakah sistem yang dibuat benar-benar dapat menghasilkan tujuan yang diinginkan dan sesuai dengan tujuan yang diharapkan.

Lingkungan Implementasi sistem ada 2 yaitu, lingkungan perangkat keras dan lingkungan perangkat lunak. Kedua lingkungan implementasi ini diharapkan dapat memenuhi persyaratan agar sistem yang dibangun dapat dijalankan dengan baik.

4.1.1 Spesifikasi Perangkat Keras

Sistem dioperasikan dengan spesifikasi minimum perangkat keras yang digunakan mempunyai spesifikasi sebagai berikut:

Tabel 4. 1 Spesifikasi Minimum Perangkat Keras

Spesifikasi Minimum Perangkat Keras	
Processor	Intel Pentium IV 1.06 GHz
Memory	128MB RAM
Harddisk	40GB SATA
VGA	512MB
Webcam	Logitech QuickCam™ 3.1 MP
Keyboard	4TECH, PS/2
Mouse	Optical K-ONE, USB
Printer	Cannon Pixma iP1200

4.1.2 Spesifikasi Perangkat Lunak

Sistem dioperasikan dengan spesifikasi minimum perangkat lunak yang digunakan mempunyai spesifikasi sebagai berikut:

Tabel 4. 2 Spesifikasi Minimum Perangkat Lunak

Spesifikasi Minimum Perangkat Lunak	
Sistem Operasi	Microsoft Windows XP Profesional Home Edition
Bahasa Pemrograman	Microsoft Visual Basic 6.0 Enterprise Edition
Database	Microsoft Office Access 2000

4.1.3 Instalasi Hardware

Sistem presensi karyawan ini dirancang menggunakan media *input* yang berupa *webcam* karena dari segi harga, *webcam* merupakan media *capture image* yang relative murah. Pada pengembangan aplikasi ini *webcam* yang digunakan adalah *webcam* dengan tipe *Logitech QuickCam*^{TM}. *Webcam* ini dihubungkan dengan komputer melalui port USB dan bekerja dengan beberapa mode, yaitu :

1. 160 x 120
2. 170 x 144
3. 320 x 240
4. 352 x 288
5. 640 x 480



Gambar 4. 1 Webcam Logitech QuickCam^{TM}

Untuk aplikasi ini mode dengan resolusi 160 x 120 sudah cukup untuk menghasilkan citra wajah yang digunakan sebagai *input* sistem presensi. Sebelum digunakan, driver untuk webcam harus diinstalasi karena nantinya aplikasi presensi akan melakukan koneksi ke *driver webcam* untuk memfungsikannya sebagai media *input*. Jika *driver webcam* sudah diinstalasi maka tinggal menghubungkan *webcam* ke port USB dan *webcam* siap digunakan sebagai media *input* untuk sistem presensi.

4.1.4 Implementasi Sistem Presensi Karyawan

Sistem presensi karyawan ini dilengkapi dengan halaman *login* untuk membatasi hak akses antara karyawan yang melakukan presensi saja dengan admin yang bertugas meng-*input* data. Untuk karyawan yang hanya melakukan presensi, langsung menekan tombol **presensi**. Pengisian *username* dan *password* hanya dilakukan oleh admin yang bertugas meng-*input* data. Saat proses *login* dilakukan maka langkah pertama yang dilakukan oleh sistem adalah menghitung *eigenface* dari semua citra wajah yang ada dalam *database* untuk disimpan dalam suatu variabel. Jadi pada tahap ini fitur diekstrasi. Hal ini dilakukan untuk kecepatan komputasi. Sistem cukup me-*load* fitur 1 kali saja dalam variabel, setelah itu bisa diakses kapan saja selama proses presensi. Tampilan halaman *login* sebagai berikut :



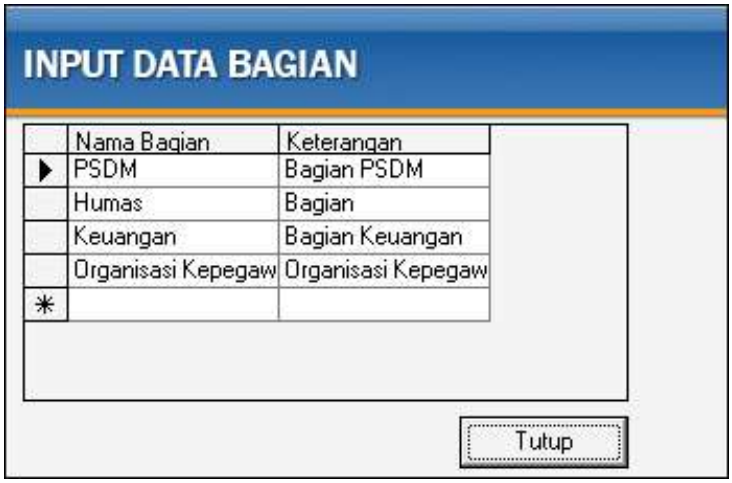
Gambar 4. 2 Form Login Sistem Presensi Karyawan

4.1.4.1 Input Data Karyawan

Untuk meng-*input* data karyawan maka harus mengisikan dulu data-data untuk tabel bagian dan jabatan. Adapun penjelasan untuk pengisian data-data tersebut dapat dijelaskan sebagai berikut :

1. Input Data Bagian

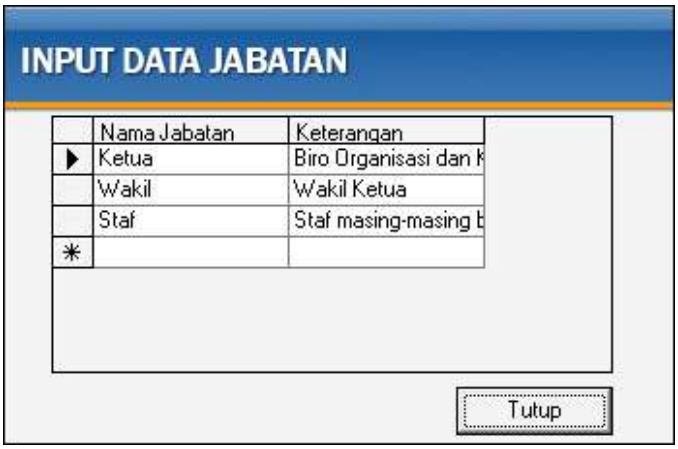
Untuk mengisi data pada tabel bagian digunakan submenu bagian (Master Data →Bagian). Pada *form* nama bagian dan keterangan deskripsi bagian di-*input*. Sedangkan id_bagian dihasilkan secara otomatis pada tabel bagian karena *field* ini betipe *autonumber*.



