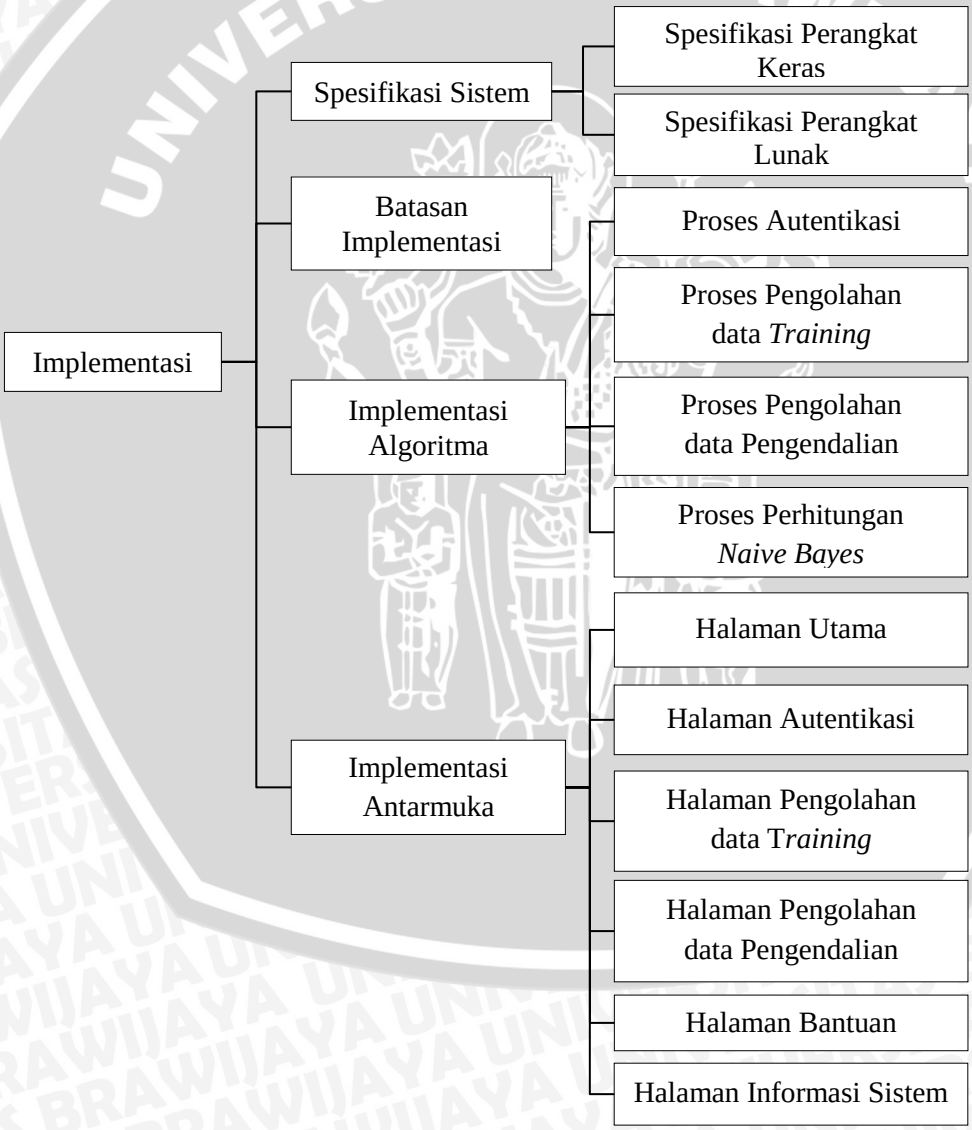
BAB V
IMPLEMENTASI

Pada bab ini membahas mengenai implementasi perangkat lunak berdasarkan hasil yang telah diperoleh dari analisis kebutuhan dan proses perancangan perangkat lunak yang telah dibuat. Pembahasan terdiri dari penjelasan tentang spesifikasi sistem, batasan-batasan dalam implementasi, implementasi pada program, implementasi antarmuka, dan implementasi metode. Gambar 5.1 merupakan pohon implementasi pada Pemodelan Sistem Pakar.



