PANDUAN INSTALASI SISTEM
KLASIFIKASI JENIS KELAMIN MANUSIA BERDASARKAN CITRA WAJAH MENGGUNAKAN 2DPCA DAN SOM

Disusun oleh :

Haris Angriawan

16.11.0339

PROGRAM SARJANA

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

2019

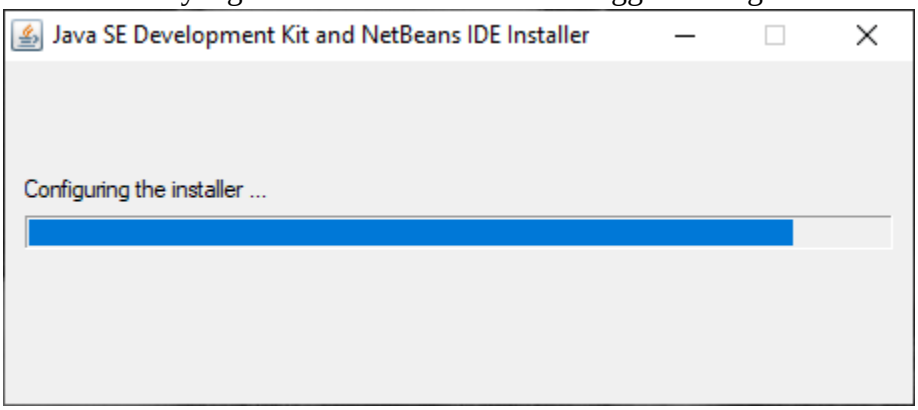
I. Pendahuluan

Sistem Klasifikasi Jenis Kelamin Manusia berdasarkan citra wajah menggunakan 2DPCA dan SOM merupakan aplikasi berbasis desktop yang berfungsi untuk pengolahan data citra wajah seorang menggukan algoritma *Two-Dimensional Principal Component Analysis* (2DPCA) dan *Self Organizing Maps* (SOM) yang berperan untuk mengambil keputusan apakah citra wajah tersebut laki-laki atau perempuan.

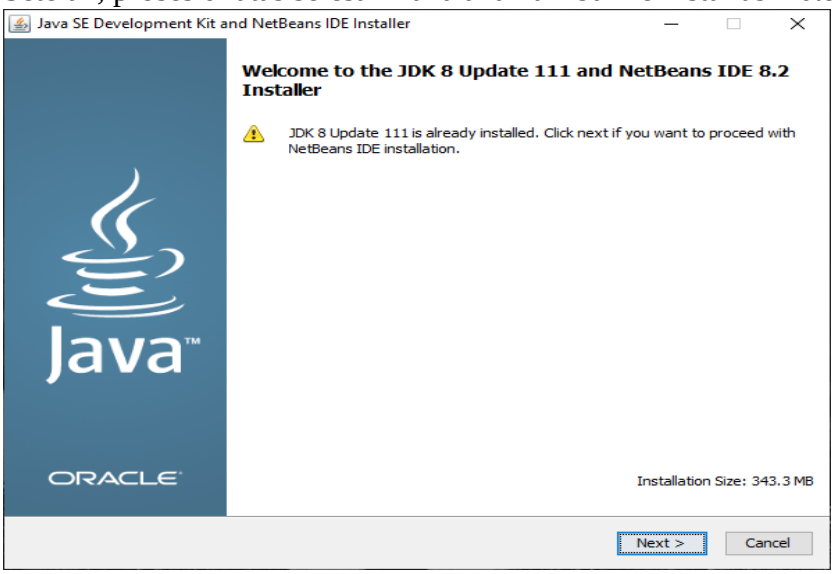
1. Instalasi *Netbeans 8.2*, yang berperan sebagai *tools* dari pembuatan aplikasi.
2. Download data citra wajah dari website http://www.facevar.com/ atau website lain sebagai data yang akan di gunakan untuk data training dan data uji.
3. Instalasi *2dpcasomgender* sebagai *source code* dari program.

II. Instalasi *Netbeans 8.2*

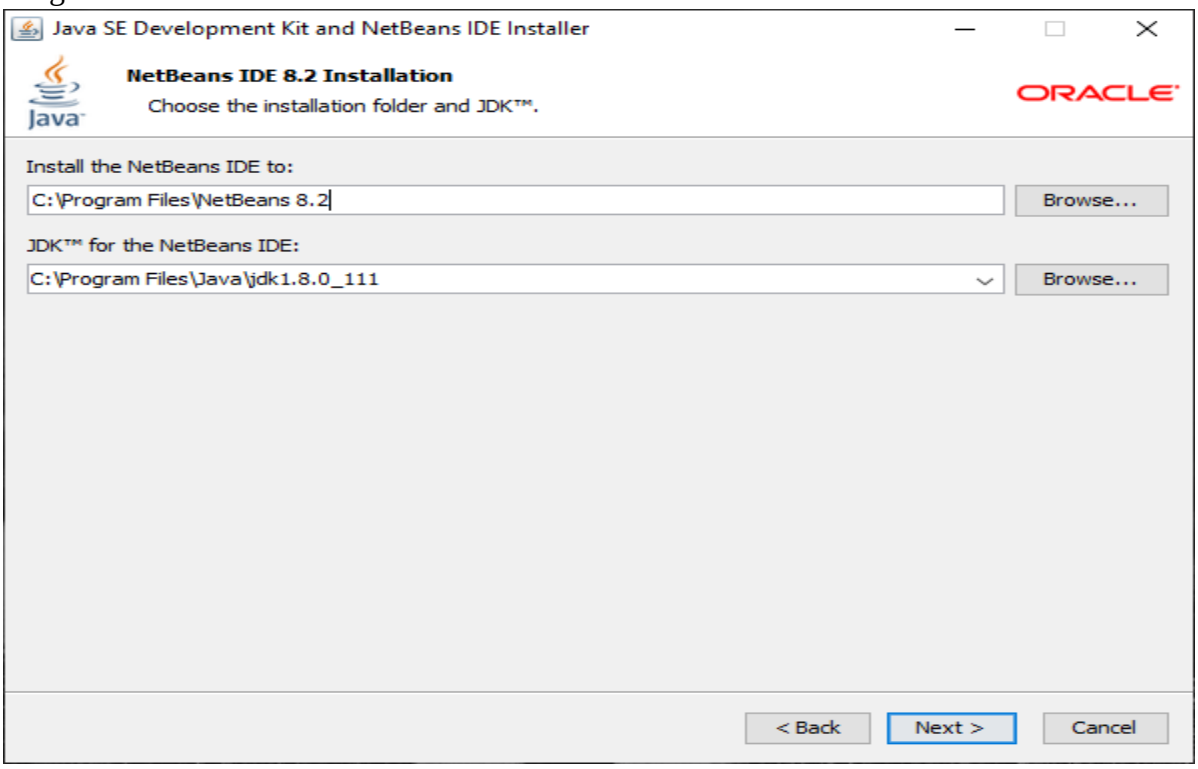
1. Download *netbeans 8.2 with jdk* melalui link berikut : https://www.oracle.com/technetwork/java/javase/downloads/jdk-netbeans-jsp-3413139-esa.html
2. Jalankan File yang sudah di download. Dan tunggu loading



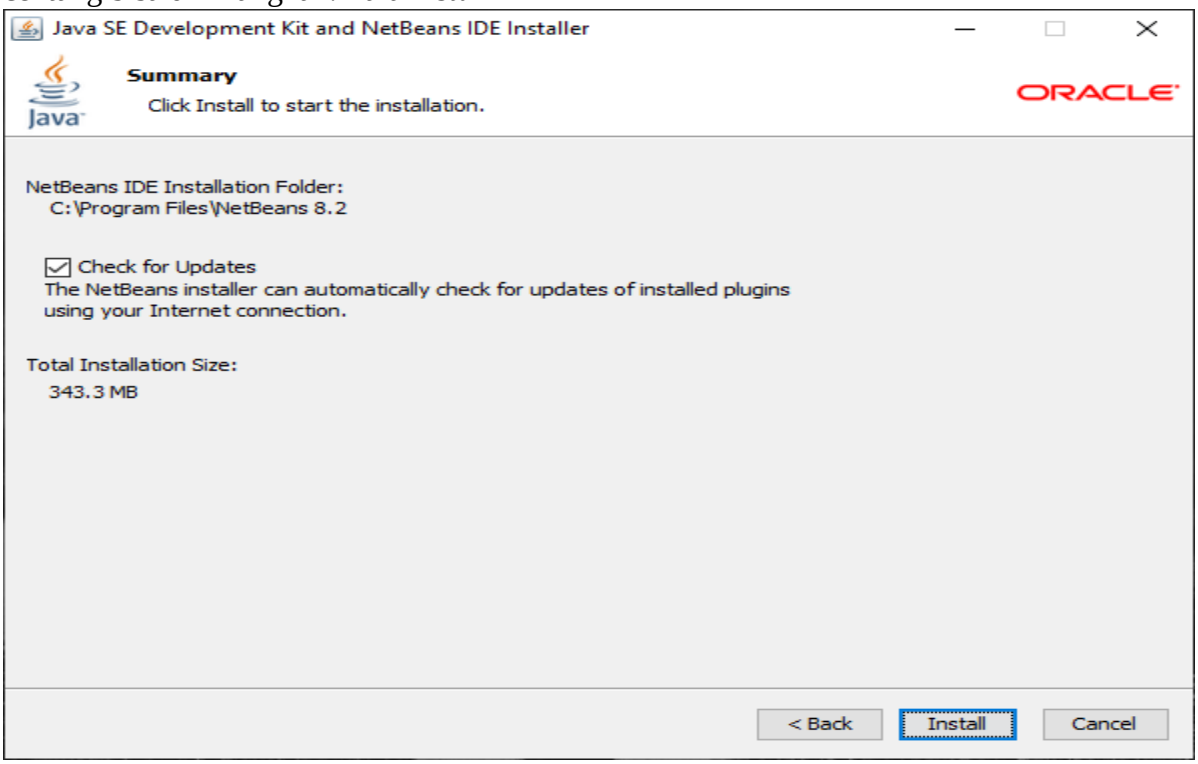
3. Setelah, proses di atas selesai maka akan timbul ke installasi *netbeans with jdk*. Lalu Next



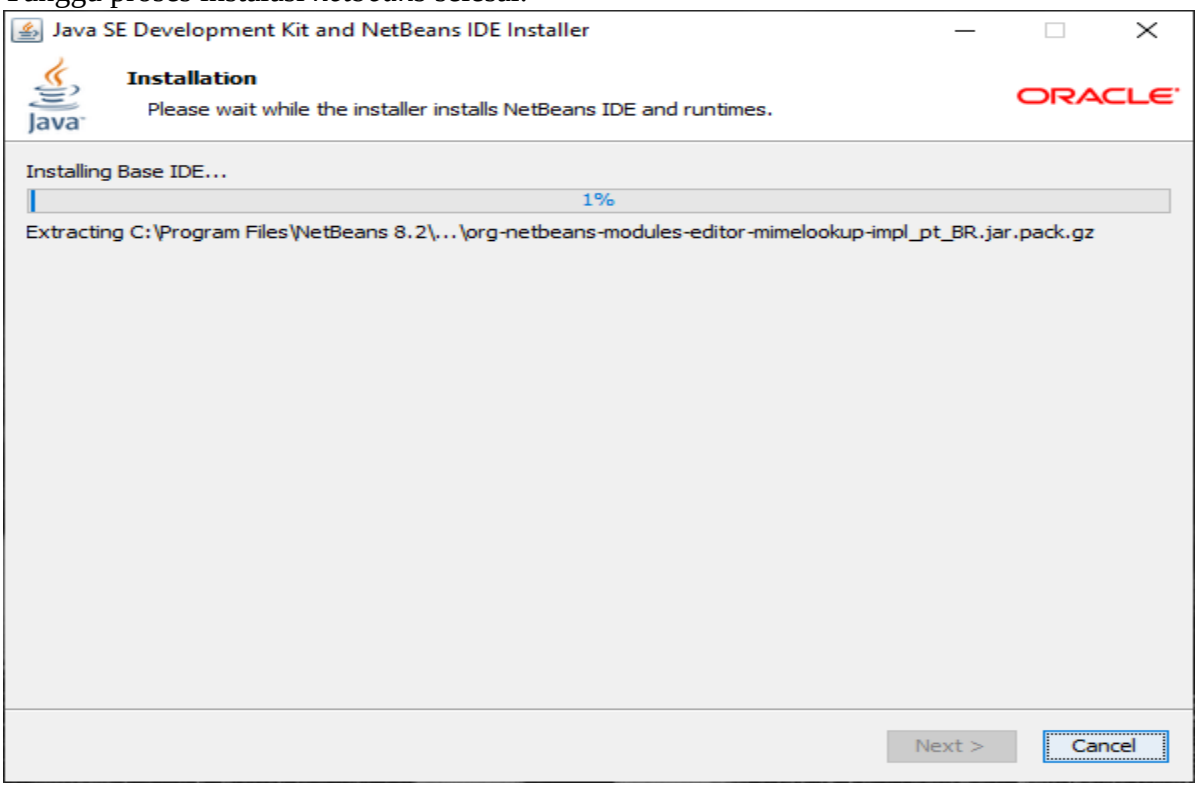
4. Pilih penyimpanan untuk *netbeans dan jdk*. Biarkan default agar *netbeans* bisa berjalan dengan baik. Lalu *Next*



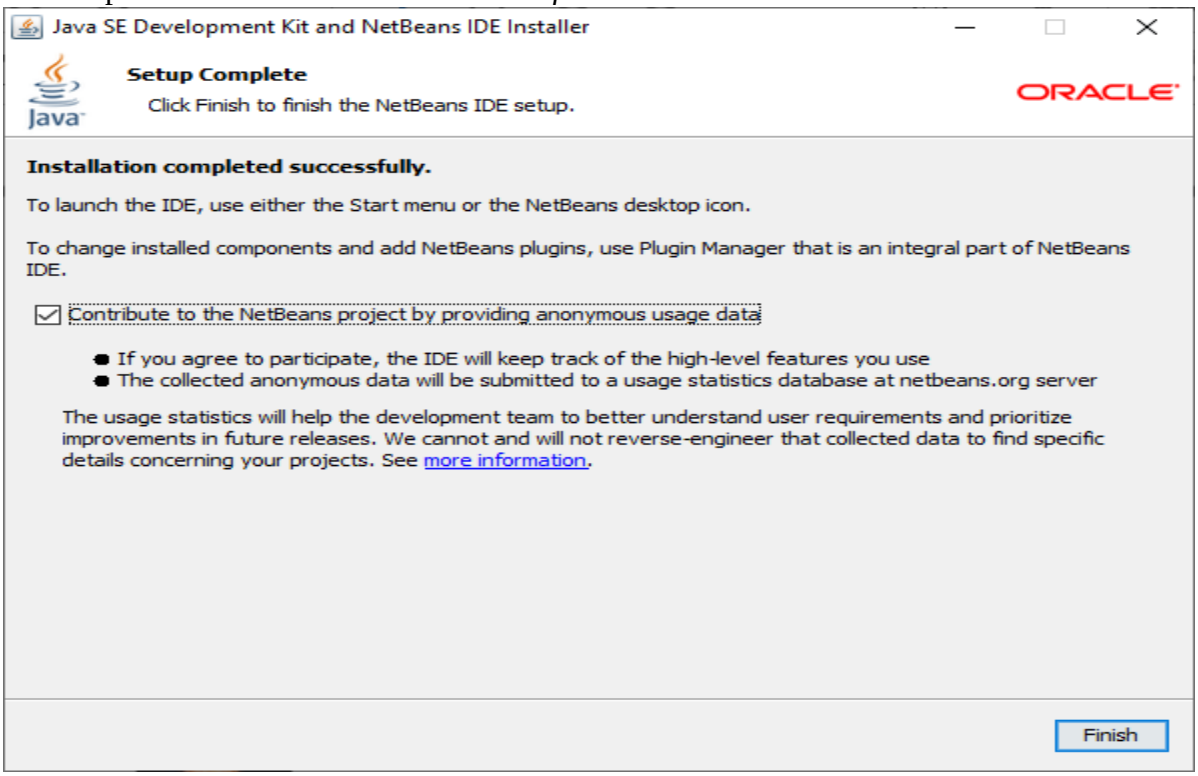
5. Lanjutkan proses dengan tombol *next.* Jika tidak ingin Melihat *update* dari netbeans, centang bisa di hilangkan. Lalu *install*



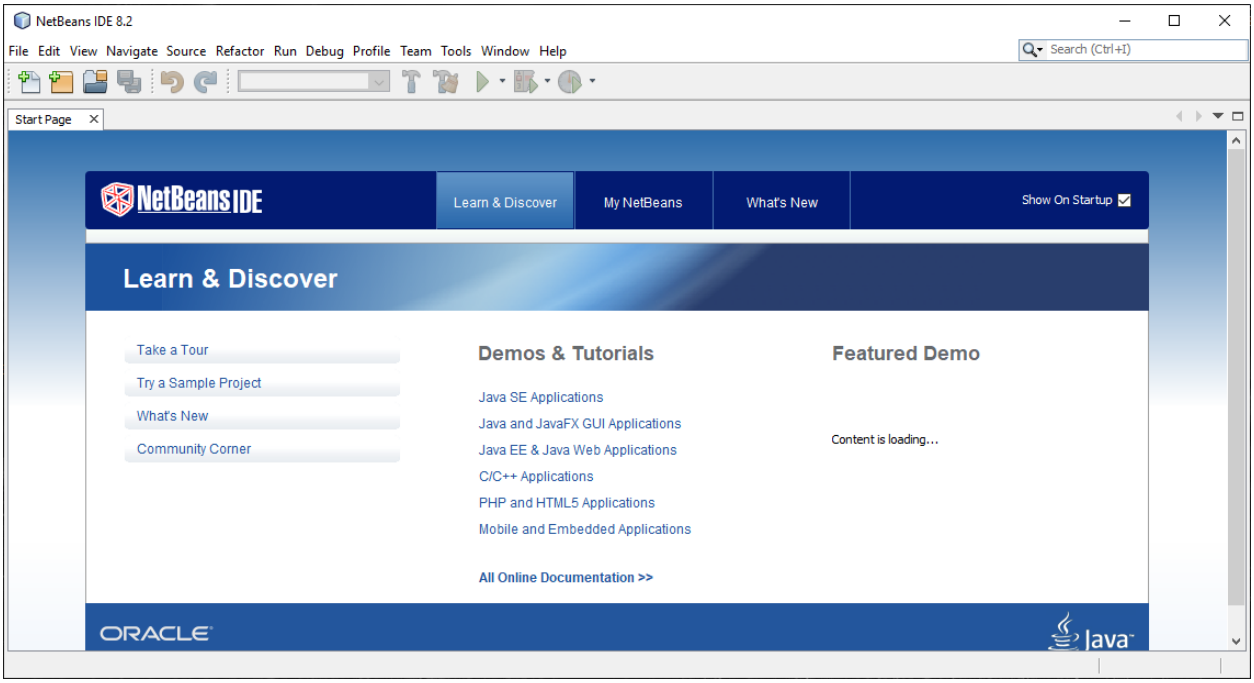
6. Tunggu proses instalasi *netbeans* selesai.



