BAB V

IMPLEMENTASI PERANGKAT LUNAK

5.1 Pendahuluan

Pada bagian ini akan dijelaskan dokumentasi implementasi perangkat lunak, struktur data, prosedur – prosedur, serta antar muka untuk pengguna.

5.2 Batasan Implementasi

Pada aplikasi internet banking dan SMS banking, yang pada akhirnya disebut bank Hamony, diasumsikan bahwa pengguna aplikasi, yaitu nasabah telah melakukan pendaftaran internet banking dan SMS banking melalui ATM . Satu nasabah dapat memiliki beberapa nomor rekening.

Platform berbasis sistem operasi Microsoft Windows XP Professional dipilih karena kemudahannya dalam mendukung database server , web server dan application server yang berguna untuk menyimpan kumpulan *business object*.

Sebagai web server, dipilih Internet Information Services karena kemudahannya dalam instalasi (IIS telah menjadi komponen optional dalam Windows XP), dukungannya terhadap ASP dan ADO.

5.3 Bahasa Pemrograman

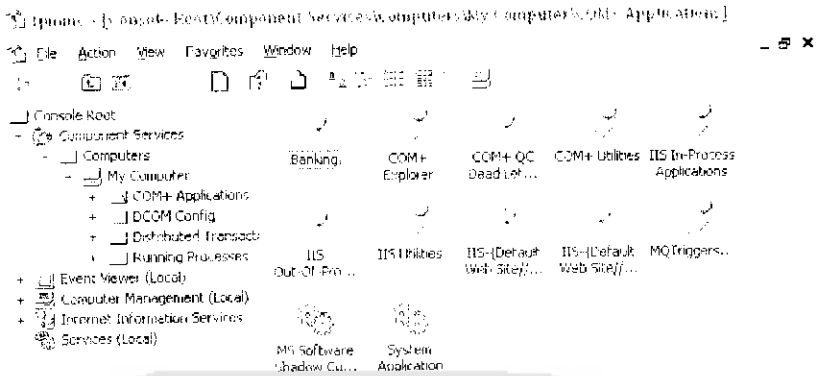
Bahasa pemrograman yang digunakan ada dua jenis, bahasa pemrograman untuk desktop dan bahasa pemrograman web.

Bahasa pemrograman untuk dekstop digunakan untuk membuat *business object*. Dalam hal ini Borland Delphi dipilih karena dukungannya terhadap teknologi COM+ dan sifatnya yang berorientasi obyek.

Untuk bahasa pemrograman web sebagai antar muka bagi pengunjung web, dipilih ASP karena dukungan penuhnya dalam memanggil beberapa *business object* pada *layer business service* yang dibuat dengan teknologi COM+.

5.4 Implementasi

Implementasi Arsitektur three-tier pada Internet Banking dan SMS Banking membutuhkan sebuah *business object*, yaitu *business object* Banking untuk memroses operasi-operasi yang berkaitan dengan aplikasi Internet Banking dan SMS Banking. *Business object* yang telah dibuat, ditampilkan pada gambar 5.1.

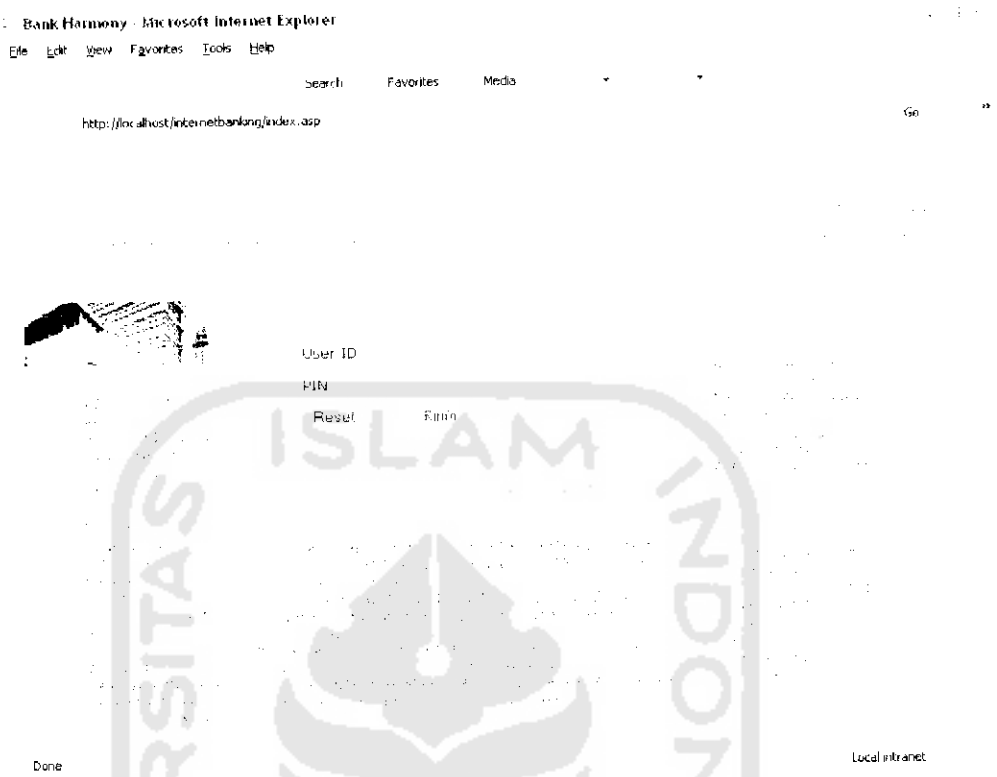


Gambar 5.1 Business object banking pada Window Component Service

5.4.1 Implementasi Tampilan dan Kode Program Internet Banking

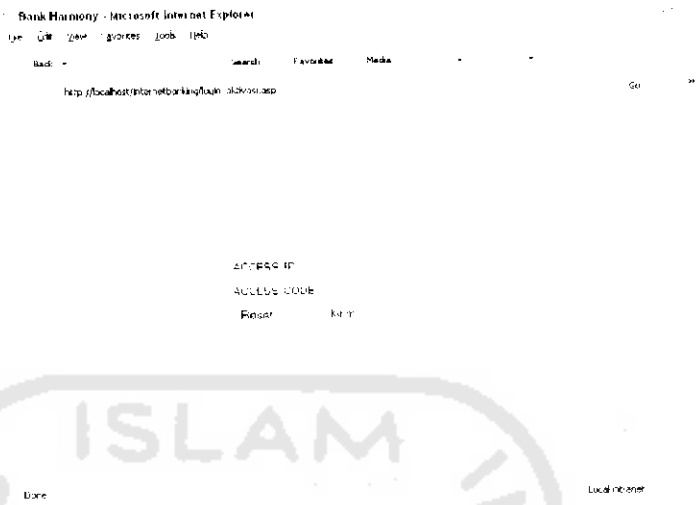
5.4.1.1 Tampilan Halaman Utama

Tampilan halaman utama berisi form login, dan beberapa petunjuk penggunaan internet banking. Selain itu, terdapat juga link untuk mengakses halaman login aktivasi jika nasabah belum melakukan aktivasi internet banking. Pada halaman ini nasabah diharuskan melakukan proses login agar dapat masuk ke dalam sistem dan melakukan transaksi perbankan di dalamnya sesuai hak akses masing-masing. Tampilan Menu Utama dapat dilihat pada gambar 5.2.