Gambar 5.1 Pohon Implementasi
Sumber: Implementasi

5.1 Spesifikasi Sistem

Hasil analisis kebutuhan dan perancangan perangkat lunak yang telah diuraikan pada bab perancangan menjadi acuan untuk melakukan implementasi menjadi sistem yang dapat berfungsi sesuai kebutuhan. Spesifikasi sistem diimplementasikan pada spesifikasi perangkat keras dan spesifikasi perangkat lunak.

5.1.1 Spesifikasi Perangkat Keras

Pengembangan Pemodelan Sistem Pakar Diagnosa Penyakit Tanaman Kopi *Arabica* menggunakan sebuah komputer dengan spesifiasi perangkat keras yang dijelaskan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
Processor	*Processor Intel® Core™ i5-3317U @1.70GHz*
Memori (RAM)	4 GB
Hardisk	500 GB
Layar LCD Monitor	14” LED

Sumber: Implementasi

5.1.2 Spesifikasi Perangkat Lunak

Pengembangan Pemodelan Sistem Pakar Diagnosa Penyakit Tanaman Kopi *Arabica* menggunakan perangkat lunak dengan spesifikasi yang dijelaskan pada Tabel 5.2.

Tabel 5.2 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
Sistem Operasi	*Microsoft Windows* 8.1 64-bit
Tools pemrograman	*NetBeans IDE 8.0*
Tools DBSM	*XAMPP v3.2.1*
Bahasa Pemrograman	Java

Sumber: Implementasi

5.2 Batasan Implementasi

Batasan implementasi adalah batasan proses yang dapat dilakukan sistem berdasarkan perancangan awal sistem. Batasan implementasi ditampilkan agar penelitian ini memiliki ruang lingkup yang jelas dalam implementasi sistem.

Beberapa batasan dalam implementasi Pemodelan Sistem Pakar Diagnosa Penyakit Tanaman Kopi *Arabica* adalah sebagai berikut:

1. Pemodelan Sistem Pakar Diagnosa Penyakit Tanaman Kopi *Arabica* dirancang dan dijalankan berbasis *Desktop.*
2. Metode inferensi yang digunakan dalam sistem ini adalah metode *Naive Bayes.*
3. Masukan yang diterima oleh sistem adalah berupa data fakta gejala penyakit yang terdapat pada tanaman kopi *arabica.*
4. Keluaran yang diterima oleh pengguna berupa jenis penyakit tanaman kopi dan pengendalian dari penyakit yang terdeteksi.
5. IDE (*Integrated Development Environment*) yang digunakan adalah *NetBeans* 8.0.

5.3 Implementasi Algoritma

Pemodelan Sistem Pakar ini mempunyai beberapa proses utama yaitu proses Autentikasi, proses pengolahan data *training,* pengolahan data pengendalian penyakit, dan proses perhitungan metode *Naive Bayes*.

5.3.1 Implementasi Algoritma Proses Autentikasi

Pada proses autentikasi ini pakar diminta untuk memasukan kata kunci pada kolom masukan yang tersedia. Setelah dimasukan maka sistem akan membadingkannya. Apabila kata kunci tersebut sama, maka akan dialihkan pada halaman pengolahan data. Apabila tidak, maka akan diberi peringatan dan akan tetap pada halaman tersebut. Untuk *source code* dapat dilihat pada Gambar 5.2.

```
1   if ("kopiarabica".equals(password.getText())){
2               halamanUtama.setVisible(false);
3               halamanLogin.setVisible(false);
4               halamanRule.setVisible(true);
5               halamanSistem.setVisible(false);
6               halamanBantuan.setVisible(false);
7           } else {
8               JOptionPane.showMessageDialog(rootPane, "Maaf, kata
9   kunci anda salah!!", "Kesalahan",JOptionPane.WARNING_MESSAGE);
10              password.setText("");
11          }
```

Gambar 5.2 Implementasi Algoritma Proses Autentikasi
Sumber: Implementasi

Penjelasan *source code* algoritma pada autentikasi tersebut adalah sebagai berikut:

1. Pada baris ke-1 sampai baris ke-6 merupakan sebuah kondisi *If* dimana jika nilai input sama dengan kondisi tersebut maka akan dialihkan ke halaman pengolahan data.
2. Pada baris ke-7 merupakan keadaan *else* apabila nilai inputan kata kunci berberda. Maka akan diberi peringatan dan akan tetap berada pada halaman tersebut.

