BAB 5 IMPLEMENTASI

Pada bab implementasi ini akan dibahas tentang implementasi sistem yang telah dibuat. Pembahasan implementasi meliputi implementasi program dan implementasi antarmuka.

5.1 Implementasi Program

Pada sub bab ini akan dijelaskan tentang implementasi program berdasarkan perancangan sistem yang sudah dijelaskan pada bab 4. Pada penjelasan implementasi program ini akan dijelaskan proses-proses dan Kode Program setiap fungsi menggunakan bahasa JAVA.

5.1.1 Implementasi inisialisasi awal

Berikut ini merupakan source code untuk inisialisasi awal seluruh variabel yang ditunjukkan oleh source code 5.1.

No	Source Code
1	private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
2	
3	int iterasi = 20;
4	int popSize = 30;
5	int stringLen = 4;
6	int lamda = 15;
7	
8	sim_annealing sa1 = new sim_annealing();
9	
10	this.getJenisKambing();
11	this.getValueComboBox3();
12	this.getValueComboBox4();
13	this.getValueComboBox5();
14	this.getValueComboBox6();
15	
16	sa1.init_populasi(popSize, stringLen, berat);
17	
18	sa1.setNilaiNutrisi(bahan1, bahan2, bahan3, bahan4);
19	sa1.hitungHarga(popSize, stringLen, cost1, cost2, cost3, cost4);
20	sa1.hitungNutrisi(popSize, stringLen);
21	sa1.hitungPenalty(berat, BK, PK, TDN, Ca, P, popSize, stringLen);
22	sa1.hitungFitness(popSize);

Source Code 5.1 Inisialisasi Awal

Method diatas adalah metod yang akan berjalan ketika button proses ditekan, method ini berfungsi untuk menentukan jumlah iterasi, jumlah populasi dan juga nilai lamda.

5.1.2 Inisialisasi populasi

Berikut ini merupakan source code untuk inisialisasi populasi seluruh variabel yang ditunjukkan oleh source code 5.2.

No	Source Code
1	public void init_populasi(int popSize, int stringLen, int berat) {
2	this.populasi = new double[popSize][stringLen];
3	double max, min, range;
4	
5	min = 1;
6	max = 0.1 * berat;
7	range = min + (max - min);
8	for (int i = 0; i < popSize; i++) {
9	for (int j = 0; j < stringLen; j++) {
10	this.populasi[i][j]= range * rand.nextDouble();
11	}
12	}
13	for (int i = 0; i < popSize; i++) {
14	for (int j = 0; j < stringLen; j++) {
15	System.out.print(this.populasi[i][j]+" ");
16	}
17	System.out.println("");
18	}
19	
20	System.out.println("Berat: "+berat);
21	
22	}

Source Code 5.2 Inisialisasi Variabel

Method diatas berfungsi untuk melakukan inisialisasi populasi dalam artian method diatas yang akan mencetak kromosom beserta berat masing-masing bahan yang dilakukan secara random.

5.1.3 Hitung Kebutuhan Minimal

Berikut ini merupakan source code untuk menghitung kebutuhan minimal nutrisi kambing yang ditunjukkan oleh source code 5.3.

No	Source Code
1	public void kebutuhanMinimal(double BB, double BK, double PK, double TDN, double Ca, double P){
2	this.BK = (BK/100) * BB;
3	this.PK = (PK/100) * (BB/10);
4	this.TDN = (TDN/100) * (BB/10);
5	this.Ca = (Ca/100) * (BB/10);
6	this.P = (P/100) * (BB/10);
7	
8	this.kebutuhanMinimal = new double[5];
9	this.kebutuhanMinimal[0] = this.BK;
10	this.kebutuhanMinimal[1] = this.PK;
11	this.kebutuhanMinimal[2] = this.TDN;
12	this.kebutuhanMinimal[3] = this.Ca;
13	this.kebutuhanMinimal[4] = this.P;
14	}

Source Code 5.3 Hitung Kebutuhan Minimal

Method diatas berfungsi untuk menghitung kebutuhan minimal nutrisi kambing, karena berbeda jenis dan berat kambing berbeda pula kebutuhan nutrisinya.