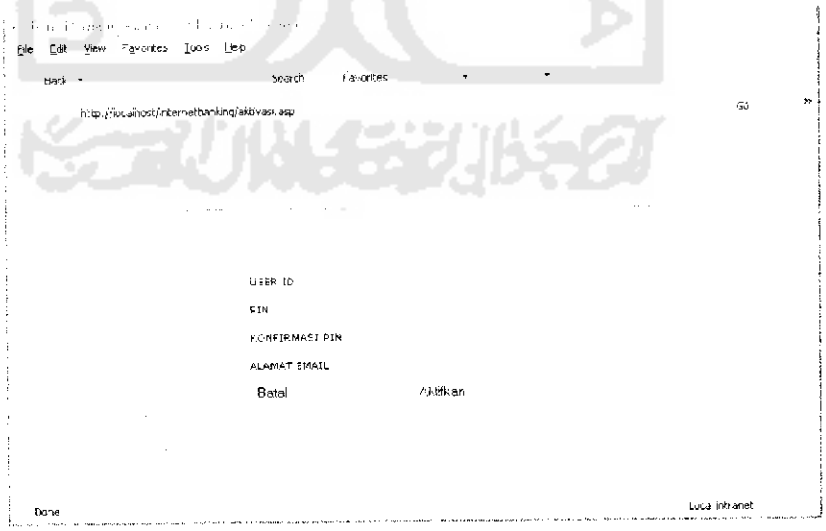
Gambar 5.2 Tampilan Menu Utama

Apabila user belum melakukan aktivasi internet, maka pada halaman utama terdapat link "silahkan klik di sini" yang dapat di-klik oleh user untuk melakukan proses login aktivasi terlebih dahulu. Login aktivasi hanya dapat dilakukan oleh user yang telah melakukan pendaftaran internet banking melalui ATM. Halaman Login Aktivasi ditunjukkan pada gambar 5.3.



Gambar 5.3 Halaman Login Aktivasi

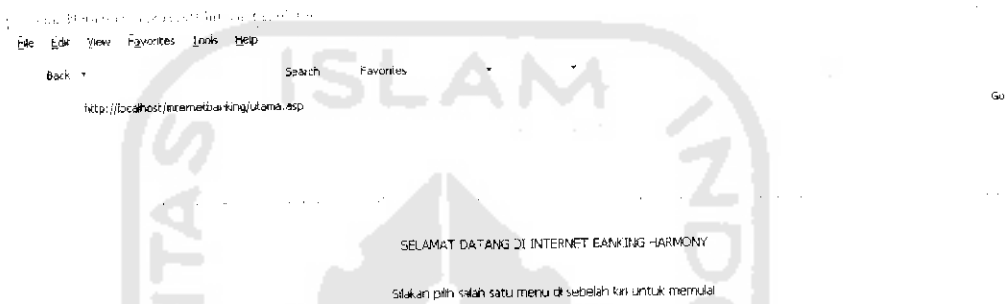
Setelah berhasil melakukan proses login aktivasi, maka nasabah disuguhkan tampilan menu form aktivasi yang yang harus diisi agar nasabah memiliki user id dan pin yang nantinya digunakan untuk melakukan transaksi perbankan. Tampilan menu aktivasi ditunjukkan pada gambar 5.4.



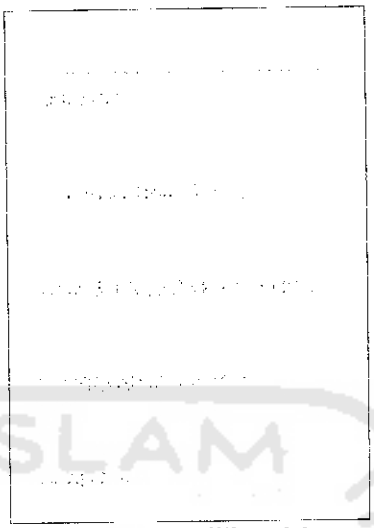
Gambar 5.4 Halaman Aktivasi

5.4.1.2 Halaman Nasabah

Seperti yang terlihat pada gambar 5.5. Halaman nasabah akan dipanggil apabila user berhasil melakukan login. Di sini terdapat beberapa menu, yaitu cek saldo, sejarah transaksi, transfer antar rekening, pembayaran listrik, dan ubah pin yang ditunjukkan pada gambar 5.6.



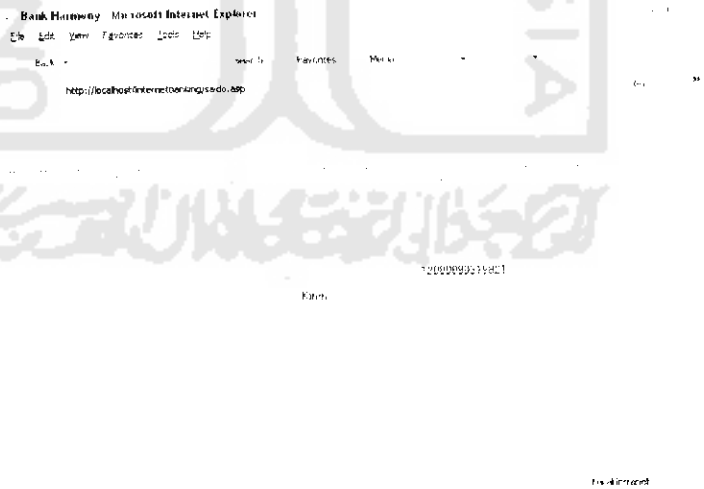
Gambar 5.5 Halaman Nasabah



Gambar 5.6 Pilihan Menu

5.4.1.3 Menu Cek Saldo

Di sini nasabah dapat melihat saldo terakhir berdasarkan nomor rekening yang dipilih. Setiap nasabah dapat memiliki satu atau lebih nomor rekening. Proses pemilihan nomor rekening ditunjukkan pada gambar 5.7.



