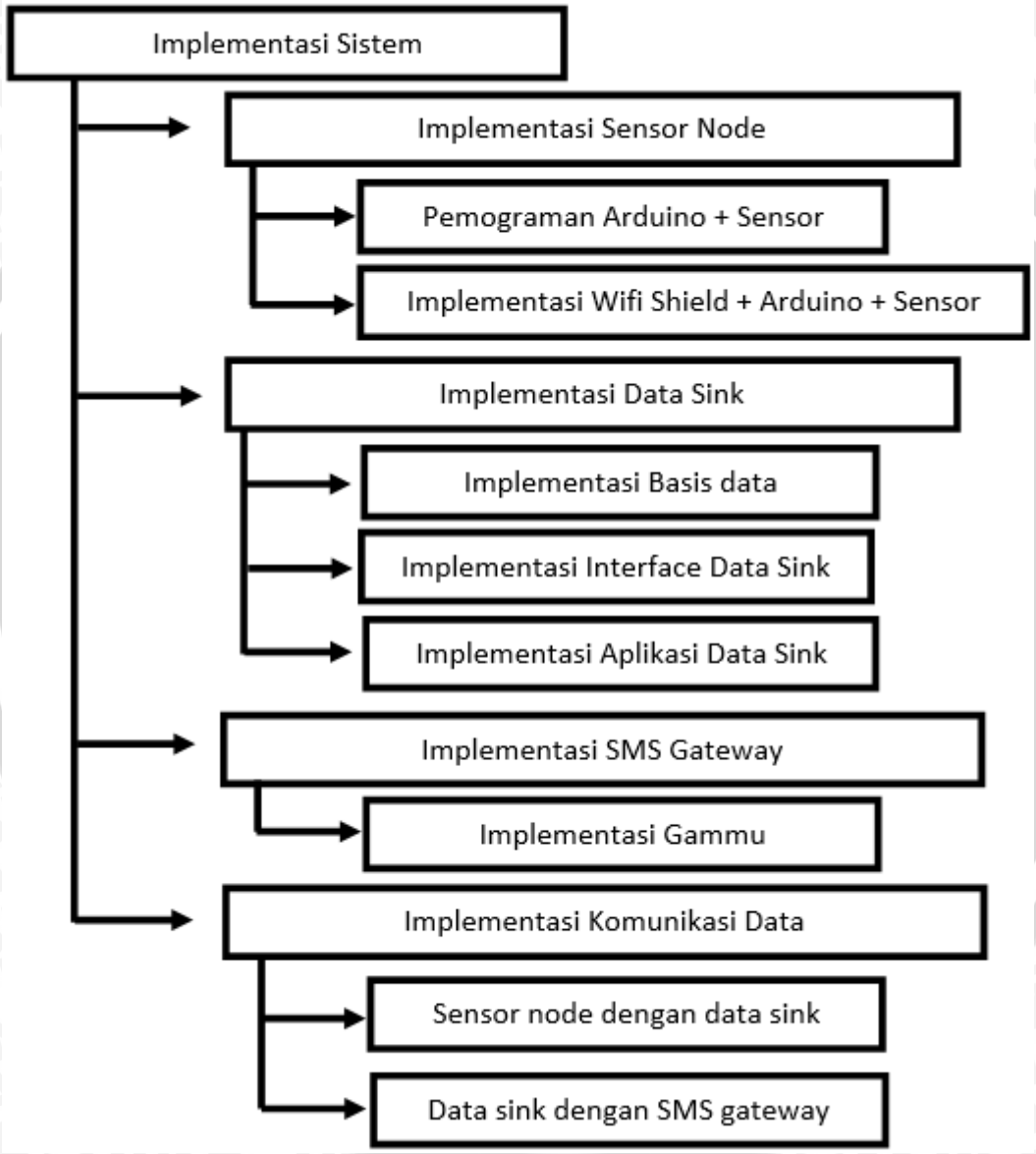
BAB V
IMPLEMENTASI SISTEM

Pada Bab ini dijelaskan mengenai implementasi sistem secara keseluruhan berdasarkan dari hasil analisis dan perancangan sistem yang telah diperoleh. Berikut pohon implementasi sistem secara keseluruhan :



Gambar 5.1 Pohon Implementasi
Sumber : Implementasi Sistem

1.1 Implementasi Sensor Node

Pada implementasi sensor node ini menggunakan bahasa pemograman C untuk modul arduino dan PIR sensor serta Arduino WifiShield. Proses kompilasi

program mengggunakan arduino IDE. Topologi yang digunakan pada sensor node menggunakan topologi *Single Hop*.

1.1.1 Pemograman Sensor Node (Arduino +Wifi Shield + Sensor)

Berdasarkan subbab 4.2.2 perancangan algoritma sensor node, sensor node diimplementasikan menggunakan bahasa pemograman arduino. Aplikasi yang digunakan untuk melakukan pemograman arduino ialah Arduino IDE versi 1.0.4. Listing program pada sensor node ditunjukkan pada gambar 5.2 :

```
1. #include <SPI.h>
2. #include <WiFi.h>
3. #include <Time.h>

4. char ssid[] = "tes";
5. int keyIndex = 0;
6. int inputPin = 2;
7. int pirState = LOW;
8. int val = 0;
9. int ledPin = 13;
10.int sensorReading;
11.int status = WL_IDLE_STATUS;

12.WiFiServer server(80);

13. void setup() {
14.setTime(1359871504);
15.pinMode(ledPin, OUTPUT);
16.pinMode(inputPin, INPUT);
17.Serial.begin(9600);
18.while (!Serial) {
19.    ;  }


20.  if (WiFi.status() == WL_NO_SHIELD) {
21.    Serial.println("WiFi shield not present");
22.    while(true);
23.  }

        // lakukan pengkoneksian sampai wifi terkoneksi
24.   while ( status != WL_CONNECTED) {
25.Serial.print("Attempting to connect to SSID: ");
26.   Serial.println(ssid);
    // Koneksi ke WPA/WPA2 network.
27.   status = WiFi.begin(ssid);

          // tunggu 10 detik untuk koneksi
28.   delay(10000);
29.}
```

```
30.server.begin();
 // cetak status wifi setelah terkoneksi
31.printWifiStatus();
32.    }


33. void loop() {
34.  // listen untuk client yang ada
35.WiFiClient client = server.available();
36.      if (client) {
37.          Serial.println("new client");
38.boolean currentLineIsBlank = true;
39.          while (client.connected()) {
40.if (client.available()) {
41.char c = client.read();
42.Serial.write(c);
            // if you've gotten to the end of the line
(received a newline
            // character) and the line is blank, the http
request has ended,
            // so you can send a reply
43.        if (c == '\n' && currentLineIsBlank) {
44.          client.println("HTTP/1.1 200 OK");
45.          client.println("Content-Type: text/html");
46.          client.println("Connnection: close");
47.          client.println();
48.          client.println("<!DOCTYPE HTML>");
49.          client.println("<html>");
         //mengganti refresh per detik..
50.           client.println("<meta http-equiv=\"refresh\"
content=\"2\">");
51.          client.println("<head>");
52.          client.println("<script>");
53.          client.println("function showUser()");
54.          client.println("{");
55.          client.println("var
x=document.getElementById(\"myHeader\");");
56.          client.println("var
y=document.getElementById(\"myHeader2\");");
57.client.println("strX=x.innerHTML");
58.client.println("strY=y.innerHTML");
59.client.println("if (strX==\"\")");
60.       client.println("{");
61.
client.println("document.getElementById(\"txtHint\").inner
HTML=\"\";");
62.          client.println("return;");
63.          client.println("} ");


64.          client.println("if (window.XMLHttpRequest)");
65.          client.println("{");
```

```
66.         client.println("xmlhttp=new XMLHttpRequest();");
67.           client.println("}");
68.           client.println("else");
69.           client.println("{");
70.                                client.println("xmlhttp=new
ActiveXObject(\"Microsoft.XMLHTTP\");");
    71.       client.println("}");

     72.client.println("xmlhttp.open(\"GET\",\"http://loc
alhost/skripsi/masuk.php?q=\"+strX+\"&w=\"+strY,true);");
73.           client.println("xmlhttp.send();");
74.           client.println("}");
75.           client.println("</script>");
76.           client.println("</head>");
77.          client.println("<body onload=\"showUser()\">");
78.           client.println("<h1 id=\"myHeader\" > ");
79.          val = digitalRead(inputPin);  // membaca nilai
input
80.           if (val == HIGH){
81.               delay(150);
82.               if(pirState == LOW) {
83.                 digitalWrite(ledPin, HIGH);
84.               Serial.println("Rumah dalam bahayaaa!!!");
85.                 sensorReading = 1;
86.                 client.print(sensorReading);
87.                 pirState = HIGH;
88.               }
89.            }
90.            else {
91.               if (pirState == HIGH){
92.                   digitalWrite(ledPin, LOW); // padamkan
LED
93.                    Serial.println("Rumaah Amaaan !");
94.                    sensorReading = 0;
95.                  client.print(sensorReading);
                    // hanya memunculkan pergantian output,
bukan status
96.                    pirState = LOW;
97.               }
98.             }


99.         client.println("</h1>");
100.
101.        client.println("<h2 id=\"myHeader2\">");
102.        client.println(digitalClockDisplay());
103.
104.        client.println("<h2>");

105.        client.println("</body>");
106.        client.println("</html>");
107.        break;
```

```
108.}
109.if (c == '\n') {
          // you're starting a new line
110.        currentLineIsBlank = true;
111.}
112. else if (c != '\r') {
          // you've gotten a character on the current line
113.        currentLineIsBlank = false;
114.        }
115.      }
116.    }
    // give the web browser time to receive the data
117.     delay(1);
118.      // close the connection:
119.      client.stop();
120.      Serial.println("client disonnected");
121.  }
122. }


123. void printWifiStatus() {

124.   Serial.print("SSID: ");
125.   Serial.println(WiFi.SSID());


126.   IPAddress ip = WiFi.localIP();
127.   Serial.print("IP Address: ");
128.   Serial.println(ip);

129.  long rssi = WiFi.RSSI();
130.  Serial.print("signal strength (RSSI):");
131.  Serial.print(rssi);
132.  Serial.println(" dBm");
133.  }


134.  String digitalClockDisplay(){
  // digital clock display of the time
135.  String text="";
136.  text+=printDigits(year());
137.  text+=("-");
138.  text+=printDigits(month());
139.  text+=("-");
140.  text+=printDigits(day());
141.  text+=(" ");

142.  text+=printDigits(hour());
143.  text+=(":");
144.  text+=printDigits(minute());
145.  text+=(":");
146.  text+=printDigits(second());
```

```
147.  text+=("");
148.  return text;

149.  }

150.  String printDigits(int digits){
  // utility function for digital clock display: prints
preceding colon and leading 0
151. String text="";
152.  if(digits < 10)
153.  text+="0";
154.  text+=digits;

155.  return text;
156.  }
```

Gambar 5.2 Tampilan *Script* Pemograman Arduino
Sumber : Implementasi Sistem

Penjelasan program arduino berdasarkan gambar 5.2 adalah sebagai berikut :

1. Baris ke- 4 : Proses inisialisasi wifi shield akan menggunakan SSID yang bernama “tes”.
2. Baris ke- 5 hingga baris ke- 11 : Proses inisialisasi variabel yang akan digunakan yang meliputi variabel dari pembacaan sensor, input digital, dan status wifi shield.
3. Baris ke-12 : Inisialisasi port wifi shield disetting pada port 80. Port 80 merupakan port yang digunakan pada protokol http.
4. Baris ke- 13 hingga baris ke- 19 : Proses setup program, dimana wifi shield disetting pada baudrate 9600, dan set waktu secara random. Selain itu juga melakukan setting pinMode.
5. Baris ke-20 hingga baris ke- 23 : Proses koneksi wifi shield ke SSID, jika tidak terkoneksi maka akan terdapat status `"WiFi shield not present"`.
6. Baris ke-24 hingga baris ke- 32 : Proses koneksi wifi shield ke SSID, jika sudah terkoneksi akan mencetak status wifi shield.
7. Baris ke- 35 : Proses listening jika ad client yang mengakses wifi shield.
8. Baris ke- 36 hingga baris ke- 42 : Jika client sudah terkoneksi dengan wifi shield dan melakukan request, maka akan ada proses reply dari wifi shield.

9. Baris ke- 42 hingga baris ke- 71 : Source code HTML yang nantinya akan di muat pada browser client.
10. Baris ke- 72 : Proses pengambilan data oleh cllient dari wifi shield, yang kemudian data akan dimasukkan ke dalam folder skripsi dalam localhost. Source code "masuk.php" digunakan sebagai *trigger* untuk mengupdate basis data yang datanya diperoleh dari wifi shield.
11. Baris ke- 73 hingga baris ke- 98 : Proses pembacaan dari sensor, jika status sensor HIGH, maka program akan mensetting ledpin menjadi low, dan status rumah aman. Sedangkan status sensor LOW, maka program akan mensetting ledpin menjadi high, dan status rumah dalam bahaya.
12. Baris ke- 119 hingga baris ke- 122 : Client mengakhiri koneksi.
13. Baris ke- 123 hingga baris ke- 133 : Mencetak wifi status.
14. Baris ke- 133 hingga baris ke- 156 : Mendisplay jam digital .

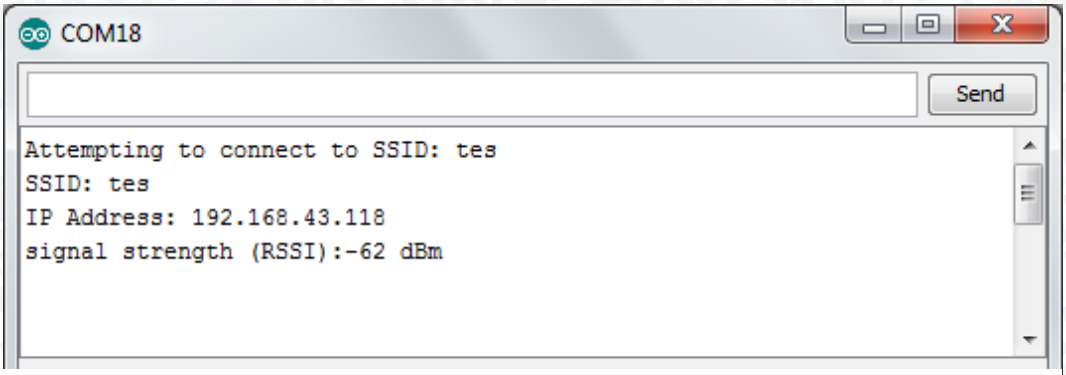
1.1.2 Implementasi Wifi Shield + Arduino + Sensor (Sensor Node)

Implementasi Arduino, Wifi Shield dan sensor dilakukan setelah kompilasi program arduino + sensor. Arduino wifi shield dapat bekerja dengan baik jika setting pada acces point bersifat open dan tidak menggunakan passsword.

Gambar 5.3 Arduino + Arduino Wifi Shield
Sumber : Implementasi Sistem

Wifi shield ini terhubung dengan SSID yang diberi nama "tes". Gambar 5.3 menunjukkan bahwa Ardunio wifi shield terpasang dengan tepat pada Arduino. Arduino Wifi Shield dipasangkan tepat pada pin Arduino Uno, dan untuk pin 2 digunakan sebagai input dari Sensor PIR, pin GND sebagai ground dari Sensor PIR dan 5V sebagai input daya untuk Sensor PIR.

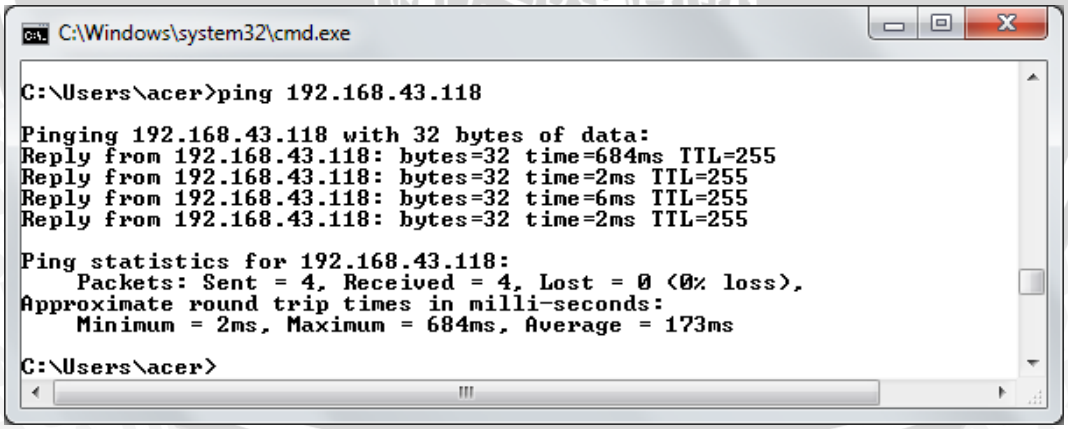
Setelah dilakukan kompilasi, maka sesnor node diimplementasikan dengan SSID yang bernama “tes”. Hasil implementasinya ditunjukkan pada gambar 5.4 :



Gambar 5.4 Implementasi Sensor Node dengan SSID
Sumber : Implementasi Sistem

Gambar 5.4 adalah serial monitor dari Arduino IDE versi 1.0.4. Berdasarkan gambar tersebut, sensor node telah terhubung dengan SSID yang bernama “tes”. Sensor node dikompilasi dengan Arduino IDE versi 1.0.4 dan terletak pada port COM 18. Baudrate yang digunakan sebesar 9600. Alamat IP yang diberikan dari *Domain Name Server* (DNS) adalah 192.168.43.118 dengan kekuatan sinyal 62 dBm.

Setelah dilakukan implementasi pada SSID tes, selanjutnya dilakukan *ping* ke alamat IP sensor node. Hasil implementasinya ditunjukkan pada gambar 5.5 :



Gambar 5.5 Implementasi Sensor Node
Sumber : Implementasi Sistem

Gambar 5.5 menunjukkan bahwa proses ping ke sensor node berhasil. Hal ini ditunjukkan dengan adanya *Reply* dari sensor node sebesar 32 *byte* dengan waktu minimum 2 *microsecond*, waktu maksimum 684 *microsecond* dan waktu rata-rata

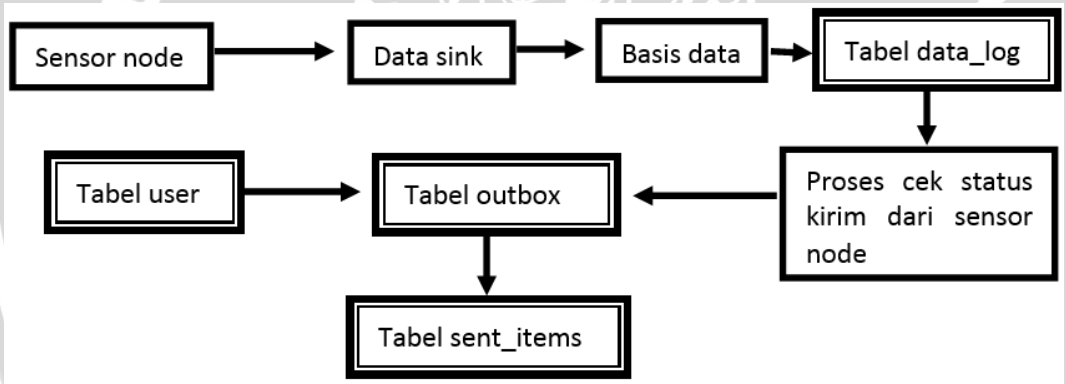
173 *microsecond*. *Time To Life* (TTL) sebesar 225. Paket data yang dikirimkan 4, data yang diterima 4. Paket data yang *loss* tidak ada.