5.1.4 Hitung Harga

Berikut ini merupakan source code untuk menghitung harga pakan yang ditunjukkan oleh source code 5.4.

No	Source Code
1	public void hitungHarga(int popSize, int stringLen, int harga1, int harga2, int harga3, int harga4){
2	this.tampungHarga = new double [popSize][stringLen];
3	this.totalHarga = new double[popSize];
4	this.setSemuaHarga(harga1, harga2, harga3, harga4);
5	this.getSemuaHarga(stringLen);
6	for (int i = 0; i < popSize; i++) {
7	for (int j = 0; j < stringLen; j++) {
8	this.tampungHarga[i][j] = this.populasi[i][j] * this.harga[j];
9	}
10	}
11	for (int i = 0; i < popSize; i++) {
12	for (int j = 0; j < stringLen; j++) {
13	this.totalHarga[i] = this.totalHarga[i] + this.tampungHarga[i][j];
14	}
15	
16	}
17	}

Source Code 5.4 Hitung Harga

Method diatas berfungsi untuk menghitung harga dari setiap gen kemudian setelah mendapatkan nilai harga setiap gen dilakukan perhitungan total harga untuk setiap individu.

5.1.5 Hitung Nutrisi

Berikut ini merupakan source code untuk menghitung nutrisi pakan yang ditunjukkan oleh source code 5.5.

No	Source Code
1	public void hitungNutrisi(int popSize, int stringLen){
2	this.setSemuaNutrisi();
3	this.getSemuaBahan();
4	this.tampungNutrisi = new double[popSize][stringLen+1];
5	
6	//System.out.println("Nutrisi");
7	//proses menghitung nilai nutrisi
8	for (int i = 0; i < popSize; i++) {
9	for (int j = 0; j < stringLen+1; j++) {
10	for (int k = 0; k < 4; k++) {
11	this.tampungNutrisi[i][j] = this.tampungNutrisi[i][j] + ((this.tampungBahan[k][j]/100) * this.populasi[i][k]) ;
12	}
13	//System.out.print(this.tampungNutrisi[i][j]+" || ");
14	}
15	//System.out.println("");
16	}
17	}

Source Code 5.5 Hitung Nutrisi

Method diatas berfungsi untuk menghitung nutrisi dari setiap gen kemudian setelah mendapatkan nilai nutrisi setiap gen dilakukan perhitungan total nutrisi untuk setiap individu.

5.1.6 Hitung Penalty

Berikut ini merupakan source code untuk menghitung penalty nutrisi pakan yang ditunjukkan oleh source code 5.6.

No	Source Code
1	public void hitungPenalty(double BB, double BK, double PK, double TDN, double Ca, double P, int popSize, int stringLen){
2	this.kebutuhanMinimal(BB, BK, PK, TDN, Ca, P);
3	this.pinalty = new double[popSize][stringLen+1];
4	this.totalPenalty = new double[popSize];

```
5
6         System.out.println(this.BK);
7         System.out.println(this.PK);
8         System.out.println(this.TDN);
9         System.out.println(this.Ca);
10        System.out.println(this.P);
11
12        System.out.println("Pinalty");
13        //proses menghitung nilai fittness
14        for (int i = 0; i < popSize; i++) {
15           for (int j = 0; j < stringLen+1; j++) {
               //                System.out.println(this.tampungNutrisi[i][j]+"                ||
16 "+this.kebutuhanMinimal[j]);
17              if (this.tampungNutrisi[i][j] > this.kebutuhanMinimal[j]) {
18                 this.pinalty[i][j] = 0;
19              }else{
                   this.pinalty[i][j]        =        Math.abs(this.kebutuhanMinimal[j]        -
20 this.tampungNutrisi[i][j]);
21              }
22              this.totalPenalty[i] = this.totalPenalty[i] + this.pinalty[i][j];
23              //System.out.print(this.pinalty[i][j]+" || ");
24           }
25           //System.out.println("");
26           //System.out.println(this.totalPenalty[i]+" ");
27        }
28
29     }
```

Source Code 5.6 Hitung Penalty

Method diatas berfungsi untuk menghitung penalty dari nutrisi yang dimiliki individu. Pada method sebelumnya telah ditunjukkan method untuk menghitung nutrisi minimal yang dibutuhkan kambing dan juga method untuk menghitung kandungan nutrisi setiap bahan pakan yang dimiliki setiap individu. Kemudian pada method ini akan dihitung penalty dengan cara nutrisi pakan dikurangi dengan nutrisi minimal yang dibutuhkan kambing.