7. Setelah proses instalasi selesai. Lalu tekan *finish*



8. Buka *netbeans IDE 8.2*

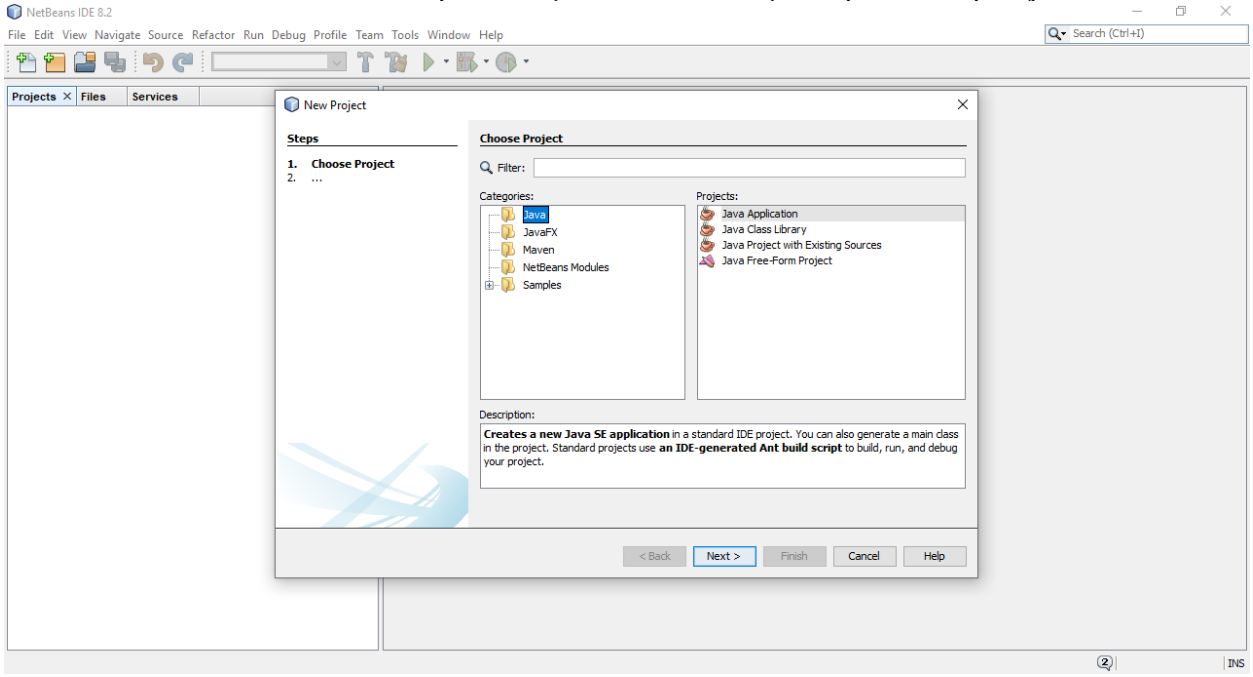


III. Data Training dan Data Uji

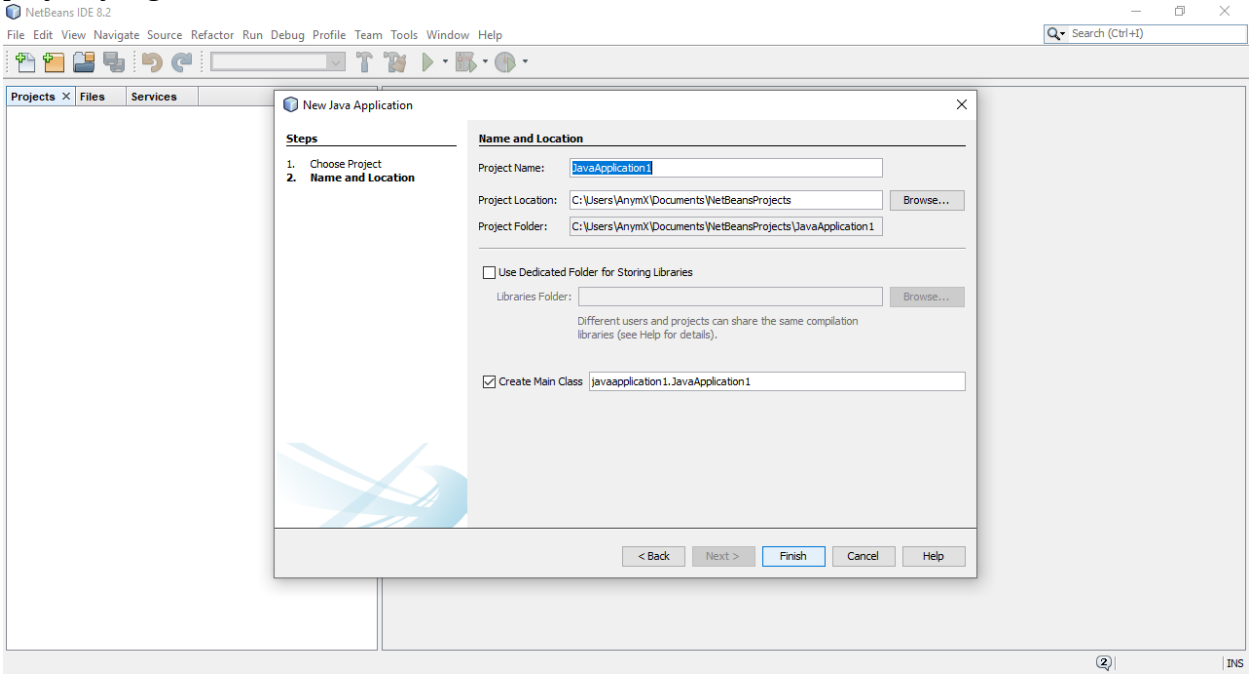
1. Unzip file GUFD yang sudah di download di website http://www.facevar.com/
2. Setelah selesai di extract, pilih beberapa data dari folder ..\3cameras_cropped\C1 yang terbagi 2 folder lagi untuk data citra dari laki-laki dan perempuan.
3. Pilih beberapa data citra wajah, misalkan 100 data citra wajah dengan rincian 50 laki-laki dan 50 citra wajah perempuan
4. Pisahkan file data citra wajah tersebut untuk data training dan data testing agar mempermudah
5. Dalam kasus ini, menggunakan 70 data training dan 30 data testing dengan rincian :
    - 70 data training merupakan data citra wajah yang akan di latih untuk mendapatkan bobot dari citra tersebut (35 data citra wajah laki-laki dan 35 citra wajah perempuan)
    - 30 data testing merupakan data citra wajah yang akan di gunakan untuk proses klasifikasi. Data tersebut merupakan data yang belum pernah di training (15 data citra wajah laki-laki dan 15 data citra wajah perempuan)

IV. Instalasi Source Code

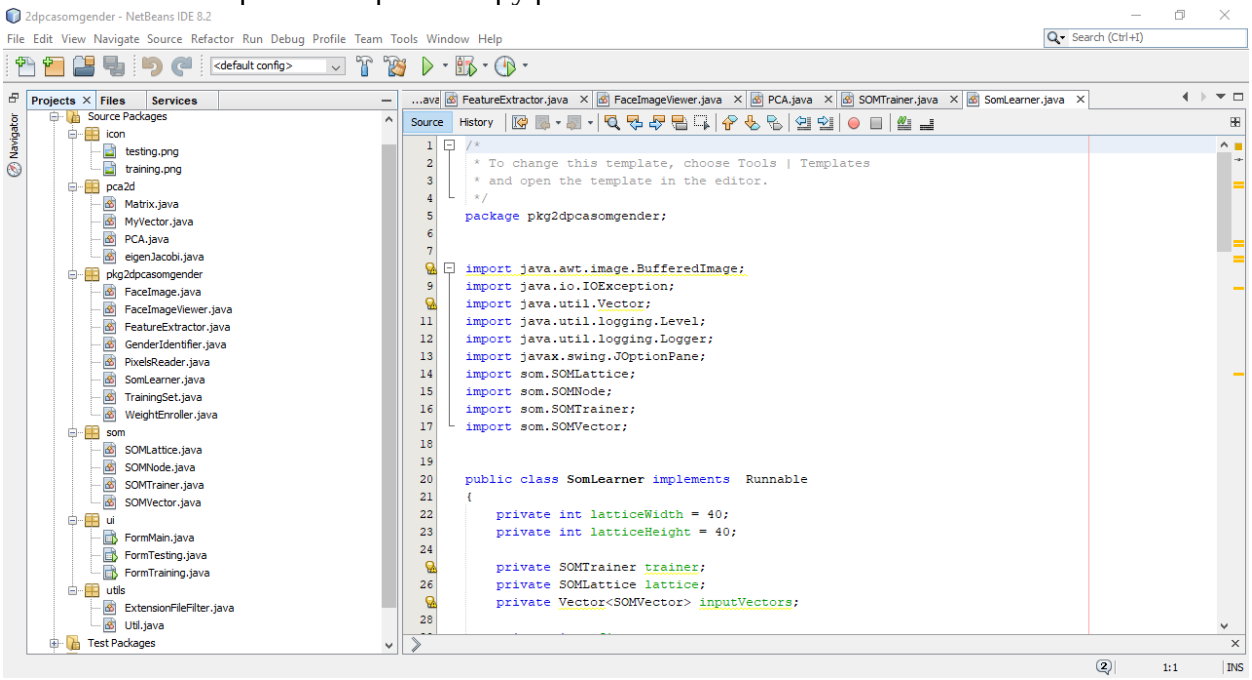
1. Buka Netbeans IDE 8.2, Lalu pilih file(sudut kanan atas) lalu pilih new peroject.



2. Pilih Seperti Gambar diatas, lalu next.
3. Buatlah nama project tersebut, misalkan 2dpcasomgender. Lalu pilih penyimpanan untuk project yang akan di buat. Lalu tekan finish.



4. Buatlah beberapa java package, dan Jframe sebagai UI untuk program. Seperti gambar di bawah untuk mempermudah proses copy paste source code.



5. Source Code 2dpcasomgender :

CLASS MATRIX

```
package pca2d;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;


public class Matrix {
   private int kolom;
   private int baris;
   private double nilai[][];

   public Matrix(int baris,int kolom)
   {
      this.baris = baris;
      this.kolom = kolom;
      nilai = new double[baris][kolom];
   }

   public Matrix(Matrix M)
   {
      nilai = new
double[M.getJumlahBaris()][M.getJumlahKolom
()];
      this.baris = M.getJumlahBaris();
      this.kolom  = M.getJumlahKolom();
      copy(M);
   }

   public Matrix(double[][] nilai)
   {
      this.baris = nilai.length;
      this.kolom = nilai[0].length;
      this.nilai = new double[baris][kolom];

      for (int x = 0; x < this.baris; x++) {
         for (int y = 0; y < this.kolom; y++)
         {
            this.nilai[x][y] = nilai[x][y];
         }
      }
   }

   public Matrix(int[] nilai, int baris, int kolom)
   {
      this.baris =baris;
      this.kolom = kolom;
      this.nilai = new double[baris][kolom];
      int i=0;
      for (int x = 0; x < this.baris; x++) {
         for (int y = 0; y < this.kolom; y++)
         {
            this.nilai[x][y] = nilai[i];
            i++;
         }
      }
   }

   public Matrix(double[] nilai, int baris, int
kolom)
   {
      this.baris =baris;
      this.kolom = kolom;
      this.nilai = new double[baris][kolom];
      int i=0;

      for (int x = 0; x < this.baris; x++) {
         for (int y = 0; y < this.kolom; y++){
            this.nilai[x][y] = nilai[i];
            i++;
         }
      }
   }

   public Matrix(String tipe, int baris,int kolom) {
      if (tipe.equals("identitas")) {
         this.baris = baris;
         this.kolom = kolom;
         nilai = new double[baris][kolom];

         for (int i = 0; i < baris; i++) {
            nilai[i][i] = 1;
         }
      }
   }

   public int getJumlahBaris() {
      return baris;
   }

   public void setJumlahBaris(int baris) {
      this.baris = baris;
```

```
    }

    public int getJumlahKolom() {
        return kolom;
    }

    public void setJumlahKolom(int kolom) {
        this.kolom = kolom;
    }

    public double[][] getNilai() {
        return this.nilai;
    }

    public double getNilai(int baris, int kolom) {
        return this.nilai[baris][kolom];
    }

    public void setNilai(double nilai, int baris, int
kolom) {
        this.nilai[baris][kolom] = nilai;
    }

    public void setNilaiBaris(double[] nilai, int
kolom) {
        for (int i = 0; i < this.baris; i++) {
            this.nilai[i][kolom] = nilai[i];
        }
    }

    public void setNilaiKolom(double[] nilai, int
baris) {
        for (int i = 0; i < this.kolom; i++) {
            this.nilai[baris][i] = nilai[i];
        }
    }

    public double[] getNilaiBaris(int baris) {
        double[] dataBaris = new double[this.baris];
            for (int i = 0; i < this.baris; i++) {
                dataBaris[i] = nilai[i][baris];
            }
        return dataBaris;
    }

    public double[] getNilaiKolom(int kolom) {
        double[] dataBaris = new
double[this.kolom];
        for (int i = 0; i < this.kolom; i++) {
            dataBaris[i] = nilai[kolom][i];
        }
        return dataBaris;
    }

    public double [] getDiagonal(){
        if(baris == kolom){
            double [] nilaiDiagonal = new
double[baris];
            for (int i = 0; i < baris; i++){
                nilaiDiagonal[i] = nilai[i][i];
            }
            return nilaiDiagonal;
        }
        else{
            return null;
        }
    }

    public Matrix transpose() {
        Matrix transpose = new Matrix(kolom,
baris);
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                transpose.nilai[j][i] = this.nilai[i][j];
            }
        }
        return transpose;
    }

    public Matrix kaliMatriks(Matrix B) {
        if (B.baris != kolom) {
            throw new
IllegalArgumentException("Tidak Memenuhi
Persyaratan Perkalian");
        }
        Matrix X = new Matrix(baris, B.kolom);
        double[][] C = X.getNilai();
        double[] Bcolj = new double[kolom];

        for (int j = 0; j < B.kolom; j++) {
            for (int k = 0; k < kolom; k++) {
                Bcolj[k] = B.nilai[k][j];
            }
            for (int i = 0; i < baris; i++) {
                double[] Arowi = nilai[i];
                double s = 0;
```

```
            for (int k = 0; k < kolom; k++) {
                s += Arowi[k] * Bcolj[k];
            }
            C[i][j] = s;
        }
    }
    return X;
  }

  public Matrix kali(double nilaiPerkalian) {
    Matrix hasil = new Matrix(baris, kolom);
    for (int i = 0; i < hasil.baris; i++) {
      for (int j = 0; j < hasil.kolom; j++) {
        hasil.setNilai(nilaiPerkalian * nilai[i][j],
i, j);
      }
    }
    return hasil;
  }

  public void copy(Matrix asal) {
    if (nilai[0].length == asal.kolom &&
nilai.length == asal.getJumlahBaris()) {
      for (int i = 0; i < nilai.length; i++) {
        for (int j = 0; j < nilai[0].length; j++) {
          nilai[i][j] = asal.getNilai(i, j);
        }
      }
    }
  }

  public Matrix PotongKolom(int mulai, int
akhir)
  {
    Matrix hasil = new Matrix(baris,akhir-
mulai);
    System.out.println("Baris : " + baris);
    for (int i = 0; i < baris; i++)
    {
      for (int j = mulai; j < akhir; j++)
      {
        hasil.setNilai(nilai[i][j], i, j-mulai);
      }
    }
    return hasil;
  }

  public Matrix PotongBaris(int mulai, int
akhir){
    Matrix hasil = new Matrix(akhir - mulai,
kolom);
    int barisPindah = 0;
    for (int i = mulai; i < akhir; i++){
      for (int j = 0; j < kolom; j++){
        hasil.setNilai(nilai[i][j], barisPindah, j);
      }
      barisPindah++;
    }
    return hasil;
  }

  public double[] getData()
  {
    double[] temppixel = new
double[kolom*baris];
    int i=0;
    for(int h=0;h<baris;h++){
      for(int w=0;w<kolom;w++){
        double pix =  nilai[h][w];
        temppixel[i] = pix;
        i++;
      }
    }
    return temppixel;
  }

   public void printData() {
    for (int i = 0; i < baris; i++) {
      for (int j = 0; j < kolom; j++) {
        System.out.print(this.nilai[i][j] + "\t");
      }
      System.out.println();
    }
  }

}
```