5.3.2 Implementasi Algoritma Proses Pengolahan Data *Training*

Proses ini digunakan oleh pakar untuk melakukan memasukan data dan menghapus data *training* dalam database. Proses untuk memasukan data *training*, pakar akan diminta untuk milih gejala-gejala yang terdapat pada tanaman beserta penyakitnya. Setelah itu klik simpan untuk menyimpan dalam *database*. Untuk menghapus data *training*, pilih salah satu *row* dalam tabel dan klik hapus untuk menghapus dalam database. Untuk *source code* proses pengolahan data *training* dapat dilihat pada Gambar 5.3.

```
1   public void setDataLatih(){
2           String daun = (String)kondisiDaun2.getSelectedItem();
3           String batang = (String)kondisiBatang2.getSelectedItem();
4           String buah = (String)kondisiBuah2.getSelectedItem();
5           String hamaserangan = (String)hama2.getSelectedItem();
6           try{
7               Statement stat = db.konek.createStatement();
8               String query = "INSERT INTO data_latih"
9               + "(kondisi_daun, kondisi_batang, kondisi_buah, hama) "
10              + "VALUES "
11              +
12  "('"+daun+"','"+batang+"','"+buah+"','"+hamaserangan+"')";
13              stat.executeUpdate(query);
14              JOptionPane.showMessageDialog(rootPane, "Data Latih Telah
15  disimpan", "Berhasil",JOptionPane.INFORMATION_MESSAGE);
16              refreshTabelKendali();
17          } catch(SQLException e){
18              e.getMessage();
19          }
20      }
21
22
23  public void hapusDataLatih(int row){
24          int count = 0;
25
26          try{
27              PreparedStatement ps =
28  db.getConnection().prepareStatement("SELECT * "
29                      + "FROM data_latih");
```

```
30              ResultSet result = ps.executeQuery();
31
32              while(count<=row){
33                  count++;
34                  result.next();
35              }
36              String hapusDaun = result.getString("kondisi_daun");
37              String hapusBatang = result.getString("kondisi_batang");
38              String hapusBuah = result.getString("kondisi_buah");
39              String hapusHama = result.getString("hama");
40
41              Statement stat = db.konek.createStatement();
42              String query = "DELETE FROM data_latih "
43                      + "WHERE kondisi_daun='"+hapusDaun+"' AND "
44                      + "kondisi_batang='"+hapusBatang+"' AND "
45                      + "kondisi_buah='"+hapusBuah+"' AND "
46                      + "hama='"+hapusHama+"'";
47              int status = stat.executeUpdate(query);
48              if(status==1){
49                  JOptionPane.showMessageDialog(rootPane, "Data Latih
50  Telah dihapus", "Berhasil",JOptionPane.INFORMATION_MESSAGE);
51              }
52          } catch(SQLException e){
53
54          }
55      }
```

Gambar 5.3 Implementasi Algoritma Proses Pengolahan Data *Training*
Sumber: Implementasi

Penjelasan *source code* algoritma proses pada pengolahan data *training* tersebut adalah sebagai berikut:

1. Baris ke-1 sampai baris ke-20 merupakan proses untuk menyimpan data *training*. Prosesnya, sistem akan mengambil nilai yang telah dimasukan oleh pakar. Setelah itu akan dimasukan kedalam statemen *database*. Apabila berhasil, akan muncul informasi data telah disimpan. Apabila gagal, sistem akan menampilkan letak error yang terjadi.
2. Baris ke-23 sampai baris ke-55 merupakan proses untuk menghapus data *training* yang terdapat dalam database. Prosesnya, sistem akan mencari baris yang telah dipilih pakar untuk dihapus. Setelah ditemukan, sistem akan mengambil nilai tersebut dan memasukannya dalam statemen database. Apabila berhasil, akan muncul informasi data telah dihapus.