Gambar 5.7 Menu Cek Saldo

Pemanggilan data pada model three-tier dengan COM+ ditunjukkan pada kode sumber 5.1 tentang penampilan informasi rekening dengan ASP.

```
<%
rsClientInfo = ""
set COM_InformasiRekening =
server.CreateObject("Banking.InformasiRekening")
set rsSaldo = COM_InformasiRekening.AmbilSaldo(rsClientInfo
,request("MenuRekening"))
set COM_InformasiRekening = nothing
%>
```

Kode Sumber 5.1 Penampilan Informasi Rekening

Pada kode sumber 5.1 variabel rsSaldo menerima hasil pengolahan dari *business object* AmbilSaldo, yang ditampilkan pada kode sumber 5.2.

```
function TInformasiRekening.AmbilSaldo(vrsClientInfo: OleVariant;
  const vNoRekening: WideString): OleVariant;

begin
  try
    result := _AmbilSaldo(vrsClientInfo,vNoRekening);
    SetComplete;
  except
    result := '';
    SetAbort;
  end;
end;
```

Kode Sumber 5.2 *Business object* AmbilSaldo

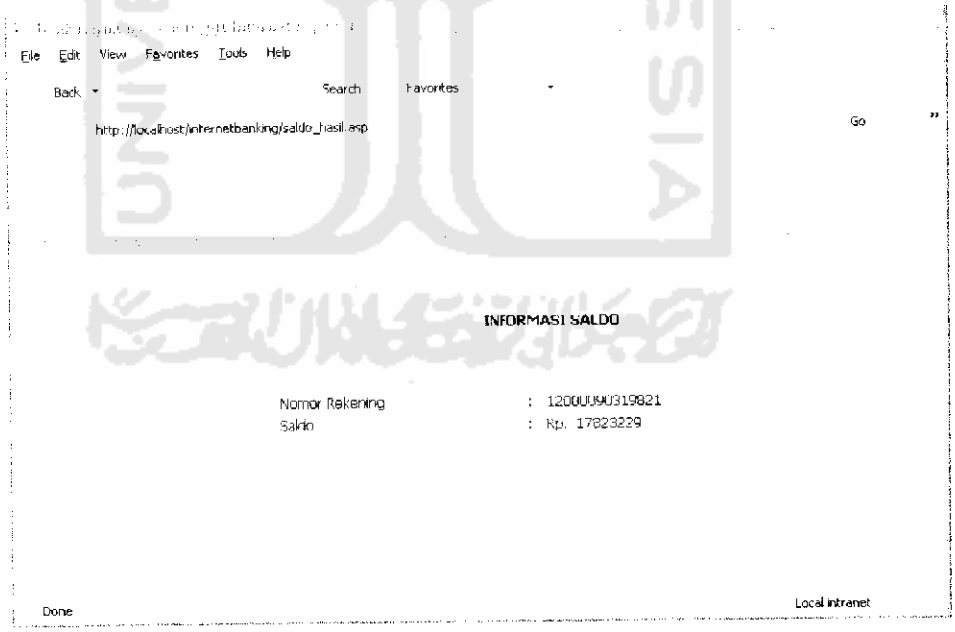
Business object AmbilSaldo pada kode sumber 5.2 melakukan panggilan ke function _AmbilSaldo pada kode sumber 5.3.

```
function TInformasiRekening._AmbilSaldo(vrsClientInfo: OleVariant;
  const vNoRekening: WideString): OleVariant;
var
  Query : TADOQuery;
begin
  try
  Query:=TADOQuery.Create(nil);
```

```
  try
  Query.ConnectionString := GetConnectionString;
  Query.SQL.Clear;
  Query.SQL.Add(' SELECT saldo FROM rekening ');
  Query.SQL.Add(' WHERE no_rekening = :no_rekening ');
  Query.Parameters.ParamByName('no_rekening').Value :=
vNoRekening;
  Query.Open;
  Query.Recordset.Set_ActiveConnection(nil);
  result := Query.Recordset;
  finally
  Query.Free;
  end;
      except
       on Ex : Exception do
       begin
        result := Null ;
       end;
    end;
end;
```

Kode Sumber 5.3 *Function* _ AmbilSaldo untuk melakukan query

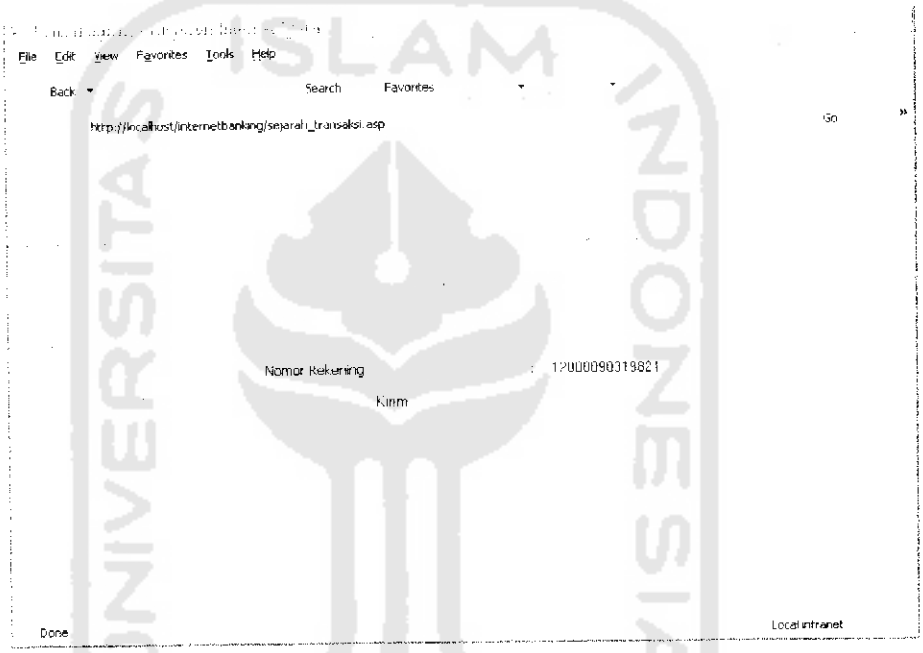
Setelah nomor rekening dipilih, maka akan muncul tampilan informasi saldo terakhir seperti ditunjukkan pada gambar 5.8.



Gambar 5.8 Informasi Saldo

5.4.1.4 Menu Sejarah Transaksi

Sama seperti menu cek saldo, pada menu sejarah transaksi nasabah terlebih dahulu memilih nomor rekening untuk melihat informasi 5 aktivitas transaksi terakhir yang pernah dilakukan. Proses pemilihan nomor rekening ditunjukkan pada gambar 5.9.



Gambar 5.9 Menu Sejarah Transaksi

Kode program penampilan sejarah transaksi ditunjukkan pada kode sumber 5.4.

```
<%
rsClientInfo = ""
set COM_InformasiRekening =
server.CreateObject("Banking.InformasiRekening")
      set rsData =
COM_InformasiRekening.AmbilSejarahTransaksi(rsClientInfo,request("
MenuRekening"))
      set COM_InformasiRekening = nothing
%>
```

Kode Sumber 5.4 Penampilan Sejarah Transaksi

Pada kode sumber 5.4, dilakukan pemanggilan fungsi AmbilSejarahTransaksi pada *business object* banking yang berada pada *application server*. Rincian kode program pada fungsi AmbilSejarahTransaksi dan _AmbilSejarahTransaksi ditunjukkan pada kode sumber 5.5.