Gambar 4. 3 Tampilan Form Input Tabel Bagian

2. Input Data Jabatan

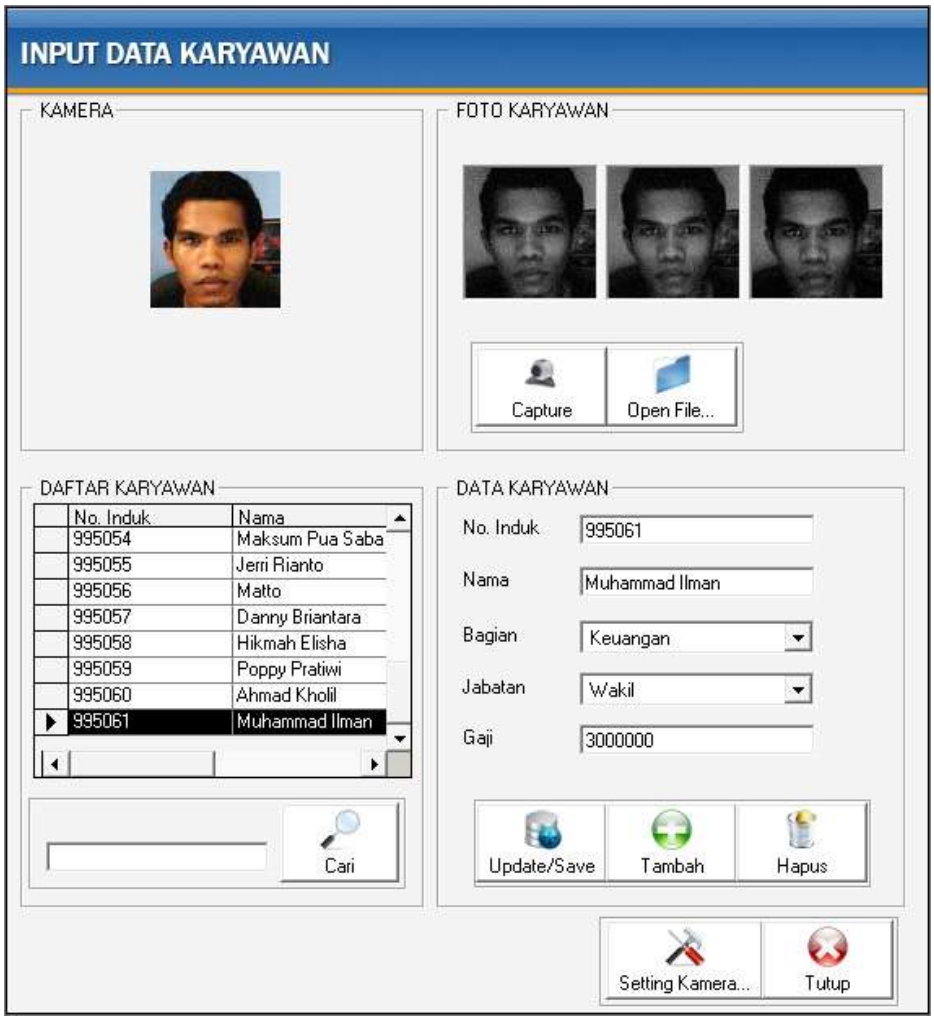
Submenu jabatan (Master Data→Jabatan) digunakan untuk mengisikan data untuk tabel jabatan. Pada *form* ini juga, nama jabatan dan keterangan deskripsi jabatan tersebut diinputkan. Sedangkan id_jabatan dihasilkan secara otomatis pada tabel jabatan karena *field* ini *betipe autonumber*.



Gambar 4. 4 Tampilan Form Pengisian Tabel Jabatan

3. Input Data Karyawan

Untuk mengisi data karyawan digunakan submenu karyawan (Master Data→Karyawan). *Form* ini juga digunakan untuk mengambil gambar wajah dari karyawan.

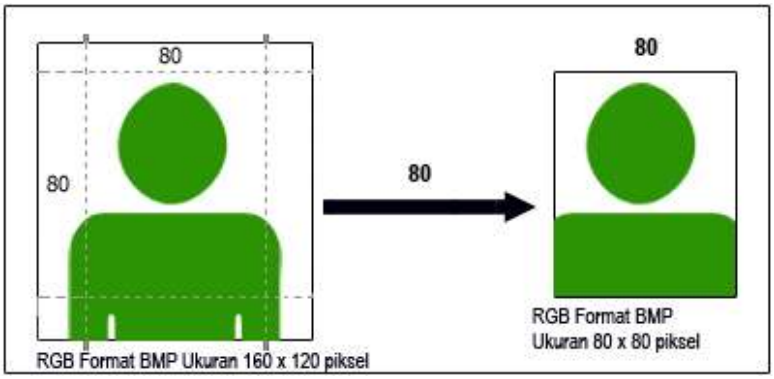


Gambar 4. 5 Tampilan Form Pengisian Data Karyawan

Terdapat dua proses yang terjadi pada *form* ini yaitu, melakukan peng-*input*-an data karyawan (data non-citra) dan peng-*input*-an data citra wajah karyawan. Data karyawan yang di-*input* akan disimpan dalam tabel karyawan sedangkan data citra wajah karyawan akan tersimpan pada *database* citra wajah. Selain dengan cara meng-*captur* juga dapat langsung diambil pada file yang telah disiapkan. Foto karyawan akan disimpan ke dalam folder *image* dengan tipe .jpg, dengan ukuran 80 x 80 piksel. Adapun urutan proses normalisasi gambarnya adalah sebagai berikut :

Gambar di-*capture* dengan menggunakan *webcam*, dihasilkan citra wajah bertipe .bmp dengan format RGB, dengan resolusi sesuai resolusi kamera pada saat proses *captur*-ing. Gambar yang disimpan harus memiliki ukuran yang seragam. Disini dipilih ukuran 80 x 80 sehinggan gambar harus dinormalisasikan.

Citra wajah yang diperoleh dari *webcam* (misalnya berukuran 160 x 120 piksel) di-*croping* sehingga mempunya ukuran 80 x 80 piksel.



Gambar 4. 6 Proses Cropping ke Ukuran 80 x 80 Piksel

Source code untuk cropping dapat dilihat dibawah ini :

```
Public Sub Crop(pict As PictureBox, filename As String)
   On Error Resume Next

   pict.Picture = LoadPicture(filename)

   x = pict.ScaleWidth
   y = pict.ScaleHeight


   For i = 20 To 99
      For j = 40 To 119
         pixel = pict.Point(j, i)
         red = pixel& Mod 256
         green = ((pixel And &HFF00) / 256&) Mod 256&
         blue = (pixel And &HFF0000) / 65536
         ImagePixels(0, i, j) = red
         ImagePixels(1, i, j) = green
         ImagePixels(2, i, j) = blue
      Next
   Next

   pict.height = 1253
   pict.width = 1253

   pict.ScaleWidth = 80
   pict.ScaleHeight = 80

   pict.Picture = Nothing

   For i = 20 To 99
      For j = 40 To 119
         pict.PSet (j - 40, i - 20), RGB(ImagePixels(0, i, j), ImagePixels(1, i, j),
ImagePixels(2, i, j))
      Next
      pict.Refresh
   Next

End Sub
```