CLASS MYVECTOR

```
package pca2d;


public class MyVector {

   private int ukuran;
   private double [] vektor;

   public MyVector(int ukuran)
   {
     this.ukuran = ukuran;
     vektor = new double[ukuran];
   }

   public MyVector(  double[] vektor) {
      this.ukuran = vektor.length;
      this.vektor = vektor;
   }

   public double [] getVector()
   {
      double[] v = new double[vektor.length];
      for(int i=0;i<vektor.length;i++)
      {
         v[i] = vektor[i];
      }
      return v;
   }

   public double getNilai(int indeks){
      return this.vektor[indeks];
   }

   public void setNilai(int indeks, double nilai){
      this.vektor[indeks] = nilai;
   }

   private void setVektor(double [] vektor){
      this.vektor = vektor;
   }


   public int getUkuran(){
      return this.ukuran;
   }

   public double getRataRata(){
      double jumlah = 0;
      for (int i = 0; i < ukuran; i++){
         jumlah = jumlah + this.vektor[i];
      }
      return jumlah/ukuran;
   }

   public void sesuaikan(){
      double ratarata = getRataRata();
      for (int i = 0; i < vektor.length; i++){
         vektor[i] = vektor[i] - ratarata;
      }
   }

   public void sesuaikanBalik(){
      double ratarata = getRataRata();
      for (int i = 0; i < vektor.length; i++){
         vektor[i] = vektor[i] + ratarata;
      }
   }

    public void printData(){
      for (int i = 0; i < ukuran; i++){
         System.out.println(vektor[i]);
      }
   }

}
```

CLASS PCA

```
package pca2d;


import pkg2dpcasomgender.FaceImage;
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.awt.image.Raster;
import java.awt.image.WritableRaster;
import java.io.BufferedInputStream;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
```

```
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.ImageIO;
import javax.swing.JOptionPane;


public class PCA
{
   private FaceImage imblock[];
   private MyVector
mVector[],mVectorAdjusted[];
   private Matrix
mxImgAsli,mxImgAdjusted,mxCov,mxEigenGam
bar;
   private Matrix eigenvalue,resultPCA;
   private eigenJacobi nilaiEigen;
   private int colblockmax;
   private int rowblockmax;
   private int blocksize;


   public PCA(){
   }

   private void CreateImageBlock(FaceImage
im,int w)
   {
      int k = im.getHeight()/w;
      int l = im.getWidth()/w;
      imblock = new FaceImage[k*l];
      rowblockmax=k;
      colblockmax=l;
      blocksize = w;

      int blockheight=w;
      int blockwidth=w;

      int startHeight=0;
      int endHeight=0;

      int startWidth=0;
      int endWidth=0;

      int jblok=0;

      for(int i=0; i<k; i++)
      {
         startHeight=blockheight*i;
         endHeight=blockheight*(i+1);
         for(int j=0; j<l; j++)
         {
            startWidth=blockwidth*j;
            endWidth=blockwidth*(j+1);
            jblok++;

            FaceImage imb = new
FaceImage(blockwidth,blockheight);
            for(int
row=startHeight;row<endHeight;row++)
            {
               for(int
col=startWidth;col<endWidth;col++)
               {
                  int pixel =
im.getPixelOutput(row,col);
                  imb.setPixelOutput(row-
startHeight,col-startWidth,pixel);
               }
            }

            imblock[jblok-1]=imb;
         }
      }
   }



   private  void turnBlocksToVector()
   {
      mVector = new MyVector[imblock.length];
      for(int i=0;i<imblock.length;i++)
      {
         mVector[i] = new
MyVector(imblock[i].OneDimensionalPixel());
      }
   }

   private void createMatrix()
   {
      mxImgAsli = new
Matrix(blocksize*blocksize,imblock.length);
      mxImgAdjusted = new
Matrix(blocksize*blocksize,imblock.length);
```

```
      mxEigenGambar = new
Matrix(blocksize*blocksize,imblock.length);
      for(int i=0;i<imblock.length;i++)
      {

mxImgAsli.setNilaiBaris(mVector[i].getVector(),
i);

mxImgAdjusted.setNilaiBaris(mVectorAdjusted[i
].getVector(), i);
      }
   }

   private void countCoVariance()
   {
      mxCov =
mxImgAdjusted.kaliMatriks(mxImgAdjusted.tra
nspose());
      mxCov = mxCov.kali((double) 1 /
(imblock.length));
   }

   private void adjustData()
   {
      mVectorAdjusted = new
MyVector[mVector.length];
      for(int i=0;i<mVector.length;i++)
      {
         double rata2 = mVector[i].getRataRata();
         mVectorAdjusted[i] = new
MyVector(mVector[i].getVector());
         for(int
j=0;j<mVectorAdjusted[i].getUkuran();j++){

mVectorAdjusted[i].setNilai(j,mVectorAdjusted[
i].getNilai(j) - rata2);
         }
      }
   }

   public void runPCA(FaceImage im)
   {

CreateImageBlock(im,(int)Math.sqrt(im.getHeig
ht()));
      turnBlocksToVector();
      adjustData();
      createMatrix();
      countCoVariance();

      nilaiEigen = new eigenJacobi(mxCov);
      eigenvalue =
nilaiEigen.getMyVectorEigenMenaik();

mxEigenGambar.copy(eigenvalue.transpose().k
aliMatriks(mxImgAdjusted.transpose()));
      resultPCA = mxEigenGambar.transpose();

      /*
      System.out.println("==================
PCA Result ====================");
      resultPCA.printData();


System.out.println(resultPCA.getJumlahBaris());

System.out.println(resultPCA.getJumlahKolom()
);
      */
   }

   public Matrix getResult() {
      return resultPCA;
   }

}
```

CLASS EIGENJACOB

```
package pca2d;


public class eigenJacobi {
   Matrix matrix;
   Matrix[] matrixU;
   Matrix[] matrixA;
   private int eppoch = 40;
   private int sweep = 0;
   boolean konvergen = false;
   double error = 0.1;
   MyVector eigen, eigenMenaik;
   Matrix MyVectorEigen,
MyVectorEigenMenaik;

   public eigenJacobi(Matrix matrix)
   {
      setMatrix(matrix);
      matrixU = new Matrix[1];
```

```
        matrixA = new Matrix[1];

        rotasi(matrix.getNilai());
        setNilaiEigen();
        setNilaiMyVectorEigen();
        urutNilaiEigen();
    }

    private int faktorial(int n) {
        if (n <= 1)
        {
            return 1;
        } else {
            return n * faktorial(n - 1);
        }
    }

    public static double[][] minus(double A[][],
double B[][], int n) {
        double[][] C = new double[n][n];
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                C[i][j] = A[i][j] - B[i][j];
            }
        }
        return C;
    }

    public static void abs(double A[][], double
B[][], int n) {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                B[i][j] = Math.abs(A[i][j]);
            }
        }
    }

    public static void maxMatrix(double A[][], int
n, int Row[], double Max[]) {
        for (int i = 0; i < n; i++) {
            int k = 0;
            Max[i] = A[k][i];
            Row[i] = k;
            for (int j = 0; j < n; j++) {
                if (A[j][i] > Max[i]) {
                    Max[i] = A[j][i];
                    Row[i] = j;
                }
            k = k + 1;
            }
        }

    public static void maxVector(double A[], int n,
int Row[], double Max[]) {
        Max[0] = A[0];
        Row[0] = 0;
        for (int i = 0; i < n; i++) {
            if (A[i] > Max[0]) {
                Max[0] = A[i];
                Row[0] = i;
            }
        }
    }

    public static void transpose(double A[][],
double B[][], int n) {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                B[i][j] = A[j][i];
            }
        }
    }

    public static double[][] diag(double A[][], int
n) {
        double[][] B = new double[n][n];
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                B[i][j] = 0;
            }
        B[i][i] = A[i][i];
        }
    return B;
    }

    public static double sumDiagElSq(double
A[][], int n) {
        double sum = 0;
        for (int i = 0; i < n; i++) {
            sum = A[i][i] * A[i][i] + sum;
        }
    return sum;
    }
```

```
   private void printArray(String label,double
A[][]){
      System.out.println(label);

      for (int i=0;i<A.length;i++){
         for(int j=0;j<A[i].length;j++){
            System.out.print(A[i][j] + "\t");
         }
      System.out.println();
      }
   }

   private void rotasi( double[][] A){
      double t, c, s;
      int p, q, icount, state, size = A.length;
      double tol = 1.e-5; // level toleransi
konvergen
      int icmax = 100; // jumlah iterasi
maksimum

      int[] colRowOfElMax = new int[size],
rowOfElMax = new int[1];
      double[][] temp = new double[size][size], D
= new double[size][size];
      double[][] V, diagD;

      double[] maxElColRow = new double[size],
maxElRow = new double[1];
      double[][] dMinusDiagD = new
double[size][size], absDminusDiagD = new
double[size][size];
      double[][] rot = new double[2][2], rotT =
new double[2][2];

      // mengubah ke matrix identitas
      V = new double[size][size];
      for (int i = 0; i < size; i++) {
         for (int j = 0; j < size; j++) {
            V[i][j] = 0;
         }
         V[i][i] = 1.0;
      }

      D = A; // menyalin A ke D
      diagD = diag(D, size);// keluaran
DiagD=diagonal dari D
      dMinusDiagD = minus(D, diagD, size); //   D-
DiagD
      abs(dMinusDiagD, absDminusDiagD,
size);//  abs(D-DiagD)
      maxMatrix(absDminusDiagD, size,
colRowOfElMax, maxElColRow);
      maxVector(maxElColRow, size,
rowOfElMax, maxElRow);
      q = rowOfElMax[0];
      p = colRowOfElMax[q];
      icount = 0;
      state = 1;

      // Iterasi
      while (state == 1 && icount < icmax) {
         icount = icount + 1;
         if (D[q][q] == D[p][p]) { // memeriksa
untuk menCegah t menjadi divergen
            D[q][q] = D[p][p] + 1.e-10;
         }
         t = D[p][q] / (D[q][q] - D[p][p]);
         c = 1 / Math.sqrt(t * t + 1);
         s = c * t;
         rot[0][0] = c;
         rot[0][1] = s;
         rot[1][0] = -s;
         rot[1][1] = c;
         transpose(rot, rotT, 2);

         for (int i = 0; i < size; i++) {
            temp[p][i] = rotT[0][0] * D[p][i] +
rotT[0][1] * D[q][i];
            temp[q][i] = rotT[1][0] * D[p][i] +
rotT[1][1] * D[q][i];
            D[p][i] = temp[p][i];
            D[q][i] = temp[q][i];
         }

         for (int i = 0; i < size; i++) {
            temp[i][p] = D[i][p] * rot[0][0] + D[i][q]
* rot[1][0];
            temp[i][q] = D[i][p] * rot[0][1] + D[i][q]
* rot[1][1];
            D[i][p] = temp[i][p];
            D[i][q] = temp[i][q];
         }

         for (int i = 0; i < size; i++) {
            temp[i][p] = V[i][p] * rot[0][0] + V[i][q]
* rot[1][0];
```

```
            temp[i][q] = V[i][p] * rot[0][1] + V[i][q]
* rot[1][1];
            V[i][p] = temp[i][p];
            V[i][q] = temp[i][q];
          }

          //menemukan array q, p baru  yang perlu
dirubah
          diagD = diag(D, size); // outputs
diagD=diagonal of D
          dMinusDiagD = minus(D, diagD, size); //
does D-DiagD
          abs(dMinusDiagD, absDminusDiagD,
size); // does abs(D-DiagD)
          maxMatrix(absDminusDiagD, size,
colRowOfElMax, maxElColRow);
          maxVector(maxElColRow, size,
rowOfElMax, maxElRow);
          q = rowOfElMax[0];
          p = colRowOfElMax[q];
          if (Math.abs(D[p][q]) < tol *
Math.sqrt(sumDiagElSq(diagD, size)) / size) {
            state = 0;
          }
        }

    matrixA[0] = new Matrix(diagD);
    matrixU[0] = new Matrix(V);
    }

    private double getNilaiRotasi(int baris, int
kolom, Matrix MatrixX) {
        double nilaiRotasi = 0;
        nilaiRotasi = 0.5 *
Math.toDegrees(Math.atan(2 *
MatrixX.getNilai(baris, kolom) /
(MatrixX.getNilai(baris, baris) -
MatrixX.getNilai(kolom, kolom))));
        return nilaiRotasi;
    }

    private void setMatrix(Matrix matrix) {
        this.matrix = matrix;
    }

    public Matrix getMatrix() {
        return this.matrix;
    }

    private void setNilaiEigen() {
        eigen = new
MyVector(matrixA[0].getDiagonal());
    }

    public MyVector getNilaiEIgen() {
        return eigen;
    }

    private void setNilaiMyVectorEigen() {
        MyVectorEigen = new
Matrix(matrixU[0].getJumlahBaris(),
matrixU[0].getJumlahKolom());
        MyVectorEigen.copy(matrixU[0]);
    }

    public Matrix getMyVectorEigen(){
        return MyVectorEigen;
    }

    public MyVector getEigenMenaik(){
        return eigenMenaik;
    }

    public Matrix getMyVectorEigenMenaik(){
        return MyVectorEigenMenaik;
    }

    private void urutNilaiEigen() {
        int iPos;
        int iMax;

        MyVector eigenTemp = eigen;
        Matrix MyVectorEigenTemp =
MyVectorEigen;

        for (iPos = 0; iPos <
matrix.getJumlahBaris(); iPos++) {
          iMax = iPos;
          for (int i = iPos + 1; i <
matrix.getJumlahBaris(); i++) {
            if (eigenTemp.getNilai(i) >
eigenTemp.getNilai(iMax)) {
              iMax = i;
            }
          }
          if (iMax != iPos) {
```

```
tukar(iMax,iPos,eigenTemp,MyVectorEigenTem
p);
          }
       }

       eigenMenaik = eigenTemp;
       MyVectorEigenMenaik =
MyVectorEigenTemp;
       }

    private void tukar(int i, int j , MyVector
tukarEigen, Matrix tukarMyVectorEigen){
       double temp;
       temp = tukarEigen.getNilai(i);
       tukarEigen.setNilai(i,tukarEigen.getNilai(j));
       tukarEigen.setNilai(j,temp);

       double [] tempMyVector;
       tempMyVector =
tukarMyVectorEigen.getNilaiBaris(i);

tukarMyVectorEigen.setNilaiBaris(tukarMyVect
orEigen.getNilaiBaris(j), i);

tukarMyVectorEigen.setNilaiBaris(tempMyVect
or, j);
    }

    private void replaceNilaiEigen(double
nilaiEigen, double [] MyVectorEigen, int pos ){
       this.eigenMenaik.setNilai(pos, nilaiEigen);

this.MyVectorEigenMenaik.setNilaiBaris(MyVec
torEigen, pos);
    }

}
```

CLASS FACEIMAGE

```
package pkg2dpcasomgender;

import java.awt.Color;


public class FaceImage
{
   private int widthOri;
   private int heightOri;
   private int pixelasli[][];
   private int pixeloutput[][];
   private double realpixelasli[][];
   private double imgpixelasli[][];
   private String name;

   public FaceImage(){
   }

   public FaceImage(int widthOri, int heightOri)
   {
      this.widthOri = widthOri;
      this.heightOri = heightOri;
      pixelasli = new int[heightOri][widthOri];
      pixeloutput = new int[heightOri][widthOri];
   }

   public void SetName(String name)
   {
      this.name = name;
   }

   public String getName()
   {
      return name;
   }


   public void setPixelAsli(int[][] pixelasli)
   {
      this.pixelasli = pixelasli;
   }

   public int[][] getPixelAsli()
   {
      return pixelasli;
   }
```

```
   public int getHeight()
   {
      return heightOri;
   }

   public int getWidth()
   {
      return widthOri;
   }

   public void setPixelOutput(int h,int w,int
output)
   {
      pixeloutput[h][w]=output;
   }

   public int getPixelOutput(int h,int w)
   {
      return  pixeloutput[h][w];
   }

   public void setPixel(int h,int w,int pixel) {
      this.pixelasli[h][w] = pixel;
   }

   public int getPixel(int h,int w)
   {
      return pixelasli[h][w];
   }

   void setPixelReal(int h,int w,double realpixel)
   {
      realpixelasli[h][w] = realpixel;
   }

   double getPixelReal(int h,int w){
      return realpixelasli[h][w];
   }

   public int[] OneDimensionalPixel2(){
      int[] onedpixel = new
int[pixelasli.length*pixelasli[0].length];
      int i=0;
      for(int
baris=0;baris<pixelasli.length;baris++){
         for(int
kolom=0;kolom<pixelasli[0].length;kolom++){
            int gray = pixelasli[baris][kolom];
            onedpixel[i] =gray;
            i++;
         }
      }
      return onedpixel;
   }

   public int[] toOneDimensionalPixelOutput()
   {
      int[] onedpixel = new
int[pixeloutput.length*pixeloutput[0].length];
      int i=0;
      for(int
baris=0;baris<pixeloutput.length;baris++){
         for(int
kolom=0;kolom<pixeloutput[0].length;kolom++)
{
            int gray = pixeloutput[baris][kolom];
            Color c = new Color(gray, gray, gray,0 );
            onedpixel[i] =c.getRGB();
            i++;
         }
      }
      return onedpixel;
   }

   public double[] OneDimensionalPixel()
   {
      double[] onedpixel = new
double[pixelasli.length*pixelasli[0].length];
      int i=0;
      for(int
baris=0;baris<pixeloutput.length;baris++)
      {
         for(int
kolom=0;kolom<pixeloutput[0].length;kolom++)
         {
            double gray =
pixeloutput[baris][kolom];
            onedpixel[i] =gray;
            i++;
         }
      }
      return onedpixel;
   }

}
```