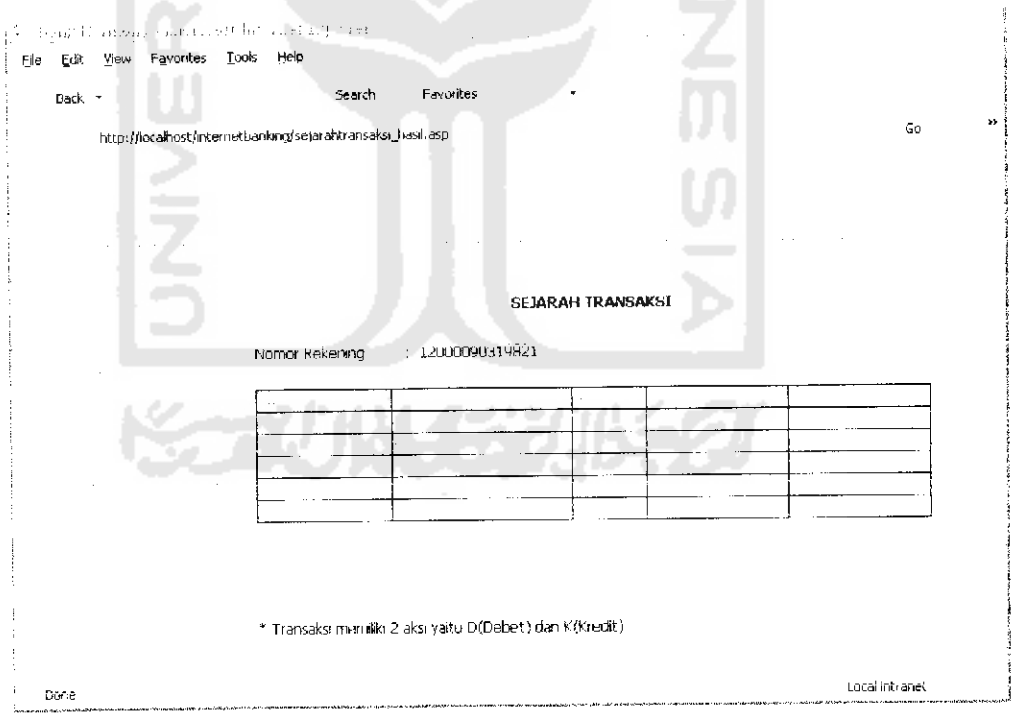
```
function TInformasiRekening._AmbilSejarahTransaksi(
  vrsClientInfo: OleVariant; const vNoRekening: WideString):
OleVariant;

var
  Query : TADOQuery;
begin
  try
  Query: TADOQuery.Create(nil);
  try
  Query.ConnectionString := GetConnectionString;
  Query.SQL.Clear;
  Query.SQL.Add(' SELECT f. * FROM ');
  Query.SQL.Add(' (SELECT a.kd_transaksi, a.no_rekening,a.aksi,
a.tanggal, a.kd_sandi, a.mutasi, a.saldo, b.nama_sandi ');
  Query.SQL.Add(' FROM transaksi a, sandi_transaksi b  WHERE
a.no_rekening= :no_rekening AND a.kd_sandi=b.kd_sandi ORDER BY
a.tanggal desc  ) f ');
  Query.SQL.Add(' WHERE rownum <= 5 ');
  Query.Parameters.ParamByName('no_rekening').Value :=
vNoRekening;
  Query.Open;
  Query.Recordset.Set_ActiveConnection(nil);
  result := Query.Recordset;
  finally
  Query.Free;
  end;
      except
       on Ex : Exception do
       begin
        result := Null ;
       end;
    end;
end;


function TInformasiRekening.AmbilSejarahTransaksi(
  vrsClientInfo: OleVariant; const vNoRekening: WideString):
OleVariant;
begin
  try
    result := _AmbilSejarahTransaksi(vrsClientInfo,vNoRekening);
    SetComplete;
  except
```

```
    result := '';
    SetAbort;
  end;
end;
```

Kode Sumber 5.5 Fungsi AmbilSejarahTransaksi dan _AmbilSejarahTransaksi

Pada fungsi AmbilSejarahTransaksi dilakukan pemanggilan fungsi _AmbilSejarahTransaksi, yang kemudian diakhiri dengan prosedur *setComplete* sebagai tanda bahwa transaksi pengambilan data disetujui. Penampilan 5 sejarah transaksi dilakukan dengan *query* ke *database* pada fungsi _AmbilSejarahTransaksi.

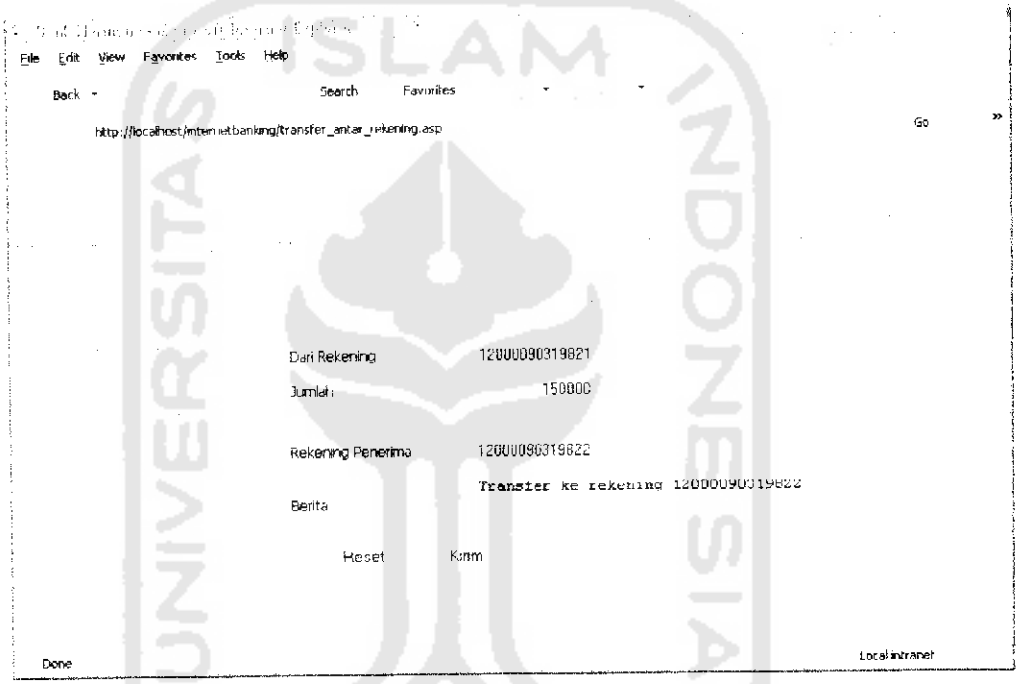
Daftar sejarah transaksi dapat dilihat setelah nomor rekening dipilih, seperti pada gambar 5.10.



Gambar 5.10 Informasi Sejarah Transaksi

5.4.1.5 Menu Transfer Antar Rekening

Menu TransferAntar Rekening digunakan untuk melakukan transfer rekening nasabah dalam satu bank. Nasabah harus mengisi form yang disediakan untuk menulis jumlah uang dan ke rekening mana uang tersebut akan ditransfer. Rekening pengirim langsung dapat dipilih. Seperti ditunjukkan pada gambar 5.11.