Setelah dihasilkan gambar berukuran 80 x 80 piksel maka gambar RGB format .bmp tadi diubah kedalam format *grayscale*, dengan tujuan agar mempunyai 1 matriks yang berkorespondensi dengan 1 image saja. Adapun *source code*-nya dapat dilihat dibawah ini :

```
Public Sub GrayScale(pict As PictureBox)
   On Error Resume Next

   x = pict.ScaleWidth
   y = pict.ScaleHeight

   For i = 0 To y - 1
      For j = 0 To x - 1
         pixel = pict.Point(j, i)
         red = pixel& Mod 256
         green = ((pixel And &HFF00) / 256&) Mod 256&
         blue = (pixel And &HFF0000) / 65536

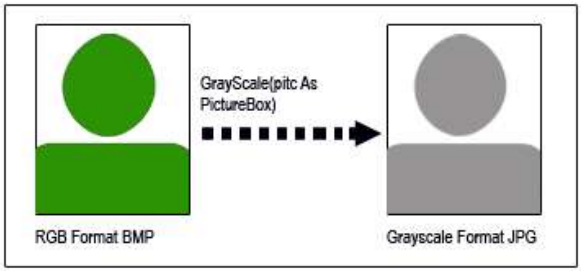
         ' my algorithm to grayscale picture
         ' is it correct?

         red = (red * 5) \ 10
         green = (green * 8) \ 10
         blue = (blue * 3) \ 10
         gray = ((red + green + blue) * 10) \ 16

         pict.PSet (j, i), RGB(gray, gray, gray)
      Next
      pict.Refresh
   Next

End Sub
```

Hasil dari langkah ini adalah citra *grayscale* ukuran 80 x 80 piksel tetapi masih dalam bentuk .bmp.



Gambar 4. 7 Proses Konversi ke format Grayscale

Langkah berikutnya tinggal menyimpan format yang telah didapat dalam format .jpg. untuk itu diperlukan *source code* sebagai berikut :

```
Public Sub SavePhoto(pict As PictureBox, index As Integer)
   On Error Resume Next
   SavePicture pict.image, App.Path & PictName & index & ".bmp"
   ConvertBMPtoJPG App.Path & PictName & index & ".bmp", 80, False
   MsgBox "File telah tersimpan", , App.Title
End Sub
```

Hal yang sama berlaku jika foto karyawan di-*input*-kan melalui pengambilan file gambar dari direktori yang telah ditetapkan.

4.1.4.2 Proses Presensi

Paroses presensi karyawan menggunakan *webcam* untuk meng-*capture* gambar wajah seperti pada saat pengisian foto untuk data karyawan. Hasil *capture* kemudian dicocokkan dengan foto karyawan yang ada pada database *image* dan dilakukan proses identifikasi dengan langkah sebagai berikut :

1. *Capture citra* wajah karyawan.

```
'stage 1 - capture camera
      Capture ezVC
      Crop Picture6, App.Path & CaptName
      Kill App.Path & CaptName
      GrayScale Picture6
```

2. Lakukan proses *matching* dengan citra wajah pada *database*.

Paroses *matching* dilakukan oleh prosedur matcpitc(picture5,picture6). Printahnya sebagai berikut :

```
Public Function MatchPict(pict1 As PictureBox, pict2 As PictureBox) As Integer
   Dim closestImage As Integer

   pict2.Picture = Nothing
   closestImage = Val(faces.Identify(pict1))

   If ((closestImage <> 0) And (Dir$(App.Path & PictName & closestImage & ".jpg") <>
"")) Then
      pict2.Picture = LoadPicture(App.Path & PictName & closestImage & ".jpg")
   Else
      MsgBox "Tidak ditemukan kemiripan", vbExclamation, App.Title
   End If
   MatchPict = closestImage
End Function
```

Hasil *capture* pada saat presensi disimpan pada variable pict1, kemudian proses *matching* dilakukan denga memanggil baris printah `closestImage = Val(faces.Identify(pict1)`. Dimana *faces* adalah objek yang merupakan *instance* dari *class face recogniser* dan *identify* adalah salah satu propertinya sedangkan *class face recogniser* adalah kelas yang digunakan untuk melakukan deteksi wajah.

Pertama kali klas akan menghitung rataan *eigenvector*. Sebelumnya semua citra wajah yang ada dalam *database image* diambil dan dibuat *eigenvector*-nya denga subprogram *face template*(). Kemudian dilakukan perhitungan rataan *eigenvector*-nya dan disimpan dalam *face template*(), dengan *source* perintahnya sebagai berikut :

```
Private Sub updateFaceTemplate()
'calculates an average face template
 Dim i As Integer
 Dim x As Integer
 Dim y As Integer

 For i = 0 To NoOfFaces - 1
  For x = 0 To imageWidth - 1
   For y = 0 To imageHeight - 1
    If (i > 0) Then
     faceTemplate(x, y) = faceTemplate(x, y) + Face(i).getPoint(x, y)
     Else
     faceTemplate(x, y) = Face(i).getPoint(x, y)
    End If
   Next
  Next
 Next

 For x = 0 To imageWidth - 1
  For y = 0 To imageHeight - 1
   faceTemplate(x, y) = Int(faceTemplate(x, y) / NoOfFaces)
  Next
 Next
End Sub
```

Rataan *eigencector* yang tersimpan dalam *face template*() akan digunakan untuk menghitung nilai *eigen* (*eigenfaces value*), untuk semua *image* yang ada dalam *database image*. Perhitungan dilakukan dengan :

```
Private Sub updateEigenFaces()
'updates all the eigenfaces
 Dim i As Integer
 Dim x As Integer
 Dim y As Integer
 Dim df As Integer

 For i = 0 To NoOfFaces - 1
  For x = 0 To imageWidth - 1
   For y = 0 To imageHeight - 1
    df = Face(i).getPoint(x, y) - faceTemplate(x, y)
    If (df < 0) Then
     df = 0
    End If
    Call EigenFace(i).setPoint(x, y, CByte(df))
   Next
  Next
 Next
End Sub
```

Nilai *eigenface* untuk semua wajah disimpan dalam *face*(i). Tinggal dilakukan identifikasi jika ada citra wajah yang masuk pada saat presensi dilakukan. Proses identifikasi dilakukan sebagai berikut :

Citra wajah yang di-*capture* pada saat presensi disimpan dalam *testface*. Citra wajah ini kemudian dikalkulasi nilai *eigenface*-nya berdasarkan *face template* yang tadi telah dihitung. Jika *eigenface* dari *testface* sudah dikalkulasi maka tinggal menghitung jarak dengan nilai yang disimpan dalam *face*(*i*). jika ditemukan bahwa *i* yang berkorespondensi maka *i* menujukan urutan citra wajah hasil identifikasi. Perhitungannya sebagai berikut :

```
'calculate the eigenface
For x = 0 To imageWidth - 1
  For y = 0 To imageHeight - 1
    df = testFace.getPoint(x, y) - faceTemplate(x, y)
    If (df < 0) Then
      df = 0
    End If
    Call testEigenFace.setPoint(x, y, CByte(df))
  Next
Next

'compare it to other eigenfaces
minDistance = ((100 - MinimumEigen) * 99999999#) / 100

'MsgBox minDistance

For i = 0 To NoOfFaces - 1
  Distance = 0
  For x = 0 To imageWidth - 1
    For y = 0 To imageHeight - 1
      a = EigenFace(i).getPoint(x, y)
      b = testEigenFace.getPoint(x, y)
      df = Abs(a - b)
      Distance = Distance + df
    Next
  Next

  If (Distance < minDistance) Then
    minDistance = Distance
    retval = NameOfFace(i)
    Set Identity = Face(i)
  End If
Next

Identify = retval
```

