Makalah Seminar Tugas Akhir

APLIKASI KARTU CERDAS TANPA KONTAK (*CONTACTLESS SMARTCARD*) PADA SISTEM PARKIR BERLANGGANAN

Subiono Ahmad[1], Aghus Sofwan[2], R Rizal Isnanto[2]
Jurusan Teknik Elektro Fakultas Teknik Universitas Diponegoro

Abstrak - Sebagaimana diketahui, kebanyakan fasilitas umum saat ini terutama di kota-kota besar seperti pusat perbelanjaan, stasiun, rumah sakit, hotel, dan sebagainya sudah menggunakan sistem parkir sebagai sarana untuk memudahkan transaksi parkir dan untuk memberikan jaminan keamanan bagi si pengguna jasa parkir. Program aplikasi sistem parkir merupakan sebuah aplikasi perangkat lunak yang digunakan untuk mengelola transaksi parkir baik di pintu masuk maupun pintu keluar. Sistem parkir yang digunakan saat ini pada umumnya masih menggunakan sistem transaksi online yang membutuhkan akses jarak jauh ke basisdata pada saat transaksi berlangsung. Melalui tugas akhir ini akan dibangun sebuah aplikasi sistem parkir yang menggunakan kartu cerdas tanpa kontak (*contactless smartcard*) sebagai media untuk bertransaksi.

Dalam pengembangan aplikasi sistem parkir ini, terlebih dahulu dilakukan perancangan sistem menggunakan diagram pemodelan (*UML*) untuk mengetahui proses bisinis dari aplikasi tersebut. Proses bisnis ini selanjutnya digunakan untuk menentukan kebutuhan dari sistem yang akan dibangun dengan menggunakan bahasa pemrograman Java™. Data yang mendukung dalam pengembangan aplikasi sistem parkir ini diperoleh dari hasil wawancara dengan narasumber, dan juga dari sumber lain seperti buku, internet, dan sebagainya.

Dengan dibangunnya sistem parkir menggunakan kartu cerdas, diharapkan dapat menjadi alternatif bagi sistem parkir pada saat ini yang mayoritas masih menggunakan struk parkir. Pada penerapannya, sistem parkir ini masih membutuhkan peran operator dalam pengoperasian dan perlu diperhatikan jarak pengaksesan kartu terhadap alat pengaksesnya (*interrogator*) kurang lebih adalah 3 cm. Kartu cerdas yang digunakan adalah Mifare® Classic dari PHILIPS™ dengan kapasitas memori sebesar 1KB.

Kata-kunci : sistem parkir, transaksi parkir, kartu cerdas tanpa kontak (*contactless smartcard*).

I. PENDAHULUAN

1.1 Latar Belakang Masalah

Program aplikasi sistem parkir merupakan sebuah aplikasi perangkat lunak yang digunakan untuk mengelola transaksi parkir baik di pintu masuk maupun pintu keluar. Jumlah transaksi atau jumlah kendaraan yang keluar masuk area parkir tidak sedikit sehingga memungkinkan terjadinya kesalahan transaksi yang akan merugikan pengelola parkir dengan nilai rupiah yang tidak sedikit.

Sebagaimana diketahui, sistem parkir pada umumnya seperti pada pusat perbelanjaan, stasiun, hotel dan sebagainya masih menggunakan sistem transaksi *online* yang membutuhkan akses jarak jauh ke basisdata pada saat transaksi berlangsung. Dengan menggunakan kartu cerdas tanpa kontak (*Contactless Smart Card*), diharapkan dapat menjadi alternatif bagi sistem parkir yang masih *online* dan menggunakan struk parkir sehingga transaksi dapat dilakukan secara *offline* karena pada kartu cerdas tanpa kontak sudah ditanamkan IC memori yang dapat digunakan untuk menyimpan informasi yang dibutuhkan.