5.3.3 Implementasi Algoritma Proses Pengolahan Data Pengendalian

Implementasi algoritma pengolahan data pengendalian mempunyai proses yang sama denga implementasi algoritma pengolahan data *training*. Perbedaannya terletak pada tipe masukan dan statemen untuk menyimpan pada database. Untuk *source code* proses pengolahan data penanganan dapat dilihat pada Gambar 5.4.

```
1   public void setDataKendali(){
2           String penyakit = (String)hama3.getSelectedItem();
3           String kendali = (String)jTextArea1.getText();
4           try{
5                   Statement stat = db.konek.createStatement();
6                   String query = "INSERT INTO data_kendali"
7                   + "(nama, kendali) "
8                   + "VALUES "
9                   + "('"+penyakit+"','"+kendali+"')";
10                  stat.executeUpdate(query);
11                  JOptionPane.showMessageDialog(rootPane, "Data Penanganan
12  Penyakit Telah disimpan",
13  "Berhasil",JOptionPane.INFORMATION_MESSAGE);
14                  refreshTabelKendali();
15          } catch(SQLException e){
16                  e.getMessage();
17          }
18      }
19
20
21  public void hapusDataKendali(int row){
22          int count = 0;
23
24          try{
25              PreparedStatement ps =
26  db.getConnection().prepareStatement("SELECT * "
27                      + "FROM data_Kendali");
28              ResultSet result = ps.executeQuery();
29
30              while(count<=row){
31                  count++;
32                  result.next();
33              }
34              String hapusPenyakit = result.getString("nama");
35              String hapusPenaganan = result.getString("kendali");
36
37              Statement stat = db.konek.createStatement();
38              String query = "DELETE FROM data_kendali "
39                      + "WHERE nama='"+hapusPenyakit+"' AND "
40                      + "kendali='"+hapusPenaganan+"'";
41              int status = stat.executeUpdate(query);
42              if(status==1){
43                  JOptionPane.showMessageDialog(rootPane, "Data Kendali
44  Telah dihapus", "Berhasil",JOptionPane.INFORMATION_MESSAGE);
45              }
46          } catch(SQLException e){
47
48          }
49      }
```

Gambar 5.4 Implementasi Algoritma Proses Pengolahan Data Pengendalian
Sumber: Implementasi

Penjelasan *source code* algoritma proses pada pengolahan data Pengendalian tersebut adalah sebagai berikut:

1. Baris ke-1 sampai baris ke-18 merupakan proses untuk menyimpan data Pengendalian. Prosesnya, sistem akan mengambil nilai yang telah dimasukan oleh pakar. Setelah itu akan dimasukan kedalam statemen database. Apabila berhasil, akan muncul informasi data telah disimpan. Apabila gagal, sistem akan menampilkan letak error yang terjadi.
2. Baris ke-21 sampai baris ke-49 merupakan proses untuk menghapus data penanganan yang terdapat dalam database. Prosesnya, sistem akan mencari baris yang telah dipilih pakar untuk dihapus. Setelah ditemukan, sistem akan mengambil nilai tersebut dan memasukannya dalam statemen database. Apabila berhasil, akan muncul informasi data telah dihapus.