Gambar 5.11 Menu Transfer Antar Rekening

```
<%
rsClientInfo = ""
set COM_InformasiRekening =
server.CreateObject("Banking.InformasiRekening")
set rsRekening =
COM_InformasiRekening.AmbilNoRekening(rsClientInfo,Session("Sessio
nID"))
set COM_InformasiRekening = nothing
%>
```

Kode Sumber 5.6 Penampilan Rekening Pengirim

Pada kode sumber 5.6, dilakukan pemanggilan fungsi AmbilNoRekening pada *business object* banking yang berada pada *application server*. Rincian kode program pada fungsi AmbilNoRekening ditunjukkan pada kode sumber 5.7.

```
function TInformasiRekening._AmbilNoRekening(vrsClientInfo:
OleVariant;
  const SessionID: WideString): OleVariant;

var
  Query : TADOQuery;

begin
   try
   Query := TADOQuery.Create(nil);

   try
     with Query do
     begin
     ConnectionString := GetConnectionString;
     SQL.Clear;
     SQL.Add(' SELECT no_rekening FROM penyetor ');
     SQL.Add(' WHERE no_atm = :no_atm ');
     Parameters.ParamByName('no_atm').Value :=
AmbilNoATM2(SessionID);
     Open;
     Recordset.Set_ActiveConnection(nil);
     result:= Recordset;
     end;

     finally
       Query.Free;
     end;

     except
       on Ex : Exception do
       begin
        result := Null ;
       end;
    end;
end;


function TInformasiRekening.AmbilNoRekening(vrsClientInfo:
OleVariant;
  const SessionID: WideString): OleVariant;
begin
  try
    result := _AmbilNoRekening(vrsClientInfo,SessionID);
    SetComplete;
  except
```

```
    result := '';
    SetAbort;
  end;
end;
```

Kode Sumber 5.7 Fungsi AmbilNoRekening dan _ AmbilNoRekening pada business object banking

Pada fungsi AmbilNoRekening dilakukan pemanggilan fungsi _ AmbilNoRekening, yang kemudian diakhiri dengan perintah *setComplete* sebagai tanda bahwa transaksi pemanggilan data disetujui. Pengambilan data nomor rekening dilakukan dengan *query* ke database pada fungsi _AmbilNoRekening.

Setelah semua form terisi, maka transaksi transfer akan diproses. Kode program transaksi transfer ditunjukkan pada kode sumber 5.8.

```
<%
Dim rsData
set rsData = server.CreateObject("ADODB.Recordset")
rsData.fields.append "NomorRekening",200,255
rsData.fields.append "Jumlah",200,255
rsData.fields.append "RekeningPenerima",200,255
rsData.fields.append "Berita",200,255
rsData.open
rsData.addnew
rsData("NomorRekening")     = request("MenuRekening")
rsData("Jumlah")            = request("jumlah")
rsData("RekeningPenerima")  = request("rekening_penerima")
rsData("Berita")            = request("berita")
rsData.update

rsClientInfo = ""
set COM_InformasiRekening =
server.CreateObject("Banking.InformasiRekening")
set COM_TransferDana = server.CreateObject("Banking.TransferDana")
set rsRekening              =
COM_InformasiRekening.AmbilNoRekening(rsClientInfo,Session("Sessio
nID"))
Data =
COM_TransferDana.TransaksiTransferAntarRekening(rsClientInfo,rsDat
a.clone)
set COM_TransferDana = nothing
set COM_InformasiRekening = nothing
vNo_Rekening  = request("MenuRekening")
vRekening_Penerima = request("rekening_penerima")
vJumlah = request("jumlah")
vBerita = request("berita")
```

```
Response.Redirect "informasi_transferdana.asp?MenuRekening=" +
vNo_Rekening  + "&rekening_penerima=" + vRekening_Penerima +
"&jumlah=" + vJumlah + "&berita=" + vBerita + "&Data=" + Data
%>
```

Kode Sumber 5.8 Transaksi Transfer Antar Rekening

Pada kode sumber 5.8, dilakukan pemanggilan fungsi TransaksiTransferAntarRekening pada *business object* banking yang berada pada *application server*. Rincian kode program pada fungsi TransaksiTransferAntarRekening dan _TransaksiTransferAntarRekening ditunjukkan pada kode sumber 5.9.

```
function
TTransferDana._TransaksiTransferAntarRekening(vrsClientInfo,
  vrsData: OleVariant): WideString;
var
  Query                : TADOQuery;
  Query1               : TADOQuery;
  Query2               : TADOQuery;
  Query3               : TADOQuery;
  vNomorRekening       : string;
  vJumlah              : longint;
  vRekeningPenerima    : string;
  vBerita              : string;
  vJmlTransaksi        : longint;
  vSaldo               : longint;
  rsSaldo              : TADODataset;
begin
 try
  Query   := TADOQuery.Create(nil);
  Query1  := TADOQuery.Create(nil);
  Query2  := TADOQuery.Create(nil);
  Query3  := TADOQuery.Create(nil);
  rsData  := TADODataset.Create(nil);
  rsSaldo := TADODataset.Create(nil);

   try
    rsData.Recordset  := iunknown(vrsData) as _Recordset;
    vNomorRekening    :=
trim(rsData.fieldbyname('NomorRekening').AsString);
    vJumlah           := rsData.fieldbyname('Jumlah').AsInteger;
    vRekeningPenerima :=
trim(rsData.fieldbyname('RekeningPenerima').AsString);
    vBerita           := rsData.fieldbyname('Berita').AsString;

    with Query3 do
     begin
      ConnectionString := GetConnectionString;
```

```
      SQL.Add(' SELECT jml_transaksi_harian(:no_rekening) jml from
dual ');
      Parameters.ParamByName('no_rekening').Value :=
vNomorRekening;
      open;
     end;

     vJmlTransaksi := Query3.fields[0].AsInteger + vJumlah ;
     rsSaldo.Recordset :=
iunknown(_AmbilSaldo(vrsClientInfo,vNomorRekening)) as _recordset;
     vSaldo := rsSaldo.fieldbyname('saldo').asinteger;

     if (vSaldo-vJmlTransaksi < 0)  then
    raise ERangeError.CreateFmt('Saldo tidak mencukupi', [Result,
0, 100]);


    result:=Inttostr(vJmlTransaksi);
    if vJmlTransaksi <= 10000000 then
      begin
      with Query1 do
       begin
        ConnectionString := GetConnectionString;
        SQL.Clear;
        SQL.Add(' INSERT INTO transaksi a (a.kd_transaksi,
a.no_rekening, a.tanggal, a.kd_sandi, a.mutasi, a.aksi, a.berita)
');
        SQL.Add(' VALUES (''TR''||SQ_TRANS.nextval, :no_rekening,
sysdate, :kd_sandi, :mutasi, :aksi, :berita) ');

        Parameters.ParamByName('no_rekening').Value :=
vNomorRekening;
        Parameters.ParamByName('kd_sandi').Value     := 'PO';
        Parameters.ParamByName('mutasi').Value       := vJumlah;
        Parameters.ParamByName('aksi').Value         := 'D';
        Parameters.ParamByName('berita').Value       := vBerita;
        ExecSQL;
       end;

      with Query2 do
       begin
        ConnectionString := GetConnectionString;
        SQL.Clear;
        SQL.Add(' INSERT INTO transaksi a (a.kd_transaksi,
a.no_rekening, a.tanggal, a.kd_sandi, a.mutasi, a.aksi, a.berita)
');
        SQL.Add(' VALUES (''TR''||SQ_TRANS.nextval, :no_rekening,
sysdate, :kd_sandi, :mutasi, :aksi, :berita) ');
        Parameters.ParamByName('no_rekening').Value :=
vRekeningPenerima;
        Parameters.ParamByName('kd_sandi').Value     := 'SO';
        Parameters.ParamByName('mutasi').Value       := vJumlah;
        Parameters.ParamByName('aksi').Value         := 'K';
        Parameters.ParamByName('berita').Value       := vBerita;
        ExecSQL;
      end;
```

```
      result := 'Transaksi transfer dana berhasil' ;
     end
    else
     result  := 'Jumlah transaksi transfer anda hari ini melebihi
batas yang ditentukan ' ;

   except
     on e: exception do result := 'GAGAL: ' + e.message;
   end;

 finally
  Query3.Free;
  Query1.Free;
  Query2.Free;
  rsData.Free;
 end;

{    try
     Query1.ExecSQL;
     Query2.ExecSQL;
     result := 'OK' ;
    except
      on e: exception do result := 'GAGAL: ' + e.message;
    end;}
end;


function
TTransferDana.TransaksiTransferAntarRekening(vrsClientInfo,
  vrsData: OleVariant): WideString;
begin
  try
    result :=
_TransaksiTransferAntarRekening(vrsClientInfo,vrsData);
    SetComplete;
  except
   result := 'error';
    SetAbort;
  end;
end;
```