CLASS FACEIMAGEVIEWER

```
package pkg2dpcasomgender;


import java.awt.Image;
import java.awt.image.BufferedImage;
import javax.swing.ImageIcon;
import javax.swing.JLabel;


public class FaceImageViewer
{
   FaceImage myimg;
   JLabel jblViewer;

   public FaceImageViewer(){
   }

   public void setImage(FaceImage myimg)
   {
      this.myimg = myimg;
   }

   public void setViewer(JLabel lblviewer)
   {
      this.jblViewer = lblviewer;
   }

   public void viewImageDefault()
   {
      int pix[] =  myimg.OneDimensionalPixel2();
      int w = myimg.getWidth();
      int h = myimg.getHeight();
      BufferedImage image = new
BufferedImage(w , h ,
BufferedImage.TYPE_INT_RGB);
      image.setRGB(0, 0, w , h ,pix, 0, w);
      jblViewer.setText("");
      jblViewer.setIcon(new
ImageIcon(image.getScaledInstance(jblViewer.g
etWidth(), jblViewer.getHeight(),
Image.SCALE_DEFAULT)));
   }
   public void viewImageOutput()
   {
      int pix[] =
myimg.toOneDimensionalPixelOutput();
      int w = myimg.getWidth();
      int h = myimg.getHeight();
      BufferedImage image = new
BufferedImage(w , h ,
BufferedImage.TYPE_INT_RGB);
      image.setRGB(0, 0, w , h ,pix, 0, w);
      jblViewer.setText("");
      jblViewer.setIcon(new
ImageIcon(image.getScaledInstance(jblViewer.g
etWidth(), jblViewer.getHeight(),
Image.SCALE_DEFAULT)));
   }

}
```

CLASS FEATUREEXTRACTOR

```
package pkg2dpcasomgender;

import pca2d.PCA;


public class FeatureExtractor {
   PCA pca;

   public FeatureExtractor(){
   }

   public void extract(FaceImage m )
   {
     pca = new PCA();
     pca.runPCA(m);
   }

   public double[] getFeature()
   {
      int kolom = 4;
      double[] feature = new
double[pca.getResult().getJumlahBaris()*kolom]
;
```

```
      int i=0;
      for(int kol=0;kol<kolom;kol++)
      {
         for(int
row=0;row<pca.getResult().getJumlahBaris();ro
w++)
         {
            feature[i] =
pca.getResult().getNilai(row, kol);
            i++;
         }
      }
      return feature;
   }

}
```

CLASS GENDERIDENTIFIER

```
package pkg2dpcasomgender;


import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.ObjectInputStream;
import java.util.Vector;
import som.SOMLattice;
import som.SOMNode;
import som.SOMVector;


public class GenderIdentifier
{
   TrainingSet tsampleuji;
   private Vector<SOMVector> testVectors;
   private SOMLattice lattice;

   public GenderIdentifier(TrainingSet
tsampleuji)
   {
      this.tsampleuji = tsampleuji;
      testVectors = new Vector<SOMVector>();
   }

   private void extract()
   {
      double[][] inputs = new
double[tsampleuji.getTotalFaces() ][];
      for(int i=0;i<tsampleuji.getTotalFaces();i++)
      {
         FaceImage faceimg =
tsampleuji.getFaceImage(i);
         FeatureExtractor fe = new
FeatureExtractor();
         fe.extract(faceimg);
         inputs[i] = fe.getFeature();
      }

      SOMVector tempVec;

      for(int i=0;i<inputs.length;i++)
      {
         //System.out.print("Data -" + i);
         tempVec = new SOMVector(
tsampleuji.getFaceImage(i).getName());
         for(int j=0;j<inputs[i].length;j++)
         {
          //System.out.print("\t" + inputs[i][j]);
          tempVec.addElement(new
Double(inputs[i][j]));
         }
         testVectors.addElement(tempVec);
         System.out.println();
      }
   }

   private SOMLattice readTrainedLattice(File
fsave)
   {

      SOMLattice somlattice = null;
      File f = fsave;

      if (f.exists())
      {
         System.out.println("Trying to read the
existing Weights");
```

```
        try{
           InputStream file = new
FileInputStream(f.getAbsolutePath());
           InputStream buffer = new
BufferedInputStream(file);
           ObjectInputStream oi = new
ObjectInputStream(buffer);
           somlattice =
(SOMLattice)oi.readObject();

        }
        catch (ClassNotFoundException ex)
        {
           System.out.println("Something went
wrong --- ClassNotFoundError: \n" +
ex.getMessage());
        }
        catch(IOException ex)
        {
           System.out.println("Something went
wrong --- IOException: \n" + ex.getMessage());
        }
     }
     return  somlattice;
  }

  public void classification()
  {
     extract();
     SOMLattice lattice = readTrainedLattice(
new File("bobot/som.haris"));
     for(int i=0;i<testVectors.size();i++)
     {
        SOMNode bmu =
lattice.getBMU(testVectors.get(i));

tsampleuji.getFaceImage(i).SetName(bmu.getCl
uster());
        //System.out.println("Gambar-" +
String.valueOf(i+1) + ",Cluster:" +
bmu.getCluster());
     }
  }

}
```

CLASS PIXELREADER

```
package pkg2dpcasomgender;


import java.awt.Color;
import java.awt.Image;
import java.awt.Rectangle;
import java.awt.image.BufferedImage;
import java.awt.image.PixelGrabber;
import javax.swing.ImageIcon;


public class PixelsReader
{
  private FaceImage img;
  public PixelsReader()
  {
     img = new FaceImage();
  }



  public void readPixelsFrom(ImageIcon imgic)
  {
     img = new
FaceImage(imgic.getIconWidth(),
imgic.getIconHeight());
     PixelGrabber pxlgrabber = new
PixelGrabber(imgic.getImage(),0,0,img.getWidt
h(), img.getHeight(),false);
     pxlgrabber.startGrabbing();
     int pixels[];
     int pixelasli[][];

     try{
        if(pxlgrabber.grabPixels())
        {
           pixels = (int[])pxlgrabber.getPixels();

           BufferedImage image = new
BufferedImage(imgic.getIconWidth() ,
imgic.getIconHeight() ,
BufferedImage.TYPE_INT_RGB);
```

```
            image.setRGB(0, 0,
imgic.getIconWidth() , imgic.getIconHeight()
,pixels, 0, imgic.getIconWidth());

            ImageIcon imgIcon = new
ImageIcon(image.getScaledInstance(100, 100,
Image.SCALE_DEFAULT));
            img = new
FaceImage(imgIcon.getIconWidth(),
imgIcon.getIconHeight());

            pxlgrabber = new
PixelGrabber(imgIcon.getImage(),0,0,img.getWi
dth(), img.getHeight(),false);
            pxlgrabber.startGrabbing();

            if(pxlgrabber.grabPixels())
            {
               pixels = (int[])pxlgrabber.getPixels();
               pixelasli  = new int
[img.getHeight()][img.getWidth()];
               int wpx = 0;
               int hpx = 0;
               for(int i =0;i<pixels.length;i++)
               {
                  int pixel = pixels[i];
                  Color c = new Color(pixel);
                  int merah = c.getRed();
                  int hijau = c.getGreen();
                  int biru = c.getBlue();
                  int gray = (merah+hijau+biru)/3 ;

                  pixelasli[hpx][wpx] = pixel;
                  img.setPixelOutput(hpx, wpx,
gray);
                  wpx++;

                  if (wpx==img.getWidth())
                  {
                     wpx=0;
                     hpx++;
                  }
               }
               img.setPixelAsli(pixelasli);
            }
         }
      }
      catch(InterruptedException ex){}
   }

   public FaceImage getFaceImage(){
      return img;
   }

}
```

CLASS SOMLEARNER

```
package pkg2dpcasomgender;


import java.awt.image.BufferedImage;
import java.io.IOException;
import java.util.Vector;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import som.SOMLattice;
import som.SOMNode;
import som.SOMTrainer;
import som.SOMVector;


public class SomLearner implements  Runnable
{
   private int latticeWidth = 40;
   private int latticeHeight = 40;

   private SOMTrainer trainer;
   private SOMLattice lattice;
   private Vector<SOMVector> inputVectors;

   private int nfiturs;
```

```
    private TrainingSet facetrainingset;
    private boolean finished;

    public SomLearner(TrainingSet
facetrainingset,int maxiterasi,double
learingrate,int latWidth,int latHeight )
    {
        this.facetrainingset = facetrainingset;
        latticeWidth = latWidth;
        latticeHeight = latHeight;
        trainer = new
SOMTrainer(maxiterasi,learingrate);
        inputVectors = new Vector<SOMVector>();
        finished = false;
    }

    private void extract()
    {
        double[][] inputs = new
double[facetrainingset.getTotalFaces() ][];
        for(int
i=0;i<facetrainingset.getTotalFaces();i++)
        {
            FaceImage faceimg =
facetrainingset.getFaceImage(i);
            FeatureExtractor fe = new
FeatureExtractor();
            fe.extract(faceimg);
            inputs[i] = fe.getFeature();
        }

        nfiturs = inputs[0].length;
        SOMVector tempVec;

        for(int i=0;i<inputs.length;i++)
        {
            //System.out.print("Data -" + i);
            tempVec = new SOMVector(
facetrainingset.getFaceImage(i).getName());
            for(int j=0;j<inputs[i].length;j++)
            {
                //System.out.print("\t" + inputs[i][j]);
                tempVec.addElement(new
Double(inputs[i][j]));
            }
            inputVectors.addElement(tempVec);
            //System.out.println();
        }
    }

    public void printCluster()
    {
        trainer.printLattice();
    }

    public void testCluster()
    {
        for(int i=0;i<inputVectors.size();i++)
        {
            SOMNode bmu =
lattice.getBMU(inputVectors.get(i));
            System.out.println("Gambar-" +
String.valueOf(i+1) + ",Cluster:" +
bmu.getCluster());
        }
    }

    public void start(){
        new Thread(this).start();
    }

    public void saveWeights()
    {
        WeightEnroller wio = new WeightEnroller();
        try {
            wio.saveWeight("bobot", "som",lattice);
        } catch (IOException ex) {

Logger.getLogger(SomLearner.class.getName()).
log(Level.SEVERE, null, ex);
        }
        JOptionPane.showMessageDialog(null,
"Bobot Berhasil
Disimpan!","Sukses",JOptionPane.INFORMATIO
N_MESSAGE);

    }

    public boolean isFinished()
    {
        return !trainer.isRunning();
    }

    public void train()
    {
      trainer.stop();
```

```
      extract();

      lattice = new SOMLattice(latticeWidth,
latticeHeight,nfiturs);
      trainer.setTraining(lattice, inputVectors);
      trainer.start();
    }

    @Override
    public void run()
    {
      train();
    }

}
```

CLASS TRAINING SET

```
package pkg2dpcasomgender;


import java.util.ArrayList;
import java.util.List;

public class TrainingSet
{
    List<FaceImage> lstfaces;
    List<String> listnamafiles;
    List<String> lsttarget;

    public TrainingSet()
    {
       lstfaces = new ArrayList<FaceImage>();
       lsttarget = new ArrayList<String>();
       listnamafiles = new ArrayList<String>();
    }

    public void addFaceImage(FaceImage
hw,String nmfile, String target)
    {
      lstfaces.add(hw);
      hw.SetName(target);
      lsttarget.add(target);
      listnamafiles.add(nmfile);
    }

    public FaceImage getFaceImage(int idx)
    {
      return lstfaces.get(idx);
    }

    public String getFileNames(int idx)
    {
      return listnamafiles.get(idx);
    }

    public int getTotalFaces()
    {
      return lstfaces.size();
    }

}
```

CLASS WEIGTHENROLLER

```
package pkg2dpcasomgender;


import java.io.BufferedInputStream;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.util.ArrayList;
import java.util.List;
import som.SOMLattice;

public class WeightEnroller
{
```

```
   public void WeightIOController(){
   }

   private static void createDir(String dir)
   {
      File filePath = new File(dir);
      filePath.mkdirs();
   }

   private static void deleteDir(String dir){
      File filePath = new File(dir);
      System.out.println(filePath.delete());
   }

   public void saveWeight(String dir,String
fileName, SOMLattice somlattice) throws
IOException
   {
      createDir(dir);
      ObjectOutputStream objectOutputStream
= null;
      try {
         objectOutputStream = new
ObjectOutputStream(new FileOutputStream(dir
+ "/" + fileName + ".haris"));

objectOutputStream.writeObject(somlattice);
      } catch (IOException e) {
         System.out.println("Could not write to
file: " + fileName+"\n"+e);
      } finally {
         try {
            if (objectOutputStream != null) {
               objectOutputStream.flush();
               objectOutputStream.close();
            }
         } catch (IOException e) {
            System.out.println("Could not write to
file: " + fileName);
         }
      }
   }

   public double[][] readWeights(String dir,
String fileName) throws FileNotFoundException,
IOException
   {
      DataInputStream dist1 = new
DataInputStream(new
BufferedInputStream(new FileInputStream(dir +
"/" + fileName + ".txt")));
      int maxw =
Integer.parseInt(dist1.readLine());
      double[][] w  = new double[maxw][];
      for(int i=0;i<w.length;i++)
      {
         String data = dist1.readLine();
         String arrdata[] = data.split(";");
         w[i] = new double[arrdata.length];
         for(int j=0;j<arrdata.length;j++){
            w[i][j]=
Double.parseDouble(arrdata[j]);
         }
      }
      return w;
   }

}
```

CLASS SOMLATTICE

```
package som;

import java.io.Serializable;

public class SOMLattice implements
Serializable{

   private int width, height;
   private SOMNode[][] matrix;

   public SOMLattice(int w, int h,int nfiturs)
   {
```

```
        width = w;
        height = h;
        matrix = new SOMNode[width][height];
        float xstep = .5f / (float)width;
        float ystep = .5f / (float)height;
        for (int x=0; x<w; x++) {
       for (int y=0; y<h; y++) {
                matrix[x][y] = new
SOMNode(nfiturs);
                matrix[x][y].setX(x);
                matrix[x][y].setY(y);
       }
        }
  }

  public SOMNode getNode(int x, int y) {
        return matrix[x][y];
  }

  public int getWidth() {
        return width;
  }

  public int getHeight() {
        return height;
  }

  public SOMNode getBMU(SOMVector
inputVector)
  {
        SOMNode bmu = matrix[0][0];
        double bestDist =
inputVector.euclideanDist(bmu.getVector());
        double curDist;

        for (int x=0; x<width; x++) {
       for (int y=0; y<height; y++) {
                curDist =
inputVector.euclideanDist(matrix[x][y].getVecto
r());
         if (curDist < bestDist)
         {
            bmu = matrix[x][y];
            bestDist = curDist;
         }
       }
        }
        return bmu;
  }

}
```

CLASS SOMNODE

```
package som;

import java.io.Serializable;


public class SOMNode implements Serializable {
  private SOMVector weights;
  private int xp, yp;

  public SOMNode(int numWeights)
  {
        weights = new SOMVector("0");
        for (int x=0; x<numWeights; x++) {
       weights.addElement(new
Double(Math.random()));
        }
  }

  public void SetCluster(String cl) {
    weights.setLabel(cl);
  }

  public String getCluster(){
    return weights.getLabel();
  }

  public void setX(int xpos) {
        xp = xpos;
  }

  public void setY(int ypos) {
        yp = ypos;
  }

  public int getX() {
        return xp;
```

```
    }

    public int getY() {
            return yp;
    }

    public double distanceTo(SOMNode n2) {
            int xleg, yleg;
            xleg = getX() - n2.getX();
            xleg *= xleg;
            yleg = getY() - n2.getY();
            yleg *= yleg;
            return xleg + yleg;
    }

    public void setWeight(int w, double value) {
            if (w >= weights.size())
            return;
            weights.setElementAt(new
Double(value), w);
    }

    public double getWeight(int w) {
            if (w >= weights.size())
           return 0;

           return
((Double)weights.elementAt(w)).doubleValue();
    }

    public SOMVector getVector() {
            return weights;
    }

    public void adjustWeights(SOMVector input,
double learningRate, double distanceFalloff)
    {
       double wt, vw;
            for (int w=0; w<weights.size(); w++) {
           wt =
((Double)weights.elementAt(w)).doubleValue();
           vw =
((Double)input.elementAt(w)).doubleValue();
           wt += distanceFalloff * learningRate *
(vw - wt);
           weights.setElementAt(new Double(wt),
w);
            }
    }

}
```