5.3.4 Implementasi Algoritma Proses Perhitungan *Naive Bayes*

Algoritma ini digunakan oleh sistem untuk melakukan inferensi ataupun diagnosa penyakit pada tanaman kopi. Prosesnya sistem akan menghitung nilai prior, likelihood, setelah itu nilai posterior dengan mengalikan nilai prior dan likelihood. Untuk *source code* proses perhitungan *Naive Bayes* dapat dilihat pada Gambar 5.5.

```
1   private double getJmlJenisHama(String hama){
2           double jml = 0;
3           try{
4               Statement stat = db.konek.createStatement();
5               String query = "SELECT COUNT(*) FROM data_latih "
6                   + "WHERE hama='"+hama+"'";
7               ResultSet result = stat.executeQuery(query);
8               while(result.next()){
9                   jml = result.getInt(1);
10              }
11          } catch(SQLException e){
12              System.err.println(e.getSQLState());
13          }
14          return jml;
15      }
16
17  private double getTotData(){
18          double tot = 0;
19          try{
20              Statement stat = db.konek.createStatement();
21              String query = "SELECT COUNT(*) FROM data_latih";
22              ResultSet result = stat.executeQuery(query);
23              while(result.next()){
24                  tot = result.getInt(1);
25              }
```

```
26          } catch(SQLException e){
27              System.err.println(e.getSQLState());
28          }
29          return tot;
30      }
31
32  private double getGejala(String kolom, String gejala, String hama){
33          double nilai = 0;
34          try{
35              Statement stat = db.konek.createStatement();
36              String query = "SELECT COUNT(*) FROM data_latih "
37                  + "WHERE hama='"+hama+"' AND "+kolom+"='"+gejala+"'";
38              ResultSet result = stat.executeQuery(query);
39              while(result.next()){
40                  nilai = result.getInt(1);
41              }
42          }catch(SQLException e){
43              System.err.println(e.getSQLState());
44          }
45          return nilai;
46      }
47
48  public double getPrior(String hama){
49          double prior = getJmlJenisHama(hama)/getTotData();
50          return prior;
51      }
52
53      public double getLikelihood(String kolom, String gejala, String
54  hama){
55          double likelihood = getGejala(kolom, gejala,
56  hama)/getJmlJenisHama(hama);
57          return likelihood;
58      }
59
50  public void getPerhitungan(String hama, List<DataGejala> list){
51          double priorHama = getPrior(hama);
52          double likelihood[] = new double[list.size()];
53          for (int i = 0; i < list.size(); i++) {
54              DataGejala dataGejala = list.get(i);
55              likelihood[i] = getLikelihood(dataGejala.getbagian(),
56  dataGejala.getGejala(), hama);
57          }
58          double finalLikelihood = likelihood[0];
59          for (int i = 1; i < likelihood.length; i++) {
50              finalLikelihood = finalLikelihood*likelihood[i];
51          }
52          double postHama = priorHama*finalLikelihood;
53          setHamaTertinggi(hama, postHama);
54      }
55
56  public void getListPerhitungan(List<DataGejala> list){
57          namaHama = null;
58          nilaiPosterior = 0;
59
60          getPerhitungan("Antraknosa", list);
61          getPerhitungan("Karat Daun", list);
62          getPerhitungan("Bercak Daun", list);
63          getPerhitungan("Jamur Upas", list);
64          getPerhitungan("Nematoda", list);
65          getPerhitungan("Normal", list);
66      }
67
68  public void setHamaTertinggi(String hama, double nilai){
69          if (nilai>nilaiPosterior){
```

```
70              this.namaHama = hama;
71              this.nilaiPosterior = nilai;
72          }
73      }
74
75      public String getPengendalian(String hama){
76          String kendali = "";
77          try{
78              Statement stat = db.konek.createStatement();
79              String query = "SELECT * FROM data_kendali "
80                  + "WHERE nama='"+hama+"'";
81              ResultSet result = stat.executeQuery(query);
82              while(result.next()){
83                  kendali = result.getString(3);
84              }
85          }catch(SQLException e){
86              System.err.println(e.getSQLState());
87          }
88          return kendali;
89      }
```

Gambar 5.5 Implementasi Algoritma Proses Perhitungan *Naive Bayes*
Sumber: Implementasi

Penjelasan *source code* algoritma proses perhitungan tersebut adalah sebagai berikut:

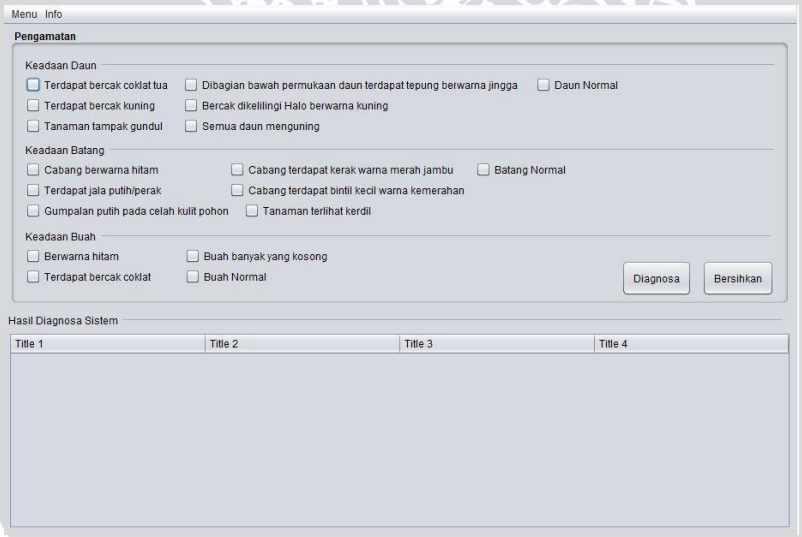
1. Pada baris ke-1 sampai baris ke-15 merupakan sebuah proses perhitungan jumlah penyakit berdasarkan jenisnya.
2. Pada baris ke-17 sampai baris ke-30 merupakan proses untuk menghitung total data *training* pada database tersebut.
3. Baris ke-32 sampai baris ke-46 merupakan proses menghitung jumlah gejala yang terdapat pada database berdasarka kolom database dan penyakitnya.
4. Baris ke-48 sampai baris ke-51 merupakan proses perhitungan nilai prior berdasarka penyakit tertentu.
5. Baris ke-53 sampai baris ke-58 merupakan proses perhitungan nilai likelihood berdasarkan gejala dan penyakit tertentu.
6. Baris ke-51 sampai baris ke-62 merupakan proses perhitungan posterior berdasarkan gejala yang tampak dan penyakitnya.
7. Baris ke-64 sampai baris ke-74 merupakan sebuah proses memasukan diagnosa penyakit ke dalam array list yang nantinya akan ditampilkan pada antaramuka diagnosa.
8. Baris ke-76 sampai baris ke-89 merupakan proses pengambilan data pengendalian penyakit berdasarkan penyakit yang menyerang.

5.4 Implementasi Antarmuka

Implementasi antarmuka Pemodelan Sistem Pakar Diagnosa Penyakit Tanaman Kopi *Arabica* ini terdiri dari halaman utama, halaman autentikasi, halaman pengolahan data *training*, halaman pengolahan data pengendalian, halaman bantuan, dan halaman informasi sistem.

5.4.1 Halaman Utama

Halaman utama adalah halaman yang digunakan pengguna untuk memasukan data gejala penyakit agar sistem dapat melakukan diagnosa dengan metode *Naive Bayes*. Data gejala yang dibutuhkan diantaranya gejala pada daun, gejala pada batang, dan gejala pada buah. Setelah itu pengguna dapat menekan tombol diagnosa untuk melihat hasil diagnosa penyakit yang telah dilakukan oleh sistem. Tampilan halaman utama dapat dilihat pada Gambar 5.6.

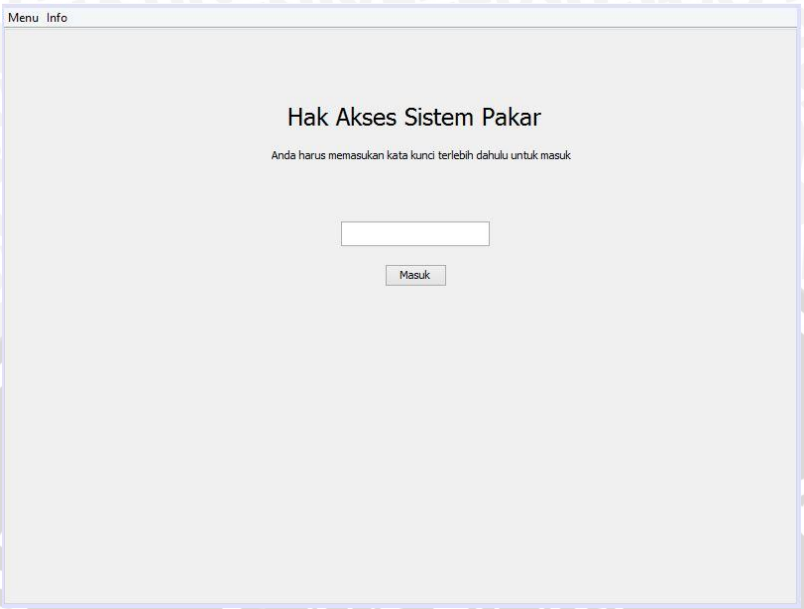


Gambar 5.6 Implementasi Halaman Utama
Sumber: Implementasi

5.4.2 Halaman Autentikasi

Halaman autentikasi merupakan halaman yang akan mengidentifikasi pengguna sistem ini. Dalam hal ini hanya pengguna yang *authorized* yang dapat mengakses sistem pengolahan data *training* maupun data pengendalian penyakit. Pengguna dapat memulai dengan memasukan kata kunci pada kolom isian yang