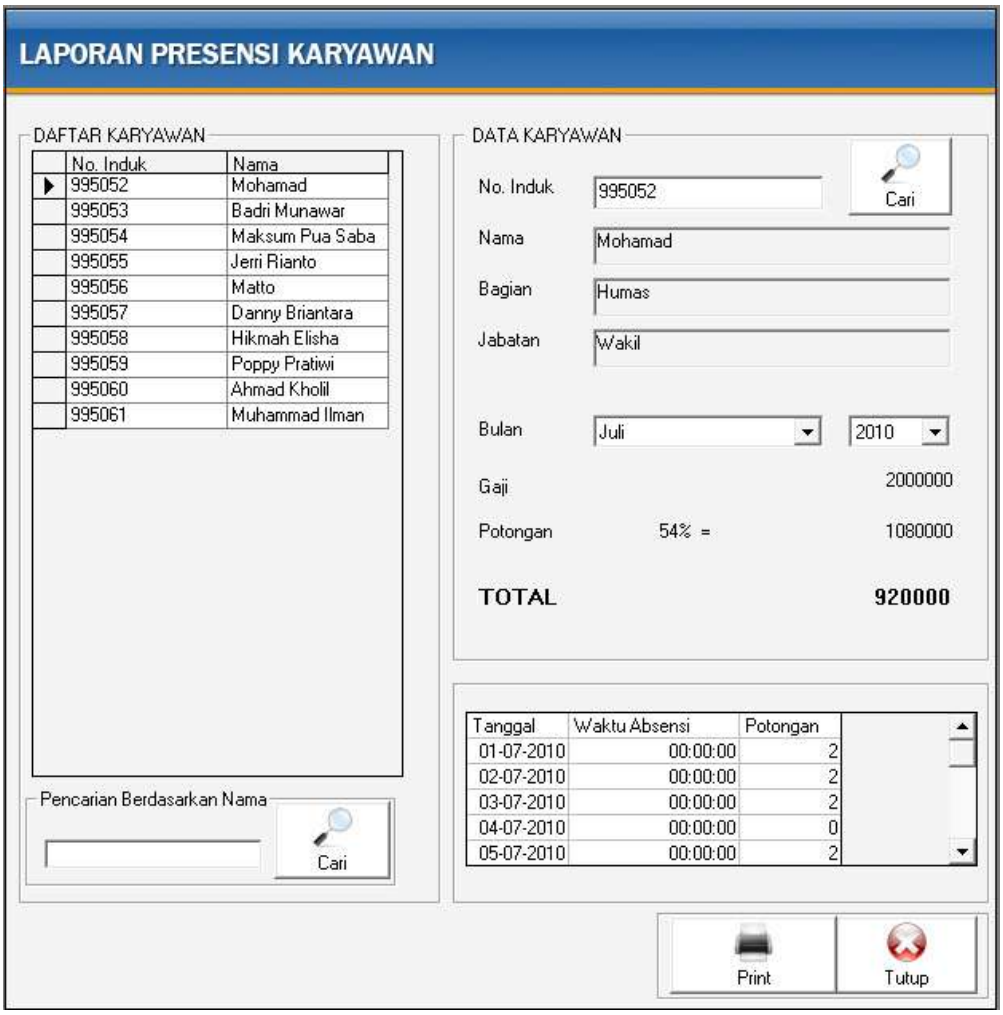
3. Identifikasikan data karyawan berdasarkan identifikasi wajah.

Proses identifikasi dilakukan saat wajah di-*capture* dan disimpan dalam *testface*. Citra wajah ini kemudian dikalkulasi nilai *eigenface*-nya berdasarkan *face template* yang tadi telah dihitung. Jika *eigenface* dan *testface* sudah dikalkulasi maka tinggal menghitung jarak dengan nilai-nilai yang telah tersimpan dalam *face*(*i*). jika ditemukan bahwa *i* yang berkorespondensi maka *i* menunjukan citra wajah hasil identifikasi.

Jika citra yang sesuai dengan *testface* sudah didapatkan maka tinggal mencari file yang sesuai dengan yang tersimpan pada *database image*. Disinilah *field* id_karyawan berguna. Nama file citra wajah sebenarnya diambil dari id_karyawan. Misalnya file pitc7.jpg akan berkorespondensi dengan karyawan yang id_karyawannya bernilai 7. Jika id_karyawan sudah teridentifikasi maka tinggal diambil semua data pada *record* karyawan dengan id_karyawan tersebut.

4.1.4.3 Proses Pembuatan Laporan

Untuk menu laporan prsensi hanya ada 1 submenu, yaitu laporan presensi. *Form* ini digunakan untuk melihat presensi yang dilakukan oleh seorang karyawan dalam waktu 1 bulan. *Form* ini juga dilengkapi dengan fasilitas pencarian karyawan berdasarkan nama karyawan. *Form* ini juga dilengkapi dengan fasilitas pencetakan rekap presensi per karyawan untuk bulan dan tahun tertentu. Berikut ini tampilan dari *form* laporan presensi :