1.2 Tujuan

Tujuan dari Tugas Akhir ini adalah untuk membangun sebuah aplikasi yang dapat mengakses kartu cerdas sehingga dapat dilakukan proses baca dan tulis pada kartu cerdas. Penerapan dari aplikasi ini adalah untuk sistem parkir berlangganan.

1.3 Batasan Masalah

Batasan masalah yang diberikan pada tugas akhir ini adalah sebagai berikut :

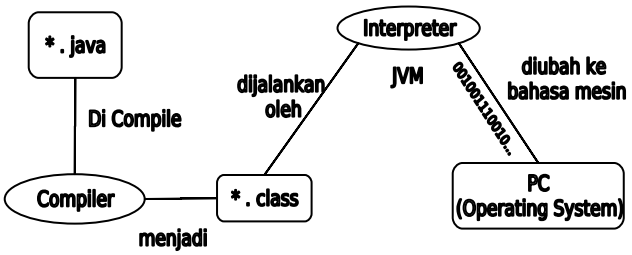
1. Tugas akhir ini dirancang menggunakan metode analisis dan perancangan berorientasi objek (*Object-Oriented Analysis and Design*) dan menggunakan bahasa pemodelan UML.
2. Aplikasi ini menggunakan Java Development Kit 1.6 sebagai bahasa pemrograman dan NetBeans 6.1 sebagai lingkungan pengembangan terintegrasi (*Integrated Developmnent Environment* - IDE).
3. Kartu cerdas yang digunakan adalah PHILIPS™ Mifare® RF Interface (ISO/IEC 14443 A) dan alat pembacanya adalah EHAG microengine 13,56 MHz Dual Reader Module.
4. Pada tugas akhir ini hal-hal yang berkaitan dengan perangkat keras pendukung seperti alat pembaca tidak akan diuraikan secara mendalam.

II. LANDASAN TEORI

2.1 Program Java

Java pertama kali dirilis secara resmi oleh perusahaan Sun Microsystem pada tahun 1996. Java menurut definisi dari Sun adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada PC *standalone* ataupun pada lingkungan jaringan. Java2 adalah generasi kedua dari Java Platform (generasi awalnya adalah Java Development Kit). Java berdiri di atas sebuah mesin

interpreter yang diberi nama Java Virtual Machine (JVM). JVM inilah yang akan membaca *bytecode* dalam file `.class` dari sebuah program sebagai representasi langsung program yang berisi bahasa mesin. Oleh karena itu bahasa Java sering disebut dengan bahasa pemrograman yang *portable*, sehingga dapat berjalan di sistem operasi apa saja, asalkan pada sistem operasi itu terdapat JVM. Gambar 2.2 menunjukkan alur eksekusi dari program java.



Gambar 2.2 Alur eksekusi program Java

Platform Java adalah kumpulan dari pustaka (*library*). Hingga saat ini terdapat tiga edisi dari platform Java teknologi terbaru, yaitu Java™2. Ketiga edisi tersebut adalah J2SE atau Java 2 Standard Edition, J2EE atau Java 2 Enterprise Edition dan J2ME atau Java 2 Micro Edition. J2SE lebih banyak ditujukan untuk desktop (*standalone*), sedangkan J2EE lebih cenderung untuk aplikasi-aplikasi *backend server*. J2ME sendiri merupakan kumpulan API (*Application Programming Interface*) yang memfokuskan diri pada konsumen atau sistem terintegrasi, mulai dari ponsel dan PDA.

2.1.1 Java 2 Standard Edition (J2SE)

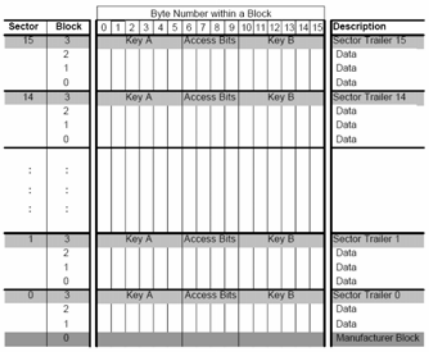
Java 2 Standard Edition (J2SE) adalah inti dari bahasa pemrograman Java untuk semua *platform*. JDK (Java Development Kit) adalah salah satu tool dari J2SE untuk mengkompilasi dan menjalankan program Java. Dalam satu paket instalasi JDK, terdapat tool untuk kompilasi dan juga JRE (Java Runtime Environment) atau sering juga dikatakan sebagai JVM untuk sistem operasi PC. Sampai saat ini JDK sudah mencapai versi 1.6.