Kode Sumber 5.9 Fungsi TransaksiTransferAntarRekening dan _ TransaksiTransferAntarRekening

Pada fungsi TransaksiTransferAntarRekening dilakukan pemanggilan fungsi _TransaksiTransferAntarRekening, yang kemudian diakhiri dengan prosedur *setComplete* sebagai tanda bahwa transaksi pengambilan data disetujui.

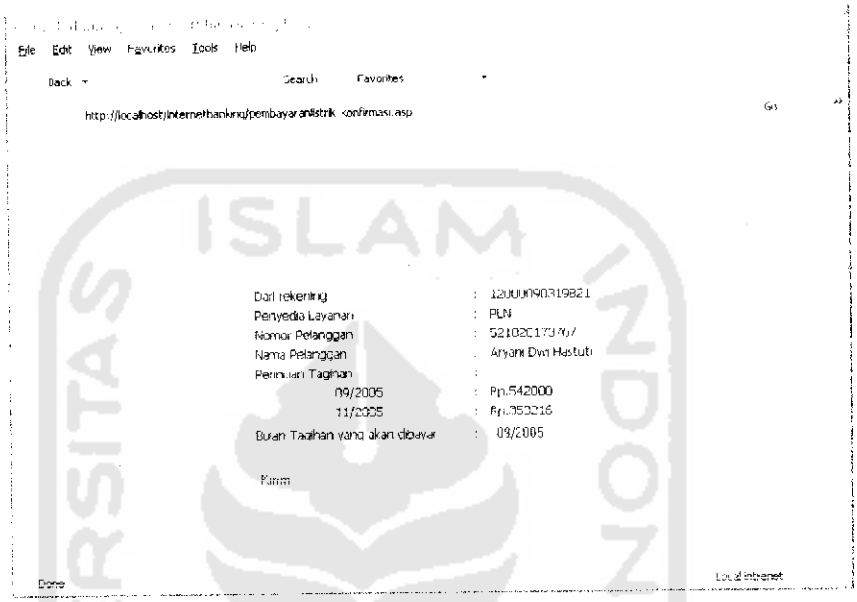
5.4.1.6 Menu Pembayaran Listrik

Internet Banking Bank Harmony juga menyediakan fasilitas pembayaran listrik atau tagihan PLN. Pada halaman ini nasabah diharuskan mengisi form dengan nomor rekening, penyedia layanan dan nomor pelanggan untuk mendapatkan informasi mengenai tagihan PLN. Seperti ditunjukkan pada gambar 5.12.

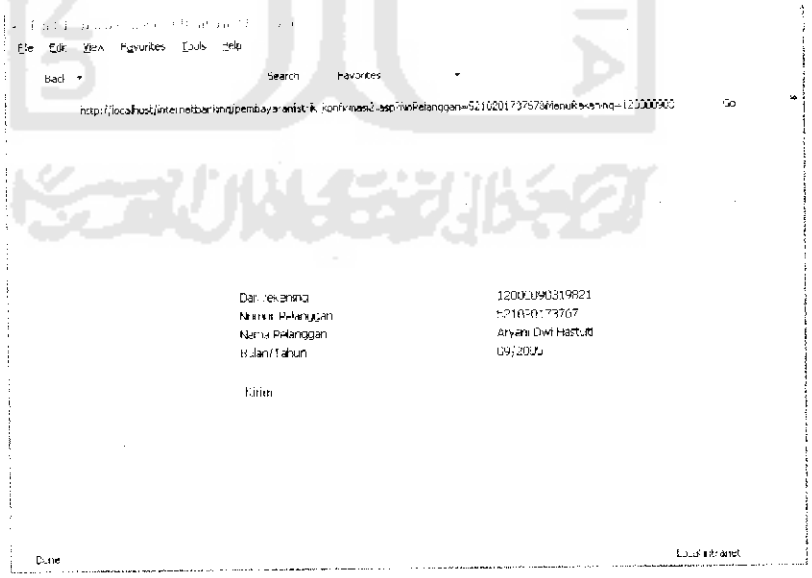
Gambar 5.12 Menu Pembayaran Listrik

Setelah mengisi form pada menu pembayaran listrik, maka nasabah akan mendapatkan konfirmasi pertama mengenai informasi tagihan yang akan dibayar. Konfirmasi pertama ditunjukkan pada gambar 5.13. Pada konfirmasi pertama ditampilkan pilihan bulan tagihan yang akan dibayar oleh nasabah. Apabila nasabah telah memilih bulan tagihan mana yang akan dibayar, maka akan tampil

konfirmasi kedua, seperti ditunjukkan gambar 5.14. Data pada konfirmasi kedua inilah yang nantinya akan diproses untuk melakukan transaksi.



