PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK INTERNET BANKING UNTUK ALAT PEMBAYARAN MENGGUNAKAN MICROSOFT.NET

TUGAS AKHIR

Disusun Oleh:

FRANCISCA MAYASARI

5197100043

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA

2002

TUGAS AKHIR

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK INTERNET BANKING UNTUK ALAT PEMBAYARAN MENGGUNAKAN MICROSOFT.NET

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer
Pada
Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya

Mengetahui/Menyetujui,

Dosen Pembimbing I

Ir. Muhammad Husni, M.Kom
NIP. 131 411 100

Dosen Pembimbing II

Ir. Suhadi Lili
NIP. 132 048 148

SURABAYA
Agustus, 2002

ABSTRAK

ABSTRAK

Layanan yang disediakan oleh bank di era internet ini adalah pembayaran tagihan-tagihan nasabah di merchant-merchant relasi bank, pembelian voucher handphone, dan transfer. Namun mekanisme yang ada saat ini masih memiliki kelemahan. Mekanisme yang ada sekarang ini menggunakan sistem batch dimana data disetor dari merchant relasi ke bank setiap kurun waktu tertentu. Hal ini mengakibatkan data merupakan cuplikan dari data yang asli maka integritas data dan sistem menjadi lemah. Selain itu hal ini mengakibatkan transaksi pembayaran hanya dapat dilakukan pada waktu-waktu tertentu setelah data diterima. Masalah yang lain yaitu merchant tidak bisa melakukan validasi transfer yang masuk ke rekeningnya secara langsung. Akibatnya proses transaksi menjadi lebih lama serta memerlukan biaya dan mengakibatkan yang lebih besar.

Pada tugas akhir ini dikembangkan suatu perangkat lunak yang mampu membantu mengatasi permasalahan ini dengan menggunakan Microsoft.NET. Web Service pada Microsoft.NET memungkinkan sistem berintegrasi dengan sistem lain. Hal ini menyebabkan integrasi data dan sistem menjadi kuat dan data masing-masing sistem yang berinteraksi bisa dibatasi pengaksesannya. Selain itu, level integrasi sekarang tidak lagi pada level data melainkan pada level bisnis (layanan). Merchant juga bisa melakukan transfer validasi secara langsung . Web Service menggunakan protokol-protokol yang standard, yaitu SOAP (simple object access protocol) dan XML sehingga komunikasi antar sistem menjadi standard.

KATA PENGANTAR

KATA PENGANTAR

Puji dan syukur saya panjatkan ke hadirat Tuhan Yang Maha Esa, karena atas perkenan-Nya lah maka saya dapat menyelesaikan segala rangkaian pengerjaan Tugas Akhir. Tugas Akhir ini merupakan kewajiban bagi setiap mahasiswa dalam menyelesaikan program strata satu (S1) pada Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember, Surabaya.

Laporan Tugas Akhir ini dengan judul **"PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK INTERNET BANKING UNTUK ALAT PEMBAYARAN MENGGUNAKAN MICROSOFT.NET"** disusun sebagai dokumentasi sekaligus merupakan penutup pelaksanaan Tugas Akhir ini.

Besar harapan saya bahwa Laporan Tugas Akhir ini dapat memberikan suatu kontribusi tersendiri bagi wahana ilmu pengetahuan dalam bidang Teknologi Informasi.

Saya menyadari bahwa penyusunan Laporan Tugas Akhir ini tidak luput dari kesalahan dan kekurangan-kekurangan lainnya. Untuk itu saya membuka diri dan akan sangat menghargai segala saran maupun kritik yang sifatnya membangun. Akhir kata, selamat membaca dan semoga Laporan Tugas Akhir ini dapat memberikan manfaat yang seluas-luasnya bagi kita semua.

Francisca Mayasari

Surabaya, Juli 2002

UCAPAN TERIMA KASIH

UCAPAN TERIMA KASIH

Dalam kesempatan ini saya ingin mengucapkan terima kasih kepada semua pihak yang telah membantu saya dalam pengerjaan tugas akhir ini, baik secara langsung maupun tidak langsung. Secara khusus saya mengucapkan terima kasih kepada :

1. Kedua orang tua dan saudara-saudara saya yang tercinta. Terima kasih atas segala dukungan, kasih sayang, arahan dan didikan serta segala bantuan yang diberikan.
2. Dave yang telah banyak mendorong, memberi bantuan serta semangat dan juga saran. Terima kasih untuk semuanya.
3. Bapak Agus Zainal Arifin M.kom, selaku ketua Jurusan Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya. Terimakasih atas dukungan dan saran yang diberikan.
4. Bapak Mohammad Husni M.Kom., selaku dosen pembimbing I Tugas Akhir ini dan sekaligus dosen wali saya. Terima kasih atas bimbingan dan saran serta dorongan yang telah banyak diberikan.
5. Bapak Ir. Suhadi Lili, selaku dosen pembimbing II Tugas Akhir ini. Terimakasih atas bimbingan, dan saran serta dorongan yang telah banyak diberikan.
6. Seluruh staf pengajar Jurusan Teknik Informatika. Terima kasih atas ilmu dan bimbingan yang telah diberikan selama perkuliahan.

7. Seluruh staf karyawan dan tata usaha Jurusan Teknik Informatika, terutama Mas Yudhi yang telah banyak membantu permasalahan administrasi selama saya kuliah.

8. Dan Semua pihak yang tidak dapat penulis sebutkan satu persatu yang telah secara langsung maupun tidak langsung membantu kelancaran penulis selama menuntut ilmu di Jurusan Teknik Informatika.

DAFTAR ISI

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR GAMBAR

BAB I

PENDAHULUAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada era internet ini bank menyediakan layanan pembayaran tagihan bulanan nasabah di merchant relasi, pembelian voucher dan transfer melalui internet. Di lain pihak dengan semakin berkembangnya pemakaian e-commerce orang-orang yang bertransaksi dengan e-commerce semakin merasakan pentingnya bank. Sebagai alat pembayaran untuk transaksi e-commerce biasa digunakan transfer melalui bank atau dengan menggunakan credit card. Namun kedua alat pembayaran ini masih memiliki banyak kelemahan. Kelemahan transfer melalui bank yaitu tidak praktis karena pihak penjual tidak tahu secara langsung apakah customer sudah melakukan pembayaran atau belum sehingga pembeli masih harus mengirimkan atau mem-fax bukti transfer dan ini memakan waktu yang lama dan atau membutuhkan dana tambahan. Akibatnya resiko yang ditanggung oleh merchant menjadi lebih besar. Sedangkan kelemahan pembayaran dengan menggunakan credit card adalah lemahnya keamanannya. Walaupun website tersebut menggunakan SSL(Secure Socket Layer), namun data credit card yang disimpan pada database perusahaan e-commerce. Hal ini memungkinkan data ini dimanfaatkan oleh orang-orang yang tidak berhak sehingga merugikan customer.

Layanan lain dari bank berkaitan dengan fungsinya sebagai alat pembayaran adalah layanan pembayaran tagihan nasabah di merchant-merchant relasi bank. Namun mekanisme pembayaran tagihan yang ada sekarang memiliki beberapa kelemahan, yaitu, menurut salah satu pegawai bank swasta, merchant relasi harus memiliki rekening di bank dan menggunakan sistem batch dimana data disetor setiap kurun waktu tertentu sehingga data tagihan disimpan di database bank akibatnya data yang dimiliki bank hanyalah cuplikan dari data asli dan bukanlah data terbaru sehingga memungkinkan pembayaran tagihan lebih dari sekali dan mempersulit proses pengeditan apabila ada data yang salah.