1.2 Implementasi Data Sink

Implementasi data sink meliputi basis data, *Interface* dan aplikasi data sink. Basis data menggunakan MySql sedangkan untuk *Interface* dan aplikasi data sink menggunakan bahasa pemograman PHP dan Java*Script*.

1.2.1 Implementasi Basis data

Berdasarkan ERD subbab 4.3.1 perancangan basis data, basis data diimplementasikan menggunakan bahasa pemrograman MySql. Selain itu juga menggunakan web server XAMPP. Alur basis data ditunjukkan dalam gambar 5.6 :



Gambar 5.6 Diagram Blok Alur Basis Data
Sumber : Implementasi Sistem

Alur data pada basis data berdasarkan gambar 5.6 adalah sebagai berikut : Data yang dikirimkan dari sensor node akan masuk ke dalam tabel data log. Kemudian data dari tabel data_log akan diproses apakah data yang dikirimkan dari sensor node berstatus “bahaya” atau “aman”. Jika data berstatus bahaya, maka aplikasi akan mengambil data dari tabel data log , kemudian data akan dimasukkan ke tabel “outbox” yang sebelumnya tabel outbox juga mengambil data dari “user”. Implementasi basis data menggunakan bahasa pemrograman MySql ditunjukkan pada tabel 5.1 sampai dengan tabel 5.13.

Tabel 5.1 Daemon

NAMA	Daemon
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan koneksi untuk memulai Gammu.
DDL	`CREATE TABLE IF NOT EXISTS `daemons` ( `Start` text NOT NULL, `Info` text NOT NULL ) ENGINE=MyISAM DEFAULT CHARSET=utf8;`

Sumber : Implementasi Sistem

Tabel daemon ditunjukkan pada Tabel 5.1. Tabel ini sudah menjadi satu pada saat Gammu *library* ini diinstal pada windows. Di dalam tabel ini menampilkan info dari Gammu *library* dan juga terdapat start untuk memulai *library* Gammu.

Tabel 5.2 Gammu

NAMA	Gammu
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan informasi versi Gammu yang digunakan.
DDL	`CREATE TABLE IF NOT EXISTS `Gammu` ( `Version` int(11) NOT NULL DEFAULT '0' ) ENGINE=MyISAM DEFAULT CHARSET=utf8; -- Dumping data for table `Gammu` INSERT INTO `Gammu` (`Version`) VALUES (10);`

Sumber : Implementasi Sistem

Tabel Gammu ditunjukkan pada Tabel 5.2. Tabel ini hanya menampilkan informasi dari versi Gammu yang digunakan pada sistem. Versi Gammu yang digunakan ialah versi 10.

Tabel 5.3 Inbox

NAMA	Inbox
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan pesan yang masuk.
DDL	`-- Table structure for table `inbox` CREATE TABLE IF NOT EXISTS `inbox` ( `UpdatedInDB` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP, `ReceivingDateTime` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00', `Text` text NOT NULL, `SenderNumber` varchar(20) NOT NULL DEFAULT '',`

	`Coding` enum('Default_No_Compression','Unicode_No_Compression','8bit','Default_Compression','Unicode_Compression') NOT NULL DEFAULT 'Default_No_Compression', `UDH` text NOT NULL, `SMSCNumber` varchar(20) NOT NULL DEFAULT '', `Class` int(11) NOT NULL DEFAULT '-1', `TextDecoded` varchar(160) NOT NULL DEFAULT '', `ID` int(10) unsigned NOT NULL AUTO_INCREMENT, `RecipientID` text NOT NULL, `Processed` enum('false','true') NOT NULL DEFAULT 'false', PRIMARY KEY (`ID`)) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=78 ; -- Dumping data for table `inbox`

Sumber : Implementasi Sistem

Tabel inbox ditunjukkan pada Tabel 5.3. Tabel ini berfungsi untuk menyimpan SMS yang masuk ke dalam sistem. Pada tabel ini terdapat primary key sebagai ID dari SMS yang masuk. TextDecoded sebagai isi dari SMS dan juga terdapat waltu dari SMS yang masuk.

Tabel 5.4 Outbox

NAMA	Outbox
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan pesan yang akan dikirimkan.
DDL	-- Table structure for table `outbox` CREATE TABLE IF NOT EXISTS `outbox` (`UpdatedInDB` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP, `InsertIntoDB` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00', `SendingDateTime` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00', `Text` text, `DestinationNumber` varchar(20) NOT NULL DEFAULT '', `Coding` enum('Default_No_Compression','Unicode_No_Compression','8bit','Default_Compression','Unicode_Compression') NOT NULL DEFAULT 'Default_No_Compression', `UDH` text, `Class` int(11) DEFAULT '-1',

	`TextDecoded` varchar(160) NOT NULL DEFAULT '', `ID` int(10) unsigned NOT NULL AUTO_INCREMENT, `MultiPart` enum('false','true') DEFAULT 'false', `RelativeValidity` int(11) DEFAULT '-1', `SenderID` varchar(255) DEFAULT NULL, `SendingTimeOut` timestamp NULL DEFAULT '0000-00-00 00:00:00', `DeliveryReport` enum('default','yes','no') DEFAULT 'default', `CreatorID` text NOT NULL, PRIMARY KEY (`ID`), KEY `outbox_date` (`SendingDateTime`,`SendingTimeOut`), KEY `outbox_sender` (`SenderID`)) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=111 ; -- Dumping data for table `outbox`

Sumber : Implementasi Sistem

Tabel outbox ditunjukkan pada Tabel 5.4. Tabel tersebut berfungsi untuk menyimpan sementara SMS yang akan dikirimkan Di dalam tabel ini terdapat field nomor pengirim, nomor tujuan, isi pesan dan waktu pengiriman. Selain itu juga terdapat field status pengiriman pesan.

Tabel 5.5 Outbox_multipart

NAMA	Outbox_multipart
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan pesan yang akan dikirimkan dalam jumlah banyak ataupun pesan grup.
DDL	-- Table structure for table `outbox_multipart` CREATE TABLE IF NOT EXISTS `outbox_multipart` (`Text` text, `Coding` enum('Default_No_Compression','Unicode_No_Compression','8bit','Default_Compression','Unicode_Compression') NOT NULL DEFAULT 'Default_No_Compression', `UDH` text, `Class` int(11) DEFAULT '-1', `TextDecoded` varchar(160) DEFAULT NULL, `ID` int(10) unsigned NOT NULL DEFAULT '0',

	`SequencePosition` int(11) NOT NULL DEFAULT '1', PRIMARY KEY (`ID`,`SequencePosition`)) ENGINE=MyISAM DEFAULT CHARSET=utf8 -- Dumping data for table `outbox_multipart`

Sumber : Implementasi Sistem

Tabel Outbox_*multy*part yang ditunjukkan pada Tabel 5.5 merupakan tabel yang digunakan untuk menyimpan sementara SMS yang akan dikeluarkan. Di dalam tabel ini terdapat field nomor pengirim, nomor tujuan, isi pesan dan waktu pengiriman. Selain itu juga terdapat field status pengiriman pesan. SMS ini ditujukan lebih dari satu penerima SMS.

Tabel 5.6 pbk

NAMA	Pbk
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan data buku telefon
DDL	-- Table structure for table `pbk` CREATE TABLE IF NOT EXISTS `pbk` (`GroupID` int(11) NOT NULL DEFAULT '-1', `Name` text NOT NULL, `Number` text NOT NULL) ENGINE=MyISAM DEFAULT CHARSET=utf8; -- Dumping data for table `pbk`

Sumber : Implementasi Sistem

Tabel pbk ditunjukkan pada tabel 5.6. Tabel tersebut berfungsi untuk menyimpan data nomor hp dan nama kontak. Hal ini mirip dengan *phonebook.*

Tabel 5.7 pbk_groups

NAMA	pbk_groups
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan pesan yang akan dikirimkan pesan grup.
DDL	-- Table structure for table `pbk_groups` CREATE TABLE IF NOT EXISTS `pbk_groups` (`Name` text NOT NULL, `ID` int(11) NOT NULL AUTO_INCREMENT, PRIMARY KEY (`ID`)) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ; -- Dumping data for table `pbk_groups`

Sumber : Implementasi Sistem

Tabel pbk_groups ditunjukkan pada tabel 5.7. Tabel ini berfungsi untuk menyimpan data nomor hp dan nama kontak. Hal ini mirip dengan *phonebook* namun dalam bentuk grup.

Tabel 5.8 sent_item

NAMA	Sentitems
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan pesan yang telah dikirimkan.
DDL	`-- Table structure for table `sentitems`` `CREATE TABLE IF NOT EXISTS `sentitems` ( `UpdatedInDB` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP, `InsertIntoDB` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00', `SendingDateTime` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00', `DeliveryDateTime` timestamp NULL DEFAULT NULL, `Text` text NOT NULL, `DestinationNumber` varchar(20) NOT NULL DEFAULT '', `Coding` enum('Default_No_Compression','Unicode_No_Compression','8bit','Default_Compression','Unicode_Compression') NOT NULL DEFAULT 'Default_No_Compression', `UDH` text NOT NULL, `SMSCNumber` varchar(20) NOT NULL DEFAULT '', `Class` int(11) NOT NULL DEFAULT '-1', `TextDecoded` varchar(160) NOT NULL DEFAULT '', `ID` int(10) unsigned NOT NULL DEFAULT '0', `SenderID` varchar(255) NOT NULL, `SequencePosition` int(11) NOT NULL DEFAULT '1', `Status` enum('SendingOK','SendingOKNoReport','SendingError','DeliveryOK','DeliveryFailed','DeliveryPending','DeliveryUnknown','Error') NOT NULL DEFAULT 'SendingOK', `StatusError` int(11) NOT NULL DEFAULT '-1', `TPMR` int(11) NOT NULL DEFAULT '-1', `RelativeValidity` int(11) NOT NULL DEFAULT '-1', `CreatorID` text NOT NULL, PRIMARY KEY (`ID`,`SequencePosition`),

```
  KEY                    `sentitems_date`
(`DeliveryDateTime`),
  KEY `sentitems_tpmr` (`TPMR`),
  KEY                    `sentitems_dest`
(`DestinationNumber`),
  KEY `sentitems_sender` (`SenderID`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
-- Dumping data for table `sentitems`
```

Sumber : Implementasi Sistem

Tabel sent_items ditunjukkan pada tabel 5.6. Tabel ini berfungsi untuk menyimpan SMS yang telah dikirimkan. Di dalam tabel ini terdapat field nomor pengirim, nomor tujuan, isi pesan dan waktu pengiriman. Selain itu juga terdapat field status pengiriman pesan.

Tabel 5.9 phones

NAMA	Phones
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan informasi perangkat telepon atau modem yang terhubung dengan server / PC.
DDL	`-- Table structure for table `phones` CREATE TABLE IF NOT EXISTS `phones` ( `ID` text NOT NULL, `UpdatedInDB` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP, `InsertIntoDB` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00', `TimeOut` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00', `Send` enum('yes','no') NOT NULL DEFAULT 'no', `Receive` enum('yes','no') NOT NULL DEFAULT 'no', `IMEI` varchar(35) NOT NULL, `Client` text NOT NULL, `Battery` int(11) NOT NULL DEFAULT '0', `Signal` int(11) NOT NULL DEFAULT '0', `Sent` int(11) NOT NULL DEFAULT '0', `Received` int(11) NOT NULL DEFAULT '0', PRIMARY KEY (`IMEI`) ) ENGINE=MyISAM DEFAULT CHARSET=utf8; -- Dumping data for table `phones`

Sumber : Implementasi Sistem

Tabel phones ditunjukkan pada tabel 5.9. Tabel ini merupakan tabel yang digunakan untuk menyimpan informasi perangkat telepon atau modem yang terhubung dengan server / PC. Tabel ini *include* pada saat Gammu *library* diinstal.

Tabel 5.10 user

NAMA	User
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan informasi user yang dapat mengakses aplikasi ini.

DDL	`-- Table structure for table `user``
	
	 `CREATE TABLE IF NOT EXISTS `user` (`
	 `  `username` varchar(32) NOT NULL,`
	 `  `pass` varchar(32) NOT NULL,`
	 `  PRIMARY KEY (`username`)`
	
	 `) ENGINE=MyISAM DEFAULT CHARSET=latin1;`

Sumber : Implementasi Sistem

Tabel user ditunjukkan pada tabel 5.10. Tabel ini berfungsi menyimpan data user yang dapat melakukan login ke dalam aplikasi. Terdapat field username dan password. Password dienkripsi menggunakan metode md5 agar password tidak diketahui oleh orang lain.

Tabel 5.11 data_log

NAMA	Data_log
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan data yang dikirimkan dari sensor node.

DDL	`-- Struktur dari tabel `data_log``
	
	 `CREATE TABLE IF NOT EXISTS `data_log` (`
	 `  `Nomer`  int(5)  NOT  NULL AUTO_INCREMENT,`
	 `  `Status` char(20) CHARACTER SET latin1 COLLATE latin1_general_ci NOT NULL,`
	 `  `Waktu_Sensor` datetime NOT NULL,`
	 `  PRIMARY KEY (`Nomer`)`
	 `)`

Sumber : Implementasi Sistem

Tabel data log yang ditunjukkan pada tabel 5.11 berfungsi sebagai penyimpanan data utama dari sensor node. Data dari tabel ini nantinya yang akan memberikan informasi terkait status sensor. Jika data bernilai 1, maka status tersebut bahaya. Jika data bernilai 0, maka status aman.

Tabel 5.12 user_ms

NAMA	Data_log
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan data admin .

DDL	`-- Struktur dari tabel `user_ms` CREATE TABLE IF NOT EXISTS `user_ms` ( `username` varchar(32) NOT NULL, `pass` varchar(32) NOT NULL, PRIMARY KEY (`username`) ) ENGINE=MyISAM DEFAULT CHARSET=latin1;`

Sumber : Implementasi Sistem

Tabel user_ms yang ditunjukkan pada tabel 5.12 berfungsi menyimpan data user admin yang dapat mengakses aplikasi data sink. User yang dapat mengakses data sink hanya admin saja.

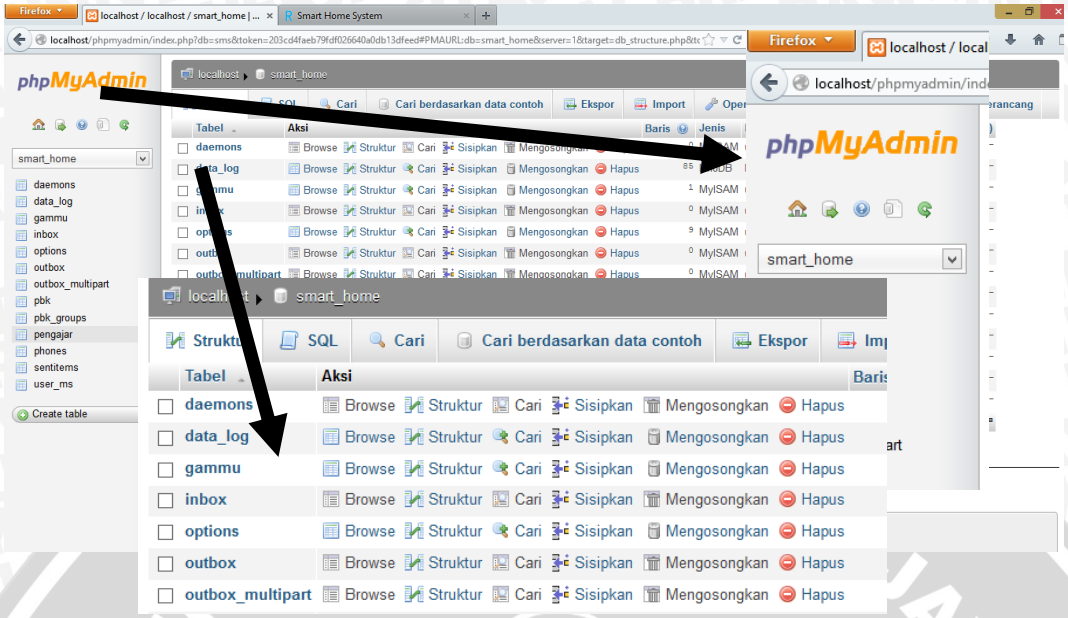
Tabel 5.13 pemilik

NAMA	Data_log
KETERANGAN	Tabel ini merupakan tabel yang digunakan untuk menyimpan data admin .
DDL	`-- Struktur dari tabel `pemilik` CREATE TABLE IF NOT EXISTS `pemilik` ( `id_pemilik` varchar(10) NOT NULL, `pemilik` varchar(100) NOT NULL, `status_dalam_kluarga` text NOT NULL, `hp` varchar(20) NOT NULL, PRIMARY KEY (`id_pengajar`)  ) ENGINE=MyISAM DEFAULT CHARSET=latin1;`

Sumber : Implementasi Sistem

Tabel pemilik ditunjukkan pada tabel 5.13. Tabel ini berfungsi menyimpan data pemilik rumah. Pemilik rumah inilah yang akan mendapatkan SMS dari aplikasi data sink jika status sensor menunjukkan status bahaya.