5.1.7 Hitung Fitness

Berikut ini merupakan source code untuk menghitung fitness setiap individu yang ditunjukkan oleh source code 5.7.

No	Source code
1	public void hitungFitness(int popSize){
2	this.fitness = new double[popSize];
3	System.out.println("Fitness");
4	for (int i = 0; i < this.fitness.length; i++) {

```
5           this.fitness[i] = 10000 / (this.totalHarga[i] + (this.totalPenalty[i]*100000));
6           System.out.println(this.fitness[i]);
7       }
8   }
```

Source Code 5.7 Hitung Fitness

Method ini berfungsi untuk menghitung fitness setiap individu parent.

5.1.8 Mutasi

Berikut ini merupakan source code untuk mendapatkan individu baru dengan proses mutasi yang ditunjukkan oleh source code 5.8.

No	Source Code
1	public void mutasi(int popSize, int stringLen, int lamda, int berat, double BK, double PK, double TDN, double Ca, double P, int harga1, int harga2, int harga3, int harga4) {
2	double sigma, rMax, rMin, rRange;
3	double fitChild, fitParent;
4	int countParent, count, trigger, cMin, cMax, cRange, exPoint1, exPoint2;
5	this.child = new double[popSize*lamda][stringLen];//inisialisasi popSize child
6	this.fitnessChild = new double[popSize*lamda];
7	
8	rMin = 1;
9	rMax = 3;
10	rRange = rMin + (rMax - rMin);
11	
12	cMin = 1;
13	cMax = 4;
14	cRange = cMax - cMin + 1;
15	
16	count = 0;
17	countParent = 0;
18	sigma = 0;
19	trigger = 0;
20	fitChild = 0;
21	fitParent = 0;
22	
23	
24	System.out.println("Tukar Posisi");
25	
26	for (int i = 0; i < (popSize * lamda); i++) {
27	System.out.println("======================");
28	if (count == lamda) {
29	count = 0;

```
30              trigger = 0;
31              countParent++;
32          }
33          if (trigger == 0) {
34              sigma = Math.round(rRange * rand.nextDouble());
35              if (sigma < 1) {
36                  sigma = 1;
37              }
38              System.out.println(sigma + " Sigma tetap");
39          } else {
40              System.out.println("FChild: "+i+" || FParent: "+countParent);
41              System.out.println(fitChild+" || "+fitParent);
42              if (fitChild > fitParent) {
43                  sigma = Math.round(sigma * 1.1);
44                  if (sigma < 1) {
45                      sigma = 1;
46                  }else if (sigma <= 2) {
47                      sigma = 2;
48                  }else {
49                      sigma = 3;
50                  }
51                  System.out.println(sigma + " Sigma lama x 1.1");
52              } else {
53                  sigma = Math.round(sigma * 0.9);
54                  if (sigma < 1) {
55                      sigma = 1;
56                  }else if (sigma <= 2) {
57                      sigma = 2;
58                  }else {
59                      sigma = 3;
60                  }
61                  System.out.println(sigma + " Sigma lama x 0.9");
62              }
63          }
64          for (int j = 0; j < sigma; j++) {
65              exPoint1 = rand.nextInt(cRange) + cMin;
66              exPoint2 = rand.nextInt(cRange) + cMin;
67              if (exPoint2 == exPoint1) {
68                  exPoint2 = rand.nextInt(cRange) + cMin;
69              }
70              System.out.println(exPoint1+" "+exPoint2);
71              for (int k = 0; k < stringLen; k++) {
72                  if (k == (exPoint1-1)) {
```

```
73                  this.child[i][k] = this.populasi[countParent][exPoint2-1];
74               }
75               else if (k == (exPoint2-1)) {
76                  this.child[i][k] = this.populasi[countParent][exPoint1-1];
77               }
78               else {
79                  this.child[i][k] = this.populasi[countParent][k];
80               }
81               System.out.print(this.child[i][k]+" || ");
82            }
83            this.hitungFitnessChild(stringLen, i, berat, BK, PK, TDN, Ca, P, harga1,
harga2, harga3, harga4);
84            fitChild = this.fitnessChild[i];
85            fitParent = this.fitness[countParent];
86         }
87         trigger++;
88         count++;
89         System.out.println("");
90      }
91
92      System.out.println("Child");
93      for (int i = 0; i < (popSize * lamda); i++) {
94         for (int j = 0; j < stringLen; j++) {
95            System.out.print(this.child[i][j]+" || ");
96         }
97         System.out.println("");
98      }
99
100   }
```