Bank juga menyediakan pembelian voucher handphone melalui bank. Namun, menurut salah satu pegawai bank swasta, pihak mekanisme pembelian voucher ini juga menggunakan sistem batch, dimana data disetor setiap kurun waktu tertentu dan data voucher handphone ini harus disimpan di bank. Akibatnya bila persediaan voucher pada bank habis, bank harus mengisikan lagi data yang baru ke database bank sehingga memboroskan waktu dan tenaga kerja.

1.2 Permasalahan

Permasalahan-permasalahan yang dihadapi adalah :

1. Bagaimana merancang dan membuat sistem yang memungkinkan customer membayar secara on-line karena tidak perlu meninggalkan data credit cardnya di merchant, di mana bila data ini disimpan di merchant ada kemungkinan data ini bisa dimanfaatkan oleh pihak yang tidak berwenang.

2. Bagaimana merancang dan membuat sistem yang memungkinkan perusahaan merchant untuk tahu dengan cepat dan yakin tentang status pembayaran customernya sehingga proses transaksi e-commerce bisa berlangsung dengan lebih cepat.
3. Bagaimana merancang dan membuat sistem bank yang menyediakan layanan perbankan khususnya pembayaran berbasis internet ke nasabah.
4. Bagaimana merancang dan membuat sistem bank dan sistem merchant terintegrasi secara on-line dengan menggunakan standard-standard protocol yang terbuka.
5. Bagaimana merancang dan membuat sistem yang otorisasi pembayaran antara bank dengan merchant dapat dilakukan secara langsung.

1.3 Batasan Permasalahan

Dalam penyusunan Tugas Akhir ini, penulis memfokuskan pada permasalahan dengan ruang lingkup sebagai berikut :

1. Pembahasan tugas akhir ini menitik beratkan pada fungsi bank sebagai alat pembayaran, bukan pada fungsi bank untuk melayani kebutuhan perbankan masyarakat.
2. Implementasi perangkat lunak mengabaikan proses-proses *back office*, seperti pengisian data nasabah, pengisian data pelanggan merchant, pengisian tagihan bulanan, pengisian data barang dan lain-lain, karena proses tersebut tidak

sesuai dengan ruang lingkup tugas akhir ini.

3. Implementasi perangkat lunak ini tidak menyediakan layanan untuk pembuatan PIN baru nasabah. Penginputan ini dianggap dilakukan di sistem lain.
4. Implementasi perangkat lunak ini tidak menitik beratkan pada sistem merchant maupun e-commerce melainkan hanya sistem bank sebagai alat pembayaran. Implementasi merchant dan e-commerce yang ada hanya membantu memperlihatkan jalannya sistem.
5. Implementasi perangkat lunak selain bank sebagai alat pembayaran, seperti registerasi merchant, ganti pin, informasi saldo, dan lain-lain, hanya untuk membantu pelaksanaan uji coba.

1.4 Tujuan dan Manfaat

Tujuan dari tugas akhir ini adalah merancang dan membuat perangkat lunak yang membantu bank untuk mempermudah dan memperlancar bank menjalankan fungsinya sebagai alat pembayaran.

Manfaat dari Tugas Akhir ini antara lain adalah :

1. Memberikan alternatif pembayaran untuk transaksi e-commerce.
2. Integrasi sistem bank dan merchant menjadi lebih kuat.
3. Membantu menekan resiko merchant.

4. Lebih murah bagi merchant karena mengurangi biaya administrasi, seperti untuk kertas, biaya mem-fax, dan lain-lain.
5. Pembeli bisa melakukan transaksi dengan lebih cepat.
6. Meningkatkan kepuasan pembeli/pelanggan.

1.5 Metodologi

Metodologi yang digunakan dalam penyusunan Tugas Akhir ini adalah sebagai berikut :

1. Studi literatur

   Mencari, mempelajari dan merangkum berbagai macam literatur yang berkaitan dengan rumusan masalah, teori-teori yang berhubungan dengan sistem yang akan dibangun, desain sistem, dan bahasa pemrogramannya.

2. Perancangan perangkat lunak

   Pada tahap ini dilakukan perancangan terhadap perangkat lunak yang meliputi diagram sistem, proses-proses yang akan dilaksanakan, dan penentuan rancangan antar muka berdasarkan studi pustaka yang telah dilakukan.

3. Pembuatan perangkat lunak

   Pada tahap ini dilakukan implementasi dari tahap perancangan sistem.

4. Pengujian dan Evaluasi perangkat lunak

   Pada tahap ini program yang telah dibuat diuji kebenarannya dengan menggunakan data simulasi yang telah dipersiapkan sebelumnya.

Selanjutnya, hasil dari pengujian program akan dievaluasi untuk menentukan keamanan, ketangguhan dan kontrol akses dari program dan menentukan perlu tidaknya dilakukan modifikasi pada program.

5. Penulisan Tugas Akhir

Pada tahap terakhir ini disusun buku sebagai dokumentasi dari pelaksanaan Tugas Akhir.

1.6 Sistematika Penulisan

Sistematika penulisan yang ada dalam buku ini adalah sebagai berikut :

Bab I Pendahuluan

Bab ini berisi berisi latar belakang, permasalahan, ruang lingkup permasalahan, tujuan dan manfaat, serta metodologi dan sistematika penulisan.

Bab II Dasar Teori

Bab ini berisi pembahasan tentang dasar ilmu dan teori-teori yang menunjang perancangan dan pembuatan sistem.

Bab IV Perancangan dan Pembuatan Lunak

Bab ini berisi penjelasan langkah-langkah dalam perancangan sistem berikut implementasi hasil perancangan yang meliputi arsitektur sistem, proses-proses yang dilakukan dan juga antar muka aplikasi.

Bab V Ujicoba dan Evaluasi

Bab ini berisi hasil-hasil uji coba yang dilakukan terhadap sistem disertai

evaluasi terhadap hasil-hasil ujicoba tersebut.

Bab VI Penutup

Bab ini berisi kesimpulan yang dapat diambil dari pelaksanaan Tugas Akhir dan saran untuk pengembangan selanjutnya.

BAB II

TEORI PENUNJANG

BAB II
TEORI PENUNJANG

Pada bab ini dibahas mengenai dasar ilmu dan teori-teori yang menunjang permasalahan dalam Tugas Akhir ini. Pembahasan pertama mengenai XML, kemudian SOAP, Web Service, dan terakhir tentang Microsoft.NET.

2.1 XML

Extensible Markup Language (XML) memberikan jalan untuk mendeskripsikan struktur data. Tidak seperti label pada HTML, dimana peran utamanya adalah untuk mengontrol tampilan dan data yang ditampilkan, label pada XML digunakan untuk mendefinisikan struktur dan tipe data dari data itu sendiri.

XML mengunakan satu set label untuk menggambarkan elemen data. Setiap elemen mengenkapsulisasi satu bagian data walaupun data ini simpel atau sangat kompleks. Anda dapat mendefinisikan sekumpulan label XML yang tidak terbatas.

XML adalah sederhana, *platform* yang mandiri dan adalah standard yang sudah diadaptasi secara luas. Keuntungan dari XML dibandingkan HTL adalah XML memisahkan tampilan antar muka dari struktur data. Pemisahan data ini dari

presentasi memungkinkan integrasi data dari berbagai macam sumber.