CLASS SOMTRAINER

```
package som;

import java.util.Vector;
import javax.swing.JOptionPane;


public class SOMTrainer implements Runnable
{
    private double START_LEARNING_RATE = 0;
    private int NUM_ITERATIONS = 0;
    private double LATTICE_RADIUS;
    private double TIME_CONSTANT;

    private SOMLattice lattice;
    private Vector<SOMVector> inputs;
    private static boolean running;
    private Thread runner;


    public SOMTrainer(int maxiteration,double
learningrate){
            running = false;
       NUM_ITERATIONS = maxiteration;
       START_LEARNING_RATE = learningrate;
    }

    private double
getNeighborhoodRadius(double iteration){
            return LATTICE_RADIUS * Math.exp(-
iteration/TIME_CONSTANT);
    }

    private double getDistanceFalloff(double
distSq, double radius){
            double radiusSq = radius * radius;
            return Math.exp(-(distSq)/(2 *
radiusSq));
    }
```

```
   public void setTraining(SOMLattice
latToTrain, Vector<SOMVector> in){
          lattice = latToTrain;
          inputs = in;
   }

   public void start(){
          if (lattice != null){
         runner = new Thread(this);

runner.setPriority(Thread.MIN_PRIORITY);
         running = true;
         runner.start();
           }
   }

   public void printLattice()
   {
      int width = lattice.getWidth();
      int height = lattice.getHeight();


System.out.println("=====================
=================== SOM MAP
====================================")
;
           for (int x=0; x<width; x++){
         for (int y=0; y<height; y++){
                    SOMNode sn =
lattice.getNode(x, y);
            System.out.print(sn.getCluster() +
"\t");
         }
         System.out.println("");
           }

System.out.println("=====================
=======================================
=========================");
   }

   @Override
   public void run() {
      int lw = lattice.getWidth();
          int lh = lattice.getHeight();
          int xstart, ystart, xend, yend;
          double dist, dFalloff;

          LATTICE_RADIUS = Math.max(lw, lh)/2;
          TIME_CONSTANT = NUM_ITERATIONS /
Math.log(LATTICE_RADIUS);

          int iteration = 0;
          double nbhRadius;
          SOMNode bmu = null, temp = null;
          SOMVector curInput = null;
          double learningRate =
START_LEARNING_RATE;

          while (iteration < NUM_ITERATIONS &&
running)
      {
         nbhRadius =
getNeighborhoodRadius(iteration);
         for (int i=0; i<inputs.size(); i++) {
                    curInput =  inputs.elementAt(i);
                    bmu =
lattice.getBMU(curInput);
            bmu.SetCluster(curInput.getLabel());
                    xstart = (int)(bmu.getX() -
nbhRadius - 1);
                    ystart = (int)(bmu.getY() -
nbhRadius - 1);
                    xend = (int)(xstart + (nbhRadius
* 2) + 1);
                    yend = (int)(ystart + (nbhRadius
* 2) + 1);

           if (xend > lw) xend = lw;
                    if (xstart < 0) xstart = 0;
                    if (yend > lh) yend = lh;
                    if (ystart < 0) ystart = 0;

                    for (int x=xstart; x<xend; x++) {
              for (int y=ystart; y<yend; y++) {
                              temp =
lattice.getNode(x,y);
                              dist =
bmu.distanceTo(temp);
                              if (dist <= (nbhRadius *
nbhRadius)) {
                        dFalloff =
getDistanceFalloff(dist, nbhRadius);
                    temp.adjustWeights(curInput,
learningRate, dFalloff);
                              }
```

```
                }
                    }
            }
                iteration++;
            learningRate = START_LEARNING_RATE *
Math.exp(-
(double)iteration/NUM_ITERATIONS);
             }

        running = false;
        JOptionPane.showMessageDialog(null,
"Train Finished!");
    }

    public boolean isRunning() {
        return running;
    }

    public void stop() {
             if (runner != null) {
            running = false;
            while (runner.isAlive()) {};
             }
    }

}
```

CLASS SOMVECTOR

```
package som;

import java.io.Serializable;
import java.util.Vector;


public class SOMVector extends java.util.Vector
implements Serializable {
    private String label;

    public SOMVector(String label)
    {
        this.label = label;
    }

    public void setLabel(String label)
    {
        this.label = label;
    }

    public String getLabel()
    {
        return label;
    }

    public double euclideanDist(SOMVector v2) {
             if (v2.size() != size())
            return -999;

              double summation = 0, temp;
              for (int x=0; x<size(); x++) {
            temp =
((Double)elementAt(x)).doubleValue() -
((Double)v2.elementAt(x)).doubleValue();
            temp *= temp;
            summation += temp;
              }

              return summation;
              }

}
```

CLASS EXTENSIONFILEFILTER

```
package utils;

import java.io.File;

import javax.swing.JFileChooser;
import javax.swing.filechooser.FileFilter;

public class ExtensionFileFilter extends FileFilter
{
  String description;

  String extensions[];

  public ExtensionFileFilter(String description,
String extension) {
    this(description, new String[] { extension });
```

```
  }

  public ExtensionFileFilter(String description,
String extensions[]) {
    if (description == null) {
      this.description = extensions[0];
    } else {
      this.description = description;
    }
    this.extensions = (String[]) extensions.clone();
    toLower(this.extensions);
  }

  private void toLower(String array[]) {
    for (int i = 0, n = array.length; i < n; i++) {
      array[i] = array[i].toLowerCase();
    }
  }

  public String getDescription() {
    return description;
  }

  public boolean accept(File file) {
    if (file.isDirectory()) {
      return true;
    } else {
      String path =
file.getAbsolutePath().toLowerCase();
      for (int i = 0, n = extensions.length; i < n; i++)
{
        String extension = extensions[i];
        if ((path.endsWith(extension) &&
(path.charAt(path.length() - extension.length() -
1)) == '.')) {
          return true;
        }
      }
    }
    return false;
  }
}
```

CLASS UTIL

```
package utils;
import java.awt.Dimension;
import java.awt.Frame;
import java.awt.Toolkit;
import java.awt.Window;
import javax.swing.JWindow;
import java.text.DateFormat;
import java.text.ParseException;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.text.SimpleDateFormat;
import javax.swing.JOptionPane;
import javax.swing.SwingUtilities;
import javax.swing.UIManager;
import
javax.swing.UnsupportedLookAndFeelException
;
import java.util.*;
import java.text.*;
import
javax.swing.UIManager.LookAndFeelInfo;

public class Util {
  public static void TengahWindow(Window f){

        // Get the size of the screen
    Dimension dim =
Toolkit.getDefaultToolkit().getScreenSize();

    // Determine the new location of the window
    int w = f.getSize().width;
    int h = f.getSize().height;
    int x = (dim.width-w)/2;
    int y = (dim.height-h)/2;

    // Move the window
    f.setLocation(x, y);
    }



  public static void LookAndFeel(Frame f){
      try{
```

```
UIManager.setLookAndFeel("com.sun.java.swing.plaf.nimbus.NimbusLookAndFeel");
    SwingUtilities.updateComponentTreeUI(f);
    }catch (ClassNotFoundException ex){

JOptionPane.showMessageDialog(f,"Kelas tak ditemukan.. ulangi installasi");
    }catch (InstantiationException ex){
    }catch (IllegalAccessException ex){
    }catch (UnsupportedLookAndFeelException ex){
       JOptionPane.showMessageDialog(f,"Dak Support");
    }

 }


 public static void initNimbusTheme()
 {
     try {
        for (LookAndFeelInfo info : UIManager.getInstalledLookAndFeels()) {

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

UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
     } catch (Exception ex) {

     }


 }


}
```

JFRAME FORMMAIN

```
package ui;

public class FormMain extends javax.swing.JFrame {

   public FormMain()
   {
      initComponents();
      utils.Util.TengahWindow(this);
      setTitle("Menu Utama");
      setResizable(false);
   }


   @SuppressWarnings("unchecked")
   // <editor-fold defaultstate="collapsed" desc="Generated Code">
   private void initComponents() {

      jPanel1 = new javax.swing.JPanel();
      jbtnShowTraining = new javax.swing.JButton();
      jbtnShowTesting = new javax.swing.JButton();
      jLabel1 = new javax.swing.JLabel();
      jLabel2 = new javax.swing.JLabel();
      jLabel3 = new javax.swing.JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
      setBackground(new java.awt.Color(51, 255, 204));
      setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));

      jPanel1.setBackground(new java.awt.Color(0, 153, 153));

      jbtnShowTraining.setBackground(new java.awt.Color(0, 153, 153));
      jbtnShowTraining.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N
      jbtnShowTraining.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icon/training.png"))); // NOI18N
      jbtnShowTraining.setText("Training");
      jbtnShowTraining.setBorder(null);
```

```
        jbtnShowTraining.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent
evt) {

jbtnShowTrainingActionPerformed(evt);
            }
        });

        jbtnShowTesting.setBackground(new
java.awt.Color(0, 153, 153));
        jbtnShowTesting.setFont(new
java.awt.Font("Comic Sans MS", 1, 24)); //
NOI18N
        jbtnShowTesting.setIcon(new
javax.swing.ImageIcon(getClass().getResource("
/icon/testing.png"))); // NOI18N
        jbtnShowTesting.setText("Testing");
        jbtnShowTesting.setBorder(null);
        jbtnShowTesting.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent
evt) {
                jbtnShowTestingActionPerformed(evt);
            }
        });

        jLabel1.setFont(new java.awt.Font("Felix
Titling", 1, 16)); // NOI18N
        jLabel1.setText("Klasifikasi Jenis Kelamin
Manusia");

        jLabel2.setFont(new java.awt.Font("Felix
Titling", 1, 16)); // NOI18N
        jLabel2.setText("Menggunakan 2DPCA dan
SOM");

        jLabel3.setFont(new java.awt.Font("Felix
Titling", 1, 16)); // NOI18N
        jLabel3.setText("Berdasarkan Citra
Wajah");

        javax.swing.GroupLayout jPanel1Layout =
new javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
jPanel1Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGrou
p()
                .addContainerGap(66,
Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.
TRAILING,
jPanel1Layout.createSequentialGroup()

.addComponent(jbtnShowTraining,
javax.swing.GroupLayout.PREFERRED_SIZE, 177,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(26, 26, 26)
                    .addComponent(jbtnShowTesting,
javax.swing.GroupLayout.PREFERRED_SIZE, 177,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(57, 57, 57))

.addGroup(javax.swing.GroupLayout.Alignment.
TRAILING,
jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jLabel1)

.addGroup(jPanel1Layout.createSequentialGrou
p()
                            .addGap(10, 10, 10)
                            .addComponent(jLabel2)))
                    .addGap(79, 79, 79))

.addGroup(javax.swing.GroupLayout.Alignment.
TRAILING,
jPanel1Layout.createSequentialGroup()
                    .addComponent(jLabel3)
                    .addGap(117, 117, 117))))
        );
        jPanel1Layout.setVerticalGroup(
```

```
jPanel1Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGrou
p()
            .addGap(34, 34, 34)
            .addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.Com
ponentPlacement.RELATED)
            .addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.Com
ponentPlacement.RELATED)
            .addComponent(jLabel2)
            .addGap(49, 49, 49)

.addGroup(jPanel1Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jbtnShowTraining,
javax.swing.GroupLayout.PREFERRED_SIZE, 87,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jbtnShowTesting,
javax.swing.GroupLayout.PREFERRED_SIZE, 87,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addContainerGap(87,
Short.MAX_VALUE))
      );

      javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
      getContentPane().setLayout(layout);
      layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)
          .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
      );
      layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)
          .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
      );

      pack();
   }// </editor-fold>

private void
jbtnShowTrainingActionPerformed(java.awt.eve
nt.ActionEvent evt) {
   FormTraining frmTraining = new
FormTraining();
   frmTraining.setVisible(true);
}

private void
jbtnShowTestingActionPerformed(java.awt.eve
nt.ActionEvent evt) {
  FormTesting frmTesting = new FormTesting();
   frmTesting.setVisible(true);
}

   public static void main(String args[]) {
      try {
         for
(javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFee
ls()) {
            if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.get
ClassName());
               break;
            }
         }
      } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(FormMain.cla
ss.getName()).log(java.util.logging.Level.SEVERE
, null, ex);
      } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(FormMain.cla
ss.getName()).log(java.util.logging.Level.SEVERE
, null, ex);
      } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(FormMain.cla
```

```
ss.getName()).log(java.util.logging.Level.SEVERE
, null, ex);
      } catch
(javax.swing.UnsupportedLookAndFeelExceptio
n ex) {

java.util.logging.Logger.getLogger(FormMain.cla
ss.getName()).log(java.util.logging.Level.SEVERE
, null, ex);
      }
      java.awt.EventQueue.invokeLater(new
Runnable() {

         public void run() {
            new FormMain().setVisible(true);
         }
      });
   }
   // Variables declaration - do not modify
   private javax.swing.JLabel jLabel1;
   private javax.swing.JLabel jLabel2;
   private javax.swing.JLabel jLabel3;
   private javax.swing.JPanel jPanel1;
   private javax.swing.JButton jbtnShowTesting;
   private javax.swing.JButton
jbtnShowTraining;
   // End of variables declaration
}
```

