BAB 5 IMPLEMENTASI

Pada bab ini akan dibahas implementasi dari sistem sesuai dengan perancangan yang telah dibuat pada bab sebelumnya. Bab ini terdiri dari beberapa sub bab yaitu implementasi kode program dan implementasi antarmuka pengguna.

5.1 Implementasi Metode

Pada sub bab ini akan dijelaskan source code dari masing-masing proses dari metode SPK AHP-TOPSIS untuk prioritas perbaikan jalan. Masing-masing proses tersebut antara lain perhitungan bobot Kriteria AHP, perhitungan bobot sub kriteria AHP, *preprocessing*, normalisasi, matriks ternormalisasi keputusan, matriks ternormalisasi pembobotan, solusi ideal positif dan solusi ideal negatif, *separation measure*, dan nilai alternatif.

5.1.1 Proses Perhitungan Bobot Kriteria AHP

```
No
 1  //--------CARI SIGMA PER KOLOM BOBOT KRITERIA AWAL
 2          System.out.println("SIGMA PER KOLOM KRITERIA");
 3          for (int i = 0; i < sigmaKolomKriteria.length; i++) {
 4              for (int j = 0; j < matriksBobotKriteria.length;
 5  j++) {
 6                  sigmaKolomKriteria[i] +=
 7  Double.valueOf(df.format(matriksBobotKriteria[j][i]));
 8              }
 9              sigmaKolomKriteria[i] =
10  Double.valueOf(df.format(sigmaKolomKriteria[i]));
11              System.out.print(sigmaKolomKriteria[i] + " ");
12          }
13          System.out.println("");
14  //CARI NORMALISASI MATRIKS KEPENTINGAN DAN NILAI EIGEN(BOBOT)
15          System.out.println("HASIL NORMALISASI MATRIKS
16  KEPENTINGAN KRITERIA");
17          for (int i = 0; i < matriksBobotKriteriaNorm.length;
18  i++) {
19              for (int j = 0; j <
20  matriksBobotKriteriaNorm[0].length; j++) {
21                  matriksBobotKriteriaNorm[i][j] =
22  matriksBobotKriteria[i][j] / sigmaKolomKriteria[j];
23                  matriksBobotKriteriaNorm[i][j] =
24  Double.valueOf(df_.format(matriksBobotKriteriaNorm[i][j]));
25
26  System.out.print(matriksBobotKriteriaNorm[i][j] + " ");
27                  eigenKriteria[i] +=
28  matriksBobotKriteriaNorm[i][j];
29              }
30              System.out.println("");
31              eigenKriteria[i] =
32  Double.valueOf(df.format(eigenKriteria[i] / 5));
```

```
33          }
34          System.out.println("NILAI EIGEN KRITERIA");
35          for (int i = 0; i < eigenKriteria.length; i++) {
36              System.out.print(eigenKriteria[i] + " ");
37          }
38          System.out.println("");
39  //CARI NILAI LAMBDA, CI DAN CR
40          for (int i = 0; i < Ax.length; i++) {
41              for (int j = 0; j <
42  matriksBobotKriteria[0].length; j++) {
43                  Ax[i] += (matriksBobotKriteria[i][j] *
44  eigenKriteria[j]);
45              }
46              AVG[i] = Ax[i] / eigenKriteria[i];
47              lambda += AVG[i];
48          }
49          lambda = lambda / 5;
50          CI_Kriteria = (lambda - 5) / 4;
51          CR_Kriteria = Double.valueOf(df.format(CI_Kriteria /
52  IR_Kriteria));
53          System.out.println("NILAI CONSISNTENCY RATIO KRITERIA
54  = " + CR_Kriteria);
55          return eigenKriteria;
56
```

***Source Code* 5.1 Hitung Bobot Kriteria AHP**

Penjelasan dari source code pada Tabel 5.1 adalah sebagai berikut :

1 Baris 2-13 adalah proses untuk memperoleh jumlah per kolom dari matriks perbandingan kepentingan.

2 Baris 17-39 adalah proses untuk menghitung normalisasi dari matriks perbandingan kepentingan, nilai eigen dan kemudian mencetaknya.

3 Baris 41-56 adalah proses untuk mencari nilai *Consistency Ratio* untuk menentukan apakah bobot / nilai eigen sudah konsisten atau tidak.

5.1.2 *Preprocessing*

No	
1	`String database_Reprocessing[][] = new String[315][10];`
2	`for (int i = 0; i < database_Reprocessing.length;`
3	`i++) {`
4	`for (int j = 0; j <`
5	`database_Reprocessing[0].length; j++) {`
6	`database_Reprocessing[i][j] = database[i][j];`
7	`if (j == 7) {`
8	`if (database[i][j].equals("K")) {`
9	`database_Reprocessing[i][j] =`
10	`String.valueOf(2);`
11	`}`
12	`if (database[i][j].equals("P")) {`
13	`database_Reprocessing[i][j] =`
14	`String.valueOf(1);`

```
15                      }
16                  }
17                  if (j == 8) {
18                      if (database[i][j].equals("JJS")) {
19                          database_Reprocessing[i][j] =
20  String.valueOf(2);
21                      }
22                      if (database[i][j].equals("LU")) {
23                          database_Reprocessing[i][j] =
24  String.valueOf(1);
25                      }
26                    }
27                  }
28                  if (j == 9) {
29                      if (database[i][j].equals("Ya")) {
30                          database_Reprocessing[i][j] =
31  String.valueOf(1);
32                      }
33                      if (database[i][j].equals("Tidak")) {
34                          database_Reprocessing[i][j] =
35  String.valueOf(0);
36                      }
37                  }
38              }
39          }
            return database_Reprocessing;
```