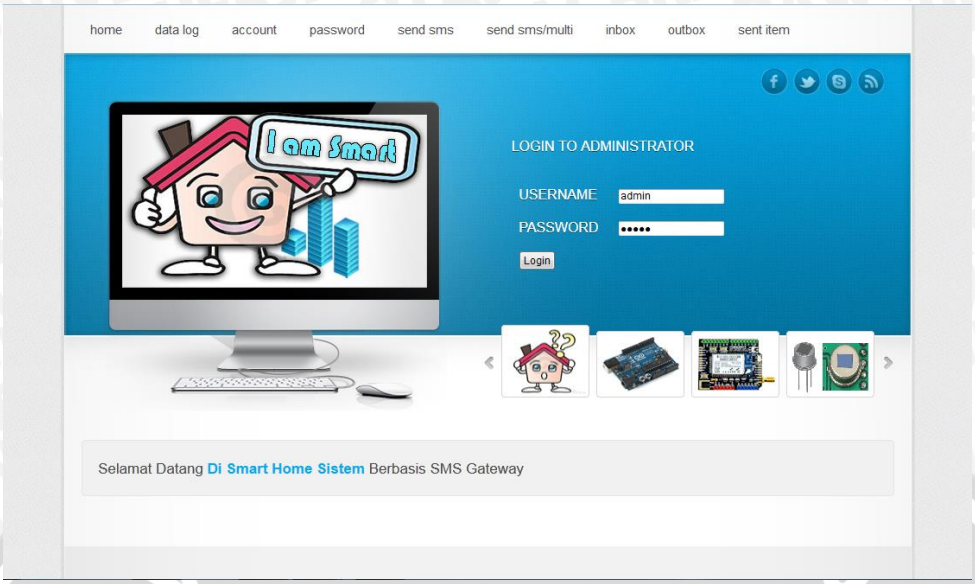
Gambar 5.7 menunjukkan hasil dari *script* basis data tersebut setelah diimplementasikan. *Script* basis data ini diimplementasikan pada web server XAMPP / Localhost dengan basis data MySql.



Gambar 5.7 Implementasi basis data
Sumber : Perancangan Sistem

1.2.2 Implementasi *Interface* Data Sink

Berdasarkan subbab 4.3.2 perancangan *Interface* data sink, implementasi *Interface* data sink menggunakan bahasa pemograman PHP dan CSS untuk memperindah tampilan. Terdapat sembilan halaman utama pada *Interface* ini. Halaman ini meliputi halaman login, halaman data log, halaman mengubah password, halaman menambah user pemilik rumah, halaman kirim pesan, halaman kirim pesan group, halaman kotak masuk pesan , halaman kotak keluar pesan, dan halaman pesan terkirim. Tampilan dari implementasi desain *Interface* data sink ditunjukkan dalam gambar 5.8 sampai dengan gambar 5.7 :



Gambar 5.8 Halaman Login
Sumber : Implementasi Sistem

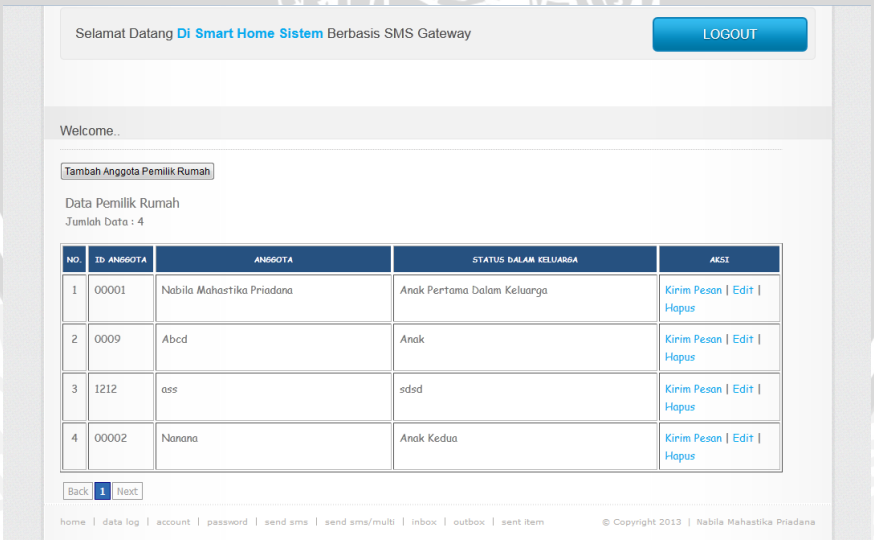
User dapat melakukan login dengan memasukkan username dan password pada kolom yang sudah ada yang ditunjukkan pada Gambar 5.8. Tombol menu pada halaman login ini tidak berfungsi, tombol ini akan berfungsi jika user telah melakukan login.

Gambar 5.9 Halaman Home
Sumber : Implementasi Sistem

Gambar 5.9 adalah halaman utama setelah user melakukan login. Di halaman ini user dapat melihat tanggal saat user melakukan login. Di halaman ini user sudah dapat menggunakan tombol menu yang ada.

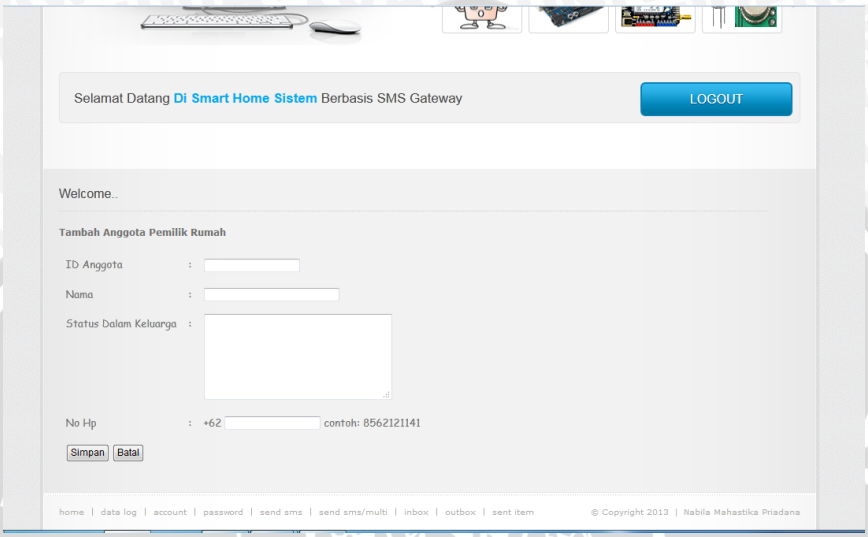
Gambar 5.10 Halaman Data Log
Sumber : Implementasi Sistem

Gambar 5.10 adalah halaman data log. Halaman tersebut merupakan halaman yang menampilkan data yang dikirimkan dari sensor node.



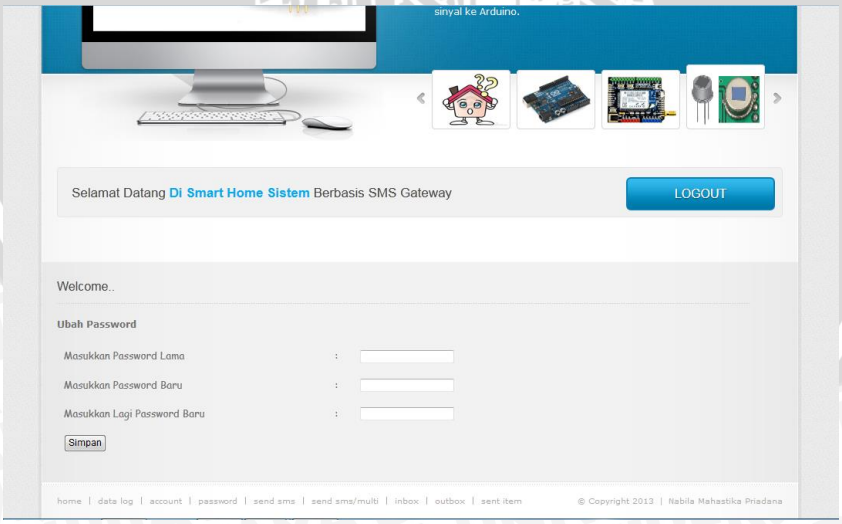
Gambar 5.11 Account
Sumber : Implementasi Sistem

Gambar 5.11 adalah halaman account. Halaman tersebut yang menampilkan user pemilik rumah. User dapat menambahkan anggota pemilik rumah, mengirim pesan ke pemilik rumah dan menghapus anggota pemilik rumah.



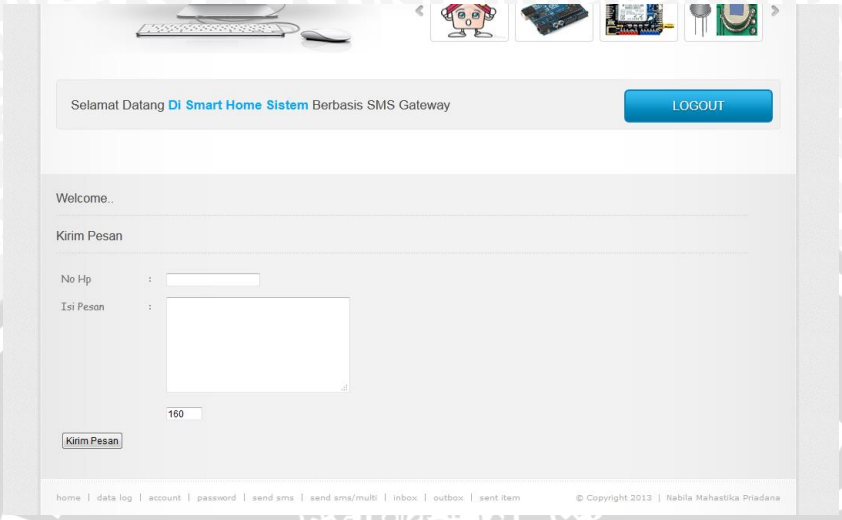
Gambar 5.12 Halaman Tambah Pemilik Rumah
Sumber : Implementasi Sistem

Gambar 5.12 adalah halaman tambah pemilik rumah. Halaman tersebut untuk menambahkan anggota pemilik rumah. SMS yang dikrimkan dari sistem akan diterima oleh semua anggota pemilik rumah.



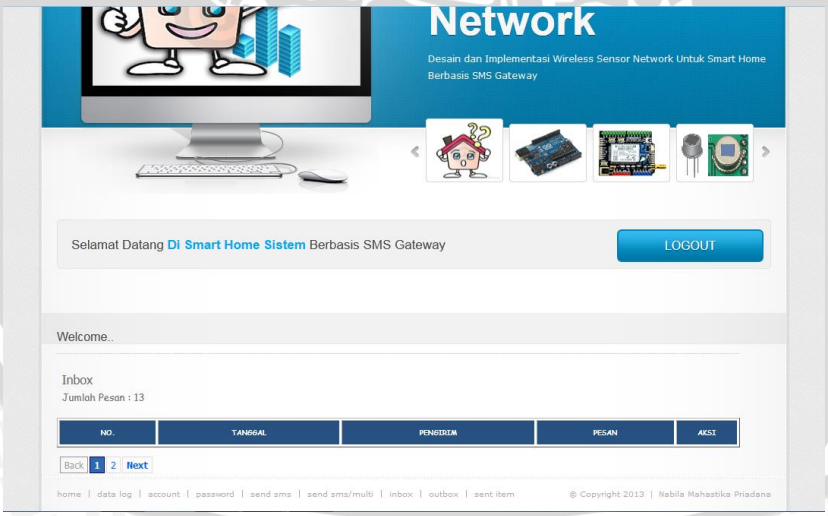
Gambar 5.13 Halaman Ubah Password
Sumber : Implementasi Sistem

Gambar 5.13 adalah halaman ubah password. Halaman tersebut merupakan halaman user untuk mengubah password yang telah dibuat sebelumnya. Tujuan dari halaman ini agar aplikasi data sink tidak mudah diakses oleh orang lain.



Gambar 5.14 Halaman Kirim SMS
Sumber : Implementasi Sistem

Gambar 5.14 adalah Halaman kirim SMS. Halaman tersebut merupakan halaman user untuk mengirimkan SMS ke nomor tujuan sesuai dengan keinginan user.

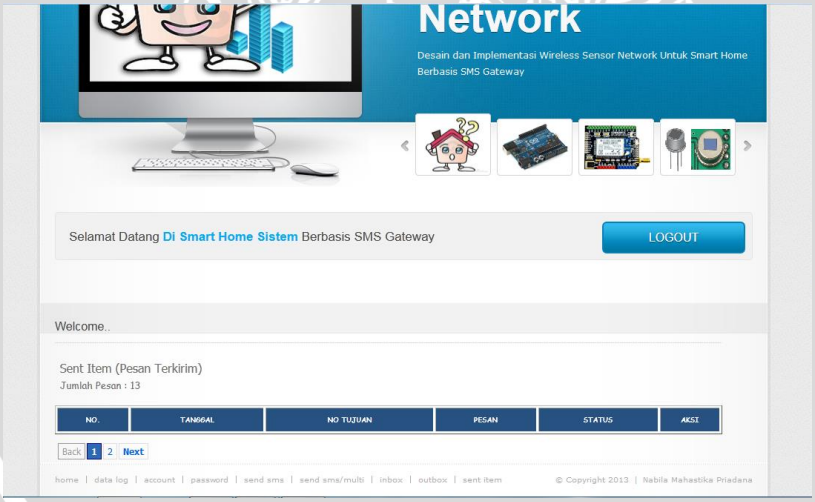


Gambar 5.15 Halaman Inbox
Sumber : Implementasi Sistem

Gambar 5.15 adalah Halaman inbox. Halaman tersebut yang menampilkan SMS yang masuk ke dalam sistem. User dapat menghapus SMS yang tidak terlalu penting pada aplikasi ini agar basis data aplikasi tidak terlalu penuh.

Gambar 5.16 Halaman Outbox
Sumber : Implementasi Sistem

Gambar 5.16 adalah Halaman outbox. Halaman tersebut yang menampilkan SMS yang akan dikirimkan ke nomor tujuan sesuai keinginan user dan nomor tujuan anggota pemilik rumah. User juga dapat menghapus data outbox pada halaman ini.



Gambar 5.17 Halaman Sent Items
Sumber : Implementasi Sistem

Gambar 5.17 adalah Halaman sent items. Halaman tersebut yang menampilkan SMS yang sudah terkirim ke nomor hp sesuai tujuannya. User juga dapat menghapus data sent items pada halaman ini.

1.2.3 Implementasi Aplikasi Data Sink

Berdasarkan subbab 4.5.1 perancangan algoritma aplikasi data sink, aplikasi tersebut diimplementasikan dengan menggunakan bahasa pemrograman PHP seperti yang ditunjukkan dalam gambar 5.18. Aplikasi data sink berperan penting dalam mengolah data yang diperoleh dari sensor node. Aplikasi bertugas menyeleksi data dari sensor node, apakah data yang diterima berstatus bahaya ataukah data yang diterima berstatus aman. Selanjutnya, jika data yang diterima berstatus bahaya, maka aplikasi akan secara otomatis mengirimkan notifikasi SMS Gateway ke user pemilik rumah.

```
1. <?php
2. error_reporting(0);
3. session_start();
4. include "koneksi/koneksi.php";
5. $username = $_POST['username'];
6. $pass = $_POST['pass'];
7. $pass = md5($pass); //Mengambil data dari basis data
8.  $ambil = mysql_query("select * from user_ms where
username='$username' and pass='$pass'");

//mengecek ada tidak file yang dicari
9. $ada = mysql_num_rows($ambil);
10. if ($ada == 1){ //Daftarkan variabel session
11. session_register("username");
12. session_register("pass"); //jumping ke index
13. include "loading_berhasil.html";
14.    echo    "<meta    http-equiv='refresh'    content='5
URL=master.php'>";
}
15. else{
16. include "loading_gagal.html";
     echo    "<meta    http-equiv='refresh'    content='5
URL=index.php'>";
}
17. ?>
```

Gambar 5.18 Tampilan *script* Cek Login.php
Sumber : Implementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma cek login , gambar 5.18 adalah *script* halaman cek login. Halaman ini berfungsi untuk mengecek, apakah user yang masuk di aplikasi adalah admin atau bukan. Jika yang melakukan login adalah admin, maka aplikasi akan meneruskan ke halaman master. Jika tidak, maka aplikasi akan meneruskan ke halaman index. Berikut penjelasan lebih lengkap dari gambar 5.18 :

1. Baris ke- 3 : Memulai session. Session adalah hak akses user yang nantinya akan di cek di dalam basis data.
2. Baris ke- 4 : Melakukan koneksi dengan basis data.
3. Baris ke- 5-7 : Mengambil data dengan method post dari username dan password yang telah dimasukkan oleh user
4. Baris ke- 8-9 : Melakukan query basis data dan mengecek username dan password yang telah dimasukkan oleh user. Jika ada, akan dilakukan *fetch* data.
5. Baris ke- 10-12 : Jika username dan password ada, maka *session* akan meregistrasi username dan password.
6. Baris ke- 13-15 : Melakukan jumping ke halaman index yang kemudian masuk ke halamn master.
7. Baris ke- 16- 17 : Jika gagal akan kembali ke halaman index.

```
1. <?php
2. $q=$_GET["q"];
3. $w = $_GET["w"];
4. $con = mysql_connect('localhost','root','');
5. if (!$con)
6.  {
7.  die('Could not connect: ' . mysql_error());
8.  }
9. mysql_select_db("smart_home", $con);

10.        $sql=        "INSERT        INTO        data_log
(Status,Waktu_Sensor)VALUES('".$q."',CURRENT_TIMESTAMP) ";
11. mysql_query($sql);
12. mysql_close($con);
```

Gambar 5.19 Tampilan *script* Masuk.php
Sumber : Implementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma koneksi sensor node, diimplementasikan seperti pada gambar 5.19. Gambar tersebut menunjukkan tampilan *script* masuk.php yang berfungsi sebagai koneksi sensor dari data yang diperoleh dari sensor node kedalam basis data . Data yang diperoleh dari sensor node akan dimasukkan ke dalam tabel data_log. Berikut penjelasan secara lengkap :

1. Baris ke- 2-4 : Menerima method dari get dari sensor node.
2. Baris ke- 4-8 : Proses menyambungkan ke basis data dengan server localhost, user root dan password kosong.