Berikut ini adalah beberapa poin penting mengenai hubungan XML dan HTML :

- XML bukanlah pengganti HTML; kenyataannya, XML dapat dianggap sebagai pelengkap HTML. XML dan HTML memliki tujuan yang berbeda : HTML dirancang untuk menampilkan data dan terfokus pada bagaimana data tersebut terlihat, sedangkan XML dirancang untuk mendeskripsikan data dan terfokus pada data itu sendiri.
- Seperti HTML, XML tidak melakukan apapun. Label XML dapat digunakan untuk mendiskripsikan struktur dari suatu barang, tapi ia tidak mengandung kode yang dapat digunakan untuk memprosesnya. Harus ada orang lain yang menulis kode untuk benar-benar melakukannya dengan format data XML.
- Tidak seperti HTML, label XML didefinisikan oleh pembuat skema atau dokumen dan tidak terbatas. Label HTML sudah didefinisikan terlebih dahulu; pembuat HTML hanya bisa menggunakan label yang sudah didukung oleh standard HTML [1].

Di dalam C# anda bisa mendokumentasikan kode yang anda tulis menggunakan XML. C# adalah satu-satunya bahasa pemrograman di Visual Studio.NET 7.0 dengan keistimewaan ini [2].

2.2 SOAP

SOAP (Simple Object Access Protocol) adalah simpel, termasuk kelas ringan protokol dasar XML untuk menukarkan struktur dan tipe informasi dalam Web. Secara keseluruhan tujuan rancangan SOAP adalah untuk membuatnya sesederhana mungkin, dan menyediakan sebuah kegunaan minimum. Protokol ini mendefinisikan sebuah rangka pemesanan yang tak mengandung semantik aplikasi atau transport. Hasilnya, protokol adalah modular kelas tinggi dan sangat luas.

Dengan menjalankan standard protokol transport, SOAP bisa mempengaruhi arsitektur Internet menarik yang dibuka dan mendapat kemudahan penerimaan oleh sistem yang berubah-ubah mampu didukung oleh kebanyakan standard dasar Internet. Anda dapat melihat infrastruktur membutuhkan untuk mendukung sebuah SOAP. Sebagai contoh Web Service masih memfasilitasi akses universal ke pembangunan layanan dengan SOAP.

Spesifikasi protokol SOAP terdiri dari empat bagian utama. Bagian pertama medefinisikan amplop *mandatory* untuk mengenkapsulasi data. Amplop SOAP mendifinisikan sebuah pesan SOAP dan merupakan unit dasar untuk pertukaran antar prosesor pesan SOAP. Ini adalah satu-satunya bagian *mandatory* dari spesifikasi.

Bagian kedua dari spesifikasi protokol SOAP mendefinisikan optional aturan data *encoding* untuk merepresentasikan aplikasi-tipe data terdefinisi dan

grafik yang ditunjuk, dan sebuah model seragam untuk menserialisasi model data yang tidak sintatik.

Bagian ketiga mendefinisikan sebuah pola pertukaran pesan RPC-*style* (*request/response*). Setiap pesan SOAP adalah transmisi searah. Meskipun akar SOAP adalah di RPC, ia tidak terbatas untuk menjadi sebuah mekanisme permintaan/respon. Webservice kadang mengkombinasikan pesan SOAP untuk mengimplementasikan pola seperti itu, tetapi SOAP tidak mengamanatkan sebuah pola pertukaran pesan dan bagian ini juga optional.

Spesifikasi keempat mendefinisikan ikatan antara SOAP dan HTTP. Bagaimanapun juga, bagian ini juga optional. Anda dapat menggunakan SOAP pada kombinasi dengan protokol transport lain atau mekanisme lain yang mungkin untuk mengangkut amplop SOAP, termasuk SMTP, FTP, atau bahkan sebuah *floopy disk* [10].

2.3 Web Service

2.3.1 Sekilas Mengenai Web Service

Sebuah Web Service adalah *entity* yang dapat diprogram yang menyediakan elemen kegunaan yang istimewa, dan ia dapat diakses untuk beberapa sistem terpisah melalui standard Internet, seperti XML dan HTTP. Web Service bergantung sekali pada sambutan luas XML dan standard Internet lainnya

untuk membuat sebuah infrastruktur yang mendukung *interoperability* aplikasi pada level yang memecahkan banyak masalah yang sebelumnya mengganggu, seperti mencoba.

Sebuah Web Service dapat digunakan juga secara internal dengan satu aplikasi atau dibuka secara eksternal melalui Internet untuk digunakan oleh banyak aplikasi. Karena ia dapat diakses melalui standard tampilan antar muka, sebuah Web Service memungkinkan sistem yang berbeda-beda untuk bekerja bersama sebagai satu web.

Malahan mengikuti kemampuan umum dari *portability* kode, Web Service menyediakan sebuah solusi aktif untuk memungkinkan *interoperability* data dan sistem.Web Service mempergunakan basis XML *messaging* sebagai arti fundamental komunikasi data untuk membantu menjembatani perbedaan yang ada antar sistem yang menggunakan model komponen, sistem operasi, dan bahasa pemrograman yang tidak sama. Pengembang dapat membuat aplikasi yang dirangkai bersama Web Service dari sebuah variasi sumber dalam jalan yang sama dengan pengembang tradisional menggunakan kompone ketika membuat aplikasi terdistribusi.

Satu dari karakteristik dari Web Service adalah abstrak dengan derajat tinggi yang tetap ada antara implementasi dan konsumsi layanan. Dengan menggunakan basis XML *messaging* sebagai mekanisme oleh dimana layanan dibuat dan diakses, kedua Web Service Client dan Web Service *Provider* dibebasikan dari kebutuhan untuk mengetahui input, output dan lokasi diluar dari

yang dibutuhkan.

Web Service memungkinkan sebuah era baru dari pengembangan distribusi aplikasi. Tidak lagi ada persoalan perang model objek atau kontes kecantikan bahasa pemrograman. Ketikan sistem digandengkan dengan erat menggunakan infrastruktur pemilik, ia dikerjakan juga dengan mengorbankan *interoperability* aplikasi. Web Service mengantarkan *interoperability* pada level yang baru sama sekali yang meniadakan seperti rival banyaknya produktivitas. Revolusi berikutnya promosi Internet, Web Service akan menjadi struktur dasar yang menghubungkan semua peralatan komputer bersama-sama [3].

2.3.2 Skenario Web Service

Untuk mengerti dengan lebih baik konstribusi berharga Web Service, akan sangat membantu untuk menguji beberapa skenario dimana Web Service dapat ikut berperan penting.

2.3.2.1 Integritas Aplikasi

Web Service dapat digunakan dalam cara sangat komposit untuk integrasi sebuah grup aplikasi menarik yang tampak terpisah. Luasnya pemakaian kebiasaan perangkat lunak di seluruh departemen dari kebanyakan perusahaan sebenarnya telah menghasilkan banyak aturan yang berguna, tapi terisolasi pada data dan logik bisnis. Seharusnya banyaknya keadaan mungkin untuk dikembangkan dan pengembangan alami teknologi, sekarang ia sudah betul-betul

menjadi sebuah permintaan yang menakutkan untuk membangun sebuah kumpulan orang-orang yang berguna dari aplikasi ini.

Dengan Web Service itu mungkin untuk membuka kegunaan dan data dari setiap aplikasi yang menarik sebagai sebuah Web Service. Anda dapat membuat sebuah gabungan aplikasi yang menggunakan koleksi Web Service ini untuk memungkinkan *interoperability* antar unsur pokok aplikasi.

2.3.2.2 Solusi Aliran Kerja

Web Service tidak terbatas untuk pemanggilan *procedure* jarak jauh. Web Service juga memungkinkan sebuah mekanisme yang sangat kuat dengan aplikasi yang merupakan *end-to-end* solusi aliran kerja dapat dibuat. Solusi serupa cocok untuk skenario jangka panjang seperti dalam transaksi bisnis-ke-bisnis.