J2SE biasanya digunakan untuk membangun aplikasi-aplikasi yang berjalan sendiri (*standalone*), seperti pengolah teks, permainan (*game*), pemutar lagu, dan sebagainya. Paket pustaka program java terbagi ke dalam 2 paket, yaitu paket `java.*` dan `javax.*`. Pustaka-pustaka inti (*core*) dar Java terdapat dalam paket `java.io, java.awt, java.math, java.net, java.sql` dan sebagainya. Tetapi terdapat pengecualian untuk paket pustaka inti seperti Swing, yang dimasukkan dalam paket `javax.*`. Paket `javax.*,` adalah kumpulan dari pustaka-pustaka tambahan yang dirilis oleh SUN Microsystem seperti *Java Communication API*, sebuah API untuk mengakses port serial dan paralel PC.

2.2 Kartu Cerdas (*Smart Card*)

2.2.1 Kartu Memori Rangkaian Terintegrasi (*Integrated Circuit Memory Card*).

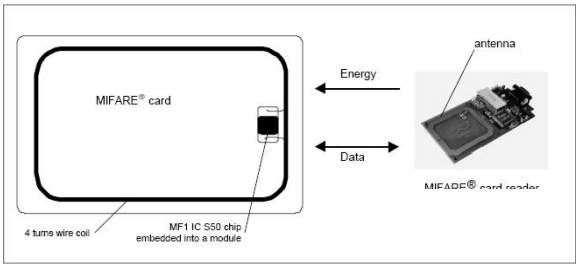
Kartu berkeping memori ini mampu menampung data berkisar dari 1 hingga 4 KB akan tetapi tidak memiliki prosesor yang digunakan untuk memanipulasi data pada kartu tersebut. Oleh karenanya kartu jenis ini bergantung pada alat pengakses kartu (*interrogator*) untuk dapat beroperasi dan jenis kartu ini sesuai untuk melaksanakan operasi yang tetap.



Gambar 2.3 Blok memori kartu cerdas dengan kapasitas 1 Kb

2.2.2 Kartu Cerdas tanpa Kontak (*Contactless Smart Card*).

Maksud tanpa kontak (*contactless*) adalah pengambilan data pada kartu dapat dilakukan atau alat pengakses dapat mengenali kartu tanpa terjadi kontak secara langsung dengan kartu pada jarak tertentu. Gambar 2.3 menunjukkan blok memori kartu cerdas dengan kapasitas 1 Kb dan Gambar 2.4 menunjukkan kartu cerdas tanpa kontak dan alat pengaksesnya.



Gambar 2.4 Kartu cerdas tanpa kontak dan alat pengakses kartu

2.3.2 *Interrogator Passive* RFId

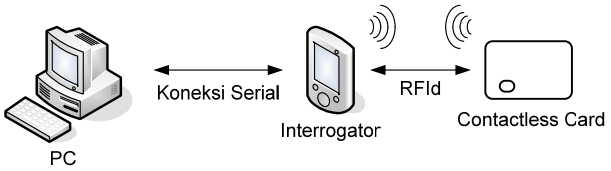
Sarana pendukung sistem berbasis RFId yang dikenal sebagai *Interrogator* adalah suatu alat pengisian data ke chip RFId (*Writer*) dan alat untuk membaca data yang terdapat dalam chip RFId (*Reader*). Dua macam fungsi tersebut dapat dikemas ke dalam satu buah alat baca dan tulis Untuk operasi statis alat ini pada umumnya berbentuk *desktop* atau bisa berupa suatu gerbang. Sementara untuk operasional di lapangan (*mobile*) alat ini bisa dalam bentuk genggam (*Handheld*).