Source Code* 5.2 *Preprocessing

Penjelasan dari source code pada Tabel 5.3 adalah sebagai berikut :

1. Baris 5 berfungsi untuk memasukkan array database kedalam array database_Reprocessing untuk nantinya dilakukan reproses terhadap data tersebut.
2. Baris 6-15 adalah proses untuk mengubah nilai dari Tabel database_Reproscessing pada kolom ke 8 menjadi nilai 1 atau 2.
3. Baris 16-26 adalah proses untuk mengubah nilai dari Tabel database_Reproscessing pada kolom ke 9 menjadi nilai 1 atau 2.
4. Baris 27-38 adalah proses untuk mengubah nilai dari Tabel database_Reproscessing pada kolom ke 10 menjadi nilai 1 atau 0.
5. Baris 39 berfungsi untuk menyimpan nilai dari database_Reprocessing.

5.1.3 Proses Normalisasi

No	
1	`public double[][] normalisasi_Max_Min(String[][] database) {`
2	`        double[][] normalisasi_Max_Min = new double[315][8];`
3	`        double max[] = new double[8];`
4	`        double min[] = new double[8];`
5	

```
6           for (int i = 0; i < normalisasi_Max_Min.length; i++)
7   {
8               for (int j = 0; j <
9   normalisasi_Max_Min[0].length; j++) {
10                  normalisasi_Max_Min[i][j] =
11  Double.parseDouble(database[i][j + 2]);
12              }
13          }
14          for (int i = 0; i < normalisasi_Max_Min[0].length;
15  i++) {
16              min[i] = normalisasi_Max_Min[0][i];
17              max[i] = normalisasi_Max_Min[0][i];
18              for (int j = 0; j < normalisasi_Max_Min.length;
19  j++) {
20                  if (normalisasi_Max_Min[j][i] < min[i]) {
21                      min[i] = normalisasi_Max_Min[j][i];
22                  }
23                  if (normalisasi_Max_Min[j][i] > max[i]) {
24                      max[i] = normalisasi_Max_Min[j][i];
25                  }
26              }
27          }
28          for (int i = 0; i < normalisasi_Max_Min.length; i++)
29  {
30              for (int j = 0; j <
31  normalisasi_Max_Min[0].length; j++) {
32                  normalisasi_Max_Min[i][j] =
33  Double.valueOf(df.format(((normalisasi_Max_Min[i][j] -
34  min[j])) / (max[j] - min[j])));
35              }
36          }
37          return normalisasi_Max_Min;
38      }
39
40      public double[] sigmaRow(double[][] normalisasi_Max_Min)
41  {
42          double sigmaRow[] = new double[315];
43          for (int i = 0; i < normalisasi_Max_Min.length; i++)
44  {
45              for (int j = 0; j <
46  normalisasi_Max_Min[0].length; j++) {
47                  sigmaRow[i] +=
48  Math.pow(normalisasi_Max_Min[i][j], 2);
49              }
50              sigmaRow[i] =
    Double.valueOf(df_.format(Math.pow(sigmaRow[i], 0.5)));
          }
          return sigmaRow;
      }
```

***Source Code* 5.3 Normalisasi**

Penjelasan dari source code pada Tabel 5.4 adalah sebagai berikut :

1 Baris 1-36 berfungsi untuk melakukan normalisasi max – min terhadap database yang sudah di reproses.

2 Baris 38-50 adalah proses untuk menghitung nilai sigma dari setiap baris data normalisasi max min yang sudah di peroleh pada baris 35.

5.1.4 Proses Matriks Ternormalisasi Keputusan

```
No
 1 public double[][] matriks_Normalisasi_Keputusan(double[][]
 2 normalisasi_Max_Min, double[] sigmaRow) {
 3         double[][] matriks_Normalisasi_Keputusan = new
 4 double[normalisasi_Max_Min.length][normalisasi_Max_Min[0].length];
 5         for (int i = 0; i < matriks_Normalisasi_Keputusan.length;
 6 i++) {
 7             for (int j = 0; j <
 8 matriks_Normalisasi_Keputusan[0].length; j++) {
 9                 matriks_Normalisasi_Keputusan[i][j] =
10 Double.valueOf(df_.format(normalisasi_Max_Min[i][j] /
11 sigmaRow[i]));
12             }
13         }
14         return matriks_Normalisasi_Keputusan;
15     }
```

***Source Code* 5.4 Proses Matriks Ternormalisasi Keputusan**

Penjelasan dari source code pada Tabel 5.5 adalah sebagai berikut :

1 Baris 1 Deklarasi method matriks_Normalisasi_Keputusan dengan parameter normalisasi_Max_Min dan sigmaRow.

2 Baris 3-15 adalah proses untuk menghitung nilai matriks_Normalisasi_Keputusan yang diperoleh dari hasil bagi antara parameter normalisasi_Max_Min dengan parameter sigmaRow.