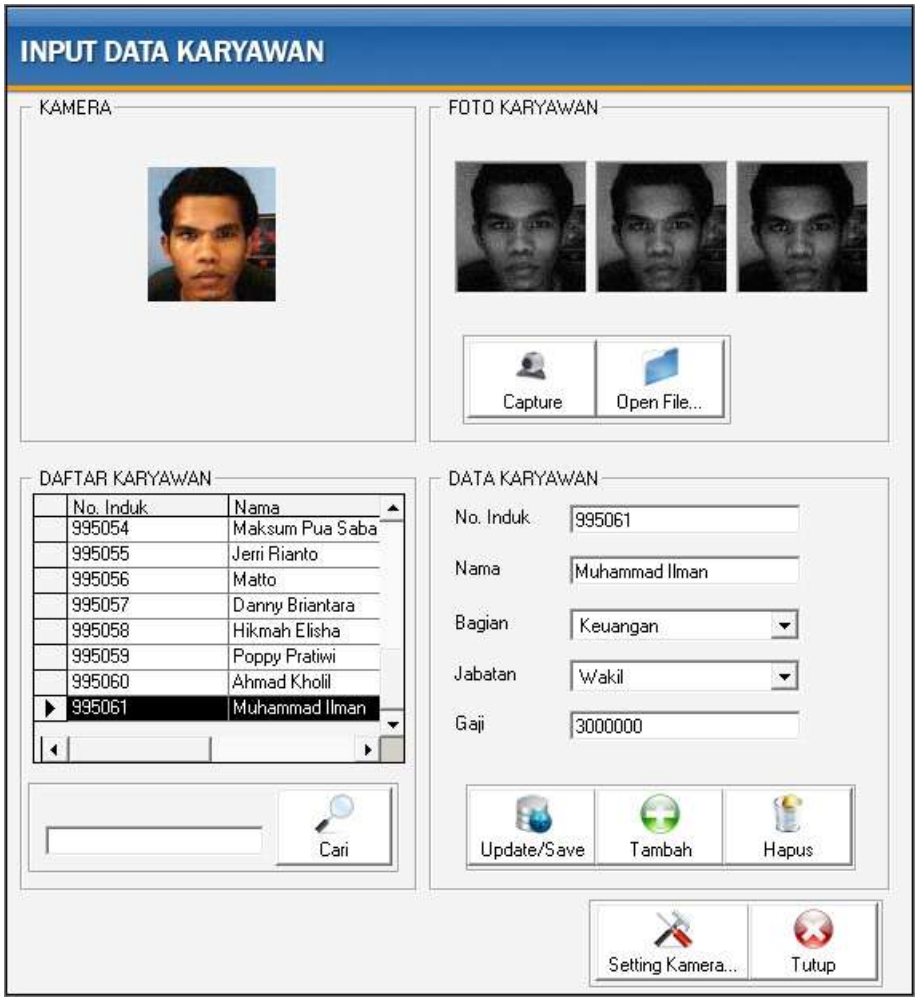
Gambar 4. 8 Form Laporan Presensi Karyawan.

4.2 Pengujian Sistem

Pengujian sistem dilakukan untuk mengintegrasikan metode perancangan *software* ke dalam langkah-langkah terencana yang tersusun rapih sehingga menghasilkan konstruksi *software* yang sukses.

4.2.1 Database Citra Wajah

Hal pertama yang harus dilakukan sebelum pengujian sistem adalah memasukkan terlebih dahulu citra wajah yang diperlukan untuk menguji dan memastikan bahwa sistem benar-benar mengenali wajah karyawan yang melakukan pencatatan presensi. Citra wajah yang disimpan adalah citra wajah dengan pose menghadap ke depan dan ekspresi standar, tidak tersenyum ataupun tertawa. Untuk memasukan data ini digunakan menu input data karyawan sebagai berikut:



Gambar 4. 9 Proses pemasukan data karyawan

Untuk pose standar yang dimaksud idealnya citra wajah yang disimpan ke dalam database memiliki pose menghadap lurus ke depan dengan pencahayaan yang cukup dan background berwarna putih/terang. Untuk peng-captur-an citra wajah yang akan dinormalisasi maka kondisi yang harus ditetapkan sebagai berikut:

1. Pose wajah lurus ke depan, ekspresi standar.
2. Bacground adalah tembok berwarna putih.
3. Pencahayaan untuk pengambilan semua citra karyawan dibuat sama.
4. Jarak kamera dan wajah yang di-capture dibuat konstan (30cm – 35cm).
5. Resolusi kamera yang dipilih adalah 120 x 160

Untuk keperluan pengetesan dilakukan beberapa percobaan dengan beberapa karakteristik. Karakteristik pertama, percobaan individual 1 subjek melakukan beberapa kali presensi dengan 8 kondisi yang berbeda. Karakteristik kedua, percobaan multisubjek, sekelompok subjek melakukan presensi secara berurutan pada 8 kondisi yang berbeda.

4.2.2 Hasil Percobaan

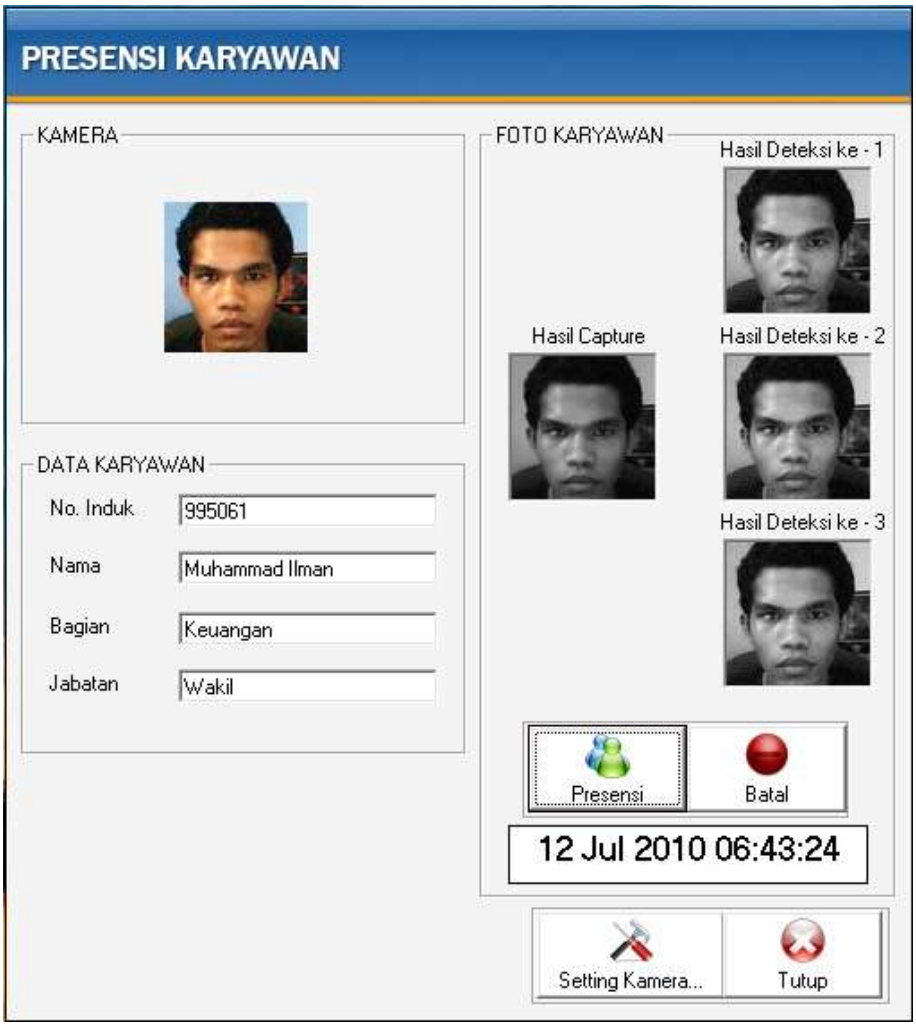
Untuk melakukan pengetesan sistem presensi ini dilakukan beberapa percobaan dengan beberapa karakteristik. Karakteristik pertama, percobaan individual, 1 subjek melakukan beberapa kali presensi dengan 8 kondisi yang berbeda dengan database berisi 20 citra wajah. Karakteristik kedua, percobaan multisubjek, sekelompok subjek melakukan presensi secara berurutan pada 8 kondisi yang berbeda.