Setiap alat mengikuti standar frekuensi tertentu, misalnya 13,56 MHz seperti yang digunakan pada tugas

akhir ini dan secara optimal hanya mendukung satu jenis standar ISO, yaitu ISO 15693 atau ISO 1444A/B.

III. PERANCANGAN SISTEM

3.1 Perancangan Perangkat Keras



Gambar 3.1 Perancangan perangkat keras

Penjelasan dari masing-masing blok pada Gambar 3.1 adalah sebagai berikut:

1. *Personal Computer* (PC)

   PC adalah pusat pengontrol pada aplikasi ini meliputi sistem registrasi, pengisian ulang, pengaturan tarif, data pengguna sistem, data pengguna teregistrasi dan data transaksi serta penagihan (*billing*). PC dapat memonitor operasi yang dilakukan oleh sistem, memberi perintah pada *interrogator* untuk membaca dan menuliskan data pada kartu (*Contactless Card*). PC terhubung melalui kabel serial dengan *interrogator*.
2. *Interrogator*

   Interrogator adalah suatu alat untuk pengisian data ke chip RFId yang terdapat pada kartu (*Contactless Card*) sekaligus untuk membaca data yang terdapat pada chip RFId. *Interrogator* ini berkomunikasi dengan chip RFId pada frekuensi 13,56 MHz.
3. *Contactless Card*

   Contactless Card terdiri dari dua bagian pokok yaitu, chip RFId dan antena (*coil*). Chip ini berfungsi untuk menyimpan data dari pengguna dan memiliki kapasitas memori sebesar 1 Kbyte yang terdiri dari 16 sektor. Masing-masing sektor terbagi menjadi 4 blok dan setiap bloknya dapat menampung data sebesar 16 byte. Data yang tersimpan pada chip ini dalam format heksadesimal.

3.2 Perancangan Perangkat Lunak

3.2.1 Perancangan Perangkat Lunak Sistem Admin

Agar perangkat lunak dapat bekerja sesuai dengan yang diinginkan, maka sebelumnya perlu dilakukan analisis kebutuhan terhadap perangkat lunak yang akan dirancang. Perangkat lunak diharapkan dapat memberikan fasilitas sebagai berikut :

1. PC dapat melakukan komunikasi dengan *interrogator*
2. Admin dapat melakukan *login* sesuai dengan nama dan kata sandinya.
3. Admin dapat mengubah data pada kartu melalui PC dengan *interrogator* sebagai perantaranya sehingga proses registrasi dan pengisian ulang dapat dilaksanakan.
4. Admin dapat melakukan pengaturan tarif.
5. Admin dapat membuat akun baru, mengubah dan menghapus data akun pada data pengguna sistem.
6. Admin dapat meninjau dan mencetak data pengguna yang teregistrasi dan data transaksi pada pintu masuk maupun pintu keluar.

3.2.2 Perancangan Perangkat Lunak Sistem Operator

Seperti pada perancangan perangkat lunak untuk PC admin, sebelum melakukan perancangan untuk PC operator terlebih dahulu dilakukan analisis kebutuhan yang diharapkan pada PC operator. Perangkat lunak diharapkan dapat memberikan fasilitas sebagai berikut :

1. PC dapat melakukan komunikasi dengan dengan *interrogator*
2. Operator dapat melakukan *login* sesuai dengan nama dan kata sandinya.
3. Operator dapat memantau proses transaksi.
4. Sistem dapat melakukan debet saldo pada kartu secara otomatis sesuai dengan tarif yang telah ditentukan.
5. Sistem dapat menyimpan dan menampilkan data transaksi.
6. Operator dapat melakukan transaksi manual apabila kartu tidak terbaca, kode akses tidak sesuai, atau saldo tidak mencukupi.