3. Baris ke- 9 : Proses memilih basis data
4. Baris ke- 10 : Proses membuat syntax insert data ke basis data.
5. Baris ke- 11 : Proses mengeksekusi syntax
6. Baris ke- 12 : Proses menutup koneksi ke basis data

```
1. <a     href="master.php?menu=tambah_anggota">     <input
   type=submit value="Tambah Anggota Pemilik Rumah"> </a>
2. <?php
3. include "koneksi/koneksi.php";

4. $offset=$_GET['offset'];
5. $totalquery=mysql_query("select * from pemilik");
6. $numrows=mysql_num_rows($totalquery);

7. //jumlah data yang ditampilkan perpage
8. $limit = 10;
9. if (empty ($offset)) {
10.   $offset = 0;
11.   }
12.   if ($numrows == 0) {
13.   echo  "<br><center>  Tidak  Ada  Data  Pemilik  Rumah
   </center>";
14.   }
15.   else {
16.   ?>
17.   <table     width="100%"     height="30"     border="0"
   cellpadding="0" cellspacing="0">
18.   <tr>
19.   <td height="30"">
20.   <div  align="left"><font  size="3"  face="tahoma">Data
   Pemilik  Rumah</font><br>  Jumlah  Data  :  <?php  echo
   "$numrows" ;?> </div>
21.   </td>
22.   </tr>
23.   </table>
24.   <?php
25.   // panggil semua data dari tabel pemilik diurutkan
   berdasarkan id pemilik, dibatasi dengan limit = 15
26.   $hasil = mysql_query("select * from pemilik order by
   pemilik limit $offset,$limit");
27.   $k = 1;
28.   $k = 1 + $offset;

29.   echo"
30.   <div align=left>
31.   <table border=1 width=100%>
32.   <tr>
33.   <th> No. </th>
34.   <th width=70> ID Anggota </th>
35.   <th> Anggota </th>
36.   <th> Status Dalam Keluarga</th>
```

```
37.   <th width=140> Aksi </th>
38.   </tr>
39.   ";

40.   while ($data = mysql_fetch_array($hasil)) {

41.   echo"
42.   <tr>
43.   <td width=10 align=center> $k </td>
44.   <td> $data[id_pemilik] </td>
45.   <td> $data[pemilik] </td>
46.   <td> $data[alamat]</td>
47.   <td>
48.   <a
      href=master.php?menu=kirim_pesan&id_pemilik=$data[id_pem
      ilik] title='Kirim Pesan'> Kirim Pesan </a> |
49.   <a
      href=master.php?menu=edit_anggota&id_pemilik=$data[id_pe
      milik] title='Edit Anggota'> Edit </a> |
50.   <a
      href=master.php?menu=hapus_anggota&id_pemilik=$data[id_p
      emilik] title='Hapus Anggota'> Hapus </a> </td>
51.   </tr>
52.   ";
53.   $k++;
54.   }
55.   //untuk tutup tabel
56.   echo "</table>";
57.   echo "<div class=paging>";

58.   if ($offset!=0) {
59.   $prevoffset = $offset-10;
60.   echo         "<span          class=prevnext>          <a
      href=$PHP_SELF?menu=anggota&offset=$prevoffset>Back</a><
      /span>";
61.   }
62.   else {
63.   echo    "<span    class=disabled>Back</span>";//cetak
      halaman tanpa link
64.   }
65.   //hitung jumlah halaman
66.   $halaman = intval($numrows/$limit);//Pembulatan

67.   if ($numrows%$limit){
68.   $halaman++;
69.   }
70.   for($i=1;$i<=$halaman;$i++){
71.   $newoffset = $limit * ($i-1);
72.   if($offset!=$newoffset){
73.   echo                                                  "<a
      href=$PHP_SELF?menu=anggota&offset=$newoffset>$i</a>";
74.   //cetak halaman
```

```
75.   }
76.   else {
77.   echo    "<span    class=current>".$i."</span>";//cetak
  halaman tanpa link
78.   }
79.   }

80.   //cek halaman akhir
81.   if(!(($offset/$limit)+1==$halaman) && $halaman !=1){

82.   //jika bukan halaman terakhir maka berikan next
83.   $newoffset = $offset + $limit;
84.   echo              "<span             class=prevnext><a
  href=$PHP_SELF?menu=anggota&offset=$newoffset>Next</a>";
85.   }
86.   else {
87.   echo    "<span    class=disabled>Next</span>";//cetak
  halaman tanpa link
88.   }
89.   }
90.   echo "</div";
91.   echo "</font>";
92.   ?>
```

Gambar 5.20 Tampilan *script* Anggota.php
Sumber : Implementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma account, diimplementasikan seperti pada gambar 5.20. Gambar tersebut menunjukkan tampilan *script* anggota.php . Halaman ini adalah halaman account. Pada halaman ini user dapat melihat siapa saja pemilik rumah yang ada didalam basis data. User dapat menambah, mengedit dan menghapus pemilik rumah yang ada pada halaman ini. Berikut penjelasan lebih rinci source code dari gambar 5.20 :

1. Baris ke- 1 : Proses melakukan koneksi dengan halaman master yang di dalamnya terdapat halaman tambah anggota.
2. Baris ke- 2 : Proses melakukan koneksi dengan basis data
3. Baris ke- 4 : Proses mengambil method get
4. Baris ke- 5-6 : Proses melakukan query basis data dari tabel pemilik rumah
5. Baris ke- 8-24 : Proses mengambil data dan menampilkan per halaman dengan limit data 10.
6. Baris ke- 26-28 : Proses memanggil semua data dari tabel pemilik dan diurutkan berdasarkan id pemilik yang dibatasi dengan limit 15
7. Baris ke- 30-38 : Proses membuat tabel

8. Baris ke- 40-47 : Proses mengambil data dari *fetching* data dan dimasukkan ke dalam tabel untuk ditampilkan pada halaman
9. Baris ke- 48 : Link kirim pesan ke pemilik rumah
10. Baris ke- 49 : Link edit anggota untuk mengedit anggota
11. Baris ke- 50 : Link hapus anggota untuk menghapus anggota pemilik rumah
12. Baris ke- 51-92 : Proses melakukan paginasi tabel anggota pemilik rumah

```
1.<h4> Ubah Password </h4>
2.<?php
3.include "koneksi/koneksi.php";
4.$ambil = mysql_query("select * from user_ms");
5.$data = mysql_fetch_array($ambil);
6.?>
7.<form method="post" action="index.php?menu=tpassword">
8.<table width="70%" bordercolor="#999999">
9.   <tr>
10.      <td>Masukkan Password Lama</td>
11.      <td>:</td>
12.      <td>  <input  name="pass_lama"  type="password">
  <input          name="username"          type="hidden"
  value="<?echo"$data[username]";?>"> </td>
13.      </tr>
14.      <tr>
15.            <td> Masukkan Password Baru </td>
16.       <td>:</td>
17.            <td>       <input       name="pass_baru"
  type="password"></td>
18.      </tr>
19.      <tr>
20.        <td> Masukkan Lagi Password Baru </td>
21.        <td>:</td>
22.        <td>        <input        name="pass_ulangi"
  type="password"></td>
23.      </tr>
24.      <tr>
25.        <td     colspan=3>     <input     type="submit"
  value="Simpan"></td>
26.      </tr>
27.  </table> </form>
```

Gambar 5.21 Tampilan *script* Ubah_password.php
Sumber : Implementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma password, diimplementasikan seperti pada gambar 5.21. Gambar tersebut menunjukkan tampilan *script* ubah_password.php yang berfungsi untuk mengubah password admin pada aplikasi data sink. Hal ini untuk mengantisipasi user lain yang dapat

masuk ke dalam aplikasi ini. Berikut penjelasan lebih detail source code berdasarkan gambar 5.21 :

1. Baris ke- 3 : Proses melakukan koneksi ke dalam basis data
2. Baris ke- 4-5 : Proses melakukan query basis data mengambil data user di dalam basis data
3. Baris ke- 7 : Proses mengambil method post dari halaman tpassword
4. Baris ke- 8-26 : Field ubah pasword

```
1. <?php
2. include "koneksi/koneksi.php";

3. $offset=$_GET['offset'];
4. $totalquery=mysql_query("select * from data_log");
5. $numrows=mysql_num_rows($totalquery);

//jumlah data yang ditampilkan perpage
6. $limit = 10;
7. if (empty ($offset)) {
8.    $offset = 0;
9. }
10. if ($numrows == 0) {
11.   echo "<br><center> Tidak Ada Data Log Sensor</center>"

12. }
13. else {
14. ?>
15.    <table    width="100%"    height="30"    border="0"
cellpadding="0" 16. cellspacing="0" align="center">
17.   <tr>
18.   <td height="30"">
19.        <div        align="left"><font        size="3"
20.face="tahoma">Data LOg Smart Home</font><br> Jumlah Data
: 21.<?php echo "$numrows" ;?></div>
22.   </td>
23.</tr>
24.</table>
25. <?php

26. $hasil = mysql_query("select * from data_log order by
Waktu_Sensor desc limit $offset,$limit");
27. $k = 1;
28. $k = 1 + $offset;

29. echo"
30.   <div align=left>
31.        <table border=>
32.             <tr>
33.                  <th> NO </th>
34.                  <th> ID </th>
```

```
35.                     <th> Status</th>
36.                     <th> Waktu </th>
37.                 </tr>
38. ";

39. while ($data = mysql_fetch_array($hasil)) {

40. echo"
41.         <tr>
42.                 <td align=center> $k </td>
43.                 <td> $data[Nomer] </td>
44.                 <td> $data[Status] </td>
45.                 <td> $data[Waktu_Sensor] </td>
46.         </tr>
47. ";
48. $k++;
49. }
//untuk tutup tabel
50. echo "</table>";
51. echo "<div class=paging>";

52. if ($offset!=0) {
53.   $prevoffset = $offset-10;
54.   echo           "<span                  class=prevnext><a
href=$PHP_SELF?menu=siswa&offset=$prevoffset>Back</a></spa
n>";
55. }
56. else {
57.   echo    "<span    class=disabled>Back</span>";//cetak
halaman tanpa link
59. }
//hitung jumlah halaman
60. $halaman = intval($numrows/$limit);//Pembulatan

61. if ($numrows%$limit){
     $halaman++;
62. }
63. for($i=1;$i<=$halaman;$i++){
64.   $newoffset = $limit * ($i-1);
65.   if($offset!=$newoffset){
66.         echo                                          "<a
href=$PHP_SELF?menu=siswa&offset=$newoffset>$i</a>";
          //cetak halaman
67.   }
68.   else {
69.         echo "<span class=current>".$i."</span>";//cetak
halaman tanpa link
70.   }
71. }

//cek halaman akhir
72.if(!(($offset/$limit)+1==$halaman) && $halaman !=1){
```

```
     //jika bukan halaman terakhir maka berikan next
73.   $newoffset = $offset + $limit;
     echo              "<span              class=prevnext><a
href=$PHP_SELF?menu=siswa&offset=$newoffset>Next</a>";

74. }
75. else {
76.   echo    "<span    class=disabled>Next</span>";//cetak
halaman tanpa link

77. }
78. }
79. echo "</div";
80. echo "</font>";
81. ?>
```

Gambar 5.22 Tampilan *script* Data_log.php
Sumber : Implementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma data log, diimplementasikan seperti pada gambar 5.22. Gambar tersebut menunjukkan tampilan *script* halaman data log yang berfungsi untuk menampilkan data yang diperoleh dari sensor node. Data akan ditampilkan dalam bentuk tabel. Berikut penjelasan lebih detail berdasarkan tampilan *script* gambar 5.22 :

1. Baris ke- 2-5 : Proses melakukan koneksi ke basis data dan melakukan query.
2. Baris ke- 6-14 : Proses mencari data di dalam basis data dengan limit data yang ditampilkan sebanyak 10.
3. Baris ke- 15-24 : Proses membuat tabel untuk menampilkan data
4. Baris ke- 26 : Proses Query basis data
5. Baris ke- 30-59 : Proses menampilkan data pada tabel setelah proses query data
6. Baris ke- 60-81 : Proses membuat paginasi pada tabel data log

```
1. <html>q
2. <head>
3.   <title> Pesan Single </title>
4.   <script>
5.   function Count(){
6.         var karakter,maksimum;
7.         maksimum = 160
8.         karakter                =                maksimum-
   (document.form1.isi_pesan.value.length);
9.         if (karakter < 0) {
10.                alert("Jumlah Maksimum Karakter:  " +
   maksimum + "");
```

```
11.                 document.form1.isi_pesan.value      =
   document.form1.isi_pesan.value.substring(0,maksimum);
12.                 karakter        =        maksimum-
   (document.form1.isi_pesan.value.length);
13.                 document.form1.counter.value        =
   karakter;
14.           }
15.           else {
16.                 document.form1.counter.value        =
   maksimum-(document.form1.isi_pesan.value.length);
17.           }
18.     }
19.     </script>
20.  </head>
21.  <body>
22.  <form          name="form1"          method="post"
   action="send_SMS_single.php">
23.  <h3> Kirim Pesan </h3>
24.  <table   width="100%"   border="0"   cellspacing="0"
   cellpadding="3">
25.  <tr>
26.       <td width="100"> No Hp </td><td width="10"> :
   </td>
27.       <td> <input type=text name=hp></td>
28.  </tr>
29.  <tr valign="top">
30.       <td> Isi Pesan </td><td> : </td>
31.       <td>
32.            <textarea    name="isi_pesan"    cols="40"
   rows="7" OnFocus="Count();"
33.            OnClick="Count();"    onKeydown="Count();"
   OnChange="Count();"
34.            onKeyup="Count();"></textarea>
35.       </td>
36.  </tr>
37.  <tr>
38.       <td colspan="2"></td>
39.       <td><input name="counter" type="text" size="5"
   maxlength="5" value="160" /></td>
40.  </tr>
41.  <tr>
42.       <td colspan=3><input type="submit" value="Kirim
   Pesan" /></td>
43.  </tr>
44.  </table>
45.  </form>
46.  </body>
47.  </html>
```

Gambar 5.23 Tampilan *script* SMS_*single*.php
Sumber : Implementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma SMS *single*, diimplementasikan seperti pada gambar 5.23. Gambar tersebut menunjukkan tampilan *script* halaman SMS_*single*.php yang berfungsi untuk megirimkan SMS ke nomer hp yang tidak tercantum pada aplikasi data sink. SMS yang dikirimkan maksimal 160 karakter. Berikut penjelasan lebih lengkap gambar 5.23 :

1. Baris ke- 4-19 : Merupakan java *script* yang berfungsi untuk menampilkan peringatan jika salah satu field yang diisikan kosong. Field yang diisikan harus terisi semua, meliputi nomer hp dan isi pesan
2. Baris ke- 22 : Method post yang menghubungkan dengan halaman send_SMS_*single*.php. Halaman ini berfungsi untuk meneruskan SMS yang akan dikrimkan
3. Baris ke- 24-41 : Proses membuat form untuk mengisi pesan yang meliputi form nomer hp tujuan dan isi pesan
4. Baris ke- 42 : Tombol submit untuk mengkonfirmasi pesan yang sudah dibuat

```
1. <?php
2. include "koneksi/koneksi.php";
3. error_reporting(7);
4. $offset = $_GET['offset'];
5. //jumlah data yang ditampilkan perpage
6. $limit = 10;
7. if (empty ($offset)) {
8. $offset = 0;
9. }
10.  if ($numrows == 0) {
11.  echo "<br><center> Tidak Ada Pesan Masuk </center>";

12.  }
13.  else {
14.  ?>
15.  <table    width="100%"    height="30"    border="0"
   cellpadding="0" cellspacing="0">
16.  <tr>
17.  <td height="30"">
18.  <div          align="left"><font          size="3"
   face="tahoma">Inbox</font><br> Jumlah Pesan : <?php echo
   "$numrows" ;?> </div>
19.  </td>
20.  </tr>
21.  </table>
22.  <?php
```

```
23.  $hasil = mysql_query("select * from inbox order by
   ReceivingDateTime DESC limit $offset,$limit");
24.  $k = 1;
25.  $k = 1 + $offset;

26.  echo"
27.  <div align=left>
28.  <table border=1 width=100%>
29.  <tr>
30.  <th> No. </th>
31.  <th> Tanggal </th>
32.  <th> Pengirim </th>
33.  <th> Pesan </th>
34.  <th width=70> Aksi </th>
35.  </tr>
36.  ";

37.  while ($data = mysql_fetch_array($hasil)) {

38.  echo"
39.  <tr>
40.  <td width=10 align=center> $k </td>
41.  <td> $data[ReceivingDateTime] </td>
42.  <td> $data[SenderNumber] </td>
43.  <td> $data[TextDecoded] </td>
44.  <td> <a href=index.php?menu=hapus_inbox&ID=$data[ID]
   title='Hapus Inbox'> Hapus </a> </td>
45.  </tr>
46.  ";
47.  $k++;
48.  }
49.  //untuk tutup tabel
50.  echo "</table>";
51.  echo "<div class=paging>";

52.  if ($offset!=0) {

53.  $prevoffset = $offset-10;

54.  echo "<span class=prevnext> <a
   href=$PHP_SELF?menu=inbox&offset=$prevoffset>Back</a></
   span>";
55.  }
56.  else {
57.  echo "<span class=disabled>Back</span>";//cetak
   halaman tanpa link
58.  }
59.  //hitung jumlah halaman
60.  $halaman = intval($numrows/$limit);//Pembulatan

61.  if ($numrows%$limit){
```

```
62.  $halaman++;
63.  }
64.  for($i=1;$i<=$halaman;$i++){
65.  $newoffset = $limit * ($i-1);
66.  if($offset!=$newoffset){
67.  echo                                    "<a
   href=$PHP_SELF?menu=inbox&offset=$newoffset>$i</a>";
68.  //cetak halaman
69.  }
70.  else {
71.  echo    "<span    class=current>".$i."</span>";//cetak
   halaman tanpa link
72.  }
73.  }

74.  //cek halaman akhir
75.  if(!(($offset/$limit)+1==$halaman) && $halaman !=1){
76.  //jika bukan halaman terakhir maka berikan next
77.  $newoffset = $offset + $limit;
78.  echo             "<span            class=prevnext><a
   href=$PHP_SELF?menu=inbox&offset=$newoffset>Next</a>";
79.  }
80.  else {
81.  echo    "<span    class=disabled>Next</span>";//cetak
   halaman tanpa link
82.  }
83.  }
84.  echo "</div";
85.  echo "</font>";
86. ?>
```

Gambar 5.24 Tampilan *script* Inbox.php
Sumber : Impelementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma kotak masuk pesan, diimplementasikan seperti pada gambar 5.24. Gambar tersebut menunjukkan tampilan *script* halaman inbox.php yang berfungsi menampilkan semua pesan yang masuk ke dalam aplikasi data sink. Berikut penjelasan lebih rinci dari gambar 5.24 :

1. Baris ke- 2 : Proses melakukan koneksi dengan basis data
2. Baris ke- 3 : Proses menghilangkan error reporting
3. Baris ke- 4 : Method get pada offset halaman yang terkait dengan halaman inbox pada menu
4. Baris ke- 6-21 : Proses membuat tabel dan melakukan perulangan menampilkan data dengan limit minimal 10