Gambar 5.13 Halaman Konfirmasi Pertama Pembayaran listrik



Gambar 5.14 Halaman Konfirmasi Kedua Pembayaran listrik

Kode program untuk menampilkan informasi tagihan terlihat pada kode sumber 5.10.

```
<%
rsClientInfo = ""
set COM_InformasiRekening =
server.CreateObject("Banking.InformasiRekening")
set COM_Pembayaran          =
server.CreateObject("Banking.Pembayaran")
set rsRekening              =
COM_InformasiRekening.AmbilNoRekening(rsClientInfo,Session("Sessio
nID"))
set rsRincianPembayaran     =
COM_Pembayaran.AmbilInformasiRincianTagihan(rsClientInfo,request("
NoPelanggan"))
set COM_InformasiRekening = nothing
set COM_Pembayaran          = nothing
no_Rekening   = request("MenuRekening")
no_Pelanggan  = request("NoPelanggan")
layanan       = request("PenyediaLayanan")
%>
```

Kode Sumber 5.10 Informasi Tagihan

Pada kode sumber 5.10 dilakukan pemanggilan dua buah fungsi, yaitu AmbilNoRekening dan AmbilInformasiRincianTagihan. Fungsi AmbilNoRekening digunakan untuk menampilkan nomor rekening untuk membayar tagihan. Sedangkan fungsi AmbilInformasiRincianTagihan digunakan untuk memanggil data-data tagihan PLN. Fungsi AmbilInformasiRincianTagihan ditunjukkan pada kode sumber 5.11.

```
function TPembayaran._AmbilInformasiRincianTagihan(vrsClientInfo
:OleVariant;
const vNoPelanggan: WideString): OleVariant;

var
    Query : TADOQuery;
begin
  try
     Query  := TADOQuery.Create(nil);
     rsData := TADODataset.Create(nil);
  try
  with Query do
    begin
    ConnectionString := GetConnectionString2;
```

```
    SQL.Clear;
    SQL.Add(' SELECT a.bulan, a.tahun, a.kd_pelanggan,
a.jml_tagihan, ');
    SQL.Add(' a.biaya_banking,a.pajak_penerangan_jalan, a.status,
');
    SQL.Add(' a.jml_tagihan + a.pajak_penerangan_jalan TOTAL,
b.nama_pelanggan ');
    SQL.Add(' FROM rincian_pembayaran a, pelanggan b ');
    SQL.Add(' WHERE a.kd_pelanggan = :kd_pelanggan ');
    SQL.Add(' AND a.kd_pelanggan = b.kd_pelanggan ');
    SQL.Add(' AND a.status = ''T'' ');
    Parameters.ParamByName('kd_pelanggan').Value := vNoPelanggan;
    open;
    Recordset.Set_ActiveConnection(nil);
    result := Query.Recordset;
    end;

    except
      result:='anonymous';
    end;

finally
  Query.free;
  rsData.Free;
end;
end;

function TPembayaran.AmbilInformasiRincianTagihan(
  vrsClientInfo: OleVariant; const vNoPelanggan: WideString):
OleVariant;
begin
  try
    result :=
_AmbilInformasiRincianTagihan(vrsClientInfo,vNoPelanggan);
    SetComplete;
  except
    result := 'iskosong';
    SetAbort;
  end;
end;
```

Kode Sumber 5.11 Fungsi _AmbilInformasiRincianTagihan dan AmbilInformasiRincianTagihan

Pada fungsi AmbilInformasiRincianTagihan dilakukan pemanggilan fungsi _ AmbilInformasiRincianTagihan, yang kemudian diakhiri dengan perintah *setComplete* sebagai tanda bahwa transaksi pemanggilan data disetujui. Pengambilan data tagihan PLN dilakukan dengan *query* ke database pada fungsi _ AmbilInformasiRincianTagihan.

Setelah bulan tagihan dipilih pada halaman konfirmasi kedua, selanjutnya transaksi pembayaran listrik diproses. Kode sumber transaksi pembayaran ditunjukkan pada kode sumber 5.12.