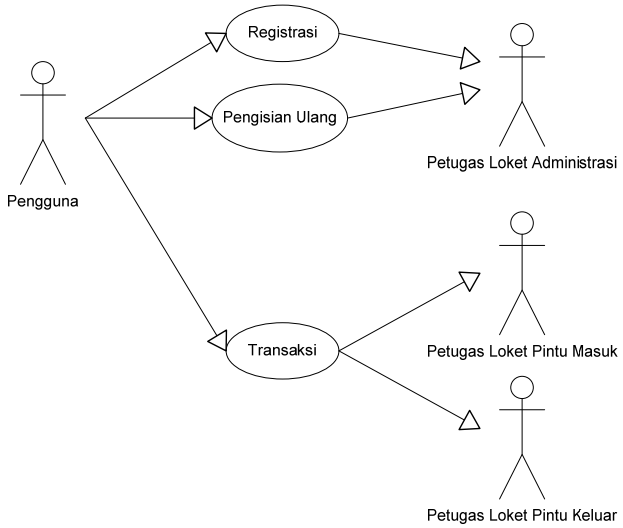
3.2.3 Perancangan Diagram *Use Case*

Berikut ini adalah *use case* dari Aplikasi Kartu Cerdas Tanpa Kontak (*Contactless Smartcard*) pada Sistem Parkir Berlangganan. Kata yang bercetak tebal menggambarkan calon objek sedangkan kata yang bercetak miring menggambarkan operasi.

Use Case:

Untuk dapat *mengakses* area parkir dapat dilakukan dengan cara melakukan **registrasi** pada tempat yang telah ditunjuk secara legal untuk *menerbitkan* kartu. **Petugas loket administrasi** akan *memasukkan* data pengguna pada kartu sekaligus *mengisikan* saldo awal dengan nominal sesuai dengan pilihan **pengguna**. Agar dapat digunakan **pengguna** harus *membayar* di muka (deposit) untuk selanjutnya *didebet* dalam setiap **transaksi**. Pada saat memasuki **loket pintu masuk**, **pengguna** yang terdaftar (registered user) harus *mendekatkan* kartu pada alat pembaca sehingga sistem dapat melakukan manipulasi data pada kartu yaitu *membaca* kode autentikasi dan nomor kendaraan sekaligus *menuliskan* jam dan tanggal masuk. Pada **loket pintu keluar** akan dilakukan proses kalkulasi yaitu dengan *menghitung* lama waktu parkir kemudian saldo pada kartu akan *didebet* sesuai dengan tarif yang telah ditentukan. Apabila saldo tidak mencukupi maka, kekurangan dari tagihan dapat *dibayar* secara tunai. Untuk melakukan **pengisian ulang** saldo kartu dapat dilakukan di tempat yang telah ditunjuk secara legal untuk melakukan **pengisian ulang**.

Skenario di atas kemudian diubah ke dalam bentuk diagram *use case* seperti yang ditunjukkan pada Gambar 3.2



Gambar 3.2 Diagram *use case* aplikasi sistem parkir berlangganan

IV. IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi dan Pengujian Proses Baca dan Tulis Kartu

Pengujian pada proses baca dan tulis kartu melibatkan kelas yang terdapat pada paket **carddumper**. Salah satu kelas pada paket tersebut adalah kelas **CardDumperAdmin**. Untuk dapat melakukan proses ini maka, pada kelas **CardDumperAdmin** dideklarasikan beberapa variabel untuk mengakses dan mengalokasikan memori pada kartu. Senarai guna mendeklarasikan variabel yang digunakan mengakses dan mengalokasikan memori pada kartu adalah sebagai berikut :

```
private static String SELECT = "s";
private static String LOGIN00 = "l00aafffffffffff";
private static String LOGIN02 = "l02aafffffffffff";
private static String LOGIN03 = "l03aafffffffffff";
private static String LOGIN04 = "l04aafffffffffff";
private static String LOGIN05 = "l05aafffffffffff";
private static String READ = "r";
private static String WRITE = "w";
private static String ID = "01";          // Sektor 00 Blok 01
private static String NO_KENDARAAN = "02";  // Sektor 00 Blok 02
        :
        :
        :
private static String AUTENTIKASI = "15";      // Sektor 05 Blok 15
```