JFRAME FORMTESTING

```
package ui;


import java.awt.image.BufferedImage;
import java.io.IOException;
import javax.imageio.ImageIO;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.filechooser.FileFilter;
import javax.swing.table.DefaultTableModel;
import pkg2dpcasomgender.GenderIdentifier;
import pkg2dpcasomgender.FaceImage;
import pkg2dpcasomgender.FaceImageViewer;
import pkg2dpcasomgender.FeatureExtractor;
import pkg2dpcasomgender.PixelsReader;
import pkg2dpcasomgender.TrainingSet;
import utils.ExtensionFileFilter;
import utils.Util;


public class FormTesting extends
javax.swing.JFrame {
 JFileChooser flchooser = new JFileChooser();
 FaceImage fitest;
 TrainingSet sampleuji;

   public FormTesting() {
      initComponents();
      Util.TengahWindow(this);
      setResizable(false);
      sampleuji = new TrainingSet();
      showFilesImages();
   }

    private void showFilesImages()
   {
      String[] header = new String[]{"No","Image
FileName","Hasil"};
      String[][] data = new
String[sampleuji.getTotalFaces()][header.length
];
      for(int i=0;i<sampleuji.getTotalFaces();i++)
      {
         String ffname =
sampleuji.getFaceImage(i).getName();

         data[i][0] = String.valueOf(i+1);
         data[i][1] = sampleuji.getFileNames(i);
         data[i][2] = ffname;
      }
      jTblTesting.setModel(new
DefaultTableModel(data, header));
}


   @SuppressWarnings("unchecked")
   // <editor-fold defaultstate="collapsed"
desc="Generated Code">
   private void initComponents() {

      jLabel1 = new javax.swing.JLabel();
      jbtnBrowse = new javax.swing.JButton();
      jPanel1 = new javax.swing.JPanel();
```

```
        jlblUji = new javax.swing.JLabel();
        jBtnRecognise = new javax.swing.JButton();
        jLabel2 = new javax.swing.JLabel();
        jlblHasil = new javax.swing.JLabel();
        jScrollPane1 = new
javax.swing.JScrollPane();
        jTblTesting = new javax.swing.JTable();
        jPanel2 = new javax.swing.JPanel();


setDefaultCloseOperation(javax.swing.Window
Constants.DISPOSE_ON_CLOSE);
        setMinimumSize(new
java.awt.Dimension(0, 0));
        setResizable(false);
        getContentPane().setLayout(new
org.netbeans.lib.awtextra.AbsoluteLayout());

        jLabel1.setFont(new
java.awt.Font("Cambria Math", 1, 24)); //
NOI18N
        jLabel1.setText("TESTING");
        getContentPane().add(jLabel1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(
310, 20, -1, -1));

        jbtnBrowse.setBackground(new
java.awt.Color(0, 153, 153));
        jbtnBrowse.setFont(new
java.awt.Font("Copperplate Gothic Bold", 1,
12)); // NOI18N
        jbtnBrowse.setText("Load Images");
        jbtnBrowse.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent
evt) {
                jbtnBrowseActionPerformed(evt);
            }
        });
        getContentPane().add(jbtnBrowse, new
org.netbeans.lib.awtextra.AbsoluteConstraints(
450, 60, 257, 40));


jPanel1.setBorder(javax.swing.BorderFactory.cr
eateLineBorder(new java.awt.Color(0, 0, 0)));
        javax.swing.GroupLayout jPanel1Layout =
new javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGrou
p()
                .addContainerGap()
                .addComponent(jlblUji,
javax.swing.GroupLayout.DEFAULT_SIZE, 232,
Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGrou
p()
                .addContainerGap()
                .addComponent(jlblUji,
javax.swing.GroupLayout.DEFAULT_SIZE, 276,
Short.MAX_VALUE)
                .addContainerGap())
        );

        getContentPane().add(jPanel1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(
450, 200, -1, 300));

        jBtnRecognise.setBackground(new
java.awt.Color(0, 153, 153));
        jBtnRecognise.setFont(new
java.awt.Font("Copperplate Gothic Bold", 1,
12)); // NOI18N
        jBtnRecognise.setText("Classification");
        jBtnRecognise.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void
mouseClicked(java.awt.event.MouseEvent evt) {
                jBtnRecogniseMouseClicked(evt);
            }
        });
```

```
        jBtnRecognise.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent
evt) {
                jBtnRecogniseActionPerformed(evt);
            }
        });
        getContentPane().add(jBtnRecognise, new
org.netbeans.lib.awtextra.AbsoluteConstraints(
450, 110, 257, 43));

        jLabel2.setFont(new java.awt.Font("Bodoni
MT Black", 1, 14)); // NOI18N
        jLabel2.setText("Gender :");
        getContentPane().add(jLabel2, new
org.netbeans.lib.awtextra.AbsoluteConstraints(
450, 160, -1, -1));

        jlblHasil.setFont(new
java.awt.Font("Sylfaen", 1, 14)); // NOI18N
        jlblHasil.setText("----------------------------------
----");
        getContentPane().add(jlblHasil, new
org.netbeans.lib.awtextra.AbsoluteConstraints(
450, 180, 260, 25));

        jTblTesting.setFont(new
java.awt.Font("Sylfaen", 0, 12)); // NOI18N
        jTblTesting.setModel(new
javax.swing.table.DefaultTableModel(
            new Object [][] {
                {null, null, null, null},
                {null, null, null, null},
                {null, null, null, null},
                {null, null, null, null}
            },
            new String [] {
                "Title 1", "Title 2", "Title 3", "Title 4"
            }
        ));
        jTblTesting.setGridColor(new
java.awt.Color(0, 102, 102));
        jTblTesting.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void
mouseClicked(java.awt.event.MouseEvent evt) {
                jTblTestingMouseClicked(evt);
            }
        });
        jScrollPane1.setViewportView(jTblTesting);

        getContentPane().add(jScrollPane1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(
10, 58, 430, 440));

        jPanel2.setBackground(new
java.awt.Color(0, 102, 102));

        javax.swing.GroupLayout jPanel2Layout =
new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)
            .addGap(0, 730, Short.MAX_VALUE)
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)
            .addGap(0, 520, Short.MAX_VALUE)
        );

        getContentPane().add(jPanel2, new
org.netbeans.lib.awtextra.AbsoluteConstraints(
0, 0, 730, 520));

        pack();
    }// </editor-fold>

    private int ShowDialogOpenImage()
    {
        FileFilter filter1 = new
ExtensionFileFilter("Bitmap and JPEG Files",
new String[] { "BMP","JPG"});
        flchooser.setFileFilter(filter1);
        flchooser.setMultiSelectionEnabled(true);
        int tanggapan =
flchooser.showOpenDialog(this);
        return tanggapan;
    }

    public void showWarningMessage(String
pesan)
```

```
   {
      JOptionPane.showMessageDialog(this,
pesan,"Peringatan",JOptionPane.WARNING_ME
SSAGE);
   }

   public void showInfoMessage(String pesan)
   {
      JOptionPane.showMessageDialog(this,
pesan,"Informasi",JOptionPane.INFORMATION_
MESSAGE);
   }


private void
jBtnRecogniseActionPerformed(java.awt.event.
ActionEvent evt) {
   if (sampleuji.getTotalFaces() >0)
   {
    GenderIdentifier er = new
GenderIdentifier(sampleuji);
     er.classification();
     showFilesImages();
   }else
   {
     showWarningMessage("Tidak Ada yang
dapat diproses!");
   }
}

private void
jbtnBrowseActionPerformed(java.awt.event.Act
ionEvent evt) {
   int tanggapan = ShowDialogOpenImage();

if(tanggapan==JFileChooser.APPROVE_OPTION)
   {
      BufferedImage img=null;
      try{
         for(int
i=0;i<flchooser.getSelectedFiles().length;i++)
         {
            img =
ImageIO.read(flchooser.getSelectedFiles()[i].get
AbsoluteFile());
            ImageIcon imgicon = new
ImageIcon(img);
            PixelsReader pxlsLoad = new
PixelsReader();
            pxlsLoad.readPixelsFrom(imgicon);
            FaceImage faceimg =
pxlsLoad.getFaceImage();
            sampleuji.addFaceImage(faceimg,
flchooser.getSelectedFiles()[i].getAbsolutePath(
), "-Belum Diketahui-");
         }

         showFilesImages();
      }catch(IOException ex)
      {

      }
   }
}

private void
jTblTestingMouseClicked(java.awt.event.Mouse
Event evt) {
   int baris =
jTblTesting.rowAtPoint(evt.getPoint());
   FaceImage faceimage =
sampleuji.getFaceImage(baris);

   FeatureExtractor fe =new FeatureExtractor();
   fe.extract(faceimage);

   FaceImageViewer fiv = new
FaceImageViewer();
   fiv.setImage(faceimage);
   jlblHasil.setText(faceimage.getName());
   fiv.setViewer(jlblUji);
   fiv.viewImageDefault();
}

private void
jBtnRecogniseMouseClicked(java.awt.event.Mo
useEvent evt) {

}


   public static void main(String args[]) {

      try {
```

```
        for
(javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFee
ls()) {
            if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.get
ClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(FormTesting.
class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
    } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(FormTesting.
class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
    } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(FormTesting.
class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
    } catch
(javax.swing.UnsupportedLookAndFeelExceptio
n ex) {

java.util.logging.Logger.getLogger(FormTesting.
class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
    }
    java.awt.EventQueue.invokeLater(new
Runnable() {

        public void run() {
            new FormTesting().setVisible(true);
        }
    });
  }
  // Variables declaration - do not modify
  private javax.swing.JButton jBtnRecognise;
  private javax.swing.JLabel jLabel1;
  private javax.swing.JLabel jLabel2;
  private javax.swing.JPanel jPanel1;
  private javax.swing.JPanel jPanel2;
  private javax.swing.JScrollPane jScrollPane1;
  private javax.swing.JTable jTblTesting;
  private javax.swing.JButton jbtnBrowse;
  private javax.swing.JLabel jlblHasil;
  private javax.swing.JLabel jlblUji;
  // End of variables declaration
}
```

JFRAME FORMTRAINING

```
package ui;


import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.ImageIO;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.filechooser.FileFilter;
import
javax.swing.filechooser.FileNameExtensionFilter
;
import javax.swing.table.DefaultTableModel;
import pkg2dpcasomgender.FaceImage;
import pkg2dpcasomgender.FaceImageViewer;
import pkg2dpcasomgender.FeatureExtractor;
import pkg2dpcasomgender.PixelsReader;
import pkg2dpcasomgender.SomLearner;
import pkg2dpcasomgender.TrainingSet;
import utils.ExtensionFileFilter;
import utils.Util;

public class FormTraining extends
javax.swing.JFrame
{

  JFileChooser flchooser = new JFileChooser();
  TrainingSet tsfaces;
  SomLearner somlearner ;

  public FormTraining()
```

```
    {
        initComponents();
        Util.TengahWindow(this);
        setResizable(false);
        setTitle("Training");

        tsfaces = new TrainingSet();
        showFilesImages();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed"
desc="Generated Code">
    private void initComponents() {

        jComboBox1 = new
javax.swing.JComboBox();
        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jLabel5 = new javax.swing.JLabel();
        jPanel3 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jtxtIterasiMaksimum = new
javax.swing.JTextField();
        jLabel3 = new javax.swing.JLabel();
        jTxtLearningRate = new
javax.swing.JTextField();
        jLabel4 = new javax.swing.JLabel();
        jTxtLatWidth = new
javax.swing.JTextField();
        jTxtLatHeight = new
javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        jScrollPane1 = new
javax.swing.JScrollPane();
        jTblTrainData = new javax.swing.JTable();
        jBtnLoadTrainingSet = new
javax.swing.JButton();
        jBtnSmpanBobot = new
javax.swing.JButton();
        jBtnTrain = new javax.swing.JButton();
        jLabel6 = new javax.swing.JLabel();

        jComboBox1.setModel(new
javax.swing.DefaultComboBoxModel(new
String[] { "Item 1", "Item 2", "Item 3", "Item 4"
}));

setDefaultCloseOperation(javax.swing.Window
Constants.DISPOSE_ON_CLOSE);

        jPanel1.setBackground(new
java.awt.Color(0, 102, 102));


jPanel2.setBorder(javax.swing.BorderFactory.cr
eateTitledBorder("Face Image"));
        jPanel2.setPreferredSize(new
java.awt.Dimension(143, 218));

        jLabel5.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void
mouseClicked(java.awt.event.MouseEvent evt) {
                jLabel5MouseClicked(evt);
            }
        });

        javax.swing.GroupLayout jPanel2Layout =
new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGrou
p()
                .addContainerGap()
                .addComponent(jLabel5,
javax.swing.GroupLayout.DEFAULT_SIZE, 205,
Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGrou
p()
                .addContainerGap()
                .addComponent(jLabel5,
javax.swing.GroupLayout.DEFAULT_SIZE, 271,
Short.MAX_VALUE))
```

```
        );


jPanel3.setBorder(javax.swing.BorderFactory.cr
eateTitledBorder("Train Parameter"));
        jPanel3.setFont(new java.awt.Font("Comic
Sans MS", 1, 12)); // NOI18N

        jLabel1.setText("Iterasi Maksimum :");

        jtxtIterasiMaksimum.setText("10000");

jtxtIterasiMaksimum.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent
evt) {

jtxtIterasiMaksimumActionPerformed(evt);
            }
        });

        jLabel3.setText("Learning Rate :");

        jTxtLearningRate.setText("0.6");

        jLabel4.setText("Map Label (pxl) :");

        jTxtLatWidth.setText("40");

        jTxtLatHeight.setText("40");

        jLabel2.setText("x");

        javax.swing.GroupLayout jPanel3Layout =
new javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.
TRAILING,
jPanel3Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel3Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGrou
p()

.addGroup(jPanel3Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.TRAILING)
                        .addComponent(jLabel1,
javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jLabel3,
javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.
LEADING,
jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(jLabel4)

.addGroup(jPanel3Layout.createSequentialGrou
p()

.addComponent(jTxtLatWidth,
javax.swing.GroupLayout.PREFERRED_SIZE, 60,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(18, 18, 18)

.addComponent(jLabel2)))

.addPreferredGap(javax.swing.LayoutStyle.Com
ponentPlacement.UNRELATED)

.addComponent(jTxtLatHeight,
javax.swing.GroupLayout.PREFERRED_SIZE, 60,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(80,
Short.MAX_VALUE))

.addGroup(jPanel3Layout.createSequentialGrou
p()

.addGroup(jPanel3Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.TRAILING)
```

```
.addComponent(jtxtIterasiMaksimum,
javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jTxtLearningRate,
javax.swing.GroupLayout.Alignment.LEADING))
                .addGap(21, 21, 21))))
        );
        jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGrou
p()
            .addContainerGap()
            .addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.Com
ponentPlacement.RELATED)
            .addComponent(jtxtIterasiMaksimum,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(33, 33, 33)
            .addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.Com
ponentPlacement.RELATED)
            .addComponent(jTxtLearningRate,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(34, 34, 34)
            .addComponent(jLabel4)

.addPreferredGap(javax.swing.LayoutStyle.Com
ponentPlacement.RELATED)

.addGroup(jPanel3Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jTxtLatWidth,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jTxtLatHeight,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jLabel2))
            .addContainerGap(73,
Short.MAX_VALUE))
        );

        jTblTrainData.setModel(new
javax.swing.table.DefaultTableModel(
            new Object [][] {
                {null, null, null, null},
                {null, null, null, null},
                {null, null, null, null},
                {null, null, null, null}
            },
            new String [] {
                "Title 1", "Title 2", "Title 3", "Title 4"
            }
        ));
        jTblTrainData.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void
mouseClicked(java.awt.event.MouseEvent evt) {
                jTblTrainDataMouseClicked(evt);
            }
        });

jScrollPane1.setViewportView(jTblTrainData);

        jBtnLoadTrainingSet.setBackground(new
java.awt.Color(0, 153, 153));
        jBtnLoadTrainingSet.setFont(new
java.awt.Font("Copperplate Gothic Bold", 1,
12)); // NOI18N
        jBtnLoadTrainingSet.setText("Load
Images");

jBtnLoadTrainingSet.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent
evt) {

jBtnLoadTrainingSetActionPerformed(evt);
            }
        });
```

```
        jBtnSmpanBobot.setBackground(new
java.awt.Color(0, 153, 153));
        jBtnSmpanBobot.setFont(new
java.awt.Font("Copperplate Gothic Bold", 1,
12)); // NOI18N
        jBtnSmpanBobot.setText("Simpan Bobot");
        jBtnSmpanBobot.setEnabled(false);
        jBtnSmpanBobot.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent
evt) {

jBtnSmpanBobotActionPerformed(evt);
            }
        });

        jBtnTrain.setBackground(new
java.awt.Color(0, 153, 153));
        jBtnTrain.setFont(new
java.awt.Font("Copperplate Gothic Bold", 1,
12)); // NOI18N
        jBtnTrain.setText("Training");
        jBtnTrain.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent
evt) {
                jBtnTrainActionPerformed(evt);
            }
        });

        jLabel6.setFont(new java.awt.Font("Cooper
Black", 0, 24)); // NOI18N
        jLabel6.setText("TRAINING");

        javax.swing.GroupLayout jPanel1Layout =
new javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGrou
p()

.addGroup(jPanel1Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGrou
p()

.addContainerGap(javax.swing.GroupLayout.DE
FAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jLabel6)
                .addGap(374, 374, 374))

.addGroup(jPanel1Layout.createSequentialGrou
p()
                .addGap(32, 32, 32)

.addGroup(jPanel1Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING,
false)
                    .addComponent(jPanel3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                    .addComponent(jBtnTrain,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.Com
ponentPlacement.RELATED, 19,
Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING,
false)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 374,
Short.MAX_VALUE)

.addComponent(jBtnLoadTrainingSet,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
                .addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING)
```

```
.addGroup(jPanel1Layout.createSequentialGrou
p()
                    .addGap(0, 0,
Short.MAX_VALUE)
                    .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 237,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(jPanel1Layout.createSequentialGrou
p()

.addComponent(jBtnSmpanBobot,
javax.swing.GroupLayout.PREFERRED_SIZE, 238,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(0, 0,
Short.MAX_VALUE)))))
            .addContainerGap())
      );
      jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.
GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.
TRAILING,
jPanel1Layout.createSequentialGroup()
            .addGap(0, 31, Short.MAX_VALUE)
            .addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.Com
ponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(j
avax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGrou
p()
                .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 305,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jBtnSmpanBobot,
javax.swing.GroupLayout.PREFERRED_SIZE, 62,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(jPanel1Layout.createSequentialGrou
p()
.addComponent(jBtnLoadTrainingSet)
                .addGap(11, 11, 11)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 413,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(jPanel1Layout.createSequentialGrou
p()
                .addComponent(jPanel3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jBtnTrain,
javax.swing.GroupLayout.PREFERRED_SIZE, 62,
javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addContainerGap())
      );


jPanel3.getAccessibleContext().setAccessibleNa
me("Training Parameter");

      javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
      getContentPane().setLayout(layout);
      layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)
          .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
      );
      layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)
          .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
      );

      pack();
   }// </editor-fold>
```

```
   private void
jBtnTrainActionPerformed(java.awt.event.Actio
nEvent evt) {
      // TODO add your handling code here:
      int imax =
Integer.parseInt(jtxtIterasiMaksimum.getText())
;
      double learnrate =
Double.parseDouble(jTxtLearningRate.getText()
);
      int latWidth =
Integer.parseInt(jTxtLatWidth.getText());
      int latHeight =
Integer.parseInt(jTxtLatHeight.getText());
      if (tsfaces.getTotalFaces()>1)
      {

        somlearner = new
SomLearner(tsfaces,imax,learnrate,latWidth,lat
Height);
        somlearner.start();
        jBtnSmpanBobot.setEnabled(true);

      }else  {
         showMessage("Data Anda Belum
Cukup");
      }

   }

   public void showMessage(String pesan)
   {
      JOptionPane.showMessageDialog(this,
pesan,"Peringatan",JOptionPane.WARNING_ME
SSAGE);
   }

   private void showFilesImages()
   {
      String[] header = new String[]{"No","Image
FileName","Label"};
      String[][] data = new
String[tsfaces.getTotalFaces()][header.length];
      for(int i=0;i<tsfaces.getTotalFaces();i++)
      {
         String ffname =
tsfaces.getFaceImage(i).getName();
         data[i][0] = String.valueOf(i+1);
         data[i][1] = tsfaces.getFileNames(i);
         data[i][2] = ffname;

      }
      jTblTrainData.setModel(new
DefaultTableModel(data, header));
}

  private int ShowDialogOpenImage()
  {
   FileFilter filter1 = new
ExtensionFileFilter("Bitmap and JPEG Files",
new String[] { "BMP","JPG"});
   flchooser.setFileFilter(filter1);
   flchooser.setMultiSelectionEnabled(true);
   int tanggapan =
flchooser.showOpenDialog(this);
   return tanggapan;
  }

private void
jLabel5MouseClicked(java.awt.event.MouseEve
nt evt) {
// TODO add your handling code here:

}

private void
jTblTrainDataMouseClicked(java.awt.event.Mou
seEvent evt) {
// TODO add your handling code here:
   int baris =
jTblTrainData.rowAtPoint(evt.getPoint());
   FaceImage faceimage =
tsfaces.getFaceImage(baris);

   FeatureExtractor fe =new FeatureExtractor();
   fe.extract(faceimage);

   FaceImageViewer fiv = new
FaceImageViewer();
   fiv.setImage(faceimage);
   fiv.setViewer(jLabel5);
   fiv.viewImageDefault();
}
```

```
private void
jBtnLoadTrainingSetActionPerformed(java.awt.
event.ActionEvent evt) {
// TODO add your handling code here:

   int tanggapan = ShowDialogOpenImage();

if(tanggapan==JFileChooser.APPROVE_OPTION)
   {
      BufferedImage img=null;
      try{
         String name =
JOptionPane.showInputDialog("Masukkan
Nama Label:");
         if (name==null)
         {
            showMessage("File yang terpilih
dibatalkan.");
         }
         else
         {
            for(int
i=0;i<flchooser.getSelectedFiles().length;i++)
            {
               img =
ImageIO.read(flchooser.getSelectedFiles()[i].get
AbsoluteFile());
               ImageIcon imgicon = new
ImageIcon(img);
               PixelsReader pxlsLoad = new
PixelsReader();
               pxlsLoad.readPixelsFrom(imgicon);
               FaceImage faceimg =
pxlsLoad.getFaceImage();

               //System.out.println(name);
               tsfaces.addFaceImage(faceimg,
flchooser.getSelectedFiles()[i].getAbsolutePath(
), name);
            }
            showFilesImages();
         }
      }
      catch(IOException ex){
      }
   }
}
```

```
private void
jBtnSmpanBobotActionPerformed(java.awt.eve
nt.ActionEvent evt) {
// TODO add your handling code here:

   if (somlearner.isFinished())
   {
      //somlearner.printCluster();
      //somlearner.testCluster();
      somlearner.saveWeights();
   }else {
      showMessage("Training Belum Selesai.");
   }

}

   private void
jtxtIterasiMaksimumActionPerformed(java.awt.
event.ActionEvent evt) {
      // TODO add your handling code here:
   }

   /**
    * @param args the command line arguments
    */
   public static void main(String args[]) {
      /*
       * Set the Nimbus look and feel
       */
      //<editor-fold defaultstate="collapsed"
desc=" Look and feel setting code (optional) ">
      /*
       * If Nimbus (introduced in Java SE 6) is not
available, stay with the
       * default look and feel. For details see
       *
http://download.oracle.com/javase/tutorial/uis
wing/lookandfeel/plaf.html
       */
      try {
         for
(javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFee
ls()) {
            if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.get
ClassName());
```