Kerangka BizTalk menyediakan sebuah lapisan ekstra yang mendefinisikan mekanisme untuk identifikasi dan pengalamatan pesan, mendefinisikan waktu hidupnya, mengepak mereka dengan tambahan-tambahan, dapat dipercaya engantarkan mereka ke tujuan mereka, dan mengamankan kandungan mereka untuk pembuktian keaslian, integritas, dan *privacy*.

2.3.3 Infrastruktur Web Service

Web Service harus menjadi agnostis berkenaan dengan pilihan sistem operasi, model objek, dan bahasa pemrograman untuk menyukseskan

keheterogenan Web. Juga, untuk Web Service untuk menikmati pengangkatan yang sama luasnya dengan teknologi berbasis Web, mereka harus :

- Bergandengan dengan longgar : Dua sistem betul-betul dipertimbangkan bergandengan dengan longgar juka satu-satunya tugas mengganggu kedua sistem untuk mengerti *self-describing* tersebut, pesan dengan basis teks. Sistem yang bergandengan dengan ketat, di lain pihak, menentukan jumlah pengeluaran tambah umum yang penting untuk memungkinkan komunikasi dan membutuhkan pengertian yang lebih besar antar sistem.
- Komunikasi di mana-mana : Tidak seperti seseorang yang akan membangun sebuah sistem operasi atau pada masa dekat-dekat ini yang tidak terhubung ke Internet, oleh karena itu menyediakan layanan chanel komunikasi di mana-mana. Sebagai contoh, kemampuan untuk konek hampir semua sistem atau peralatan ke Internet akan memastikan sistem dan peralatan seluruhnya bisa ke sistem atau peralatan terkoneksi ke Internet.
- Format Data Universal : Dengan pemakaian yang ada, membuka standard lebih dari pemilik, metode komunikasi dalam lingkaran tertutup, beberapa sistem mendukung standard pembukaan yang sama mampu mengerti Web Service. Menggunakan *self-describing*, pesan berbasis teks yang Web Service dan kliennya dapat berbagi tanpa khawatir tentang apa aturan dari sistem lain memungkinkan komunikasi antara sistem tak dikenal dan berbeda. Web Service mencapai kemampuan ini dengan menggunakan

XML.

Web Service menggunakan sebuah infrastruktur yang menyediakan layanan berikut :

1. Mekanisme penemuan untuk melokasikan Web Service
2. Sebuah deskripsi service untuk mendefinisian bagaimana menggunakan service tersebut.
3. Standard *wire format* dengan komunikasi yang mana [8].

Gambar 2.1 : Infrastruktur Web Service

2.3.4 Deskripsi Web Service

Infrastruktur Web Service ditemukan pada komunikasi via pesan berbasis XML yang menuruti dengan publikasi deskripsi Web Service. Deskripsi layanan adalah sebuah dokumen XML yang ditulis dengan sebuah *grammar* XML, yang disebut WSDL *(Web Service Description Language)* yang mendefiniskan format pesan yang dimengerti oleh Web Service. Deskripsi layanan melayani sebagai persetujuan yang mendefinisikan kelakuan sebuah Web Service dan instruksi potensial *client* bagaimana berinteraksi dengannya. Kelakuan sebuah Web Service dideterminisikam dengan pola penyuratan yang *service* definisikan dan dukung. Pola ini secara konsep mengatur apa yang service konsumen dapat harapkan terjadi ketika sebuah pesan dengan format yang pantas diserahkan ke Web Service.

Skema yang mendefiniskan format pesan SOAP dapat didefinisikan secara internal untuk deskripsi aktual Web Service atau, mereka dapat mendefiniskan secara eksternal dan diimpor ke deskripsi Web Service.

Sebagai tambahan untuk definisi format pesan dan pola pesan, deskripsi *service* juga mengandung alamat yang berhubungan dengan poin masuk Web Service. Format alamat ini akan cocok untuk protokol untuk akses *service*, seperti URL untuk HTTP atau sebuah alamat e-mail untuk SMTP [10].

2.4 Microsoft.NET

2.4.1 Pendahuluan

Pada konferensi *developer profesional* di Orlando Juli 2001, Microsoft mengeluarkan arsitektur terbarunya, .NET. Fiturnya yang beragam dipresentasikan oleh beberapa pembicara. Namun fitur yang paling menarik dari .NET adalah pada platform pengembangan software, bahasa dan protokol yang digunakan. Dengan memperkenalkan .NET, Microsoft menyediakan platform pengembangan yang memungkinkan pengembangan aplikasi web yang dapat saling bekerjasama antara satu dan yang lain, menggunakan arsitektur yang baru.

2.4.2 Maksud dan Tujuan .NET

.NET dibangun oleh Microsoft dengan ambisius baik dari segi teknis maupun strategis. Teknologi .NET pada dasarnya berbeda jauh dari teknologi DNA 2000 yang sudah ada sebelumnya; namun cenderung merupakan teknologi baru yang canggih. Namun teknologi .NET yang keseluruhan baru ini tidak menjamin kompatibilitas dengan teknologi yang sudah ada sebelumnya. .NET menawarkan dukungan lebih dari dua puluh tujuh bahasa pemrograman, yang berbagi hirarki *class* yang menyediakan layanan dasar. Aplikasi yang dibangun dengan .NET tidak akan lagi berjalan pada *native-code*, mengabaikan dominasi *code* pada Intel x86 dan menggantinya dengan bahasa penengah yang disebut MSIL (Microsoft Intermediary Language) yang merupakan kumpulam *virtual*

machine yang disebut Common Language Runtime (CLR).

Sebagai tambahan, .NET menggunakan standar format data XML secara intensiv, dan memperkuat penggunaan protokol SOAP. Berkat SOAP, Microsoft berharap membuat era baru pemrograman yang beralih dari orientasi berbasis komponen kepada aplikasi yang berbasis layanan (*services*). SOAP dan WebService adalah kunci dari platform .NET ini.

Teknologi Microsoft DNA yang ada sekarang ini nampaknya akan banyak berubah dengan kehadiran teknologi .NET, antara lain :

- Web Server IIS

  Konsep *multi-threaded* Web server IIS tidak lagi akan digunakan, dan akan diganti dengan konsep *multi-process* yang telah lama digunakan pada Web server Apache.

- ASP

  Teknologi ASP yang telah ada ikut mendukung teknologi ASP.NET (dulu ASP+), dimana *script* yang diterjemahkan akan diganti dengan program yang telah terkompilasi pada waktu pertama kali *script* tersebut dieksekusi.

- Win32 (ATL & MFC)

  Win32 termasuk ATL dan MFC akan diganti dengan sekumpulan kelas-kelas Framework dasar yang koheren, dan berjalan diatas platform CLR.

- VB

  Kelahiran *tools* baru VB.NET tidak lagi menjamin kompatibilitas dengan VB6, seiring bahasa ini menerima banyak kontribusi perubahan seperti

inheritance, ... , dalam rangka mengikuti standar *Common Language Specification (CLS).*

- COM+ 2.0

  COM+ 2.0 adalah model komponen terdistribusi yang sepenuhnya baru yang tidak mewarisi elemen dari COM/DCOM/COM+. Untuk ini COM+ 2.0 tidak lagi menggunakan *Registry Windows* untuk meregister komponen lokal atau remote; *deployment* komponen di .NET kembali seperti saat lampau dimana menginstall program berarti menyalin file pada suatu directory dan uninstall semudah menghapus file tersebut.

- C#

  Bahasa pemrograman baru yang disebut C# (C sharp) telah hadir; ini adalah bahasa pemrograman modern yang berorientasi objek. C# merupakan penggabungan dari bahasa C++ dan Java, didesain oleh Aders Hejlberg, arsitek dari beberapa bahasa dan tools pada perusahaan Borland, termasuk bahasa Delphi yang terkenal.