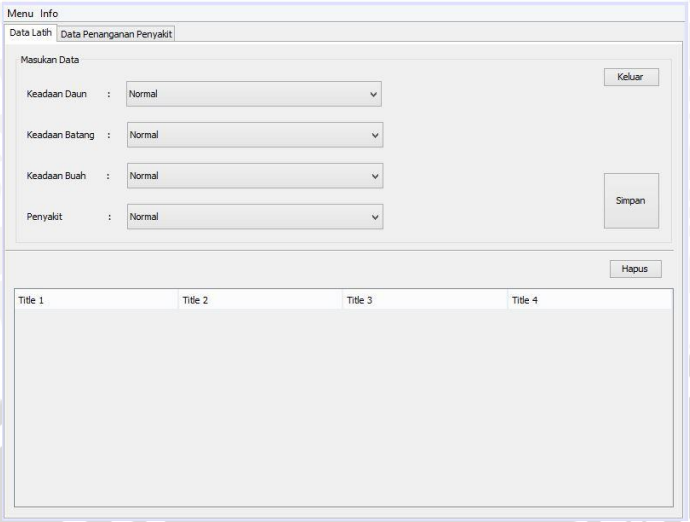
tersedia. Klik tombol masuk untuk mengidentifikasi apakah user ini *authorized* atau tidak. Tampilan halaman autentikasi dapat dilihat pada Gambar 5.7.



Gambar 5.7 Implementasi Halaman Autentikasi
Sumber: Implementasi

5.4.3 Halaman Pengolahan Data *Training*

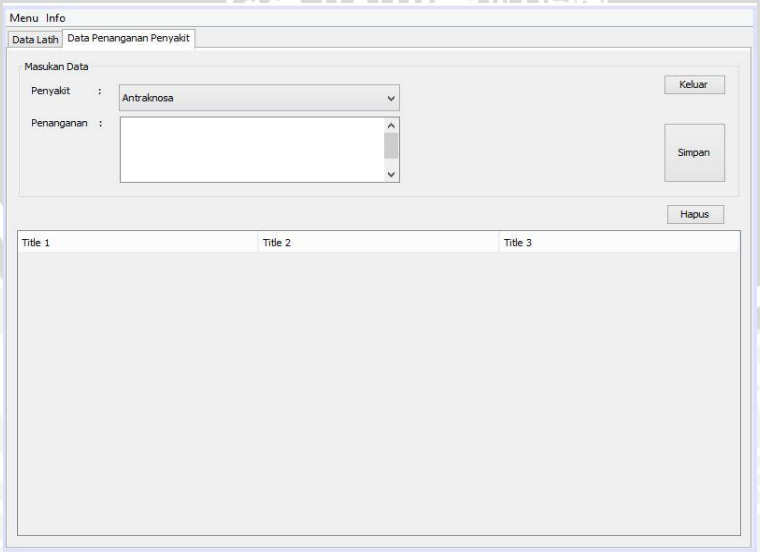
Halaman pengolahan data *training* merupakan halaman yang digunakan oleh pakar untuk memasukan data *training* ataupun melakukan penghapusan data. Untuk menambah data, pakar akan diminta untuk memilih gejala-gejala yang tersedia beserta penyakit berdasarkan gejala yang timbul karena penyakit tersebut. Setelah itu klik tombol simpan untuk menyimpannya dalam database. Sedangkan untuk menghapus data pengguna cukup memilih baris data pada tabel lalu tekan tombol hapus. Tampilan halaman pengolahan data *training* dapat dilihat pada Gambar 5.8.