5.1.5 Proses Matriks Ternormalisasi Pembobotan

```
No
 1 public double[][] matriks_Normalisasi_Pembobotan(double[][]
 2 matriks_Normalisasi_Keputusan, double[] bobot_kriteria,
 3 double[]
 4  bobot_SubKriteria) {
 5         double[][] matriks_Normalisasi_Pembobotan = new
 6 double[matriks_Normalisasi_Keputusan.length]
 7 [matriks_Normalisasi_Keputusan[0].length];
 8         for (int i = 0; i <
 9 matriks_Normalisasi_Pembobotan.length; i++) {
10             for (int j = 0; j <
11 matriks_Normalisasi_Pembobotan[0].length; j++) {
12                 matriks_Normalisasi_Pembobotan[i][j] =
13 Double.valueOf(df_.format(matriks_Normalisasi_Keputusan[i][j] *
14 (bobot_kriteria[j])));
15             }
16         }
17         return matriks_Normalisasi_Pembobotan;
     }
```

***Source Code* 5.5 Proses Matriks Ternormalisasi Pembobotan**

Penjelasan dari source code pada Tabel 5.6 adalah sebagai berikut :

1 Baris 1 Deklarasi method matriks_Normalisasi_Pembobotan dengan parameter matriks_Normalisasi_Keputusan, bobot kriteria dan bobot subkriteria.

2 Baris 4-17 adalah proses perhitungan matriks_Normalisasi_Pembobotan dengan langkah mengalikan antara matriks_Normalisasi_Keputusan dengan bobot kriteria.

5.1.6 Proses Solusi Ideal Positif dan Negatif

```
No
 1  public double[] solusi_Ideal_Positif(double[][]
 2 matriks_Normalisasi_Pembobotan) {
 3         double[] solusi_Ideal_Positif = new
 4 double[matriks_Normalisasi_Pembobotan[0].length];
 5         for (int i = 0; i <
 6 matriks_Normalisasi_Pembobotan[0].length; i++) {
 7             solusi_Ideal_Positif[i] =
 8 matriks_Normalisasi_Pembobotan[0][i];
 9             for (int j = 0; j <
10 matriks_Normalisasi_Pembobotan.length; j++) {
11                 if (i == 0 || i == 1) {
12                     if (matriks_Normalisasi_Pembobotan[j][i] <
13 solusi_Ideal_Positif[i]) {
14                         solusi_Ideal_Positif[i] =
15 Double.valueOf(df_.format(matriks_Normalisasi_Pembobotan[j][i]));
16                     }
17
18                 } else {
19                     if (matriks_Normalisasi_Pembobotan[j][i] >
20 solusi_Ideal_Positif[i]) {
21                         solusi_Ideal_Positif[i] =
22 Double.valueOf(df_.format(matriks_Normalisasi_Pembobotan[j][i]));
23                     }
24
25                 }
26             }
27         }
28         return solusi_Ideal_Positif;
29     }
30
31 public double[] solusi_Ideal_Negatif(double[][]
32 matriks_Normalisasi_Pembobotan) {
33         double[] solusi_Ideal_Negatif = new
34 double[matriks_Normalisasi_Pembobotan[0].length];
35         for (int i = 0; i <
36 matriks_Normalisasi_Pembobotan[0].length; i++) {
37             solusi_Ideal_Negatif[i] =
38 matriks_Normalisasi_Pembobotan[0][i];
39             for (int j = 0; j <
40 matriks_Normalisasi_Pembobotan.length; j++) {
```

```
41                 if (i == 0 || i == 1) {
42                     if (matriks_Normalisasi_Pembobotan[j][i] >
43 solusi_Ideal_Negatif[i]) {
44                         solusi_Ideal_Negatif[i] =
45 Double.valueOf(df_.format(matriks_Normalisasi_Pembobotan[j][i]));
46                     }
47                 } else {
48                     if (matriks_Normalisasi_Pembobotan[j][i] <
49 solusi_Ideal_Negatif[i]) {
50                         solusi_Ideal_Negatif[i] =
51 Double.valueOf(df_.format(matriks_Normalisasi_Pembobotan[j][i]));
52                     }
53                 }
54             }
55         }
56         return solusi_Ideal_Negatif;
57     }
```

***Source Code* 5.6 Proses Solusi Ideal Positif dan Negatif**

Penjelasan dari source code pada Tabel 5.7 adalah sebagai berikut :

1. Baris 1 Deklarasi method solusi_Ideal_Positif dengan parameter matriks_Normalisasi_Pembobotan.
2. Baris 2-29 adalah proses perhitungan nilai solusi ideal positif dari matriks normalisasi pembobotan.
3. Baris 31 Deklarasi method solusi_Ideal_Negatif dengan parameter matriks_Normalisasi_Pembobotan.
4. Baris 32-57 adalah proses perhitungan nilai solusi ideal negatif dari matriks normalisasi pembobotan.