Maksud dari senarai di atas adalah *tag* yang dikenali oleh kartu disimpan ke dalam variabel dengan tipe data s*tring*. *Tag* tersebut merupakan standar yang dibuat oleh produsen kartu. Algoritma untuk mengakses kartu dimulai dengan *tag* **s** yang berarti *select*. Langkah kedua adalah *login* pada kartu dengan *tag* **l**. *Tag* **l** mempunyai format **l_sektor(2digit)_tipe kunci(2digit)_kunci(12digit)**. *Tag* **r** digunakan untuk membaca data per blok dan *tag* **w** digunakan untuk menuliskan data per blok karena akses memori pada kartu hanya dapat dilakukan per blok.

4.1.1 Pengujian Tulis Kartu

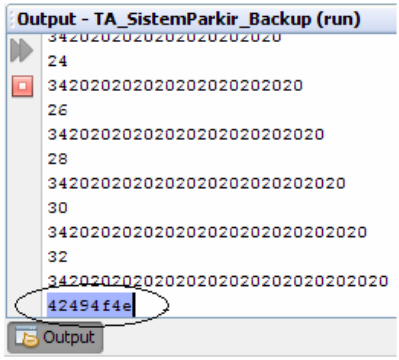
Untuk melakukan penulisan data pada kartu maka, dibutuhkan metode untuk mengubah data dalam bentuk *string* ke heksadesimal. Sebagai contoh senarai dari metode **setNama()** pada kelas **CardDumperAdmin** di bawah ini :

```
public void setNama(String Nama){
   try{
      OutputStream os = sp.getOutputStream();
      os.write(SELECT.getBytes());
      os.flush();
      Thread.sleep(100);
      os.write(LOGIN02.getBytes());
      os.flush();
      Thread.sleep(100);
      StringBuffer sb = new StringBuffer();
      for(int i = 0; i < Nama.length(); i++){
         String hexStr = Integer.toHexString(Nama.charAt(i));
         sb.append(hexStr);
           System.out.println(hexStr);
      }
      while(sb.length() != 32){
         sb.append("20");
      }
      Nama = sb.toString();
      os.write((WRITE + NAMA + Nama).getBytes());
      os.flush();
      Thread.sleep(100);
   }catch(Exception ex){
      ex.printStackTrace();
   }
}
```

Maksud dari senarai di atas adalah proses penulisan data dimulai dengan menjalankan **SELECT** diikuti dengan **LOGIN** sektor ke-2 pada memori kartu. Kemudian mengubah data bentuk *string* ke heksadesimal dan hasil dari konversi tersebut ditampung ke dalam penyangga. Apabila panjang dari karakter heksadesimal tersebut tidak sama dengan 32 maka, blok memori yang kosong akan disisipi dengan spasi. Selanjutnya melalui aliran keluaran dari port serial, data tersebut dituliskan pada kartu dengan menjalankan **WRITE + NAMA + Nama**. Perintah **NAMA** menunjukkan lokasi blok pada memori kartu di mana *string* yang variabel Nama akan disimpan. Di bawah ini merupakan metode untuk menggunakan metode **setNama()** pada kelas **CardDumperAdmin**.

```
CardDumperAdmin cda = new CardDumperAdmin();
cda.setNama("BION");
```

Pada Gambar 4.1 ditunjukkan hasil eksekusi senarai dari metode **setNama()** dengan masukkan *string* "BION" sebagai parameternya yang telah berhasil diubah ke dalam bentuk heksadesimal sehingga dapat dituliskan ke dalam kartu.