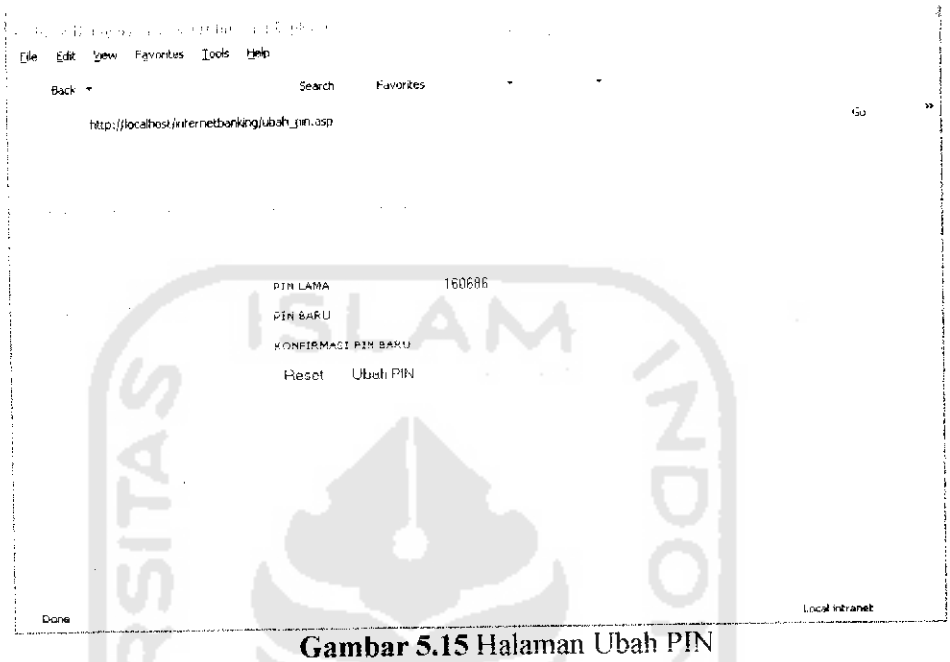
```
function TPembayaran._TransaksiBayarListrik(vrsClientInfo,
  vrsData: OleVariant): WideString;
var
  Query1              : TADOQuery;
  Query2              : TADOQuery;
  Query3              : TADOQuery;
  Query4              : TADOQuery;
  vNomorRekening      : string;
  vNoPelanggan        : string;
  vJumlahTagihan      : string;
  vRekeningPLN        : string;
  vBulanTahun         : string;
  vBulan              : string;
  vTahun              : string;
  vTotal              : longint;
  listBulanTahun      : TStringList;
begin
    try
    Query1  := TADOQuery.Create(nil);
    Query2  := TADOQuery.Create(nil);
    Query3  := TADOQuery.Create(nil);
    Query4  := TADOQuery.Create(nil);
    rsData  := TADODataset.Create(nil);
    listBulanTahun := TStringList.Create;

    try
      rsData.Recordset  := iunknown(vrsData) as _Recordset;
      vNomorRekening    :=
rsData.fieldbyname('NomorRekening').AsString;
      vNoPelanggan      :=
rsData.fieldbyname('NoPelanggan').AsString;
      vRekeningPLN      := '1111111111111';
      vBulanTahun       :=
rsData.fieldbyname('BulanTahun').AsString;
      SplitIt( vBulanTahun , '/' , listBulanTahun);
      vBulan            := listBulanTahun[0];
      vTahun            := listBulanTahun[1];

      with Query4 do
      begin
      ConnectionString := GetConnectionString;
      SQL.Clear;
      SQL.Add(' SELECT distinct a.jml_tagihan +
a.pajak_penerangan_jalan + a.biaya_banking TOTAL ');
      SQL.Add(' FROM pln.rincian_pembayaran a, pln.pelanggan b ');
      SQL.Add(' WHERE a.kd_pelanggan = :kd_pelanggan ');
```

```
      SQL.Add(' AND a.bulan = :bulan ');
      SQL.Add(' AND a.tahun = :tahun ');
      Parameters.ParamByName('kd_pelanggan').Value :=
vNoPelanggan;
      Parameters.ParamByName('bulan').Value := vBulan;
      Parameters.ParamByName('tahun').Value := vTahun;
      open;
      vTotal := fieldbyname('TOTAL').Value;
      result := GetConnectionString;
      end;

      with Query1 do
      begin
       ConnectionString := GetConnectionString;
       SQL.Clear;
       SQL.Add(' INSERT INTO transaksi a (a.kd_transaksi,
a.no_rekening, a.kd_sandi, a.mutasi, a.aksi, a.berita) ');
       SQL.Add(' VALUES (''TR''||SQ_TRANS.nextval, :no_rekening,
:kd_sandi, :mutasi, :aksi, :berita) ');

       Parameters.ParamByName('no_rekening').Value :=
vNomorRekening;
       Parameters.ParamByName('kd_sandi').Value      := 'PO';
       Parameters.ParamByName('mutasi').Value        := vTotal;
       Parameters.ParamByName('aksi').Value          := 'D';
       Parameters.ParamByName('berita').Value        := 'Pembayaran
Rekening PLN';
       ExecSQL;
      end;

      with Query2 do
      begin
       ConnectionString := GetConnectionString;
       SQL.Clear;
       SQL.Add(' INSERT INTO transaksi a (a.kd_transaksi,
a.no_rekening, a.kd_sandi, a.mutasi, a.aksi) ');
       SQL.Add(' VALUES (''TR''||SQ_TRANS.nextval, :no_rekening,
:kd_sandi, :mutasi, :aksi) ');

       Parameters.ParamByName('no_rekening').Value :=
vRekeningPLN;
       Parameters.ParamByName('kd_sandi').Value      := 'SO';
       Parameters.ParamByName('mutasi').Value        := vTotal;
       Parameters.ParamByName('aksi').Value          := 'K';
       ExecSQL;
      end;

      with Query3 do
      begin
       ConnectionString := GetConnectionString;
       SQL.Clear;
       SQL.Add(' UPDATE pln.rincian_pembayaran SET status =
:status ');
       SQL.Add(' WHERE bulan = :bulan AND tahun = :tahun ');
       SQL.Add(' AND kd_pelanggan = :kd_pelanggan ');
       Parameters.ParamByName('kd_pelanggan').Value :=
```

```
vNoPelanggan;
        Parameters.ParamByName('status').Value        := 'L';
        Parameters.ParamByName('bulan').Value         := vBulan;
        Parameters.ParamByName('tahun').Value         := vTahun;
        ExecSQL;
       end;
//     end;


            result := 'OK' ;

          except
          on e: exception do result := 'GAGAL: ' + e.message;
          end;

    finally
      Query1.Free;
      Query2.Free;
      Query3.Free;
      Query4.Free;
      rsData.Free;
      listBulanTahun.Free;
    end;
end;


function TPembayaran.TransaksiBayarListrik(vrsClientInfo,
  vrsData: OleVariant): WideString;
begin
   try
    result := _TransaksiBayarListrik(vrsClientInfo,vrsData);
    SetComplete;
  except
    result := 'error';
    SetAbort;
  end;

end;
```

Kode Sumber 5.12 Fungsi _TransaksiBayarListrik dan TransaksiBayarListrik

5.4.1.7 Menu Ubah PIN



Gambar 5.15 Halaman Ubah PIN

5.4.2 Implementasi Tampilan SMS Banking

5.4.2.1 Cek Saldo

SMS Cek Saldo digunakan untuk mengetahui saldo terakhir dari nasabah yang bersangkutan. Pengiriman SMS dan balasan ditunjukkan pada gambar 5.16 dan 5.17.

Gambar 5.16 SMS Cek Saldo

Gambar 5.17 Balasan SMS Cek Saldo

5.4.2.2 Sejarah Transaksi

SMS Sejarah Transaksi digunakan untuk mengetahui 5 sejarah transaksi terakhir dari nasabah yang bersangkutan. Pengiriman SMS dan balasan ditunjukkan pada gambar 5.18 dan 5.19.

SEJ 12000090319821
160686

Gambar 5.18 SMS Sejarah Transaksi

30/10/2005 13:29:13 D
100000
28/10/2005 13:29:13 K
35000
18/10/2005 13:29:13 D
5000
18/10/2005 13:29:13 K
2000

Gambar 5.19 Balasan SMS Sejarah Transaksi

5.4.2.3 Transfer Antar Rekening

SMS Transfer antar rekening dilakukan untuk melakukan transfer rekening dalam satu bank. Pengiriman SMS dan balasan ditunjukkan pada gambar 5.20 dan 5.21.

TRA 12000090319821
12000090319822 160686

Gambar 5.20 SMS Transfer Antar Rekening

TRANSFER ke nomor
rekening :
12000090319822
BERHASIL

Gambar 5.21 Balasan SMS Transfer Antar Rekening

5.4.2.4 Tagihan PLN

SMS Tagihan PLN terdiri dari SMS info tagihan PLN dan bayar tagihan PLN. SMS info tagihan digunakan untuk melihat banyaknya tagihan PLN. SMS bayar tagihan digunakan untuk membayar tagihan PLN. Pengiriman SMS dan balasan ditunjukkan pada gambar 5.22, 5.23, 524, dan 5.25.

INF 12000090319821
521020173767 160686

Gambar 5.22 SMS Info Tagihan PLN

Jumlah tagihan PLN No
521020173767/Aryani
Dwi Hastuti :
09/2005 Rp.542000
11/2005 Rp.353216

Gambar 5.23 Balasan SMS Info Tagihan PLN

BYR 12000090319821
521020173767 08/2005
160686

Gambar 5.24 SMS Bayar Tagihan PLN

Pembayaran tagihan
BERHASIL ke PLN no
521020173767

Gambar 5.25 Balasan SMS Info Tagihan PLN

5.4.3 Perangkat Lunak Pendukung dalam Implementasi

Berikut ini beberapa perangkat lunak yang mendukung dalam pengerjaan penelitian ini :

1. Delphi 7 Enterprise Edition
2. Windows XP Professional
3. Active Server Page (ASP)
4. Database Server Oracle 9i
5. Rational Rose untuk membuat diagram UML
6. Editor ASP menggunakan Macromedia Dreamweaver
7. IIS

8. Beberapa patch Oracle dan Windows XP untuk mendukung teknologi COM+
9. Microsoft Web Application Stress Tool