5. Baris ke- 23 : Proses query basis data untuk mengambil data inbox pada basis data untuk ditampilkan
6. Baris ke- 24-47 : Proses menampilkan data setelah dilakukan proses query data ke dalam tabel
7. Baris ke- 52-85 : Proses paginasi pada tabel

```
1. <?php
2. include "koneksi/koneksi.php";
3. error_reporting(7);
4. $offset = $_GET['offset'];
5. $totalquery = mysql_query("select * from outbox");
6. $numrows = mysql_num_rows($totalquery);
7.
8. //jumlah data yang ditampilkan perpage
9. $limit = 10;
10.  if (empty ($offset)) {
11.        $offset = 0;
12.  }
13.  if ($numrows == 0) {
14.        echo "<br><center> Tidak Ada Pesan Keluar
   </center>";
15.  }
16.  else {
17.  ?>
18.  <table width="100%" height="30" border="0"
   cellpadding="0" cellspacing="0">
19.        <tr>
20.        <td height="30"">
21.                  <div align="left"><font size="3"
   face="tahoma">Outbox</font><br> Jumlah Pesan : <?php
   echo "$numrows" ;?> </div>
22.             </td>
23.      </tr>
24.  </table>
25.  <?php
26.  $hasil = mysql_query("select * from outbox order by
   ID DESC limit $offset,$limit");
27.  $k = 1;
28.  $k = 1 + $offset;
29.
30.  echo"
31.        <div align=left>
32.             <table border=1 width=100%>
33.                  <tr>
34.                       <th> No. </th>
35.                       <th> Tanggal </th>
36.                       <th> No Tujuan </th>
37.                       <th> Pesan </th>
38.                       <th width=70> Aksi </th>
39.                  </tr>
```

```
40.  ";
41.
42.  while ($data = mysql_fetch_array($hasil)) {
43.
44.  echo"
45.              <tr>
46.                   <td width=10 align=center> $k </td>
47.                   <td> $data[SendingDateTime] </td>
48.                   <td> $data[DestinationNumber] </td>
49.                   <td> $data[TextDecoded] </td>
50.                   <td> <a
   href=index.php?menu=hapus_outbox&ID=$data[ID]
   title='Hapus outbox'> Hapus </a> </td>
51.              </tr>
52.  ";
53.  $k++;
54.  }
55.  //untuk tutup tabel
56.  echo "</table>";
57.  echo "<div class=paging>";
58.
59.  if ($offset!=0) {
60.        $prevoffset = $offset-10;
61.        echo "<span class=prevnext> <a
   href=$PHP_SELF?menu=outbox&offset=$prevoffset>Back</a><
   /span>";
62.  }
63.  else {
64.        echo "<span class=disabled>Back</span>";//cetak
   halaman tanpa link
65.  }
66.  //hitung jumlah halaman
67.  $halaman = intval($numrows/$limit);//Pembulatan
68.
69.  if ($numrows%$limit){
70.        $halaman++;
71.  }
72.  for($i=1;$i<=$halaman;$i++){
73.        $newoffset = $limit * ($i-1);
74.        if($offset!=$newoffset){
75.              echo "<a
   href=$PHP_SELF?menu=outbox&offset=$newoffset>$i</a>";
76.              //cetak halaman
77.        }
78.        else {
79.              echo "<span
   class=current>".$i."</span>";//cetak halaman tanpa link
80.        }
81.  }
82.
83.  //cek halaman akhir
84.  if(!(($offset/$limit)+1==$halaman) && $halaman !=1){
```

```
85.
86.         //jika bukan halaman terakhir maka berikan next
87.         $newoffset = $offset + $limit;
88.         echo "<span class=prevnext><a
   href=$PHP_SELF?menu=outbox&offset=$newoffset>Next</a>";
89.   }
90.   else {
91.         echo "<span class=disabled>Next</span>";//cetak
   halaman tanpa link
92.   }
93.   }
94.   echo "</div";
95.   echo "</font>"; ?>
```

Gambar 5.25 Tampilan *script* Outbox.php
Sumber : Implementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma outbox, diimplementasikan seperti pada gambar 5.25. Gambar tersebut menunjukkan tampilan *script* halaman outbox.php yang berfungsi menampilkan semua pesan sementara yang akan dikirimkan. Berikut penjelasan lebih rinci dari gambar 5.25 :

1. Baris ke- 2 : Proses melakukan koneksi dengan basis data
2. Baris ke- 3 : Proses menghilangkan error reporting
3. Baris ke- 4 : Method get pada offset halaman yang terkait dengan halaman inbox pada menu
4. Baris ke- 5-6 : Proses query pada basis data outbox
5. Baris ke- 9-25 : Proses membuat tabel dan melakukan perulangan menampilkan data dengan limit minimal 10
6. Baris ke- 26 : Proses query basis data untuk mengambil data inbox pada basis data untuk ditampilkan
7. Baris ke- 28-57 : Proses menampilkan data setelah dilakukan proses query data ke dalam tabel
8. Baris ke- 58-96 : Proses paginasi pada tabel

```
1. <?php
2. include "koneksi/koneksi.php";
3. error_reporting(0);
4. $offset = $_GET['offset'];
5. $totalquery = mysql_query("select * from wilayah2");
6. $numrows = mysql_num_rows($totalquery);
7.
8. //jumlah data yang ditampilkan perpage
9. $limit = 10;
```

```
10.  if (empty ($offset)) {
11.        $offset = 0;
12.  }
13.  if ($numrows == 0) {
14.        echo "<br><center> Tidak Ada Pesan Terkirim
   </center>";
15.  }
16.  else {
17.  ?>
18.  <table width="100%" height="30" border="0"
   cellpadding="0" cellspacing="0">
19.        <tr>
20.        <td height="30"">
21.                  <div align="left"><font size="3"
   face="tahoma">Sent Item (Pesan Terkirim)</font><br>
   Jumlah Pesan : <?php echo "$numrows" ;?> </div>
22.              </td>
23.      </tr>
24.  </table>
25.  <?php
26.  $hasil = mysql_query("select * from sentitems order
   by ID DESC limit $offset,$limit");
27.  $k = 1;
28.  $k = 1 + $offset;
29.
30.  echo"
31.        <div align=left>
32.             <table border=1 width=100%>
33.                  <tr>
34.                       <th> No. </th>
35.                       <th> Tanggal </th>
36.                       <th> No Tujuan </th>
37.                       <th> Pesan </th>
38.                       <th> Status </th>
39.                       <th width=70> Aksi </th>
40.                  </tr>
41.  ";
42.
43.  while ($data = mysql_fetch_array($hasil)) {
44.
45.  echo"
46.             <tr>
47.                  <td width=10 align=center> $k </td>
48.                  <td> $data[SendingDateTime] </td>
49.                  <td> $data[DestinationNumber] </td>
50.                  <td> $data[TextDecoded] </td>
51.                  <td> $data[Status] </td>
52.                  <td> <a
   href=index.php?menu=hapus_sent_item&ID=$data[ID]
   title='Hapus Pesan Terkirim'> Hapus </a> </td>
53.             </tr>
54.  ";
```

```
55.  $k++;
56.  }
57.  //untuk tutup tabel
58.  echo "</table>";
59.  echo "<div class=paging>";
60.
61.  if ($offset!=0) {
62.        $prevoffset = $offset-10;
63.        echo "<span class=prevnext> <a
   href=$PHP_SELF?menu=sent_item&offset=$prevoffset>Back</
   a></span>";
64.  }
65.  else {
66.        echo "<span class=disabled>Back</span>";//cetak
   halaman tanpa link
67.  }
68.  //hitung jumlah halaman
69.  $halaman = intval($numrows/$limit);//Pembulatan
70.
71.  if ($numrows%$limit){
72.        $halaman++;
73.  }
74.  for($i=1;$i<=$halaman;$i++){
75.        $newoffset = $limit * ($i-1);
76.        if($offset!=$newoffset){
77.              echo "<a
   href=$PHP_SELF?menu=sent_item&offset=$newoffset>$i</a>"
   ;
78.              //cetak halaman
79.        }
80.        else {
81.              echo "<span
   class=current>".$i."</span>";//cetak halaman tanpa link
82.        }
83.  }
84.
85.  //cek halaman akhir
86.  if(!(($offset/$limit)+1==$halaman) && $halaman !=1){
87.
88.        //jika bukan halaman terakhir maka berikan next
89.        $newoffset = $offset + $limit;
90.        echo "<span class=prevnext><a
   href=$PHP_SELF?menu=sent_item&offset=$newoffset>Next</a
   >";
91.  }
92.  else {
93.        echo "<span class=disabled>Next</span>";//cetak
   halaman tanpa link
94.  }
95.  }
96.  echo "</div";
97.  echo "</font>";
```

```
98. ?>
```

Gambar 5.26 Tampilan *script* Sent_item.php
Sumber : Implementasi Sistem

Berdasarkan pada subbab 4.5.1 perancangan algoritma pesan terkirim, diimplementasikan seperti pada gambar 5.26. Gambar tersebut menunjukkan tampilan *script* halaman sent_item.php yang berfungsi menampilkan semua pesan sementara yang akan dikirimkan. Berikut penjelasan lebih rinci dari gambar 5.26 :

1. Baris ke- 2 : Proses melakukan koneksi dengan basis data
2. Baris ke- 3 : Proses menghilangkan error reporting
3. Baris ke- 4 : Method get pada offset halaman yang terkait dengan halaman inbox pada menu
4. Baris ke- 5-6 : Proses query pada basis data outbox
5. Baris ke- 9-25 : Proses membuat tabel dan melakukan perulangan menampilkan data dengan limit minimal 10
6. Baris ke- 26 : Proses query basis data untuk mengambil data inbox pada basis data untuk ditampilkan
7. Baris ke- 27-57 : Proses menampilkan data setelah dilakukan proses query data ke dalam tabel
8. Baris ke- 58-96 : Proses paginasi pada tabel

```
1. <html xmlns="http://www.w3.org/1999/xhtml">
2. <head>
3.  <meta  http-equiv="Content-Type"  content="text/html;
charset=utf-8" />
4. <title>Untitled Document</title>
5. </head>
6. <h1> SMS server running <h1>
7. <body>

8. <?php
9.   echo   "<meta   http-equiv='refresh'   content='2;
url=gateway_test.php '>";
10. ?>
11. <?php

//koneksi ke mysql dan db nya
12. $con = mysql_connect("localhost", "root", "");
13. mysql_select_db("smart_home");

// query untuk membaca SMS yang belum diproses
```

```
14. $query = "SELECT * FROM data_log where Status = '1'
order by Waktu_Sensor DESC";
15. $hasil = mysql_query($query);
16. while ($data = mysql_fetch_array($hasil))
17. {

18.  $state = $data['Status'];
19.  $time = $data['Waktu_Sensor'];
20.   $reply = "Rumah anda dalam bahaya pada pukul :
".$time.". Segera hubungi tetangga terdekat !";
21.  echo "Rumah anda dalam bahaya pada pukul : ".$time.".
Segera hubungi tetangga terdekat !";

22. $user = mysql_query("select * from pengajar");

23. while($semua = mysql_fetch_array($user)) {
24.        $hp = $semua['hp'];
25.              $query3    =    "insert    into    outbox
(InsertIntoDB,SendingDateTime,DestinationNumber,TextDecode
d,SendingTimeOut,DeliveryReport,CreatorID)
                 values
(sysdate(),sysdate(),'$hp','$reply',sysdate(),'yes','syste
m')";
          $hasil3 = mysql_query($query3);
26.        }
27. }
28. ?>
29. </body>
30. </html>
```

Gambar 5.27 Tampilan *script* Gateway_test.php
Sumber : Implementasi Sistem

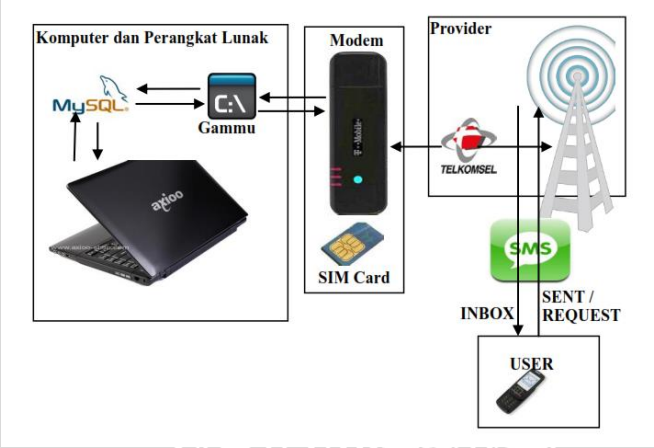
Berdasarkan pada subbab 4.5.1 perancangan algoritma SMS gateway, diimplementasikan seperti pada gambar 5.27. Gambar tersebut menunjukkan tampilan *script* halaman gateway_test yang berfungsi sebagai SMS gateway. Sehingga *script* ini dapat mengirimkan SMS ke user jika sensor node mengalami status bahaya. Data sebelumnya diambil dari tabel data_log, kemudian aplikasi akan memanggil data user dan selanjutnya semua data user dan data log akan dimasukkan sementara ke dalam tabel outbox dan selanjutnya akan dikirimkan melalui SMS. Berikut penjelasan lebih dari gambar 5.27 :

1. Baris ke- 9 : Melakukan *refresh* pada halaman yang sama.
2. Baris ke- 12-13 : Proses melakukan koneksi ke dalam basis data
3. Baris ke- 14-17 : Proses query basis data untuk membaca status data log.

4. Baris ke- 18-19 : Proses membuat variabel untuk membaca data status dan waktu sensor.
5. Baris ke- 20 : Proses membuat variabel reply SMS yang akan dikrimkan ke user
6. Baris ke- 22-24 : Proses query pada tabel pemilik rumah . Data yang diambil adalah nomor hp pemilik rumah.
7. Baris ke- 25 : Proses query untuk mengirimkan SMS. Query ini untuk memasukkan variabel reply dan nomor pemilik rumah ke dalam tabel outbox .

1.3 Implementasi SMS Gateway

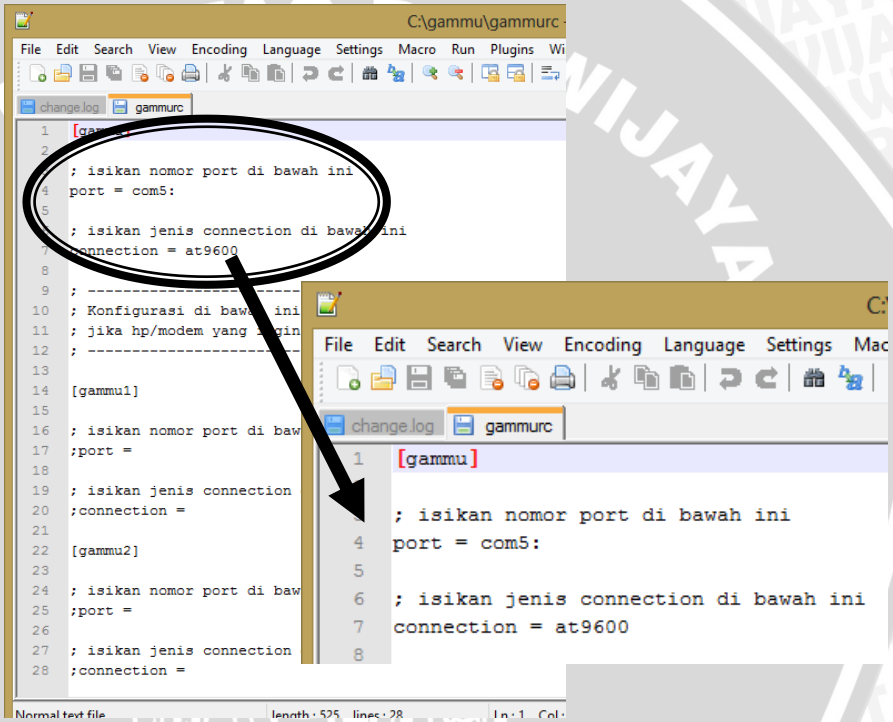
Berdasarkan subbab 4.4 perancangan SMS gateway, implementasi SMS gateway menggunakan Gammu. Gammu berperan sebagai *library* SMS gateway pada sistem ini.



Gambar 5.28 Diagram Alur SMS Gateway
Sumber : Implementasi Sistem

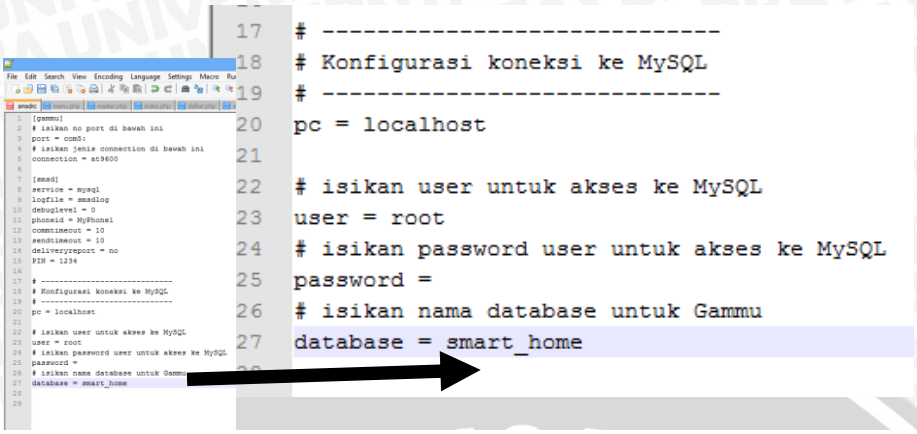
Berdasarkan pada subbab 4.4 perancangan SMS gateway, diimplementasikan seperti pada gambar 5.28. Gammu berperan sebagai *library* SMS gateway. Gammu terhubung dalam basis data aplikasi data sink. Pada basis data aplikasi data sink, SMS yang akan dikirim akan masuk ke tabel outbox. Data SMS dari tabel outbox akan diteruskan ke tabel sent item dan selanjutnya *library* Gammu akan mengirimkan SMS ke user pemilik rumah. Pada implementasi Gammu, langkah pertama yang harus dilakukan adalah menginstal *library* Gammu ke dalam windows. Versi Gammu yang digunakan adalah versi 2.57. Selanjutnya dilakukan aplikasi basis data pada Gammu.

Langkah pertama implementasi SMS gateway ialah mengimplementasi port yang akan digunakan modem pada Gammu *library*. Selain implementasi port, juga dilakukan implementasi *baudrate* modem dengan kecepatan at9600. Implementasi tersebut diimplementasikan seperti pada gambar 5.29. Pada gambar tersebut port yang digunakan pada modem SMS gateway adalah "com5". Sedangkan untuk baudratenya sebesar at9600.



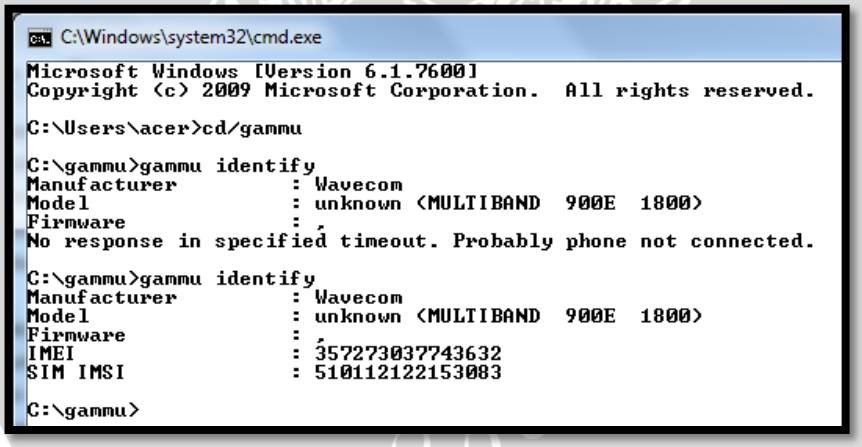
Gambar 5.29 Implementasi port Gammu dan baudrate modem pada gammurc
Sumber : Implementasi Sistem

Langkah kedua menentukan basis data yang dapat diakses oleh Gammu yang ditunjukkan pada gambar 5.30. Pada gambar tersebut ditunjukkan bahwa basis data yang digunakan adalah smart_home. Gammu akan mengirimkan data yang diperoleh ke dalam basis data yang telah dibuat sebelumnya. Implementasinya ditunjukkan pada gambar 5.30 :



Gambar 5.30 Implementasi basis data pada Gammu
Sumber : Implementasi

Setelah melakukan implementasi Gammu *library*, selanjutnya mengimplementasikan modem. Sebelum modem dapat digunakan pada aplikasi, peneliti harus menginstal *driver* modem terlebih dahulu. Setelah proses instalasi modem selesai, langkah selanjutnya menghubungkan modem dengan Gammu.