Gambar 4.1 Hasil pengujian penulisan data ke kartu pada kompilator

4.1.2 Pengujian Baca Kartu

Untuk melakukan pembacaan data pada kartu maka, metode yang dibutuhkan adalah mengubah data dari heksadesimal ke dalam bentuk *string*. Sebagai contoh senarai dari metode **getNama()** pada kelas **CardDumperAdmin** di bawah ini :

```
private void getNama(){
   try{
      OutputStream os = sp.getOutputStream();
      os.write(SELECT.getBytes());
      os.flush();
      Thread.sleep(100);
      os.write(LOGIN02.getBytes());
      os.flush();
      Thread.sleep(100);
      cardBuffer = new StringBuffer();
      cardBuilder = new StringBuffer();
      os.write((READ + NAMA).getBytes());
      os.flush();
      Thread.sleep(100);
      cardBuilder.append(cardBuffer);
      nama = parse(new String(cardBuilder));
      cardBuffer = null;
      cardBuilder = null;
   }catch(Exception ex){
      ex.printStackTrace();
   }
}
```

Maksud dari senarai di atas adalah setelah perintah **SELECT** dan **LOGIN** pada sektor ke-2 pada memori kartu dilaksanakan, melalui aliran keluaran port serial perintah **READ** + **NAMA** dilaksanakan. Kemudian data ditampung dalam penyangga dan proses selanjutnya adalah mengubah data dalam bentuk heksadesimal ke *string* dengan memanggil metode **parse()**. Senarai dari metode **parse()** adalah sebagai berikut :

```
private String parse(String str) {
      str = str.substring(0, 32);
      int i = 0;
      int j = 0;
      char[] c = new char[str.length() / 2];
      for (i = 0; i < str.length(); i += 2) {
            c[j] = (char) (Integer.parseInt(str.substring(i, i + 2), 16));
            j++;
      }
      String newStr = String.valueOf(c);
      return newStr;
   }
```

Di dalam metode **parse()**, data dalam bentuk heksadesimal dipisahkan setiap satu Byte sehingga pada saat dilakukan konversi akan menghasilkan satu karakter ASCII. Setelah data tersebut dikonversi kemudian ditampung kembali ke dalam penyangga. Untuk menampilkan data yang telah diubah tersebut dapat dilakukan dengan menggunakan metode **dumpCard()** yang terdapat pada setiap kelas yang tersimpan dalam paket **carddumper**. Senarai dari metode **dumpCard()** adalah sebagai berikut :

```
public void dumpCard() {
   try {
      getId();
      Thread.sleep(100);
      getNoKendaraan();
      Thread.sleep(100);
         :
         :
         :
      getTanggalKeluar();
      Thread.sleep(100);

      getKodeAutentikasi();
      Thread.sleep(100);
            String outputStr = id + "#"+ no_kendaraan + "#"
               + nama + "#" + alamat1 + "#" + alamat2 + "#"
               + no_telp + "#" + saldo_awal + "#" +
               saldo_akhir + "#" + jam_masuk + "#" +
               tanggal_masuk + "#" + jam_keluar + "#" +
               tanggal_keluar + "#" + autentikasi;
      File f = new File("C:/CardDumperAdmin.temp");
      FileWriter fw = new FileWriter(f);
      BufferedWriter bw = new BufferedWriter(fw);
      bw.write(outputStr);
      bw.flush();
      bw.close();
      sp.close();
   } catch (Exception ex) {
      ex.printStackTrace();
   }
}
```

Fungsi dari metode **dumpCard()** adalah untuk menampung data yang telah diubah ke dalam bentuk *string* ke dalam sebuah berkas **CardDumperAdmin.temp** sehingga dapat ditampilkan ke antarmuka pengguna. Perintah untuk menjalankan metode tersebut adalah sebagai berikut :

```
CardDumperAdmin cda = new CardDumperAdmin();
cda.dumpCard();
```

Gambar 4.2 menunjukkan hasil yang ditampilkan dengan menjalankan senarai tersebut melalui antarmuka

pengguna dalam kasus ini antarmuka yang dieksekusi adalah halaman **Admin** pada panel **Registrasi.**