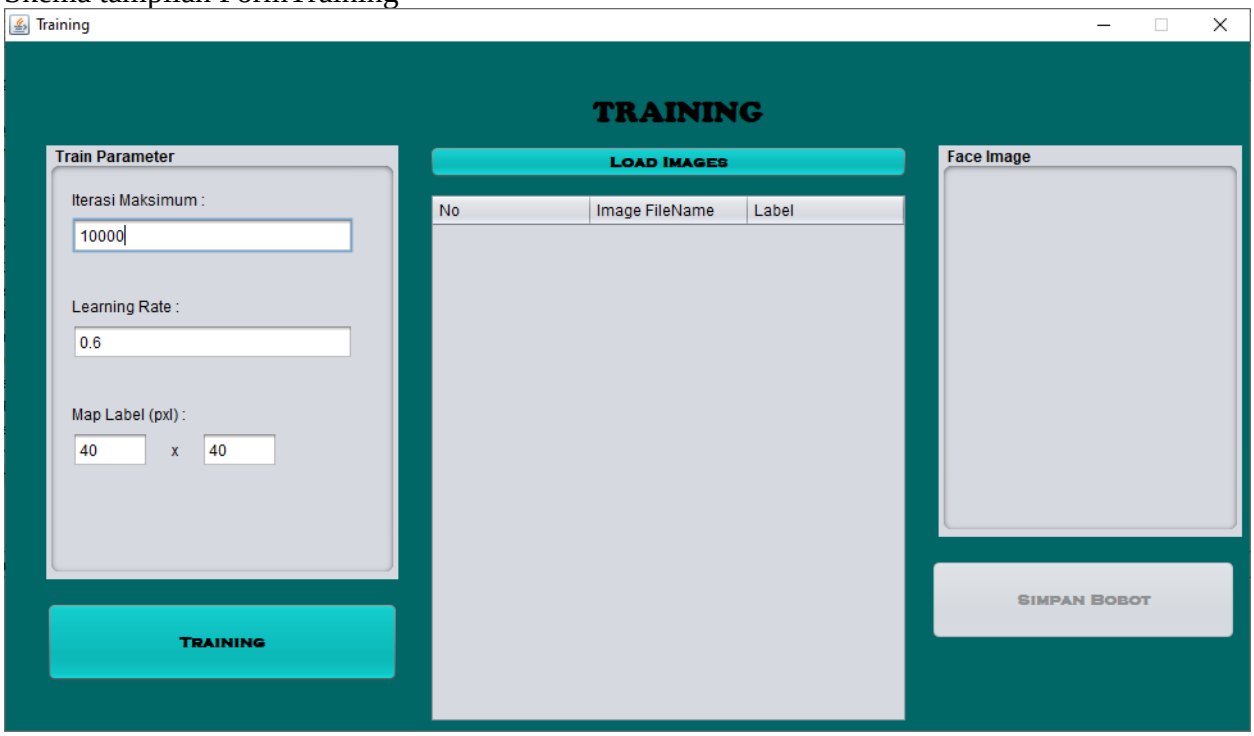
```
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(FormTraining
.class.getName()).log(java.util.logging.Level.SEV
ERE, null, ex);
    } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(FormTraining
.class.getName()).log(java.util.logging.Level.SEV
ERE, null, ex);
    } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(FormTraining
.class.getName()).log(java.util.logging.Level.SEV
ERE, null, ex);
    } catch
(javax.swing.UnsupportedLookAndFeelExceptio
n ex) {

java.util.logging.Logger.getLogger(FormTraining
.class.getName()).log(java.util.logging.Level.SEV
ERE, null, ex);
    }
    //</editor-fold>

    /*
     * Create and display the form
     */
    java.awt.EventQueue.invokeLater(new
Runnable() {

        public void run() {
            new FormTraining().setVisible(true);
        }
    });
  }
  // Variables declaration - do not modify
  private javax.swing.JButton
jBtnLoadTrainingSet;
  private javax.swing.JButton jBtnSmpanBobot;
  private javax.swing.JButton jBtnTrain;
  private javax.swing.JComboBox jComboBox1;
  private javax.swing.JLabel jLabel1;
  private javax.swing.JLabel jLabel2;
  private javax.swing.JLabel jLabel3;
  private javax.swing.JLabel jLabel4;
  private javax.swing.JLabel jLabel5;
  private javax.swing.JLabel jLabel6;
  private javax.swing.JPanel jPanel1;
  private javax.swing.JPanel jPanel2;
  private javax.swing.JPanel jPanel3;
  private javax.swing.JScrollPane jScrollPane1;
  private javax.swing.JTable jTblTrainData;
  private javax.swing.JTextField jTxtLatHeight;
  private javax.swing.JTextField jTxtLatWidth;
  private javax.swing.JTextField
jTxtLearningRate;
  private javax.swing.JTextField
jtxtIterasiMaksimum;
  // End of variables declaration
}
```

V. Penutup

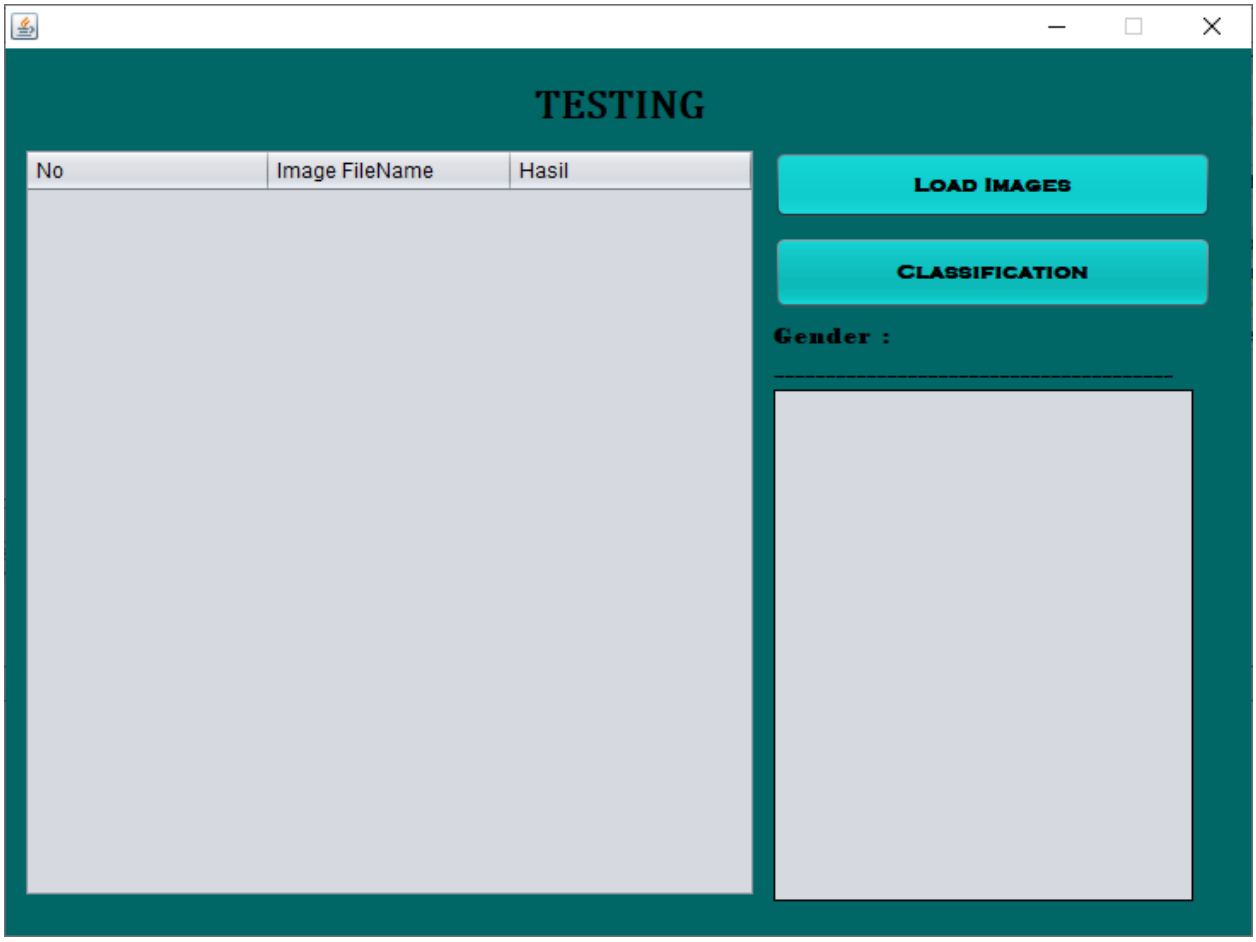
Setelah proses coding selesai. Dapat dikatan Aplikasi berjalan dan berhasil mengklasifikasi jenis kelamin seseorang. Melalui netbeans 8.2. konsep pemrograman menggunakan konsep Pemrograman berorientasi objek(PBO) dengan menggunakan bahasan pemrograman JAVA.

1. Skema tampilan FormMain

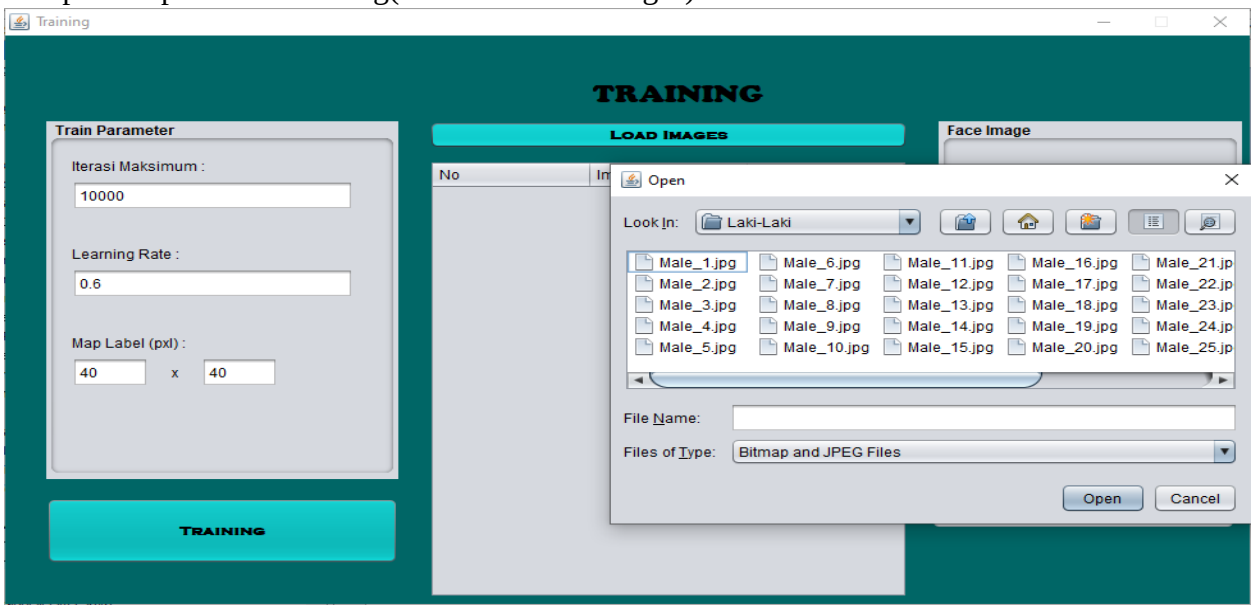
2. Skema tampilan FormTraining



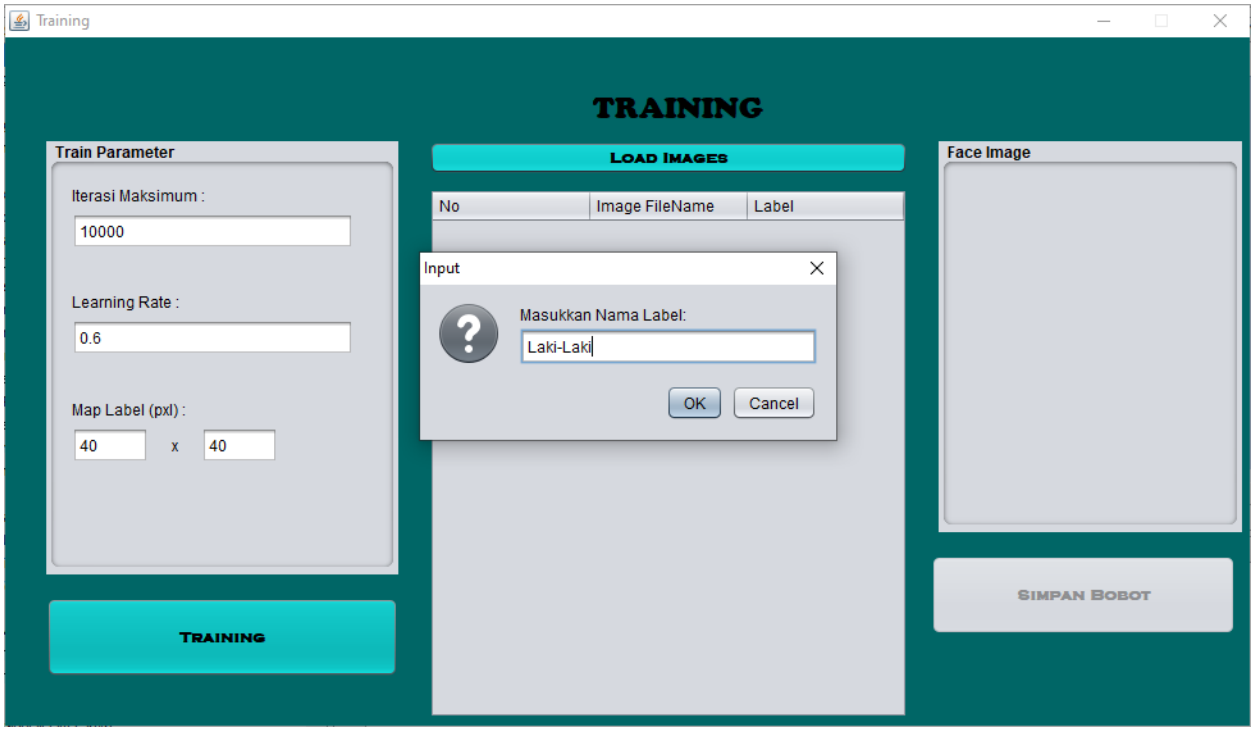
3. Skema Tampilan FormTesting



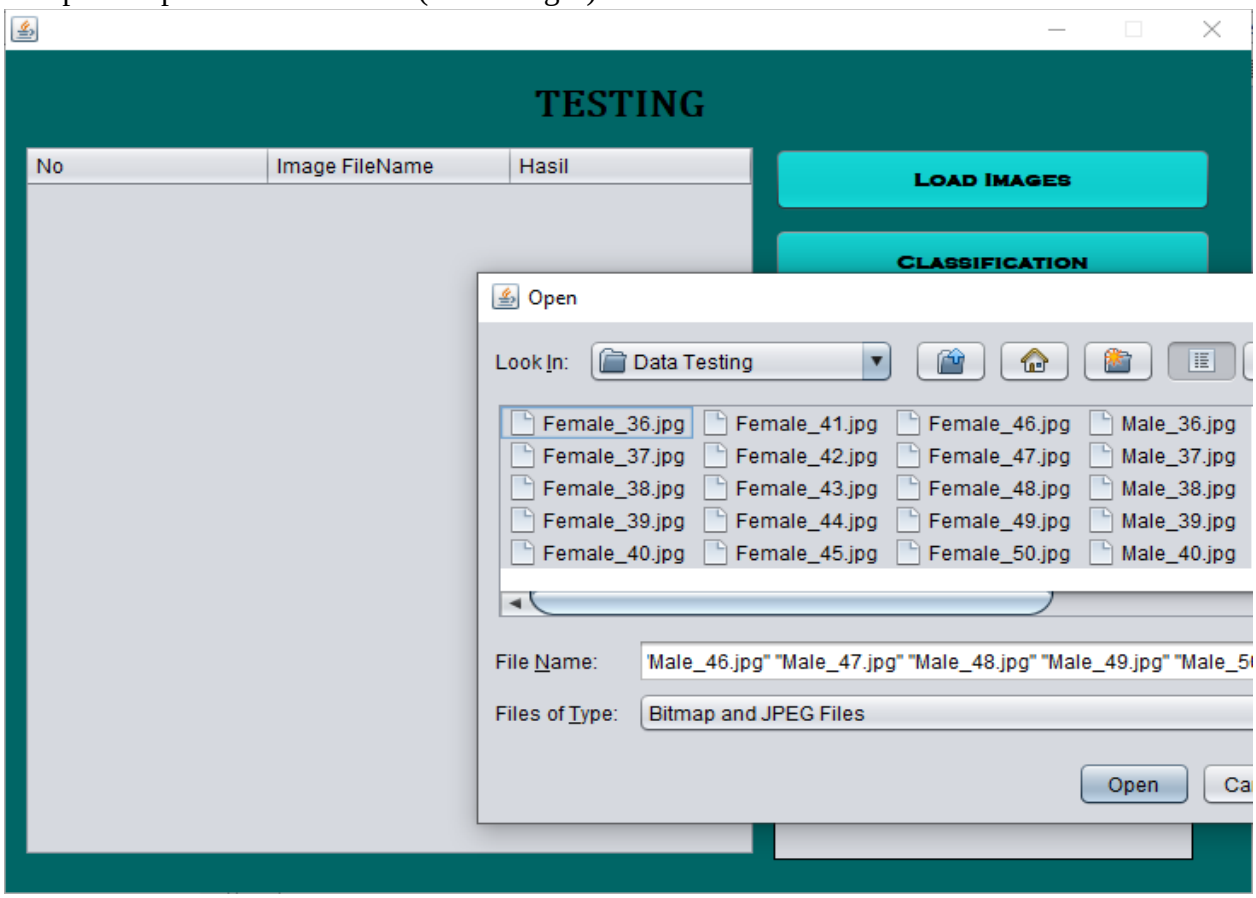
4. Tampilan Input Data Training(Menekan LoadImages)