Gambar 5.31 Implementasi modem Gammu
Sumber : Implementasi

Implementasi modem dengan gammu ditunjukkan seperti pada gambar 5.31. Pada gambar tersebut terlihat bahwa modem SMS gateway dikenali dengan cara masuk ke *command promt* (cmd) pada Windows. Pada cmd pertama-tama peneliti mengetikkan “gammu”, kemudian mengetikkan “gammu identify”. Pada gambar tersebut modem telah dikenali dengan Wavecom sebagai manufakturnya. Agar modem dapat bekerja dengan Gammu, langkah selanjutnya menjalankan

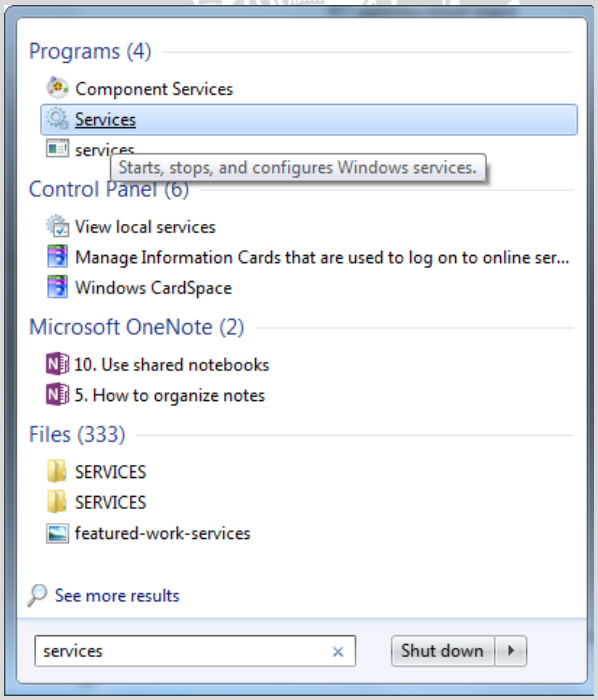
Gammu service. Masih pada cmd, peneliti mengetikkan “gammu-SMSd.exe –c SMSdrc –i” untuk menjalankan SMSdrc sebagai service.

```
Manufacturer          : Wavecom
Model                 : unknown (MULTIBAND  900E  1800)
Firmware              : ,
IMEI                  : 357273037743632
SIM IMSI              : 510112122153083

C:\gammu>gammu-smsd.exe -c smsdrc -i
```

Gambar 5.32 Implementasi Gammu service
Sumber : Implementasi

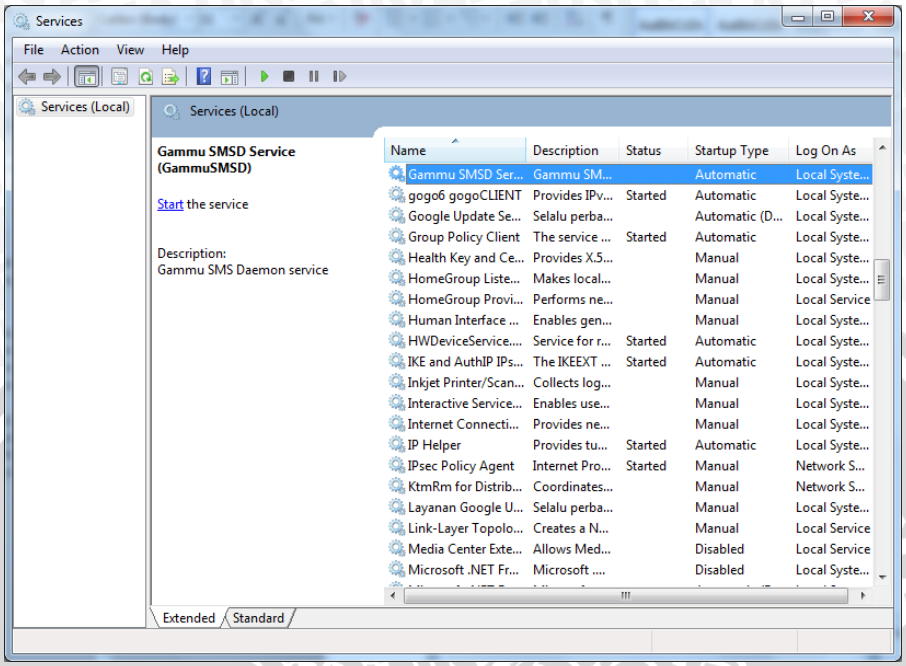
Implementasi gammu service ditunjukkan seperti pada gambar 5.32. Pada gambar tersebut gammu service akan berjalan jika service pada windows telah diaktifkan. Cara mengaktifkannya dengan cara mengetikkan “service” pada start menu, kemudian muncul seperti pada gambar 5.33 :



Gambar 5.33 Menjalankan Gammu service
Sumber : Implementasi

Implementasi menjalankan gammu service ditunjukkan seperti pada gambar 5.33. Setelah muncul jendela seperti gambar tersebut, selanjutnya menjalankan gammu service dengan cara mengklik service. Selanjutnya akan muncul jendela

seperti pada gambar 5.34. Setelah gambar tersebut muncul, selanjutnya mengklik start pada gammu SMSd. Service gammu sudah dapat berjalan.



Gambar 5.34 Menjalankan Gammu Service pada Windows
Sumber : Implementasi

Setelah gammu service diaktifkan, langkah selanjutnya dilakukan pengujian yang akan dibahas pada bab selanjutnya.

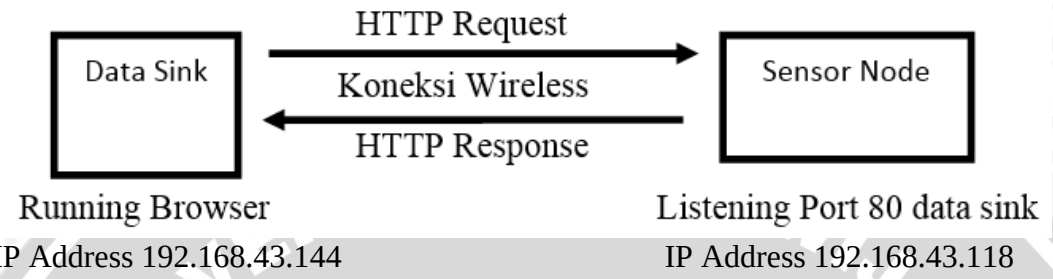
1.4 Implementasi Komunikasi Data

Implementasi komunikasi data mengimplementasikan gabungan dari implementasi sensor node dengan implementasi data sink. Selain itu juga mengimplementasikan data sink dengan SMS gateway. Pada implementasi komunikasi data, hal yang diimplementasikan adalah proses komunikasi data antar sensor node dengan data sink, kemudian data sink dengan sensor node.

1.4.1 Implementasi Komunikasi Data Sensor Node dengan Data Sink

Pada implementasi sensor node sebelumnya, sensor node sudah dapat terhubung dengan data sink dan *Acces Point* yang bernama "tes". Selanjutnya diimplementasikan komunikasi data antar sensor node dengan data sink. Data yang dikirimkan dari data sink adalah data yang diperoleh dari sensor.

Pada tahap implementasi sebelumnya, sensor node sudah mendapatkan alamat IP dari DNS *Access Point*. Kemudian alamat IP dari sensor node diakses dengan browser pada windows. Browser yang digunakan adalah Mozilla Firefork. Diagram blok komunikasi data sink dengan sensor node ditunjukkan pada gambar 5.35 :

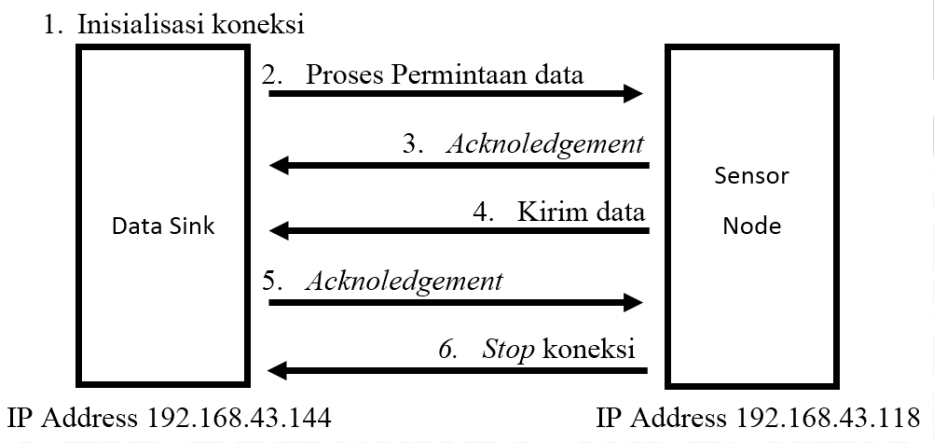


Gambar 5.35 Diagram Blok Komunikasi Data Sensor Node dan Data Sink
Sumber : Implementasi Sistem

Implementasi komunikasi data sensor node dengan data sink ditunjukkan pada gambar 5.35. Berdasarkan gambar tersebut dapat dijelaskan bahwa secara umum proses komunikasi data melibatkan komponen data yaitu :

1. Alamat IP tujuan sensor node dan data sink
2. Jenis Protokol yang digunakan adalah HTTP
3. Data yang dikirimkan
4. Port sensor node dan data sink

Dalam proses pengiriman data HTTP antara data sink dan sensor node, terbagi menjadi beberapa sesi pengiriman yang ditunjukkan pada gambar 5.36 :

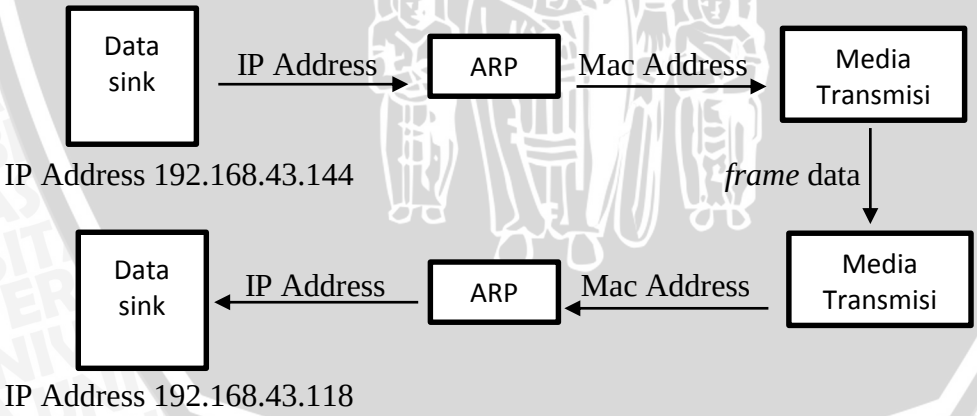


Gambar 5.36 Diagram Blok Alur Pengiriman Data Sensor Node dan Data Sink
Sumber : Implementasi Sistem

Proses pengiriman data dengan protokol HTTP antara sensor node dengan data sink ditunjukkan pada gambar 5.36. Proses pengiriman data pada gambar tersebut adalah sebagai berikut :

1. Inisialisasi koneksi dilakukan oleh data sink
2. Proses permintaan data dari sensor node (GET / HTTP /1.1)
3. Proses pengakuan (*Acknoledgement*) dari sensor node bahwa permintaan sudah diterima
4. Proses pengiriman data dari sensor node ke data sink
5. Proses pengakuan (*Acknoledgement*) dari data sink mengenai data yang dikirimkan oleh sensor node
6. Penghentian koneksi jika data sudah seluruhnya dikirim dari sensor node ke data sink

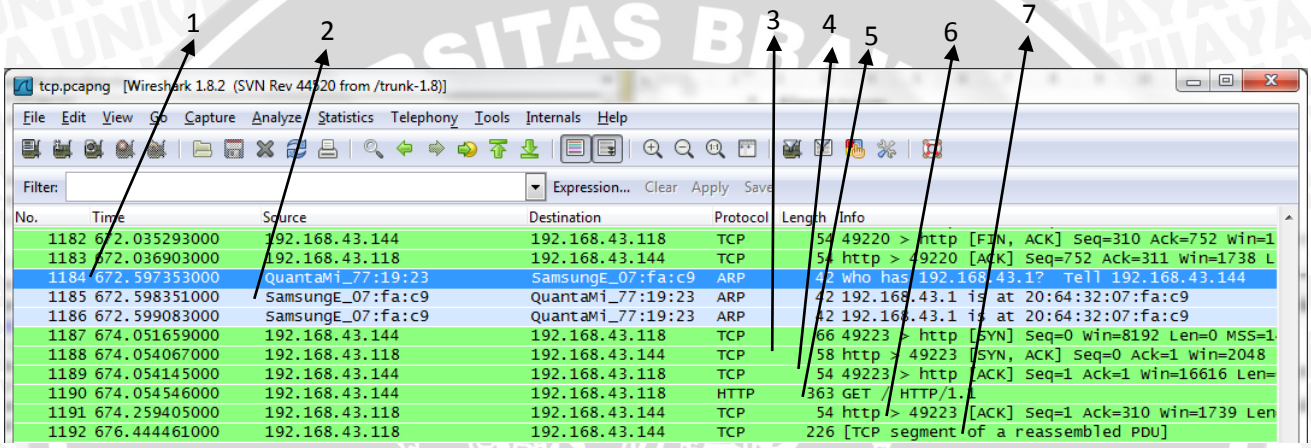
Proses komunikasi data dari sensor node ke data sink terjadi secara *wireless*. Proses pertukaran data tersebut dapat dilihat dan dianalisa dengan mengunakan tools *free Software* yaitu Wireshark. Dari Wireshark dapat terlihat ada beberapa transmisi pertukaran data yang dibedakan berdasarkan warna font di dalam tabel wireshark.



Gambar 5.37 Diagram Blok Alur Proses Koneksi HTTP Sensor Node dan Data Sink
Sumber : Implementasi Sistem

Implementasi proses HTTP sensor node dengan data sink ditunjukkan pada gambar 5.37. Berdsarkan gambar tersebut dapat dijelaskan bahwa pada saat proses koneksi HTTP antara data sink dengan sensor node, proses transmisi *Address*

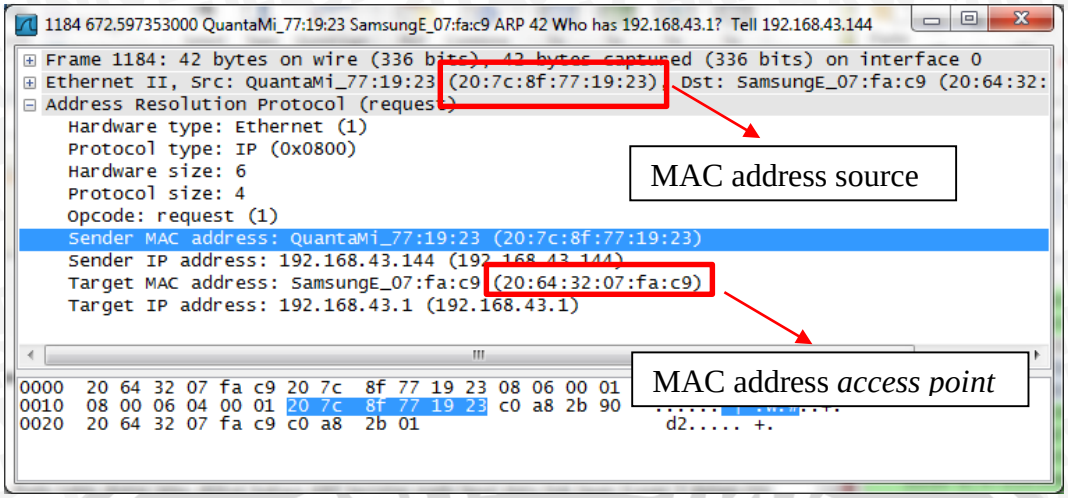
Resolution Protocol (ARP) harus dilakukan dahulu sebelum proses transmisi HTTP dimulai. ARP bertanggung jawab dalam melakukan perubahan alamat IP ke dalam alamat *Media Access Control* (MAC Address). Ketika sebuah aplikasi yang menggunakan protokol TCP IP yang mengakses sebuah *host*, maka alamat IP yang dimiliki ini harus diterjemahkan dahulu ke dalam MAC Address. Hal ini bertujuan agar *frame-frame* data dapat diteruskan ke tujuan dan diletakkan di atas media transmisi. Berikut hasil *capturing* komunikasi data sensor node dengan data sink dengan menggunakan Wireshark yang ditunjukkan pada gambar 5.38:



Gambar 5.38 Hasil Capture Komunikasi Data Sensor Node dan Data Sink
Sumber : Implementasi

Hasil *capture* komunikasi data sensor node dan data sink ditunjukkan pada gambar 5.38. Pada gambar terdapat beberapa protokol yang memproses komunikasi data. Protokol tersebut adalah ARP, TCP dan HTTP. Berikut penjelasannya :

1. Pada gambar 5.38 yang ditunjukkan oleh anak panah nomor 1, dapat dilihat pada kolom source bahwa kolom tersebut terdapat data " QuantaMi_77:19:23 " dengan MAC Addres seperti yang ditunjukkan pada gambar 5.39 :

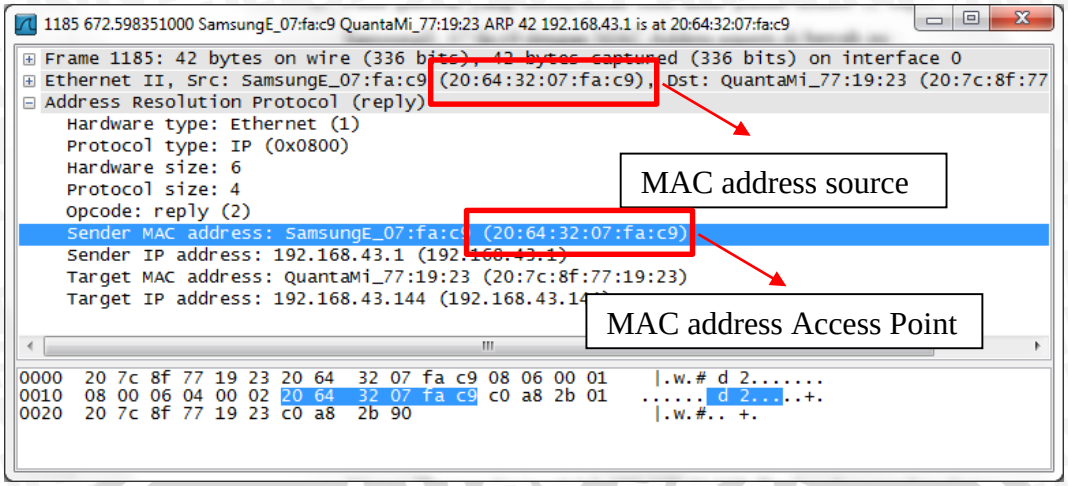


Gambar 5.39 Hasil Capture ARP Komunikasi Data Sensor Node dan Data Sink
Sumber : Implementasi

Hasil capture ARP komunikasi data sensor node dan data sink ditunjukkan pada gambar 5.39. Berdasarkan gambar tersebut MAC address Source adalah (20:7c:8f:77:19:23). Ini merupakan MAC address pada data sink. Data sink akan melakukan komunikasi HTTP pada sensor node. Data sink sebelumnya sudah melakukan koneksi pada *Access Point* dengan MAC Address (20:64:32:07:fa:c9). Data sink menginformasikan bahwa " *Who has 192.168.43.1? Tell 192.168.43.144*" , hal ini berarti data sink mengumumkan bahwa komputer yang memiliki MAC Address (20:7c:8f:77:19:23) adalah 192.168.43.144.