Source Code 5.8 Mutasi

Method diatas berfungi untuk membuat individu baru dengan cara melakukan proses mutasi dimana jumal individu yang dihasilkan tergantung pada nilai sigma yang di set pada inisialisasi awal.

5.1.9 Hitung Fitness Child

Berikut ini merupakan source code untuk mendapatkan nilai fitness child yang ditunjukkan oleh source code 5.9.

No	Source Code
1	public void hitungFitnessChild(int stringLen, int x, int berat, double BK, double PK, double TDN, double Ca, double P, int harga1, int harga2, int harga3, int harga4){
2	double totHarga = 0;
3	double[] nutrisi = new double[stringLen+1];

```
4       this.setSemuaHarga(harga1, harga2, harga3, harga4);
5       this.getSemuaHarga(stringLen);
6
7       System.out.println("Child: " + x);
8       for (int i = 0; i < stringLen; i++) {
9          System.out.print(this.child[x][i]+" || ");
10      }
11
12      for (int i = 0; i < stringLen; i++) {
13         totHarga = totHarga + (this.child[x][i] * this.harga[i]);
14      }
15      this.setSemuaNutrisi();
16      this.getSemuaBahan();
17
18         for (int i = 0; i < stringLen+1; i++) {
19            for (int j = 0; j < 4; j++) {
20               nutrisi[i] = nutrisi[i] + ((this.tampungBahan[j][i]/100) * this.child[x][j])
;
21            }
22
23         }
24
25      this.kebutuhanMinimal(berat, BK, PK, TDN, Ca, P);
26      double[] pinalty = new double[stringLen+1];
27      double totPenalty = 0;
28
29      for (int i = 0; i < stringLen+1; i++) {
30         if (nutrisi[i] > this.kebutuhanMinimal[i]) {
31            pinalty[i] = 0;
32         }else{
33            pinalty[i] = Math.abs(this.kebutuhanMinimal[i] - nutrisi[i]);
34         }
35         totPenalty = totPenalty + pinalty[i];
36
37      }
38
39      this.fitnessChild[x] = 10000 / (totHarga + (totPenalty*100000));
40      System.out.println("Fitness Child: "+this.fitnessChild[x]);
41
42    }
```

Source Code 5.9 Hitung Fitness Child

Method diatas adalah method untuk menghitung nilai fitness masing-masing individu child yang dihasilkan dari proses mutasi.

5.1.10 Gabung Populasi

Berikut ini merupakan source code untuk menggabungkan populasi yang ditunjukkan oleh source code 5.10.

No	Source Code
1	public void gabungPopulasi(int popSize, int stringLen, int lamda){
2	int hasil, count;
3	hasil = popSize + (popSize * lamda);
4	this.himpunanIndividu = new double [hasil][stringLen];
5	this.himpunanFitness = new double [hasil];
6	count = 0;
7	for (int i = 0; i < hasil; i++) {
8	if (i < popSize) {
9	for (int j = 0; j < stringLen; j++) {
10	this.himpunanIndividu[i][j] = this.populasi[i][j];
11	}
12	} else {
13	for (int j = 0; j < stringLen; j++) {
14	this.himpunanIndividu[i][j] = this.child[count][j];
15	}
16	count++;
17	}
18	}
19	count = 0;
20	for (int i = 0; i < hasil; i++) {
21	if (i < popSize) {
22	this.himpunanFitness[i] = this.fitness[i];
23	} else {
24	this.himpunanFitness[i] = this.fitnessChild[count];
25	count++;
26	}
27	}
28	System.out.println("Himpunan individu");
29	for (int i = 0; i < hasil; i++) {
30	for (int j = 0; j < stringLen; j++) {
31	System.out.print(this.himpunanIndividu[i][j]+" || ");
32	}
33	System.out.println("");
34	}
35	}