5.1.7 *Separation Measure*

No	
1	`public double[] separation_Max(double[][]`
2	`matriks_Normalisasi_Pembobotan, double[]`
3	`solusi_Ideal_Positif) {`
4	`        double separation_Max[] = new`
5	`double[matriks_Normalisasi_Pembobotan.length];`
6	`        for (int i = 0; i <`
7	`matriks_Normalisasi_Pembobotan.length; i++) {`
8	`            for (int j = 0; j <`
9	`matriks_Normalisasi_Pembobotan[0].length; j++) {`
10	`                separation_Max[i] +=`
11	`Math.pow((matriks_Normalisasi_Pembobotan[i][j] -`
12	`solusi_Ideal_Positif[j]), 2);`
13	`            }`
14	`            separation_Max[i] =`
15	`Double.valueOf(df_.format(Math.pow(separation_Max[i], 0.5)));`
16	`        }`
17	`        return separation_Max;`
18	`    }`
19	

```
20  public double[] separation_Min(double[][]
21  matriks_Normalisasi_Pembobotan, double[]
22  solusi_Ideal_Negatif) {
23          double separation_Min[] = new
24  double[matriks_Normalisasi_Pembobotan.length];
25          for (int i = 0; i <
26  matriks_Normalisasi_Pembobotan.length; i++) {
27              for (int j = 0; j <
28  matriks_Normalisasi_Pembobotan[0].length; j++) {
29                  separation_Min[i] +=
30  Math.pow((matriks_Normalisasi_Pembobotan[i][j] -
31  solusi_Ideal_Negatif[j]), 2);
32              }
33              separation_Min[i] =
34  Double.valueOf(df_.format(Math.pow(separation_Min[i], 0.5)));
35          }
            return separation_Min;
        }
```

Source Code* 5.7 *Separation Measure

Penjelasan dari source code pada Tabel 5.8 adalah sebagai berikut :

1. Baris 1 Deklarasi method separation_Measure_Max dengan parameter matriks_Normalisasi_Pembobotan dan solusi_Ideal_Positif.
2. Baris 2-17 adalah proses perhitungan nilai separation measure max dari matriks normalisasi pembobotan.
3. Baris 19 Deklarasi method separation_Measure_Min dengan parameter matriks_Normalisasi_Pembobotan dan solusi_Ideal_Negatif.
4. Baris 20-35 adalah proses perhitungan nilai separation measure min dari matriks normalisasi pembobotan.

5.1.8 Proses Perhitungan Nilai Alternatif

```
No
 1  public double[][] nilai_Alternative(double[] separation_Max,
 2  double[] separation_Min) {
 3          double[][] nilai_Alternative = new
 4  double[separation_Max.length][2];
 5          for (int i = 0; i < nilai_Alternative.length; i++) {
 6              nilai_Alternative[i][1] = separation_Min[i] /
 7  (separation_Min[i] + separation_Max[i]);
 8
 9              nilai_Alternative[i][0] = i + 1;
10          }
11          return nilai_Alternative;
12      }
```

***Source Code* 5.8 Proses Perhitungan Nilai Alternatif**

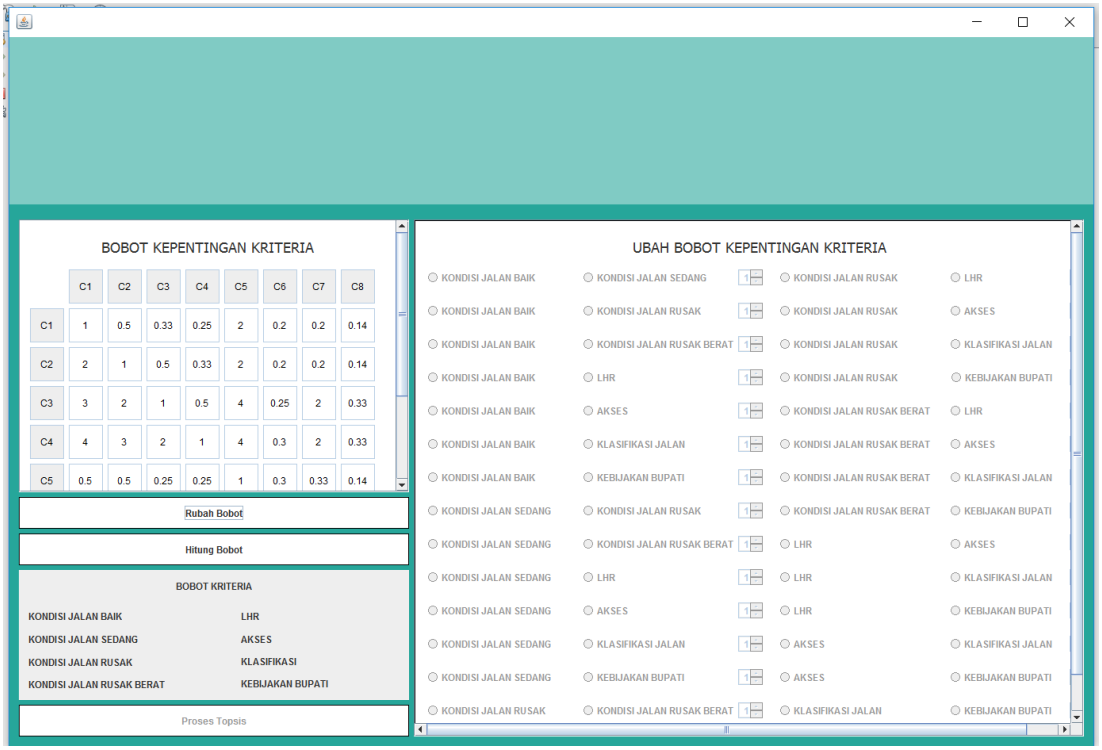
Penjelasan dari source code pada Tabel 5.9 adalah sebagai berikut :