Data sink menanyakan komputer dengan MAC Address berapa yang memiliki IP 192.168.43.1 (IP Gateway) kepada *client* di jaringan dengan IP 192.168.43.xx. Data sink dalam implementasi ini akan melakukan komunikasi data dengan sensor node yang melalui Gateway.

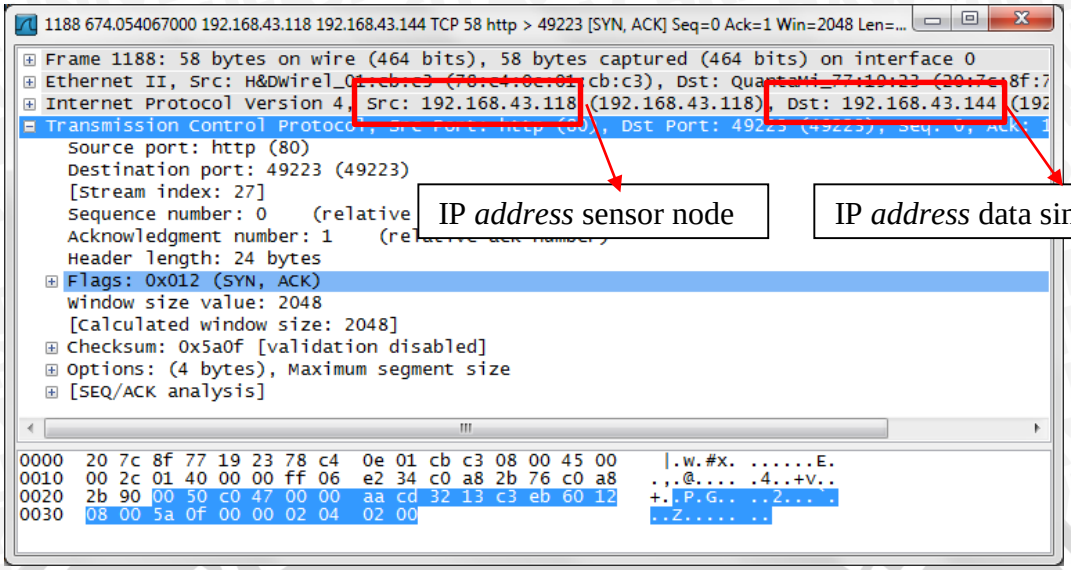
2. Pada gambar 5.38 yang ditunjukkan oleh anak panah nomor 2, dapat dilihat SamsungE_07:fa:c9 dengan MAC Addres seperti yang ditunjukkan pada gambar 5.40 :



Gambar 5.40 Hasil Capture ARP Komunikasi Data Sensor Node dan Data Sink
Sumber : Implementasi Sistem

Hasil *capture* ARP komunikasi data sensor node dan data sink ditunjukkan pada gambar 5.40. Berdasarkan gambar tersebut MAC *address* Source adalah (20:64:32:07:fa:c9). Ini merupakan MAC *address* pada *Access Point* sekaligus Gateway. *Access Point* yang digunakan adalah HP Samsung dengan tipe S2. *Access Point* menginformasikan bahwa komputer dengan MAC *address* (20:64:32:07:fa:c9) adalah komputer dengan IP 192.168.43.1 . Komputer dengan MAC *address* tersebut adalah *access point. Access Point* juga berperan sebagai Gateway.

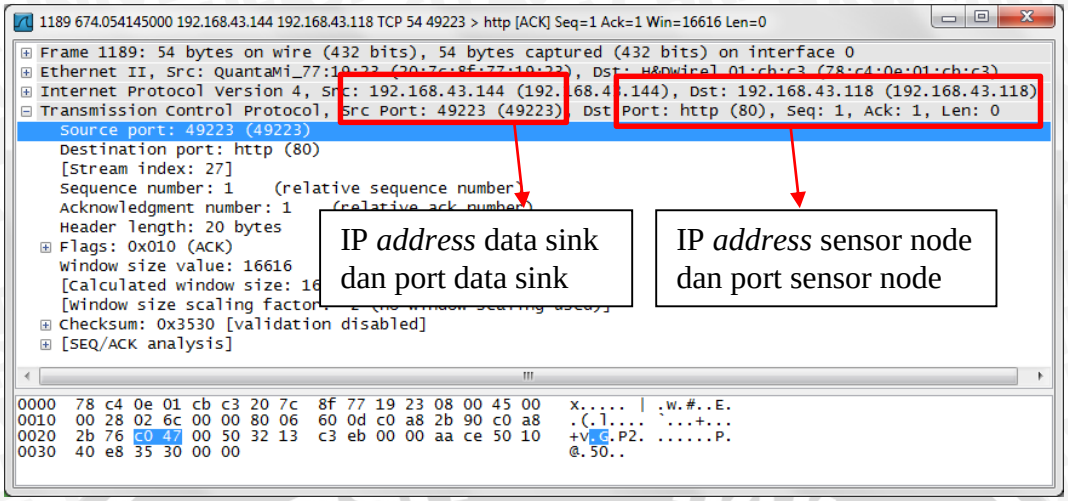
3. Pada gambar 5.38 yang ditunjukkan oleh anak panah nomor 3 adalah proses request HTTP. Hasil *capture* HTTP komunikasi data sensor node dan data sink ditunjukkan pada gambar 5.41. Pada gambar tersebut terjadi proses respond dari request yang berasal dari dari sensor node (192.168.43.118) ke data sink (192.168.43.144).



Gambar 5.41 Hasil Capture HTTP Komunikasi Data Sensor Node dan Data Sink
Sumber : Implementasi

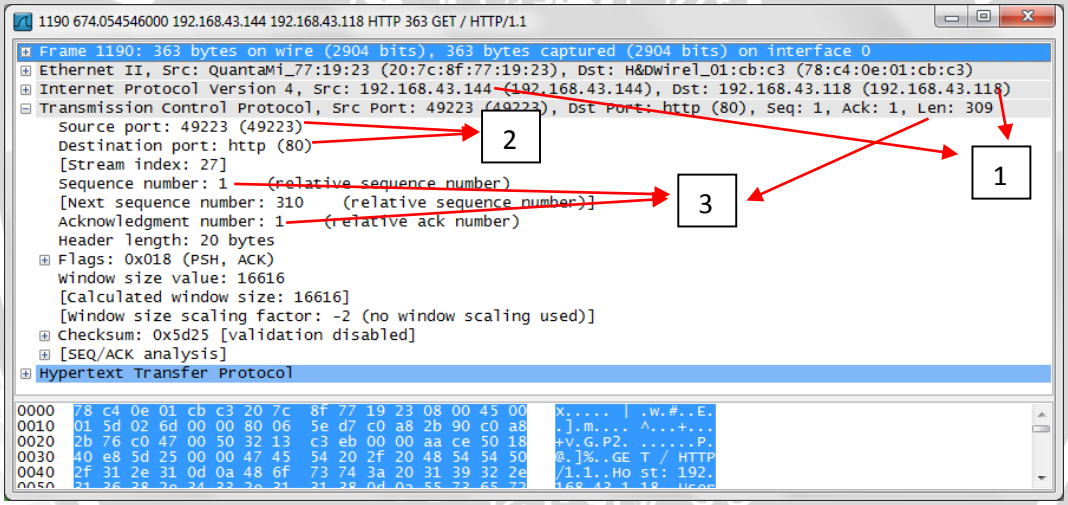
Proses tersebut melalui port 80 menuju port 49223 dengan nomor sequence 0, dan acknowledge 1 (Acknowledge Request(1) + Len Request(0)). Hal ini berarti request dari data sink sudah diterima oleh sensor node secara baik dan sensor node memberitahukan ke pada data sink.

4. Pada gambar 5.38 yang ditunjukkan oleh anak panah nomor 4 terjadi proses request yang berasal dari data sink (192.168.43.144) ke sensor node (192.168.43.118). Proses tersebut melalui port 49223 menuju port 80 dengan nomor sequence 1, dan acknowledge 1 (Acknowledge Request(1) + Len Request(0)). Hal ini berarti request dari data sink sudah diterima oleh sensor node secara baik dan sensor node memberitahukan ke pada data sink. Semua proses tersebut ditunjukkan pada gambar 5.42 :



Gambar 5.42 Hasil Capture HTTP Komunikasi Data Sensor Node dan Data Sink
Sumber : Implementasi

5. Pada gambar 5.38 yang ditunjukkan oleh anak panah nomor 5 adalah proses untuk mendapatkan dokumen dari sensor node.



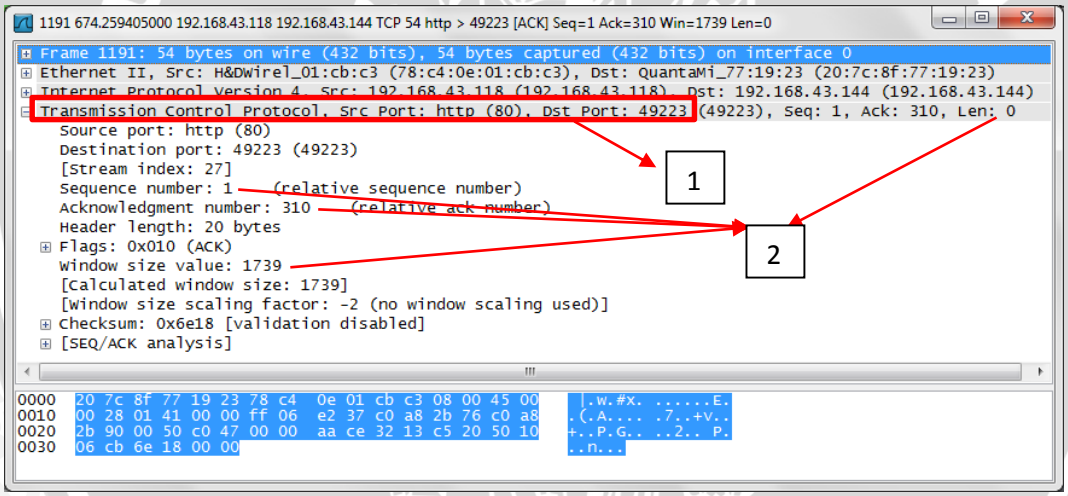
Gambar 5.43 Hasil Capture HTTP GET Komunikasi Data Sensor Node dan Data Sink
Sumber : Implementasi

Berdasarkan gambar 5.43 proses yang terjadi pada hasil *capture* HTTP GET komunikasi data sensor node dan data sink adalah :

1. Berdasarkan gambar 5.43 yang ditunjukkan oleh anak panah nomor 1 adalah proses Data sink mengirimkan HTTP request (192.168.43.144) kepada sensor node (192.168.43.118)

2. Berdasarkan gambar 5.43 yang ditunjukkan oleh anak panah nomor 2 adalah *Request* dikirimkan dengan menggunakan Protocol Transport TCP dari nomor port 49223 menuju port 80.
3. Dengan menggunakan Application Layer protocol HTTP
4. Info yang dikirimkan ke sensor node adalah GET/HTTP/1.1 yang berarti permintaan data sink untuk mendapatkan dokumen root (GET /) dari sensor node dengan menggunakan protocol HTTP/1.1
5. Berdasarkan gambar 5.43 yang ditunjukkan oleh anak panah nomor 3 adalah Sequence number = 1, Acknowledge Number = 1 dan Len (Panjang data) = 309

6. Pada gambar 5.38 ditunjukkan oleh anak panah nomor 6 adalah proses response dari sensor node yang memperoleh request dari data sink untuk mendapatkan dokumen.



Gambar 5.44 Komunikasi Data Sensor Node dan Data Sink
Sumber : Implementasi

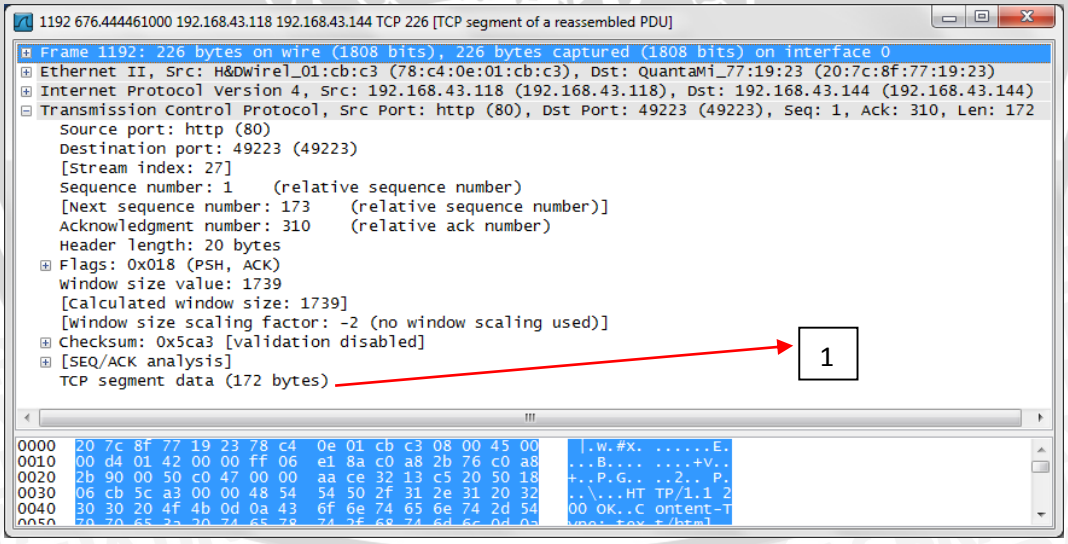
Proses yang terjadi pada gambar 5.44 hasil *capture* komunikasi data sensor node dan data sink adalah :

1. Sensor node mengirimkan response kepada data sink
2. Berdasarkan gambar 5.44 yang ditunjukkan oleh anak panah nomor 1 adalah proses request yang dikirimkan dengan menggunakan Protocol Transport TCP dari nomor port 80 menuju

port 49223 . Proses ini menggunakan Application Layer protocol TCP

3. Berdasarkan gambar 5.44 yang ditunjukkan oleh anak panah nomor 2 adalah proses info yang dikirimkan ke data sink adalah Seq = 1, Ack = 310, Win = 1739, Len = 0.

7. Pada gambar 5.38 yang ditunjukkan oleh anak panah nomor 7 adalah proses pengiriman data yang terjadi setelah proses *handshaking connection* antara sensor node dengan data sink. Proses ini biasanya berulang kali terjadi berdasarkan dari besar data yang dikirimkan dari sensor node ke data sink. Data sink akan melakukan *acknowledgment* dari setiap data yang dikirimkan oleh sensor node . Hal ini bertujuan untuk mengecek apakah data yang diterima sesuai dengan data yang dikirimkan (*check sum*). Jika paket data yang dikirimkan terjadi kerusakan maka data sink akan mengirimkan permintaan kepada sensor node untuk mengirimkan ulang data yang rusak atau hilang tersebut. Proses komunikasi data sensor node dengan data sink setelah proses *handshaking connection* ditunjukkan dalam gambar 5.45.



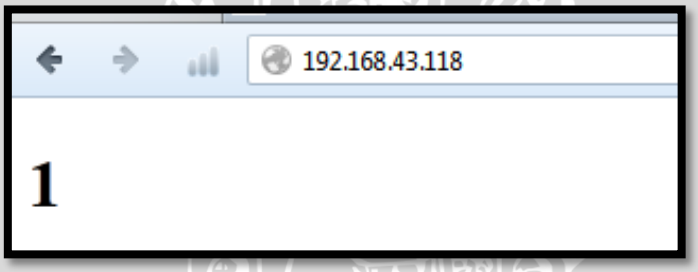
Gambar 5.45 Komunikasi Paket Data Unit Sensor Node dan Data Sink
Sumber : Implementasi

Proses yang terjadi pada gambar 5.45 hasil capture komunikasi paket data unit sensor node dan data sink adalah sebagai berikut :

IP sensor node (192.168.43.118) mengirimkan response ke data sink (192.168.43.114) dengan menggunakan transport protocol TCP dari port 80 menuju port 49223. TCP adalah *Transmission Control Protocol* yang berada pada layer *transportation.* Hal ini seperti yang dijelaskan di dalam dasar teori subbab 2.4 OSI Layer.

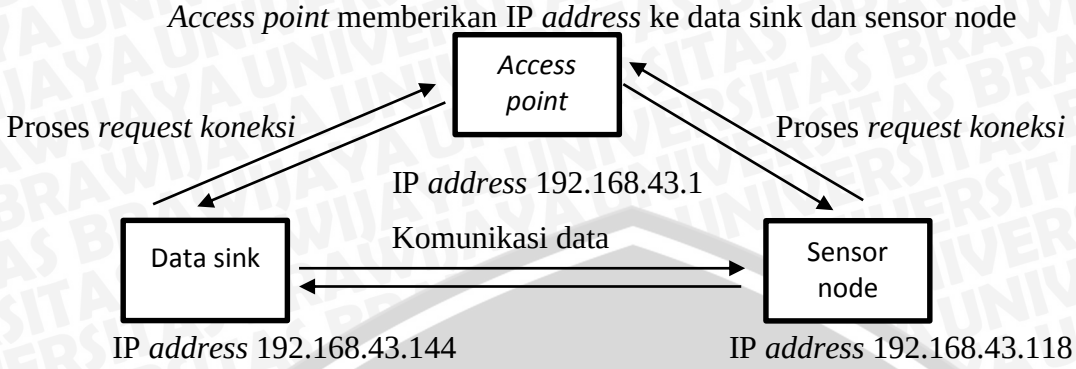
Informasi yang dikirimkan adalah TCP segment of a *reassembled* PDU. Sensor node mengirimkan *fragment* data paket menuju data sink dengan besar *fragmen* sebesar 172 *byte* untuk tiap *fragmen* data paket . Hal ini ditunjukkan dalam gambar 5.45 yang ditunjukkan oleh anak panah nomor 1.