Gambar 4.2 Hasil pengujian pembacaan data dari kartu

V. PENUTUP

5.1 Kesimpulan

1. Telah dapat dikembangkan aplikasi kartu cerdas tanpa kontak (*contactless smartcard*) pada sistem parkir berlangganan dengan menggunakan bahasa pemrograman Java.
2. Sistem masih membutuhkan peran operator dalam pengoperasian.
3. Jarak maksimal pengaksesan kartu terhadap *interrogator* kurang lebih adalah 3cm.
4. Pengaksesan data pada memori kartu baik operasi baca maupun tulis adalah per blok. Tiap blok memori pada kartu mempunyai kapasitas sebesar 16 bytes atau hanya dapat menyimpan 16 karakter.
5. Data yang akan disimpan ke dalam kartu harus diubah ke dalam bentuk heksadesimal.

5.2 Saran

1. Perlu dilakukan pengembangan pada proses baca kontinu setelah data berhasil ditampilkan di antarmuka pengguna sehingga dapat mengantisipasi terjadinya penumpukan data yang diakibatkan oleh kartu yang terbaca lebih dari satu kali pada saat transaksi dilaksanakan.
2. Diharapkan dengan dikembangkannya aplikasi kartu cerdas tanpa kontak ini, satu kartu dapat dimanfaatkan ke berbagai macam fungsi seperti kartu presensi, SIM, dan lain sebagainya.

DAFTAR PUSTAKA

[1]. Komputer, Wahana, *Membuat Aplikasi Profesional dengan Java*, PT Elex Media Komputindo, Jakarta, 2005.

[2]. Sanchez, J, *Java™ 2 Weekend Crash Course*, PT Elex Media Komputindo Kelompok Gramedia, Jakarta, 2002.

[3]. Horton, I, *Beginning Java™ 2, JDK™ 5 Edition*, Wiley Publishing Inc, Indianapolis, 2005.

[4]. Sweeney, P.J, *RFId for Dummies*, Wiley Publishing Inc, Indianapolis, 2005.

[5]. Peak, P. and N. Heudecker, *Hibernate Quickly*, Manning Publication Co., 2006.

[6]. Darwin, I.F, *Java™ Cookbook 2nd Edition*, O'Reilly, 2004.

[7]. Cole, B, *Java™ Swing 2nd Edition*, O'Reilly, 2002.

[8]. Quatrani, T, *Visual Modelling with Rational Rose 2000 and UML*, Addison Wesley, 1999.

[9]. Keegan, P, *NetBeans™ IDE Field Guide*, Prentice Hall, 2006.

[10]. Kathy, W, *JFC Swing Tutorial, The: A Guide to Constructing GUIs, Second Edition*, Addison Wesley, 2004.

[11]. ---, http://developers.sun.com, Agustus 2007.

[12]. ---, www.kodejava.org, Februari 2008.

[13]. ---, http://snippets.dzone.com/tag/java, April 2008.

[14]. ---, www.science.uva.nl, September 2007.

[15]. ---, http://smart.puragroup.com, Juli 2008.

[16]. ---, http://en.wikibooks.org, Juli 2008.

[17]. ---, *Mifare® Standard Card IC MF1ICS50 Product Specification*, PHILIPS Semiconductor, 2001

[18]. ---, *MicroEngine 13,56MHz Dual Reader Module*, EHAG, www.ehag.ch, 2004.

BIODATA PENULIS

Subiono Ahmad (L2F305240) dilahirkan di Banjarnegara, 11 Agustus 1981. Mahasiswa Teknik Elektro Ekstensi 2005, Bidang konsentrasi Teknik Informatika, Universitas Diponegoro Semarang. Email : barkenceil@gmail.com

Menyetujui dan mengesahkan,

Pembimbing I

Aghus Sofwan, S.T., M.T.
NIP. 132 163 757
Tanggal.....

Pembimbing II

R. Rizal Isnanto, S.T., M.M., M.T.
NIP. 132 288 515
Tanggal.....