Source Code 5.10 Gabung Populasi

Method diatas berfungsi untuk menggabungkan populasi antara parent dengan child.

5.1.11 Seleksi

Berikut ini merupakan source code untuk melakukan seleksi yang ditunjukkan oleh source code 5.11.

No	Source Code
1	public void seleksi(int popSize, int stringLen, int lamda) {
2	this.gabungPopulasi(popSize, stringLen, lamda);
3	int hasil = popSize + (popSize * lamda);
4	int[] tempIndex = new int [hasil];
5	int index;
6	double tempNilai = 0;
7	
8	//System.out.println("");
9	for (int i = 0; i < hasil; i++) {
10	tempIndex[i] = i;
11	//System.out.println(tempIndex[i]);
12	}
13	
14	
15	System.out.println("");
16	for (int i = 0; i < hasil; i++) {
17	System.out.println(this.himpunanFitness[i]);
18	}
19	
20	
21	boolean selesai = true;
22	while(selesai){
23	selesai = false;
24	for (int i = 1; i < hasil; i++) {
25	if (this.himpunanFitness[i-1] < this.himpunanFitness[i]) {
26	tempNilai = this.himpunanFitness[i-1];
27	index = tempIndex[i-1];
28	this.himpunanFitness[i-1] = this.himpunanFitness[i];
29	tempIndex[i-1] = tempIndex[i];
30	this.himpunanFitness[i] = tempNilai;
31	tempIndex[i] = index;
32	selesai = true;
33	}
34	}
35	}
36	/*
37	System.out.println("");
38	for (int i = 0; i < hasil; i++) {

```
39          System.out.println(tempIndex[i]);
40      }
41
42      System.out.println("");
43      for (int i = 0; i < hasil; i++) {
44          System.out.println(this.himpunanFitness[i]);
45      }
46
47      */
48      System.out.println("Fitness");
49      for (int i = 0; i < hasil; i++) {
50          System.out.println(this.himpunanFitness[i]);
51      }
52
53      this.individuBaru = new double [popSize][stringLen];
54      this.fitnessBaru = new double [popSize];
55
56      System.out.println("Populasi Baru");
57      for (int i = 0; i < popSize; i++) {
58          int pointer;
59          for (int j = 0; j < stringLen; j++) {
60              pointer = tempIndex[i];
61              this.individuBaru[i][j] = this.himpunanIndividu[pointer][j];
62              this.populasi[i][j] = this.individuBaru[i][j];
63              System.out.print(this.individuBaru[i][j]+" || ");
64          }
65          System.out.println("");
66      }
67
68      for (int i = 0; i < popSize; i++) {
69          this.fitnessBaru[i] = this.himpunanFitness[i];
70          this.fitness[i] = this.fitnessBaru[i];
71          System.out.println(this.fitnessBaru[i]);
72      }
73
74      this.fitnessTerbaik = this.fitnessBaru[0];
75      this.individuTerbaik = new double[stringLen];
76
77      //System.out.println("");
78      System.out.println(this.fitnessTerbaik);
79      //System.out.println("");
80
81      for (int i = 0; i < stringLen; i++) {
```

```
82            this.individuTerbaik[i] = this.individuBaru[0][i];
83            System.out.print(this.individuTerbaik[i]+" || ");
84        }
85        System.out.println("");
86        System.out.println("==========================");
87    }
```