Setelah melalui proses 1 hingga 7, maka data yang dikirimkan dari sensor node dapat dilihat pada browser. Pada browser diketikkan alamat salah satu sensor node, yaitu " 192.168.43.118 " yang ditunjukkan pada gambar 5.46.



Gambar 5.46 Alamat IP sensor node diakses dari browser
Sumber : Implementasi Sistem

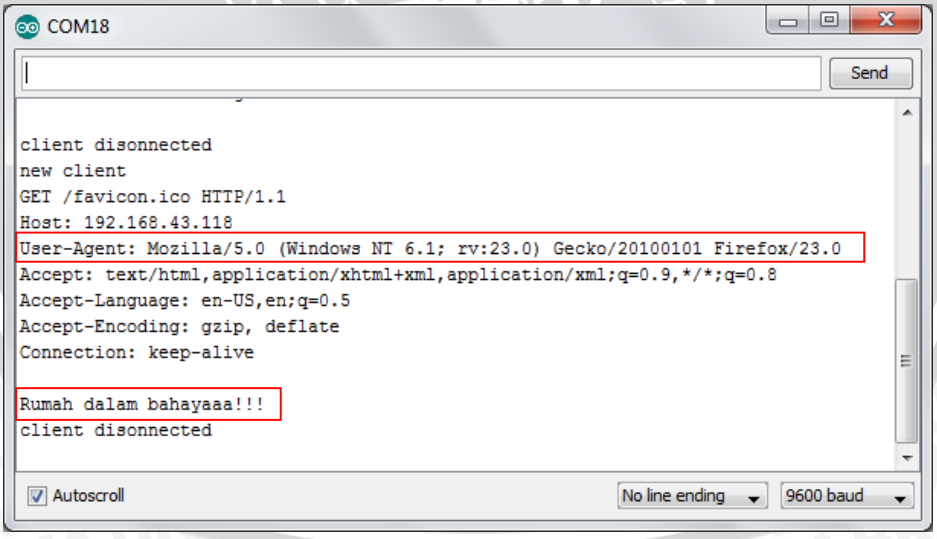
Alamat IP yang dituju adalah 192.168.43.118. Berdasarkan gambar 5.46 alamat IP dari sensor node berhasil diakses. Pada browser menunjukkan angka satu, hal ini berarti sensor node mendeteksi adanya bahaya atau sensor node mendeteksi adanya seseorang yang berada pada jangkauan sensor. Proses komunikasi data ini terjadi pada protokol HTTP dan TCP IPv4. Proses komunikasi data ini ditangkap oleh aplikasi Wireshark yang sudah dibahas sebelumnya. Diagram blok proses komunikasi data ini ditunjukkan pada gambar 5.47.



Gambar 5.47 Proses komunikasi data sensor node dan data sink
Sumber : Implementasi Sistem

Proses komunikasi data sensor node dan data sink ditunjukkan dalam gambar 5.47. Data sink dan sensor node memperoleh IP dari *Access point*. Selanjutnya data sink dan sensor node dapat langsung saling berkomunikasi.

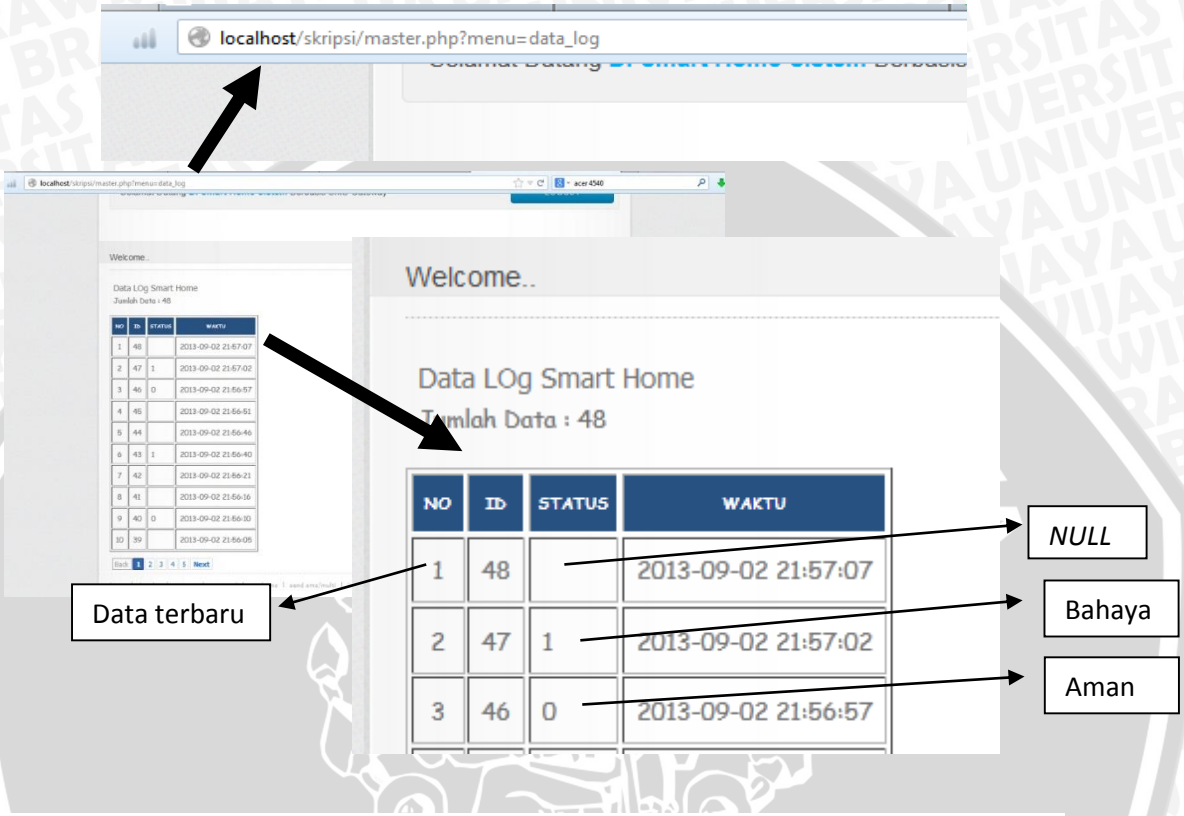
Kondisi sensor node yang telah diakses oleh data sink menggunakan browser Mozilla ditunjukkan dalam gambar 5.48. Untuk mengetahui kondisi sensor node tersebut, digunakan kabel serial monitor antara sensor node dengan data sink. Aplikasi yang digunakan untuk melihat status serial sensor node digunakan aplikasi Arduino IDE versi 1.0.4.



Gambar 5.48 Serial monitor sensor node
Sumber : Implementasi Sistem

Serial monitor sensor node ditunjukkan dalam gambar 5.48. Gambar tersebut menunjukkan bahwa sensor node sudah ada yang mengakses dengan browser Mozilla dan platform Windows. Pada gambar 5.46 browser menunjukkan

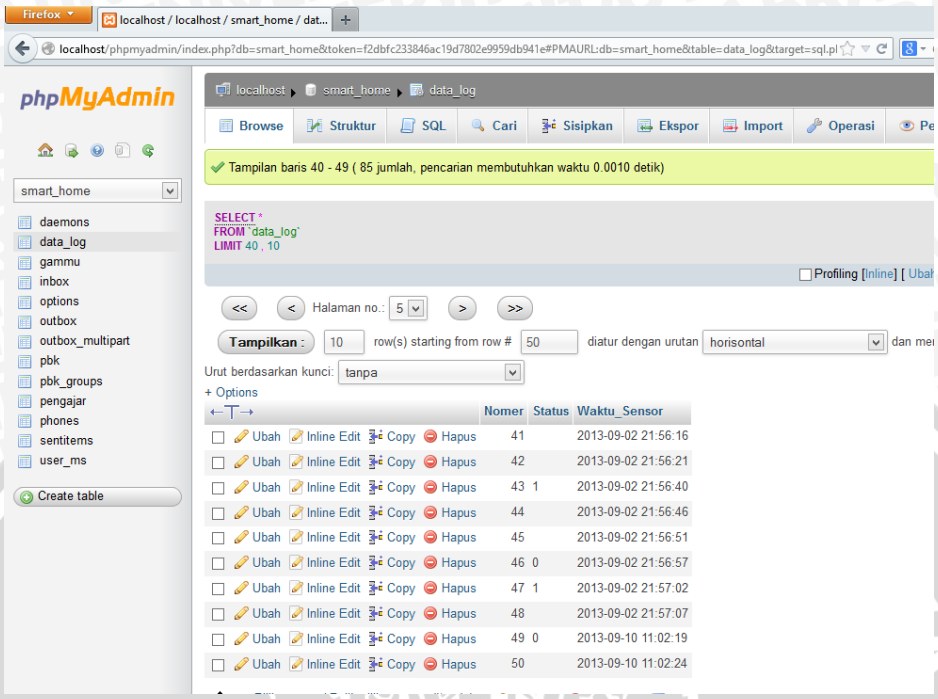
angka 1, maka pada serial monitor juga menunjukkan status bahaya. Hal ini menunjukkan bahwa sensor node dan data sink sudah terjadi komunikasi. Data dari sensor node ini nantinya akan masuk ke dalam data sink.



Gambar 5.49 Basis data sensor node
Sumber : Implementasi Sistem

Data yang dikirimkan dari sensor node masuk ke dalam basis data aplikasi data sink yang ditunjukkan dalam gambar 5.49. Data yang ditampilkan pada data sink sebelumnya sudah masuk di dalam basis data. Terdapat tiga status pada sensor, pertama dengan status *NULL* yang berarti sensor tidak membaca obyek sensor. Kedua dengan status 1 yang berarti bahaya. Sedangkan status 0 yang berarti aman. Status aman ini diperoleh setelah sensor mengalami status bahaya. Sehingga secara *default* status 1 akan berubah menjadi 0 jika sensor sudah tidak membaca obyek sensor.

Data dari sensor node langsung masuk ke dalam basis data dan langsung mengupdate tabel data log. Tabel data log inilah yang ditampilkan pada aplikasi data sink. Basis data tabel data log ditunjukkan dalam gambar 5.50.



Gambar 5.50 Implementasi Tabel Data Log
Sumber : Implementasi Sistem

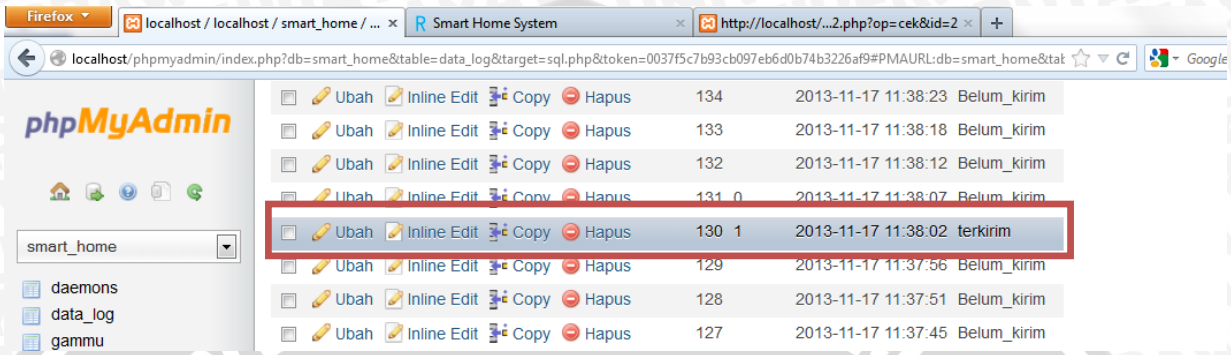
Tabel data log di dalam basis data ditunjukkan dalam gambar 5.50. Basis data ini menggunakan bahasa pemrograman MySql. Alur data pada basis data tersebut sudah dijelaskan pada diagram blok gambar 5.6 subbab 5.2.1.

5.4.2 Implementasi Komunikasi Data Sink dengan SMS Gateway

Pada implementasi SMS Gateway sebelumnya pada subbab 5.3, gammu service sudah diimplementasikan pada Windows dan berjalan pada Windows. Selanjutnya diimplementasikan komunikasi data antar SMS Gateway dengan data sink.

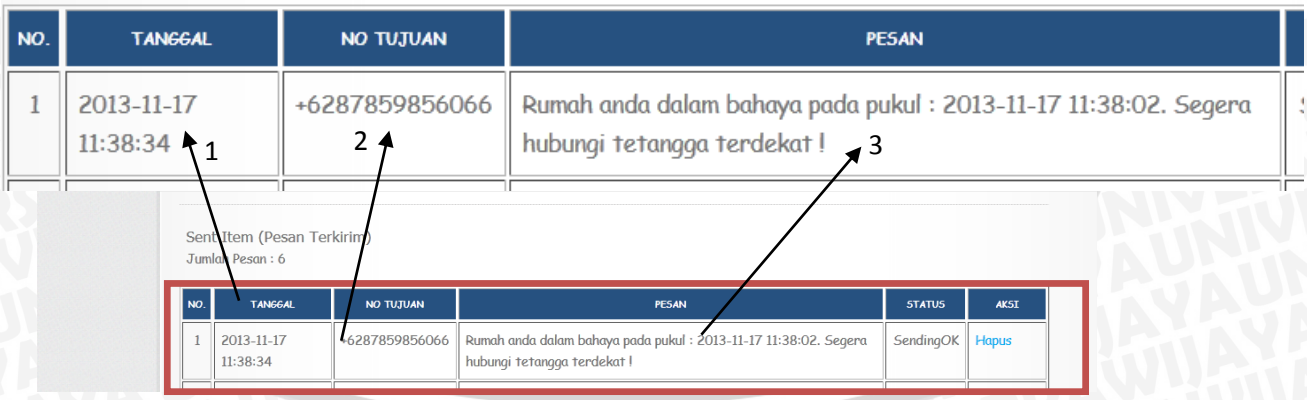
Data yang dikirimkan dari data sink adalah data yang diperoleh dari sensor node dengan status bahaya atau ditunjukkan dengan angka "1". Basis data sensor pada aplikasi data sink ditunjukkan pada gambar 5.51 basis data data log. Pada tabel data log terdapat status kirim. Status kirim adalah status dari sensor apakah data sensor ini sudah dikirimkan atau belum ke user. Data status kirim secara *default* adalah "belum_kirim". Namun data yang dikirimkan adalah data sensor yang bernilai "1". Sedangkan data yang tidak dikirimkan bernilai "0" atau *NULL*. Data

yang bernilai "1" yang sudah dikirimkan akan berubah statusnya menjadi "terkirim", sedangkan data yang lainnya akan berubah statusnya menjadi " tunggu "



Gambar 5.51 Basis Data Data Log
Sumber : Implementasi Sistem

basis data data log ditunjukkan dalam gambar 5.51. Berdasarkan gambar tersebut tanda yang berwarna merah adalah data sensor dengan status "1", data ini sudah berubah statusnya menjadi "terkirim". Data ini sudah terkirim dan di proses pada tanggal " 17 November 2013 jam 11:38:02". Selanjutnya data sensor ini akan diteruskan ke tabel outbox, selanjutnya data akan diteruskan ke tabel sent items sesuai dengan diagram blok basis data pada gambar 5.6.



Gambar 5.52 Halaman Pesan Terkirim
Sumber : Implementasi Sistem

Halaman pesan terkirim ditunjukkan dalam gambar 5.52. Pada gambar tersebut SMS sudah dikirimkan ke nomor tujuan "08785985066" yang ditunjukkan pada gambar anak panah nomor 2. Waktu kirim data ditunjukkan pada gambar anak

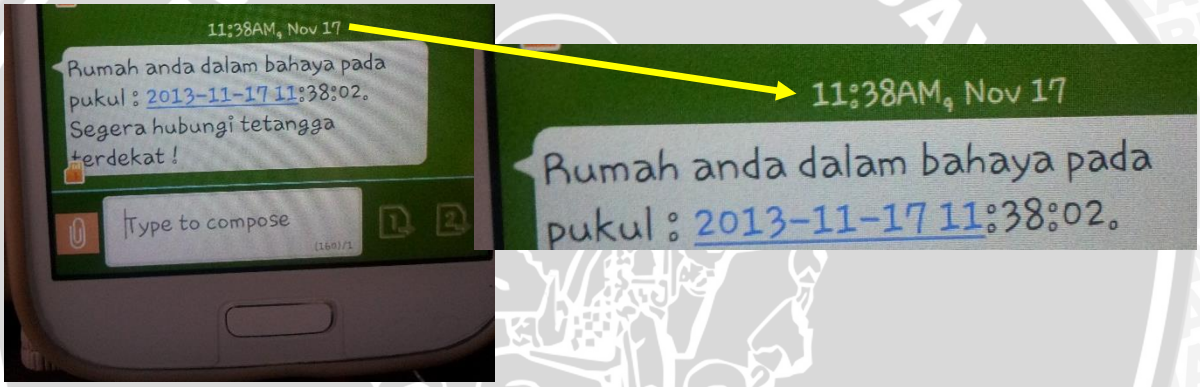
panah nomor 1 pada tanggal 17 November 2013 jam 11:38:34. Sedangkan pada waktu tangkap sensor adalah tanggal 17 November 2013 jam 11:38:02 yang ditunjukkan pada gambar anak panah nomor 3. Selisih waktu dari data sensor adalah sebagai berikut :

Tabel 5.14 Selisih waktu kirim data sensor

Waktu kirim	Waktu data sensor	Selisih waktu
11:38:34	11:38:02	32 detik

Sumber : Implementasi Sistem

Selisih waktu kirim data sensor ditunjukkan pada tabel 5.14. Berdasarkan tabel tersebut selisih waktu adalah sebesar 32 detik. Hal ini cukup efektif dalam memberikan peringatan dini kepada pemilik rumah.



Gambar 5.53 Hp User Setelah Mendapatkan Kiriman SMS Dari Data Sink
Sumber : Implementasi Sistem

HP user setelah mendapatkan kiriman SMS dari data sink ditunjukkan dalam gambar 5.53.Berdasarkan gambar tersebut data dikirmkan juga pada waktu yang sama, yaitu pada jam 11:38. Sehingga data dapat dikrimkan dengan cepat oleh sensor node.