Adapun kondisi-kondisi yang telah ditetapkan adalah sebagai berikut:

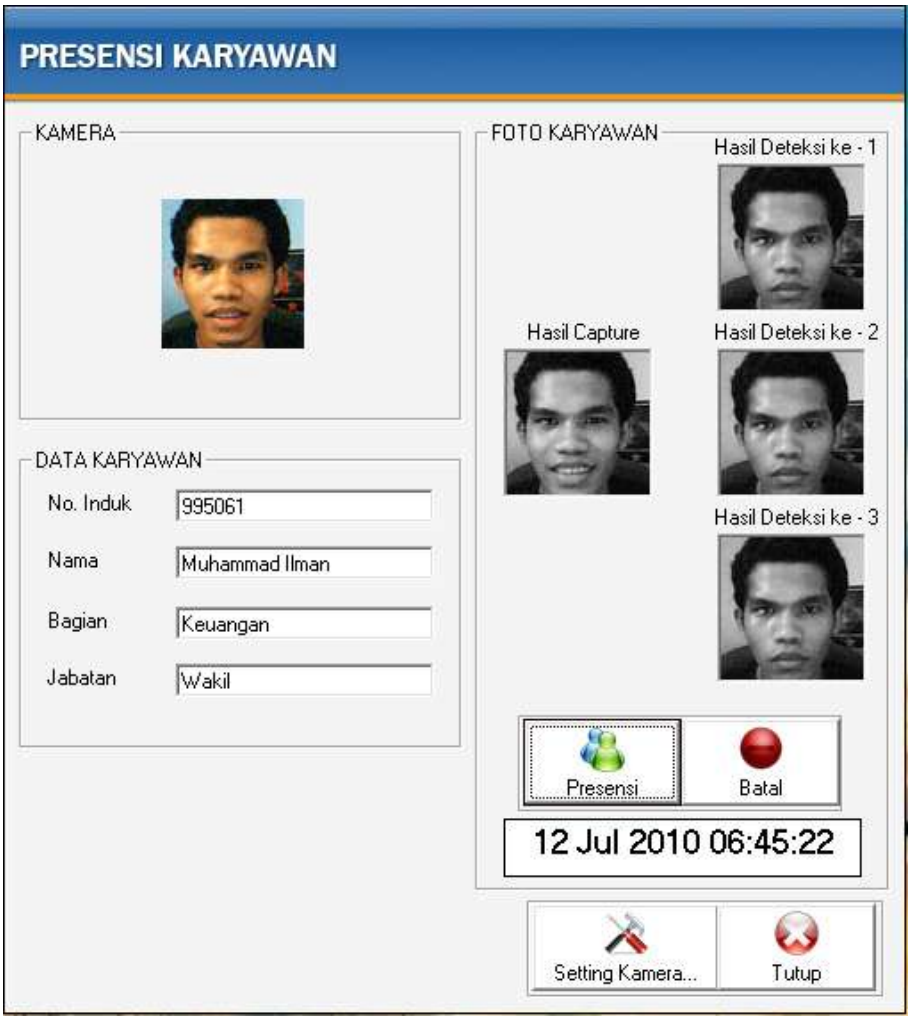
1. Kondisi 1: Pencahayaan tinggi, pose standar.
2. Kondisi 2: Pencahayaan tinggi, pose ekspresif.
3. Kondisi 3: Pencahayaan tinggi, pose miring ke kiri.
4. Kondisi 4: Pencahayaan tinggi, pose miring ke kanan.

5. Kondisi 5: Pencahayaan normal, pose standar.
6. Kondisi 6: Pencahayaan normal, pose ekspresif.
7. Kondisi 7: Pencahayaan normal, pose miring ke kiri.
8. Kondisi 8: Pencahayaan normal, pose miring ke kanan.

Kondisi–kondisi di atas dipilih dengan pertimbangan bahwa pengenalan wajah untuk keperluan presensi sangat dipengaruhi oleh pose wajah dan pencahayaan pada saat presensi dilakukan sehingga perubahan kondisi bisa mempengaruhi akurasi pengenalan wajah. Proses pengenalan dilakukan dengan form presensi sebagai berikut:



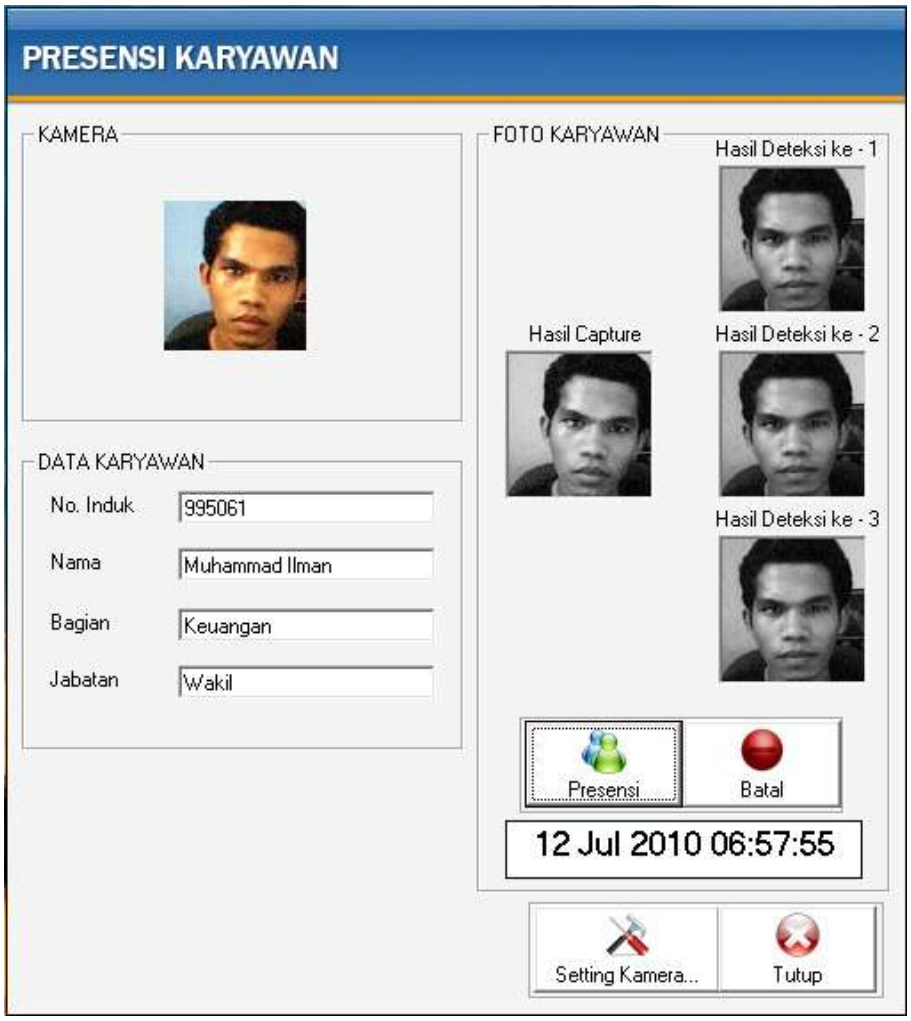
Gambar 4. 10 Pencahayaan Tinggi Pose Standar