1 Baris 1 Deklarasi method nilai_Alternative dengan parameter separation_Max dan separation_Min.

2 Baris 2-12 adalah proses perhitungan nilai alternatif dari setiap alternatif yang ada.

5.2 Implementasi Antarmuka

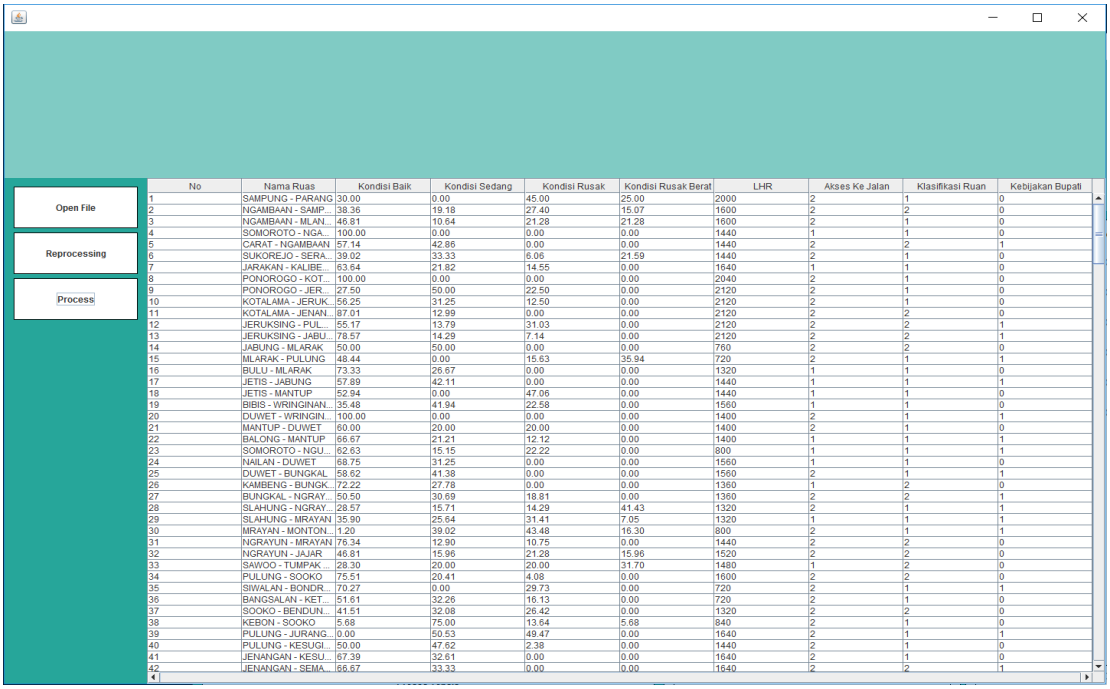
5.2.1 Antarmuka Awal / Antarmuka Matriks Bobot Kepentingan



Gambar 5.1 Antarmuka Awal / Antarmuka Matriks Bobot Kepentingan

Antarmuka awal atau antarmuka matriks bobot kepentingan digunakan untuk mendeklarasikan bobot awal dari setiap kriteria dan sub kriteria yang sudah di tentukan serta digunakan juga untuk mengubah bobot awal kriteria dan sub kriteria yang ada.