Source Code 5.11 Seleksi

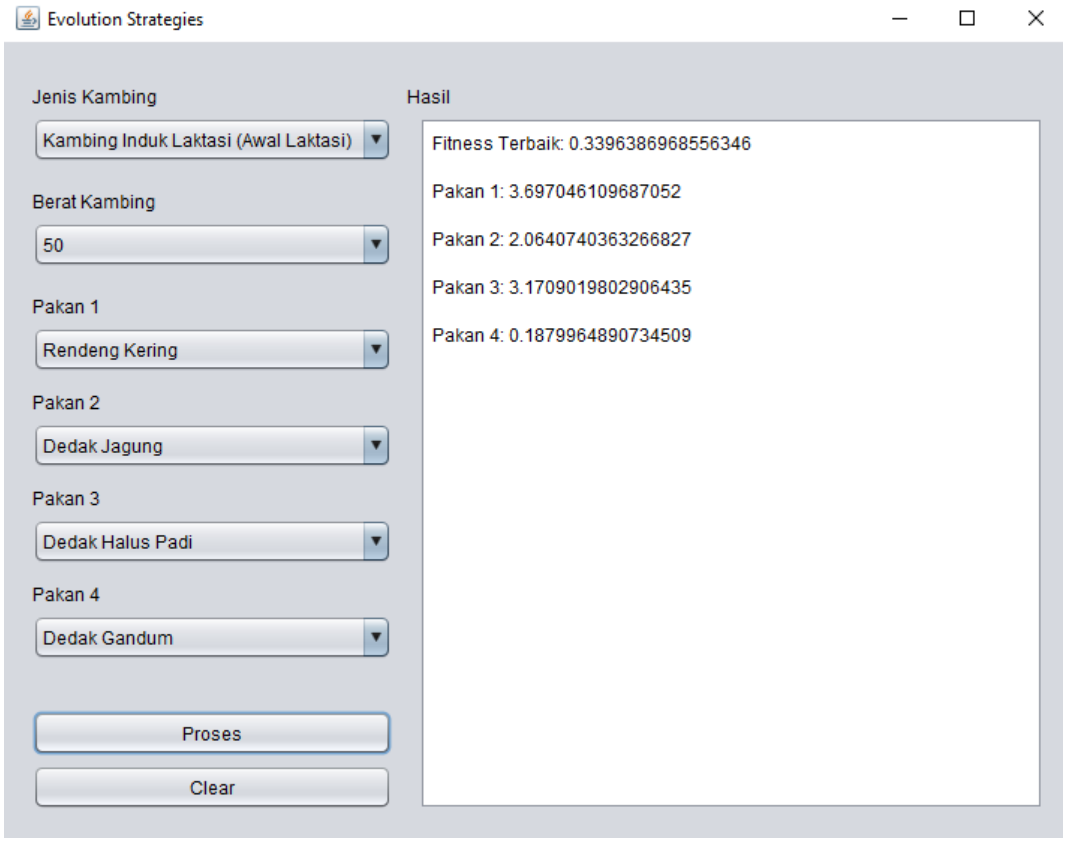
Setelah sebelumnya populasi parent dan child digabung kemudian dilakukan proses seleksi dengan cara mengambil individu dengan fitness terbaik. Individu terbaik yang diambil jumlahnya sebanyak jumlah popsize yang diset saat inisialisasi awal.

5.2 Implementasi Antarmuka

Implementasi antarmuka program komposisi pakan ternak kambing menggunakan algoritme evolution strategies nantinya akan digunakan oleh *user* untuk berinteraksi dengan sistem. Berikut hasil implementasi antarmuka pada masing-masing halaman pada aplikasi ini.

5.2.1 Implementasi Antarmuka Halaman Utama

Pada halaman ini akan ditampilkan combobox sebanyak enam buah, dua button dan sebuah textfield kosong, combo box adalah pilihan-pilihan inputan pengguna berupa jenis kambing, berat kambing dan 4 jenis pakan yang akan dioptimasi, tombol proses digunakan untuk memulai perhitungan optimasi, tombol reset digunakan untuk melakukan set ulang semua combobox dan texfield berfungsi untuk menampilkan hasil akhir perhitungan. Tampilan antarmuka halaman utama ditunjukkan oleh gambar 5.1.



Gambar 5.1 Implementasi Antarmuka Halaman Utama