- SOAP dan WebService

  Pada .NET diperkenalkan model pemrograman baru yang berubah secara fundamental menggunakan SOAP dan WebService; membuka paradigma baru bagaimana suatu aplikasi didesain dan membuka jalan bagi profesi yang baru : mendapatkan provisi secara online dengan menyediakan layanan WebServices.

Perubahan secara teknis, berkaitan dengan fakta bahwa plaftorm .NET akan berada dibawah standar pengawasan organisasi independen seperti W3C, IETF dan ECMA, menimbulkan banyak analisis bahwa teknologi ini bersifat terbuka. Dengan menggunakan .NET, Internet akan terdiri dari aplikasi web yang terhitung banyaknya, yang saling dapat berkomunikasi membentuk jaringan pertukaran layanan global. Webservice yang berbasiskan XML dan SOAP ini pada awalnya diajukan oleh IETF pada DevelopMentor, Microsoft dan Userland Software. Saat ini sejumlah besar vendor termasuk IBM sangat intensive terlibat dengan SOAP. WebService tidak hanya akan digunakan untuk mengembangkan pada Internet, namun juga dapat diaplikasikan untuk mengembangkan Sistem Informasi pada perusahaan (*enterprises*). Sistem Informasi perusahaan dapat juga dipandang sebagai jaringan (*network*) antara *front-office* dan *back-office* yang saling berkomunikasi menggunakan SOAP.

Kehadiran WebService akan merubah pola pendapatan vendor-vendor besar seperti Microsoft dari model yang mengutamakan pendapatan dari penjualan produk software dalam boks menjadi model langganan, dan sewa layanan. Oleh karena itu SDK dan kompiler .NET akan disediakan secara gratis untuk dipergunakan secara luas seperti Java.

2.4.3 Arsitektur .NET

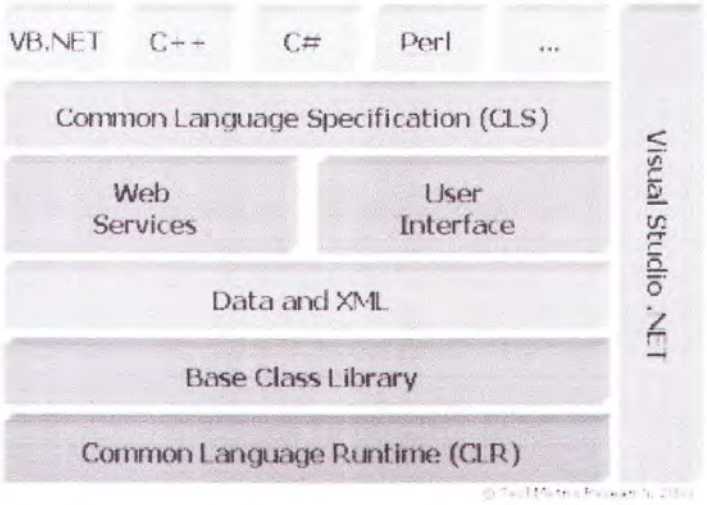
Apa yang dimaksud dengan .NET ? Produk-produk Microsoft baik pada segmen dekstop maupun server akan mengunakan label ".NET", tak terkecuali Office, SQLServer dan Biztalk server. Kita akan melihat secara detail arsitektur komponen dan tools yang digunakan untuk mendesain dan membuat aplikasi web *enterprise.*

Arsitektur .NET dapat digambarkan sebagai berikut:

- Merupakan satu set layanan dasar (*common services*) yang dapat digunakan oleh beberapa bahasa pemrograman
- Layanan-layanan ini dieksekusi dalam bentuk *intermediate code* yang independen terhadap arsitektur dibawahnya.
- Beroperasi diatas *runtime (Common Language Runtime)* yang bertugas menangani *resource* dan memonitor proses eksekusi aplikasi.

Dengan membaca deskripsi diatas, seseorang mungkin mengatakan bahwa ini mirip dengan Java, dan memang diakui oleh Microsoft bahwa inspirasinya datang lewat Java.

Tujuan utama .NET adalah menyediakan bagi sarana bagi *developer* agar dapat mengembangkan aplikasi yang dapat saling berkomunkasi dengan WebService dari semua jenis terminal apapun seperti : Komputer , PDA, Handphone dan lainnya.



Gambar2.2 : Arsitektur umum .NET

2.4.3.1 .NET adalah *multi-language*

Dengan platform .NET, Microsoft akan menyediakan beberapa bahasa dan kompiler yang bersangkutan seperti C++, Jscript, VB.NET (alias VB7) dan C#, sebuah bahasa pemrograman baru yang muncul bersama dengan .NET.

Vendor-vendor pihak ketiga juga bekerjasama dengan Microsoft untuk mengembangkan sejumlah bahasa pemrograman agar dapat digunakan pada platform .NET, termasuk Cobol, Eiffel, CAML, Lisp, Python, SmallTalk. Rational Rose, perusahaan pengembang tool UML yang terkenal, juga bersedia memporting Java agar dapat digunakan pada .NET.

2.4.3.2 Aplikasi independen terhadap hardware

Semua bahasa pemrograman ini dikompile menjadi kode binari *intermediate*, yang independen terhadap hardware dan sistem operasi. Bahasa ini adalah MSIL (Microsoft Intermediate Language). MSIL dieksekusi pada *Common Language Runtime (CLR)*, yang pada dasarnya berfungsu seperti JVM pada platform Java. MSIL kemudian akan diterjemahkan menjadi kode mesin oleh kompiler Just in Time (JIT).

2.4.3.3 Aplikasi bersifat *portable*

Aplikasi yang dikompile sebagai kode *intermediate* disebut *Portable Executables* (Pes). Di masa mendatang Microsoft akan menawarkan implementasi penuh atau sebagian dari platform .NET pada sebagian besar arsitektur hardware dan software: PC Intel dengan sistem operasi Windows9x, Windows NT, Windows 2000, atau Windows versi 64bit mendatang, PDA berbasis mikrokontroller, PocketPC dengan sistem operasi WindowsCE, dan beberapa sistem operasi yang lain, tidak diragukan.

2.4.3.4 Semua bahasa pada .NET harus mengacu pada standar umum yang digunakan

Bahasa pemrograman sangat beragam. Awalnya, bahasa pemrograman baru diciptakan untuk memenuhi kebutuhan yang baru, seperti menyelesaikan

persoalan scientific, melakukan kalkulasi pada riset, atau untuk memenuhi aspek kehandalan dan keamanan aplikasi. Akibatnya, bahasa pemrograman yang ada sangat beragam: beberapa bersifat prosedural, yang lain berorientasi object, dan banyak perbedaan lainnya. Agar suatu bahasa dapat didukung oleh platform .NET, bahasa tersebut harus memenuhi syarat-syarat kemungkinan dan pembentukan sesuai pada daftar yang telah ditentukan yang disebut *Common Language Specification* (CLS). Untuk menambahkan suatu bahasa pada .NET harus memenuhi persyaratan CLS dan berikutnya mengembangkan kompiler yang dapat mengkompile dari bahasa tersebut menjadi MSIL.

Fakta bahwa semua bahasa pada .NET dikompile dalam format code *intermediate* juga berarti bahwa sebuah *class* yang ditulis dalam suatu bahasa dapat diturunkan pada bahasa pemrograman yang lain. Dimungkinkan pada satu bahasa untuk menginstankan suatu kelas yang dibuat menggunakan bahasa yang lain.