5.2.2 Antarmuka Input *Data* dan *Pre-processing*

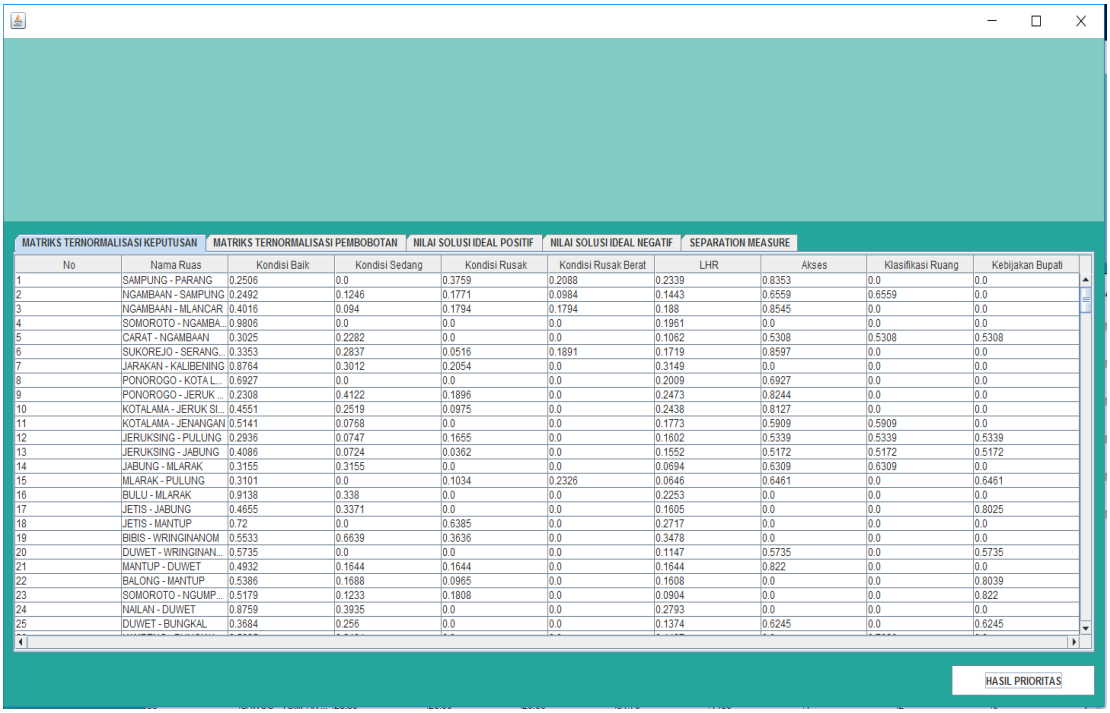


No	Nama Ruas	Kondisi Baik	Kondisi Sedang	Kondisi Rusak	Kondisi Rusak Berat	LHR	Akses Ke Jalan	Klasifikasi Ruan	Kebijakan Bupati
1	SAMPUNG - PARANG	30.00	0.00	45.00	25.00	2000	2	1	0
2	NGAMBAAN - SAMP...	38.36	19.18	27.40	15.07	1600	2	2	0
3	NGAMBAAN - MLAN...	46.81	10.64	21.28	21.28	1600	2	1	0
4	SOMOROTO - NGA...	100.00	0.00	0.00	0.00	1440	1	1	0
5	CARAT - NGAMBAAN	57.14	42.86	0.00	0.00	1440	2	2	1
6	SUKOREJO - SERA...	39.02	33.33	6.06	21.59	1440	2	1	0
7	JARAKAN - KALIBE...	63.64	21.82	14.55	0.00	1640	1	1	0
8	PONOROGO - KOT...	100.00	0.00	0.00	0.00	2040	2	1	0
9	PONOROGO - JER...	27.50	50.00	22.50	0.00	2120	2	1	0
10	KOTALAMA - JERUK...	56.25	31.25	12.50	0.00	2120	2	1	0
11	KOTALAMA - JENAN...	87.01	12.99	0.00	0.00	2120	2	2	0
12	JERUKSING - PUL...	55.17	13.79	31.03	0.00	2120	2	2	1
13	JERUKSING - JABU...	78.57	14.29	7.14	0.00	2120	2	2	1
14	JABUNG - MLARAK	50.00	50.00	0.00	0.00	760	2	2	0
15	MLARAK - PULUNG	48.44	0.00	15.63	35.94	720	2	1	1
16	BULU - MLARAK	73.33	26.67	0.00	0.00	1320	1	1	0
17	JETIS - JABUNG	57.89	42.11	0.00	0.00	1440	1	1	1
18	JETIS - MANTUP	52.94	0.00	47.06	0.00	1440	1	1	0
19	BIBIS - WRINGINAN...	35.48	41.94	22.58	0.00	1560	1	1	0
20	DUWET - WRINGIN...	100.00	0.00	0.00	0.00	1400	2	1	1
21	MANTUP - DUWET	60.00	20.00	20.00	0.00	1400	2	1	0
22	BALONG - MANTUP	66.67	21.21	12.12	0.00	1400	1	1	1
23	SOMOROTO - NGU...	62.63	15.15	22.22	0.00	800	1	1	1
24	NAILAN - DUWET	68.75	31.25	0.00	0.00	1560	1	1	0
25	DUWET - BUNGKAL	58.62	41.38	0.00	0.00	1560	2	1	1
26	KAMBENG - BUNGK...	72.22	27.78	0.00	0.00	1360	1	2	0
27	BUNGKAL - NGRAY...	50.50	30.69	18.81	0.00	1360	2	2	1
28	SLAHUNG - NGRAY...	28.57	15.71	14.29	41.43	1320	2	1	1
29	SLAHUNG - MRAYAN	35.90	25.64	31.41	7.05	1320	1	1	1
30	MRAYAN - MONTON...	1.20	39.02	43.48	16.30	800	2	1	1
31	NGRAYUN - MRAYAN	76.34	12.90	10.75	0.00	1440	2	2	0
32	NGRAYUN - JAJAR	46.81	15.96	21.28	15.96	1520	2	2	0
33	SAWOO - TUMPAK ...	28.30	20.00	20.00	31.70	1480	1	2	0
34	PULUNG - SOOKO	75.51	20.41	4.08	0.00	1600	2	2	0
35	SIWALAN - BONDR...	70.27	0.00	29.73	0.00	720	2	1	1
36	BANGSALAN - KET...	51.61	32.26	16.13	0.00	720	2	1	0
37	SOOKO - BENDUN...	41.51	32.08	26.42	0.00	1320	2	2	0
38	KEBON - SOOKO	5.68	75.00	13.64	5.68	840	2	1	0
39	PULUNG - JURANG...	0.00	50.53	49.47	0.00	1640	2	1	1
40	PULUNG - KESUGI...	50.00	47.62	2.38	0.00	1440	2	1	0
41	JENANGAN - KESU...	67.39	32.61	0.00	0.00	1640	2	1	0
42	JENANGAN - SEMA...	66.67	33.33	0.00	0.00	1640	2	2	1