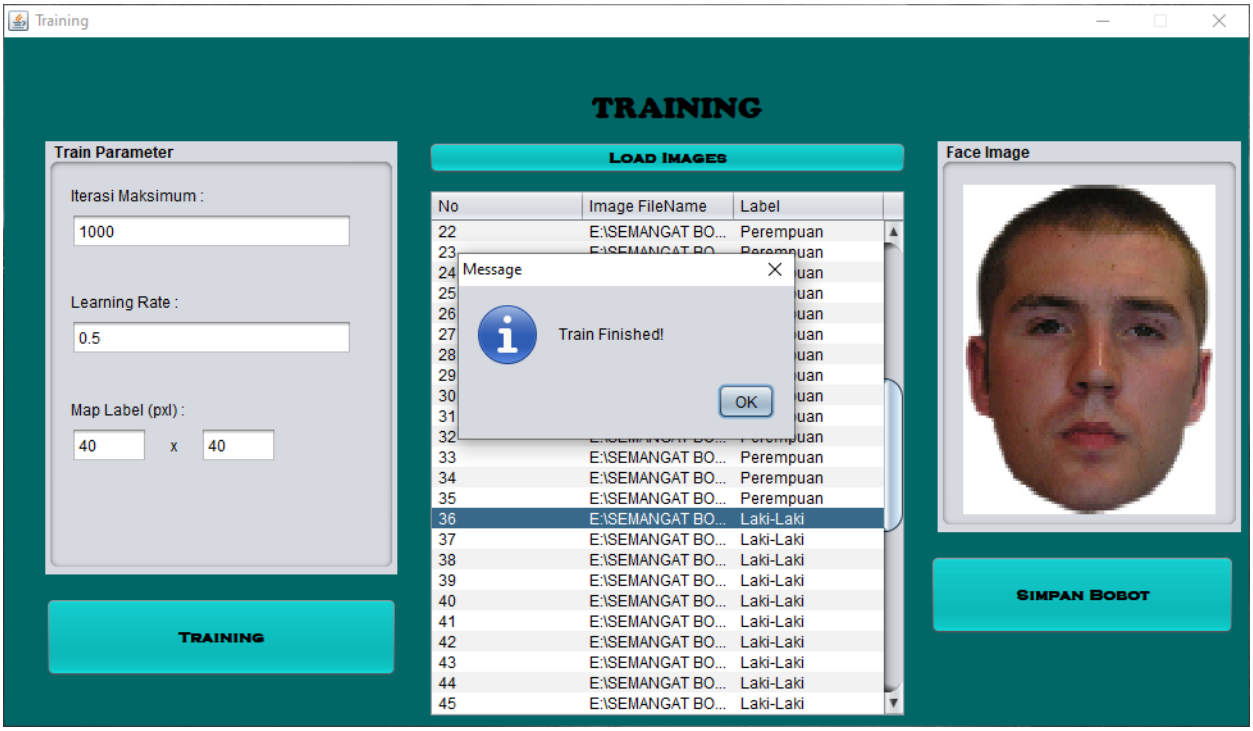
5. Tampilan Input Label untuk Data Training



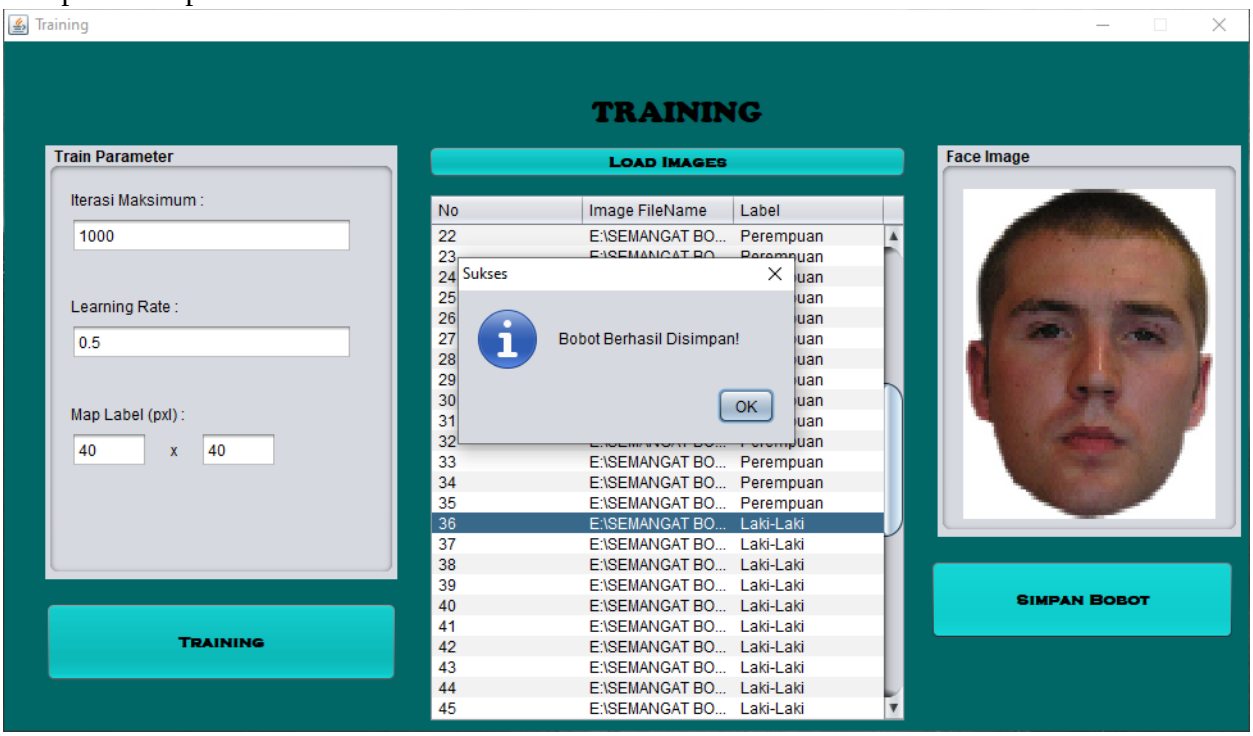
6. Tampilan Input Data Klasifikasi(LoadImages)



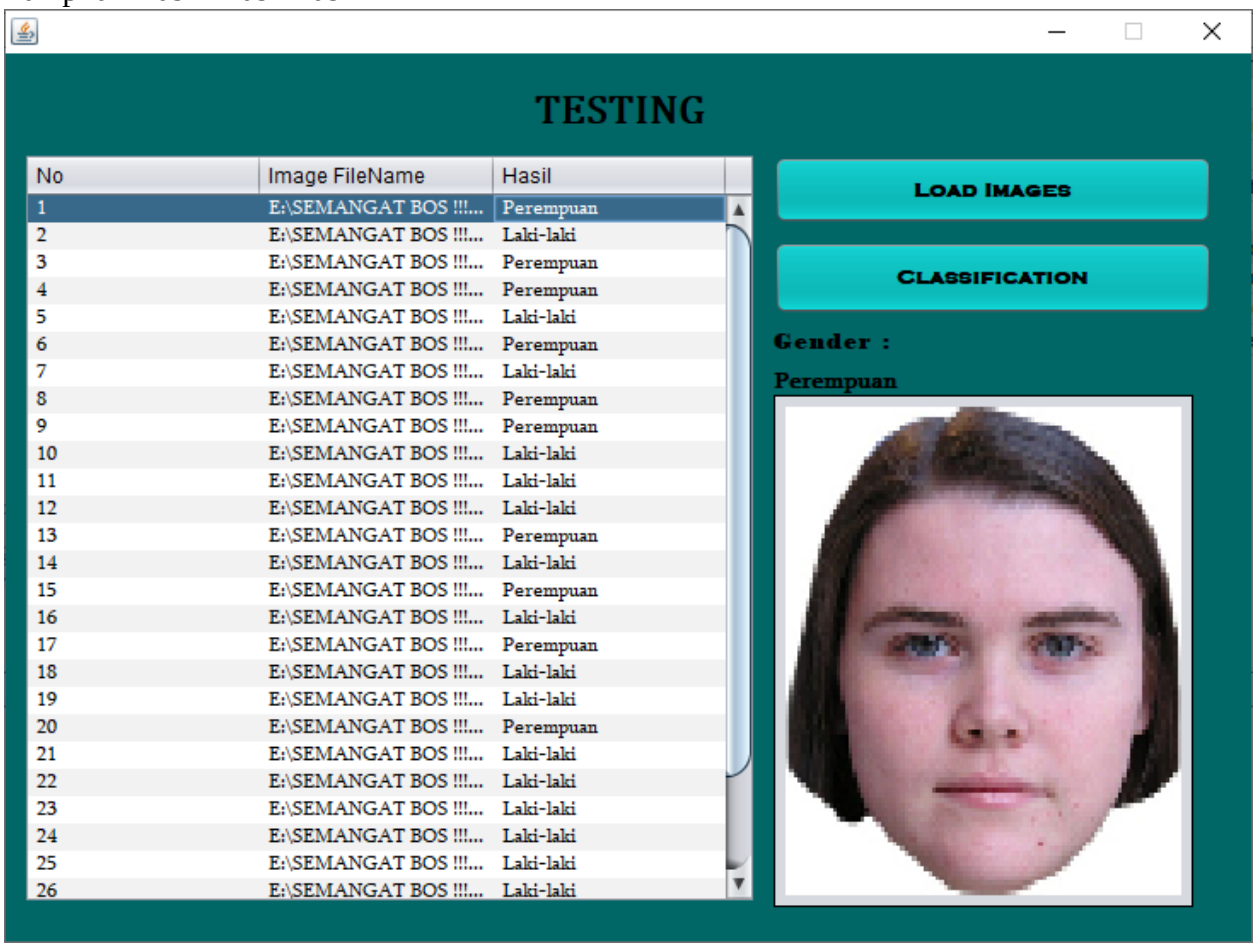
7. Tampilan Training Selesai



8. Tampilan Simpan Bobot



9. Tampilan Hasil Klasifikasi



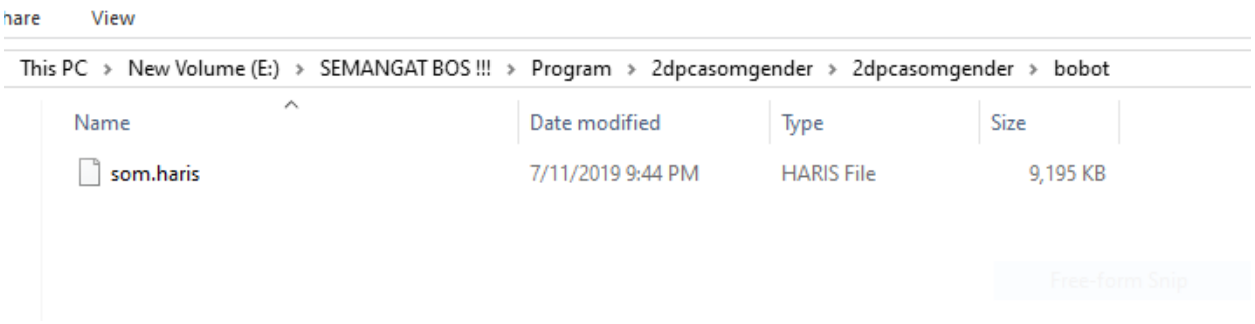
➢ Langkah Penggunaan Perangkat Lunak :

- *User* menjalankan Running perangkat lunak melalui netbeans.
- *User* memilih Training di tampilan FormMain
- Di tampilan form training *User* menekan "*Load Images*" untuk menginput data training yang sudah disiapkan. Dengan catatan citra wajah training yang di masukkan lebih dari 1 gambar.
- *User* menginput "Parameter Train" untuk batasan training data, dalam kasus ini dengan start learning rate 0.5, 0.6 dan 0.7 dengan iterasi maksimum 1000, 2000, 5000, 10000.
- *User* menekan tombol "*Training*", tunggu proses training selesai sampai timbul pop-up "*Train Finished !*" lalu Tekan OK.
- Lalu *User* Menekan tombol "Simpan Bobot" untuk menyimpan hasil training yang digunakan untuk proses klasifikasi. Proses Training Selesai.
- Keluar dari tampilan form training, dan masuk ke tampilan testing
- *User* memilih Testing di tampilan FormTesting
- Di tampilan form testing *User* menekan "*Load Images"* untuk menginput data testing yang sudah disiapkan untuk melakukan proses klasifikasi. Dengan catatan citra wajah yang ingin di testing belum pernah dilakukan proses training.

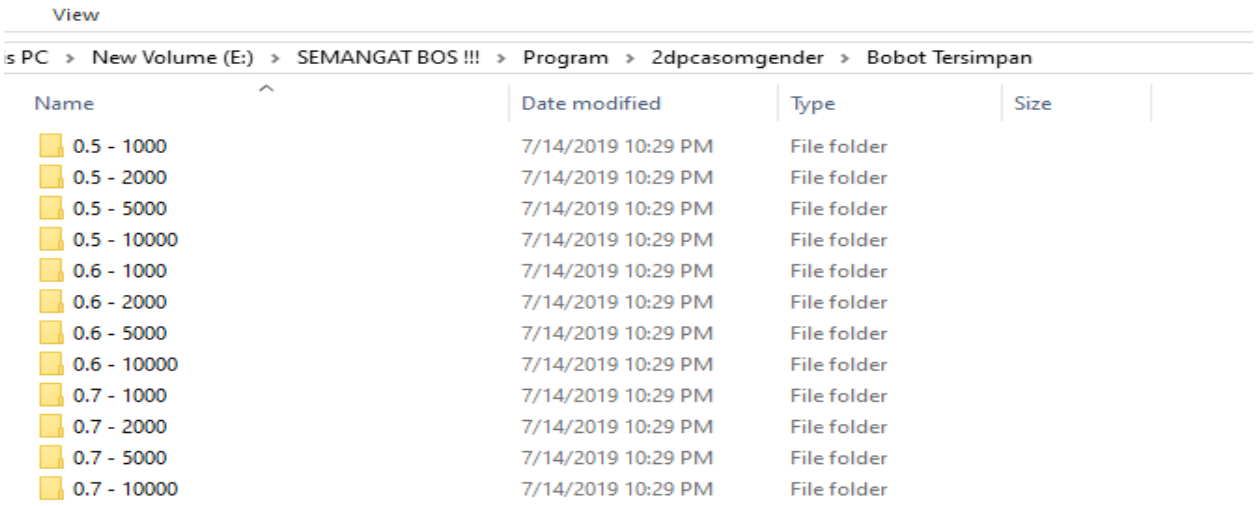
- Setelah data testing di masukkan, *User* menekan tombol "*Clasification*", perangkat lunak akan memperlihatkan hasil dari proses klasifikasi dari citra wajah seseorang.

#. Catatan Perangkat lunak :

a. Pada proses untuk penggunaan perangkat lunak, harus melakukan training terlebih dahulu sebelum melakukan testing.
b. Setelah proses training selesai, *user* harus memindahkan hasil klasifikasi yang paling optimal ke folder yang berbeda agar dapat digunakan untuk proses klasifikasi yang menghasilkan hasil yang baik. Saat penyimpanan bobot, bobot akan tersimpan ke dalam folder bobot di dalam program dengan file name **som.**(namaektensididalamprogram) Lalu pindahkan bobot tersebut kedapan folder yang berbeda agar pada proses training selanjutnya file tidak terhapus. Pada proses ini bisa dilihat pada gambar dibawah.



Bobot tersimpan di dalam aplikasi dengan folder bobot.



Bobot yang sudah di susun dan di pisahkan dari folder bobot di dalam aplikasi