Gambar 4. 11 Pencahayaan Tinggi Pose Ekspresif



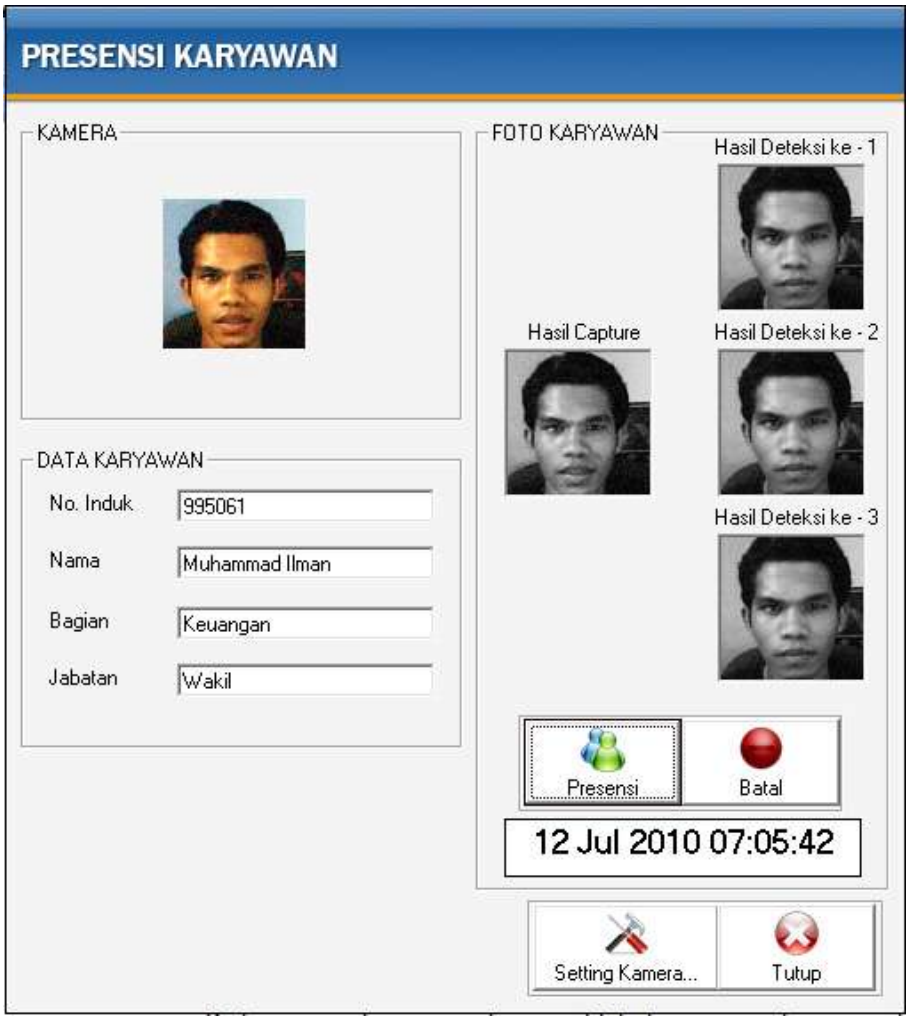
Gambar 4. 12 Pencahayaan Tinggi Pose Miring ke Kanan



Gambar 4. 13 Pencahayaan Tinggi Pose Miring ke Kiri



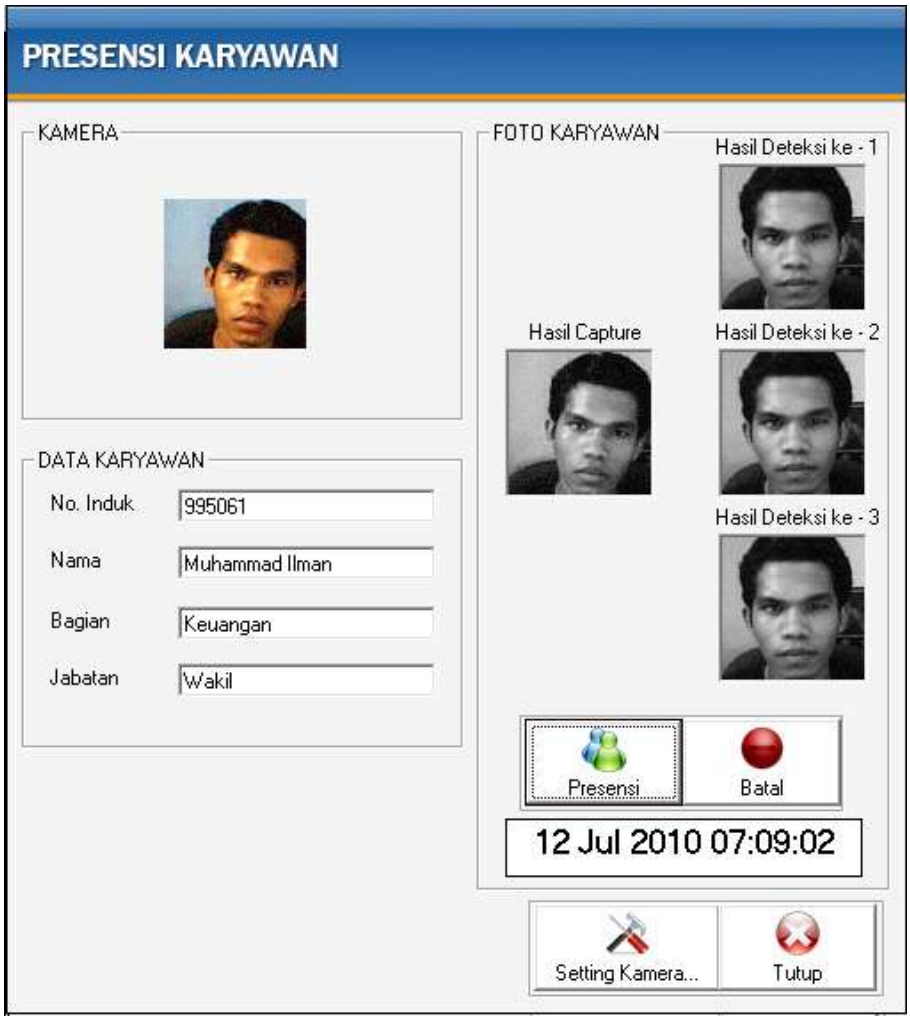
Gambar 4. 14 Pose Standar Dengan Pencahayaan Normal