Gambar 5.2 Antarmuka Input *Database* dan *Pre-processing*

Antarmuka input *data* dan *Pre-processing* digunakan untuk memasukkan database jalan yang ditangani oleh dinas PU kota Ponorogo serta berfungsi untuk melakukan reprocessing terhadap database tersebut.

5.2.3 Antarmuka Perhitungan TOPSIS

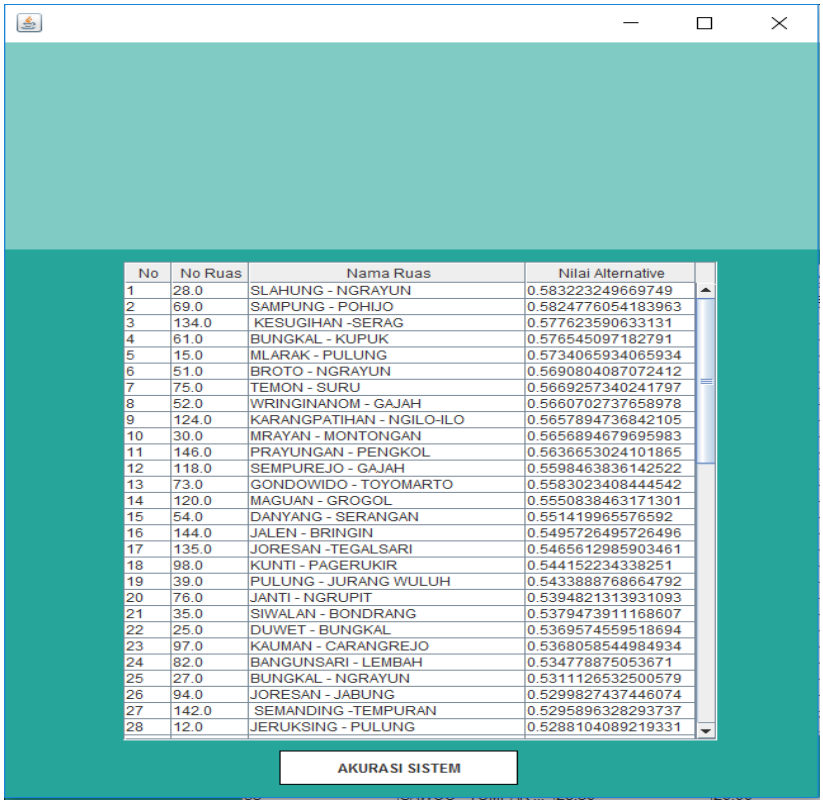


No	Nama Ruas	Kondisi Baik	Kondisi Sedang	Kondisi Rusak	Kondisi Rusak Berat	LHR	Akses	Klasifikasi Ruang	Kebijakan Bupati
1	SAMPUNG - PARANG	0.2506	0.0	0.3759	0.2088	0.2339	0.8353	0.0	0.0
2	NGAMBAAN - SAMPUNG	0.2492	0.1246	0.1771	0.0984	0.1443	0.6559	0.6559	0.0
3	NGAMBAAN - MLANCAR	0.4016	0.094	0.1794	0.1794	0.188	0.8545	0.0	0.0
4	SOMOROTO - NGAMBA...	0.9806	0.0	0.0	0.0	0.1961	0.0	0.0	0.0
5	CARAT - NGAMBAAN	0.3025	0.2282	0.0	0.0	0.1062	0.5308	0.5308	0.5308
6	SUKOREJO - SERANG...	0.3353	0.2837	0.0516	0.1891	0.1719	0.8597	0.0	0.0
7	JARAKAN - KALIBENING	0.8764	0.3012	0.2054	0.0	0.3149	0.0	0.0	0.0
8	PONOROGO - KOTA L...	0.6927	0.0	0.0	0.0	0.2009	0.6927	0.0	0.0
9	PONOROGO - JERUK ...	0.2308	0.4122	0.1896	0.0	0.2473	0.8244	0.0	0.0
10	KOTALAMA - JERUK SI...	0.4551	0.2519	0.0975	0.0	0.2438	0.8127	0.0	0.0
11	KOTALAMA - JENANGAN	0.5141	0.0768	0.0	0.0	0.1773	0.5909	0.5909	0.0
12	JERUKSING - PULUNG	0.2936	0.0747	0.1655	0.0	0.1602	0.5339	0.5339	0.5339
13	JERUKSING - JABUNG	0.4086	0.0724	0.0362	0.0	0.1552	0.5172	0.5172	0.5172
14	JABUNG - MLARAK	0.3155	0.3155	0.0	0.0	0.0694	0.6309	0.6309	0.0
15	MLARAK - PULUNG	0.3101	0.0	0.1034	0.2326	0.0646	0.6461	0.0	0.6461
16	BULU - MLARAK	0.9138	0.338	0.0	0.0	0.2253	0.0	0.0	0.0
17	JETIS - JABUNG	0.4655	0.3371	0.0	0.0	0.1605	0.0	0.0	0.8025
18	JETIS - MANTUP	0.72	0.0	0.6385	0.0	0.2717	0.0	0.0	0.0
19	BIBIS - WRINGINANOM	0.5533	0.6639	0.3636	0.0	0.3478	0.0	0.0	0.0
20	DUWET - WRINGINAN...	0.5735	0.0	0.0	0.0	0.1147	0.5735	0.0	0.5735
21	MANTUP - DUWET	0.4932	0.1644	0.1644	0.0	0.1644	0.822	0.0	0.0
22	BALONG - MANTUP	0.5386	0.1688	0.0965	0.0	0.1608	0.0	0.0	0.8039