Bagaimana ini bisa dilakukan ? sebenarnya .NET yang dianggap memiliki dukungan banyak bahasa tersebut dapat dianggap hanya mendukung satu bahasa yakni MSIL. Karena bahasa apapun yang anda gunakan untuk memprogram akan dikompile menjadi bahasa MSIL [11].

BAB III

KEBUTUHAN DAN ANALISIS

BAB III
KEBUTUHAN DAN ANALISIS

Pada bab ini diidentifikasikan kebutuhan-kebuttuhan pada sistem bank dalam fungsinya sebagai alat pembayaran.

3.1 KEBUTUHAN

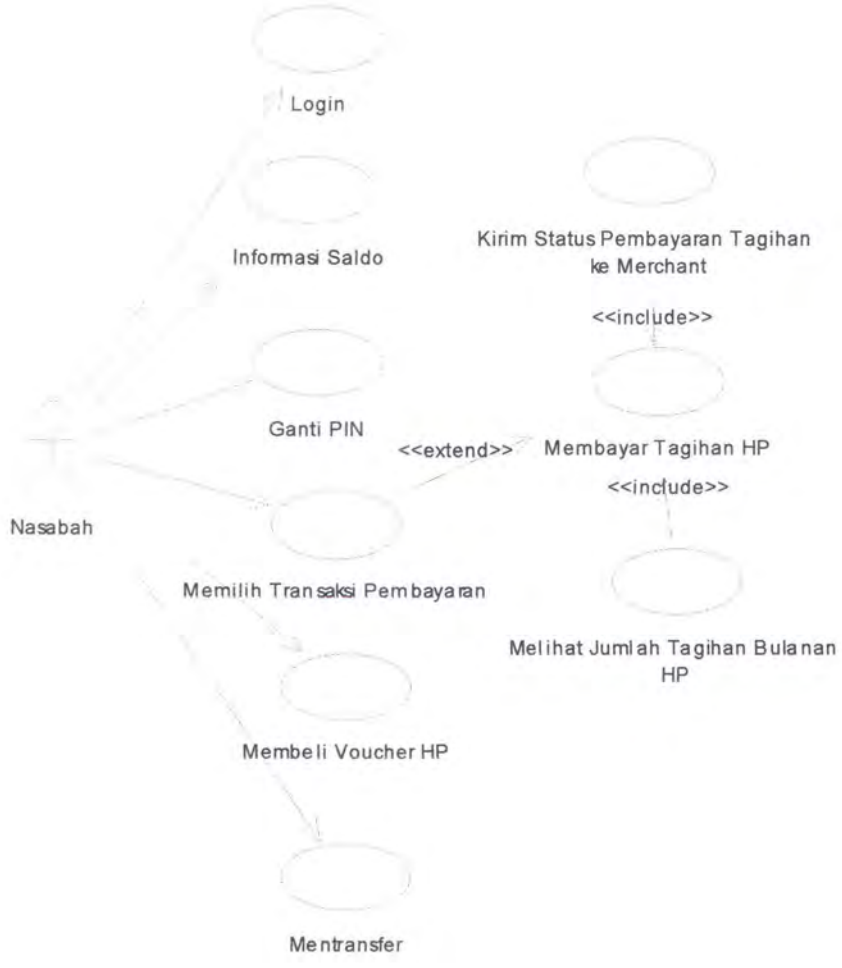
1. Untuk memperkuat integritas keseluruhan dalam sistem pembayaran dan pembelian voucher dibutuhkan integrasi antar sistem secara on-line, sehingga bila bank membutuhkan data, bank selalu mendapatkan data terbaru.
2. Untuk pembelian voucher, bank tidak perlu menyimpan data voucher di bank, melainkan langsung mengambil data voucher dari merchant.
3. Dalam transaksi e-commerce, pihak pembeli membutuhkan rasa aman dalam melakukan pembayaran, tidak seperti credit card yang datanya disimpan di database merchant melainkan yang data rekeningnya disimpan hanya di database bank, sehingga pembeli merasa lebih aman.
4. Dalam transaksi e-commerce, pihak merchant membutuhkan mekanisme validasi pembayaran oleh bank secara instan/saat itu juga, sehingga merchant dapat merasa yakin dalam menyetujui transaksi.

3.2 ANALISIS

Setelah diidentifikasikan kebutuhan-kebutuhan dari sistem ini maka berikut ini adalah analisa dari kebutuhan-kebutuhan tersebut :

1. Bagi nasabah bank :

   Sistem pembayaran harus menjaga agar tidak terjadi kondisi dimana nasabah membayar lebih dari sekali. Selain itu nasabah membutuhkan suatu mekanisme pembayaran yang aman untuk transaksi e-commerce, tanpa perlu memberikan data credit card/rekeningnya, cukup dengan nomer referensi transfer.



Gambar 3.1 : Use case diagram untuk nasabah bank

Dari use case pada gambar 3.1 dapat dilihat bahwa nasabah bank bisa melakukan :

a. Login, dilakukan nasabah untuk masuk ke menu khusus nasabah.

b. Informasi saldo, dilakukan oleh nasabah untuk mengetahui jumlah saldo rekeningnya saat itu.

c. Ganti pin, dilakukan oleh nasabah untuk mengganti pin website

nasabah.

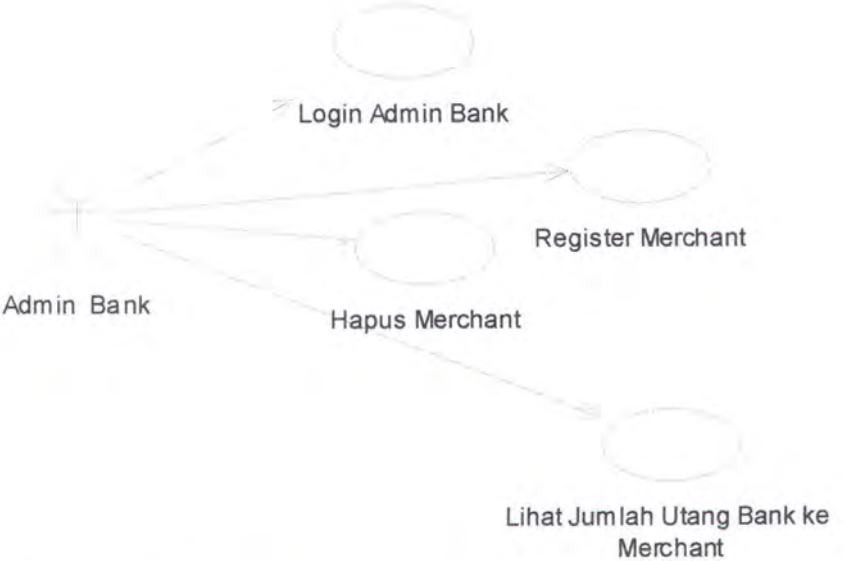
d. Membayar tagihan HP, dilakukan oleh nasabah untuk membayar tagihan handphonenya di merchant.

e. Membeli voucher HP, dilakukan oleh nasabah untuk membeli voucher handphone dengan memilih nama kartu handphone.

f. Mentransfer, dilakukan oleh nasabah untuk mentransfer sejumlah dana dari rekeningnya ke rekening nasabah lain di bank tersebut.

Dari sekian banyak hal tersebut tidak semuanya merupakan ruang lingkup dari sistem ini. Layanan ganti pin dan informasi saldo hanya merupakan layanan tambahan untuk membantu pelaksanaan uji coba.