Gambar 5.8 Implementasi Halaman Pengolahan Data *Training*
Sumber: Implementasi

5.4.4 Halaman Pengolahan Data Pengendalian

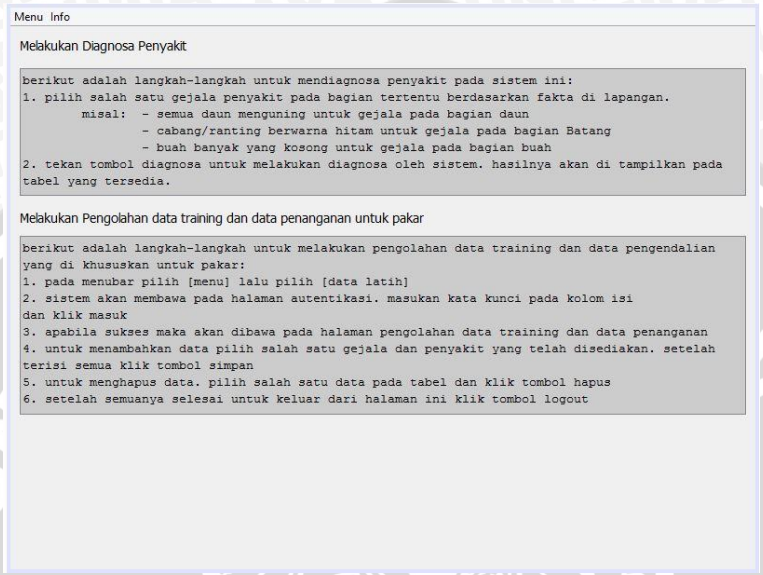
Halaman pengolahan data pengendalian merupakan halaman yang digunakan oleh pakar untuk memasukan data pengendalian ataupun melakukan penghapusan data. Untuk menambah data, pakar akan diminta untuk memilih penyakit yang tersedia dan bagaimana cara mengendalikan penyakit tersebut. Setelah itu klik tombol simpan untuk menyimpannya dalam database. Sedangkan untuk menghapus data pengguna cukup memilih baris data pada tabel lalu tekan tombol hapus. Tampilan halaman pengolahan data pengendalian dapat dilihat pada Gambar 5.9.



Gambar 5.9 Implementasi Halaman Pengolahan Data Pengendalian
Sumber: Implementasi

5.4.5 Halaman Bantuan

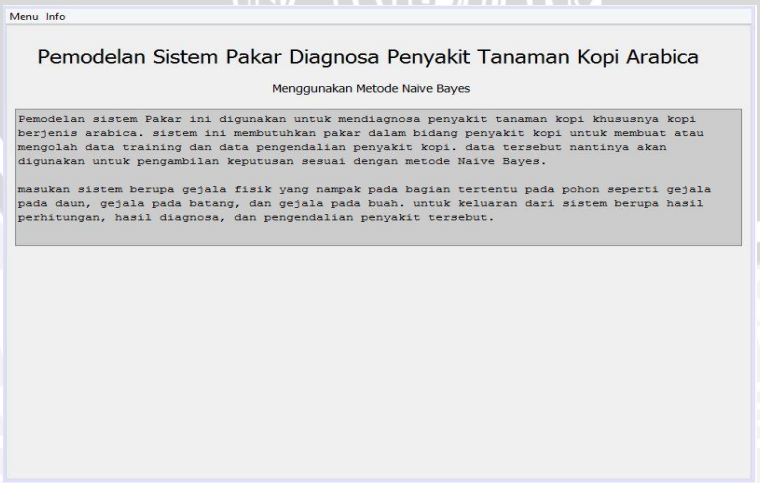
Halaman bantuan merupakan halaman tentang informasi penggunaan aplikasi pemodelan sistem pakar diagnosa penyakit kopi *arabica* menggunakan metode *Naive Bayes.* Tampilan halaman bantuan dapat dilihat pada Gambar 5.10.



Gambar 5.10 Implementasi Halaman Bantuan
Sumber: Implementasi

5.4.6 Halaman Informasi Sistem

Halaman informasi sistem merupakan halaman yang menampilkan informasi yang berkaitan dengan sistem yang dibuat. Seperti halnya metode inferensi, masukan yang dibutuhkan, dan keluaran dari sistem tersebut. Tampilan halaman informasi sistem dapat dilihat pada Gambar 5.11.



Gambar 5.11 Implementasi Halaman Informasi Sistem
Sumber: Implementasi