Gambar 4. 15 Pose Ekspresif Pencahayaan Normal

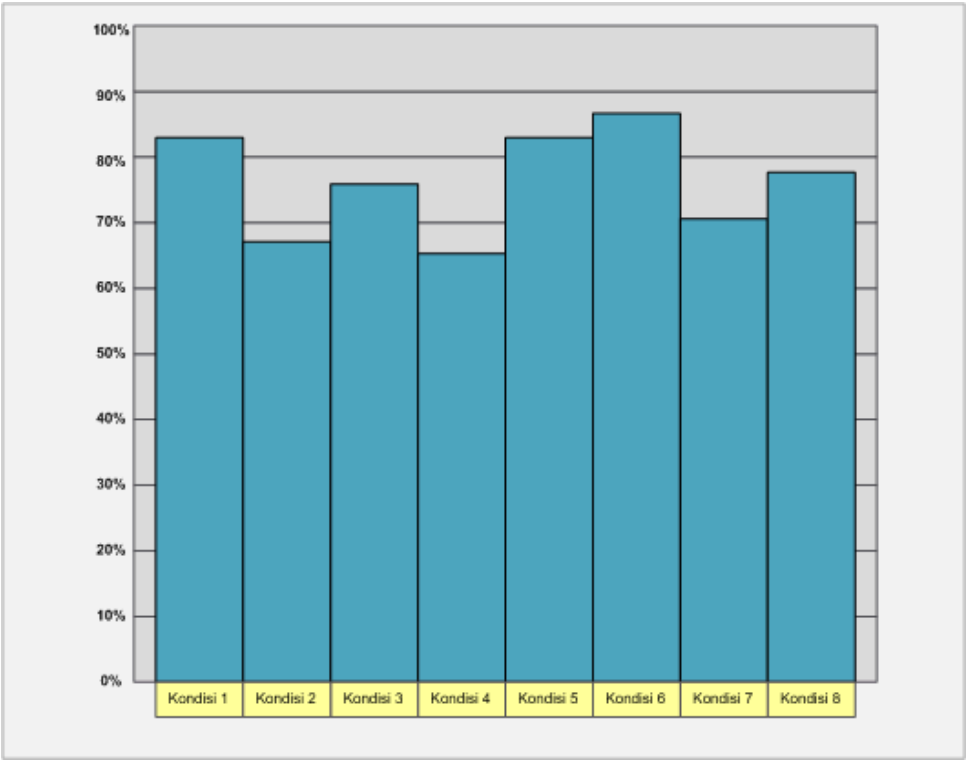


Gambar 4. 16 Pencahayaan Normal Pose Miring ke Kanan



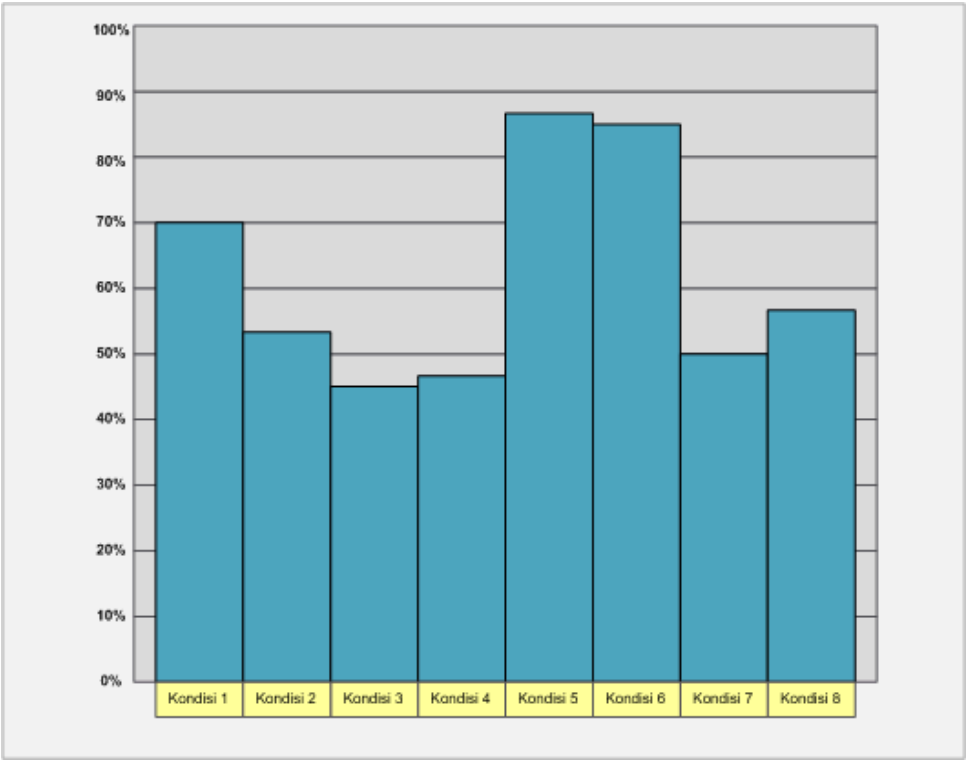
Gambar 4. 17 Pencahayaan Normal Pose Miring ke Kiri

Pada pengujian ini dilakukan percobaan individual dengan menggunakan database yang terdiri dari 20 citra wajah. Pada pengetesan ini jarak subjek dengan kamera dibuat konstan yaitu 30 cm. Adapun grafik hasil percobaan dapat dilihat sebagai berikut:



Gambar 4. 18 Grafik Hasil Pengujian Subjek Pertama

Adapun untuk subjek kedua didapatkan grafik hasil percobaan sebagai berikut:



Gambar 4. 19 Grafik Hasil Pengujian Subjek Kedua

Dengan percobaan yang telah dilakukan didapatkan beberapa analisis awal sebagai berikut:

1. Perubahan pose tidak terlalu mempengaruhi akurasi pengenalan, yang terlihat pada persentase pengenalan relatif sama
2. Pengenalan pada pencahayaan yang normal ternyata memberikan hasil yang lebih baik, ini disebabkan pada pencahayaan yang tinggi, detail wajah yang lebih jelas, sehingga kompleksitas warna wajah lebih tinggi. Kompleksitas ini menyebabkan sensitivitas pengenalan meningkan karena algoritma *eigenface* menitikberatkan pada ciri warna dari objek

Dari sini bisa ditentukan hasil analisisnya, yaitu bahwa keberhasilan pengenalan dipengaruhi beberapa hal, diantaranya:

1. Jarak antara subjek dan kamera .

   Untuk pengenalan yang baik, jarak antara subjek dan kamera saat pengisian data dan proses presensi sebaiknya dibuat konstan

2. Pencahayaan.

   Pencahayaan yang terlalu tinggi akan membuat detail gambar yang lebih rumit, mengakibatkan matriks dari gambar menjadi lebih kompleks, sehingga pengenalan menjadi lebih sensitive dan menurunkan kualitas pengenalan

3. Pose Wajah.

   Untuk pengenalan pose wajah dari diam ke ekspresif tidak terlalu mempengaruhi hasil pengenalan. Jika distorsi pose (miring ke kiri atau ke kanan), jika sudut kemiringan cukup banyak, akan menurunkan kualitas pengenalan.