2. Bagi administrator bank :



Gambar 3.2 : Use case diagram untuk administrator bank

Dari use case pada gambar 3.2 dapat dilihat bahwa administrator bank bisa melakukan :

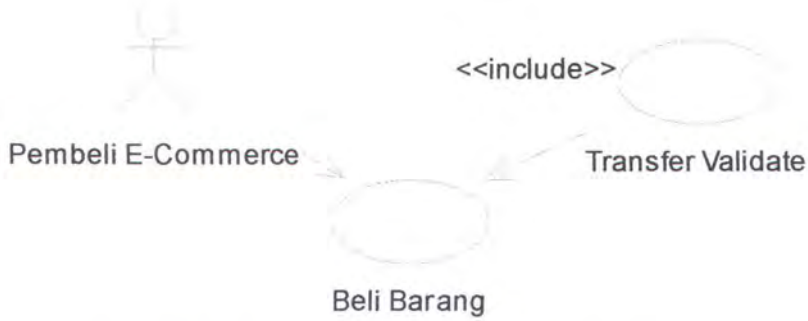
a. Login admin bank, dilakukan administrator bank untuk masuk ke menu khusus administrator bank.
b. Register merchant, dilakukan administrator bank untuk register merchant ke tabel merchant di database merchant.
c. Hapus merchant, dilakukan administrator bank untuk mengahapus data merchant dari database merchant.
d. Lihat jumlah utang bank ke merchant, dilakukan administrator bank untuk melihat berapa jumlah dana merchant di bank dari hasil transaksi pembayaran dan atau pembelian untuk merchant di bank tersebut untuk bulan itu.

3. Bagi pelanggan merchant/HP :

Gambar 3.3 : Use case diagram untuk pelanggan merchant/HP

Dari use case pada gambar 3.3 dapat dilihat bahwa pelanggan merchant/HP bisa melihat jumlah tagihan bulanan pelanggan.

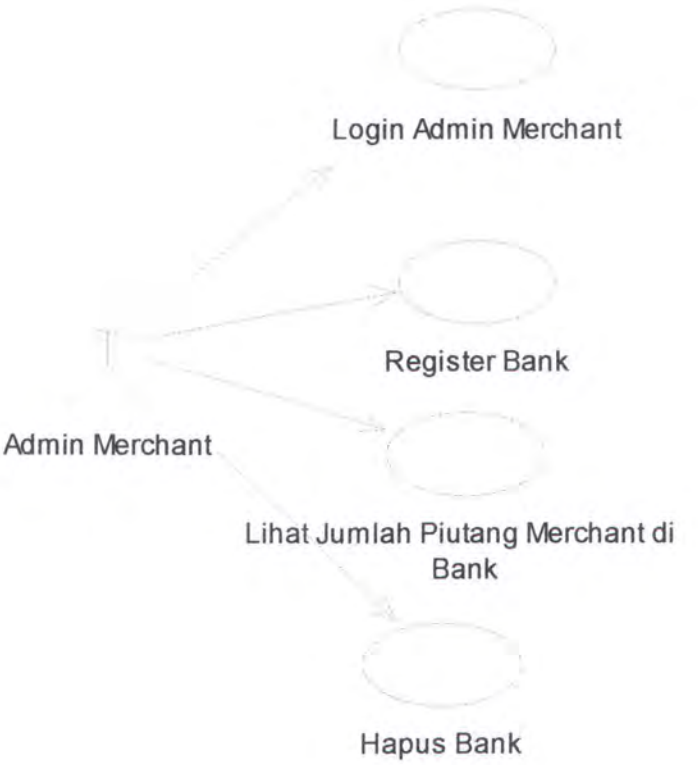
4. Bagi pembeli e-commerce :



Gambar 3.4 : Use case diagram untuk pembeli e-commerce

Dari use case pada gambar 3.4 dapat dilihat bahwa pembeli e-commerce bisa melakukan pembelian barang dan untuk itu dilakukan transfer validate untuk memvalidasi transfer pembeli ke rekening merchant sesuai dengan jumlah transaksi.

5. Bagi administator merchant :



Gambar 3.5 : Use case diagram untuk administrator merchant

Dari use case pada gambar 3.5 dapat dilihat bahwa administrator merchant bisa melakukan :

a. Login admin merchant, dilakukan oleh administrator merchant untuk masuk ke menu khusus administrator merchant.

b. Register bank, dilakukan oleh administrator merchant untuk meregister bank dan memasukkannya ke database merchant.

c. Hapus bank, dilakukan oleh administrator merchant untuk menghapus data bank dari database merchant.

BAB IV

PERANCANGAN DAN IMPLEMENTASI PERANGKAT LUNAK

BAB IV
PERANCANGAN DAN IMPLEMENTASI PERANGKAT LUNAK

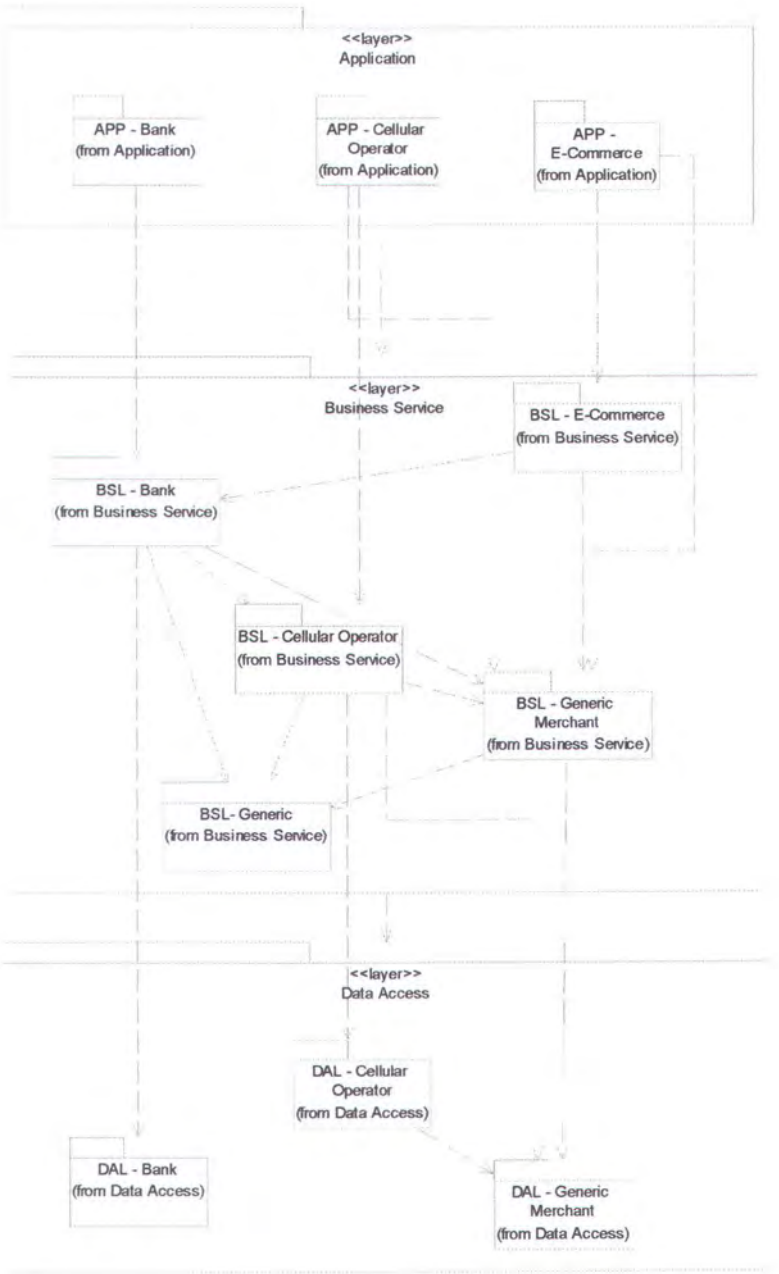
4.1 Perancangan

4.1.1 Arsitektur Sistem

Secara umum arsitektur sistem ini dapat dilihat pada gambar 4.1 di bawah ini. Ada tiga layer pada sistem ini, yaitu :