23	SOMOROTO - NGUMP...	0.5179	0.1233	0.1808	0.0	0.0904	0.0	0.0	0.822
24	NAILAN - DUWET	0.8759	0.3935	0.0	0.0	0.2793	0.0	0.0	0.0
25	DUWET - BUNGKAL	0.3684	0.256	0.0	0.0	0.1374	0.6245	0.0	0.6245

Gambar 5.3 Antarmuka Perhitungan TOPSIS

Antarmuka perhitungan TOPSIS berfungsi untuk menampilkan hasil perhitungan dari langkah-langkah metode TOPSIS.

5.2.4 Antarmuka Hasil Prioritas Perbaikan Jalan



No	No Ruas	Nama Ruas	Nilai Alternative
1	28.0	SLAHUNG - NGRAYUN	0.583223249669749
2	69.0	SAMPUNG - POHIJO	0.5824776054183963
3	134.0	KESUGIHAN -SERAG	0.577623590633131
4	61.0	BUNGKAL - KUPUK	0.576545097182791
5	15.0	MLARAK - PULUNG	0.5734065934065934
6	51.0	BROTO - NGRAYUN	0.5690804087072412
7	75.0	TEMON - SURU	0.5669257340241797
8	52.0	WRINGINANOM - GAJAH	0.5660702737658978
9	124.0	KARANGPATIHAN - NGILO-ILO	0.5657894736842105
10	30.0	MRAYAN - MONTONGAN	0.5656894679695983
11	146.0	PRAYUNGAN - PENGKOL	0.5636653024101865
12	118.0	SEMPUREJO - GAJAH	0.5598463836142522
13	73.0	GONDOWIDO - TOYOMARTO	0.5583023408444542
14	120.0	MAGUAN - GROGOL	0.5550838463171301
15	54.0	DANYANG - SERANGAN	0.551419965576592
16	144.0	JALEN - BRINGIN	0.5495726495726496
17	135.0	JORESAN -TEGALSARI	0.5465612985903461
18	98.0	KUNTI - PAGERUKIR	0.544152234338251
19	39.0	PULUNG - JURANG WULUH	0.5433888768664792
20	76.0	JANTI - NGRUPIT	0.5394821313931093
21	35.0	SIWALAN - BONDRANG	0.5379473911168607
22	25.0	DUWET - BUNGKAL	0.5369574559518694
23	97.0	KAUMAN - CARANGREJO	0.5368058544984934
24	82.0	BANGUNSARI - LEMBAH	0.534778875053671
25	27.0	BUNGKAL - NGRAYUN	0.5311126532500579
26	94.0	JORESAN - JABUNG	0.529982743744607 4
27	142.0	SEMANDING -TEMPURAN	0.5295896328293737
28	12.0	JERUKSING - PULUNG	0.5288104089219331

AKURASI SISTEM

Gambar 5.4 Antarmuka Hasil Prioritas Perbaikan Jalan dan Akurasi Sistem

Antarmuka hasil prioritas dan akurasi sistem berfungsi untuk menampilkan prioritas perbaikan jalan yang sudah dilakukan perhitungan oleh metode AHP-TOPSIS, serta menampilkan akurasi dari sistem yang sudah dibuat.