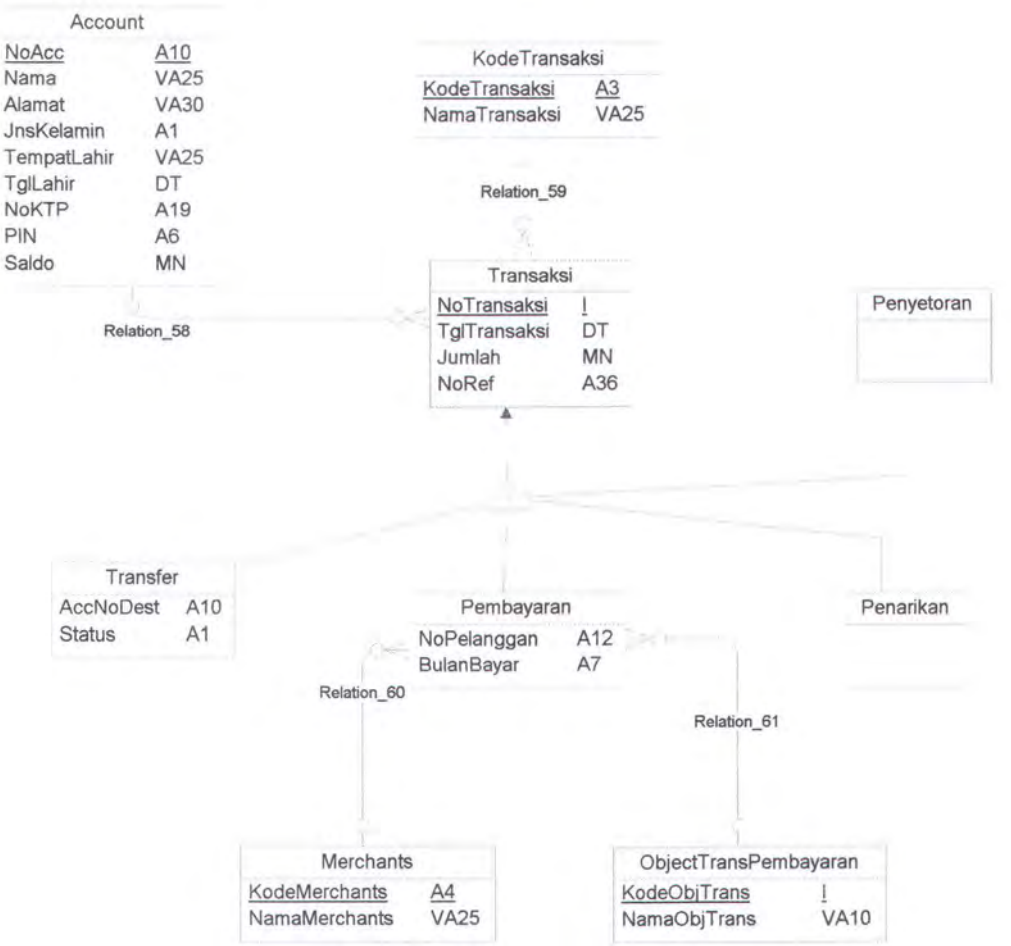
1. Application Layer, yaitu aplikasi-aplikasi yang ada pada sistem ini, yang menyediakan interaksi antarmuka dengan pengguna sistem ini.
2. Business Service Layer, yaitu class-class dan Web Service dengan Microsoft Visual C#.NET yang mengatur aspek-aspek bisnis dalam sistem ini. Pada business service layer ini dibagi menjadi lima kelompok, yaitu : BSL-Bank, BSL-E-Commerce, BSL-Cellular Operator (milik operator selular/Merchant Subscibe), BSL-Generic Merchant (business service yang terdapat pada merchant pada umumnya, baik merchant subscribe,contoh dalam kasus ini Cellular Operator, maupun merchant e-commerce, dan BSL-Generic (business service yang terdapat pada bank maupun merchant subscibe dan e-commerce).
3. Data Access Layer, yaitu tabel, *triger*, *stored procedure*, *function*, dan *view*, yang dibuat dengan menggunakan Microsoft SQL Server 2000, yang

berhubungan dengan data. Ada tiga kelompok pada Data Access Layer ini, yaitu DAL Bank, DAL Cellular Operator (data access yang hanya ada pada cellular operator), dan DAL Generic Merchant (data access yang terdapat pada semua merchant).



Gambar 4.1 : Arsitektur Sistem

4.1.2 Perancangan Data



Gambar 4.2 : Conceptual Data Model untuk Sistem Bank

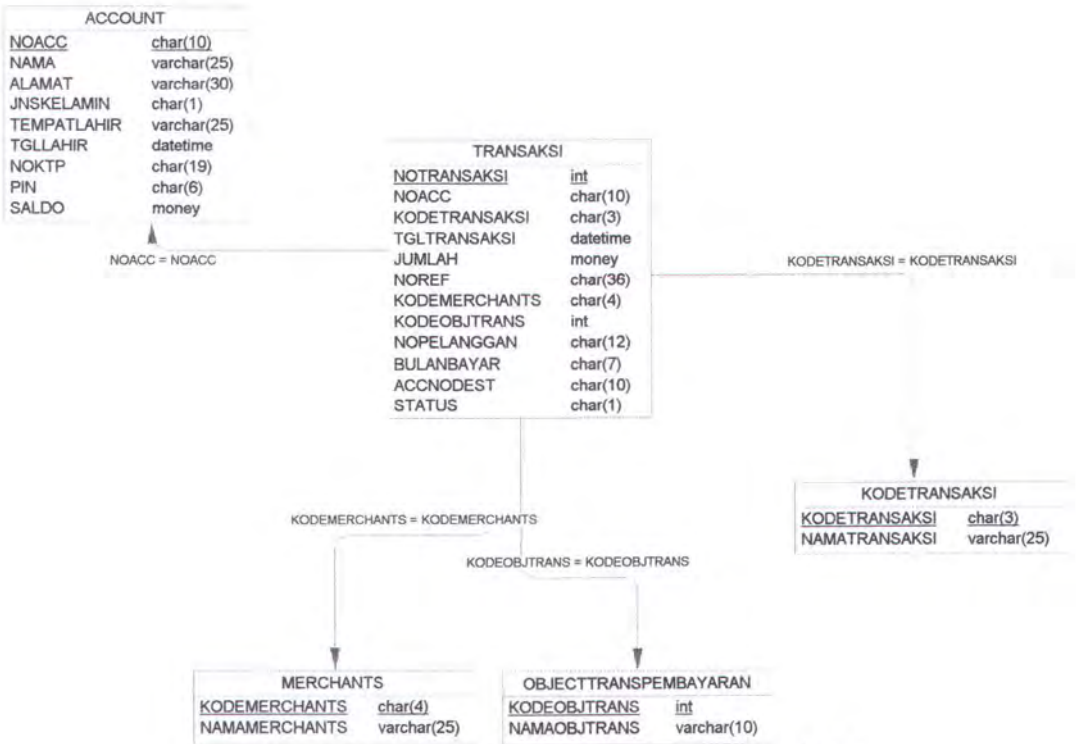
Pada database bank terdapat lima buah tabel, seperti bisa dilihat pada gambar 4.3. Tabel-tabel tersebut adalah :

1. Account, merupakan tabel yang menyimpan data *account*/rekening dari nasabah bank, termasuk saldo rekening dan pin nasabah untuk transaksi on-line melalui web.

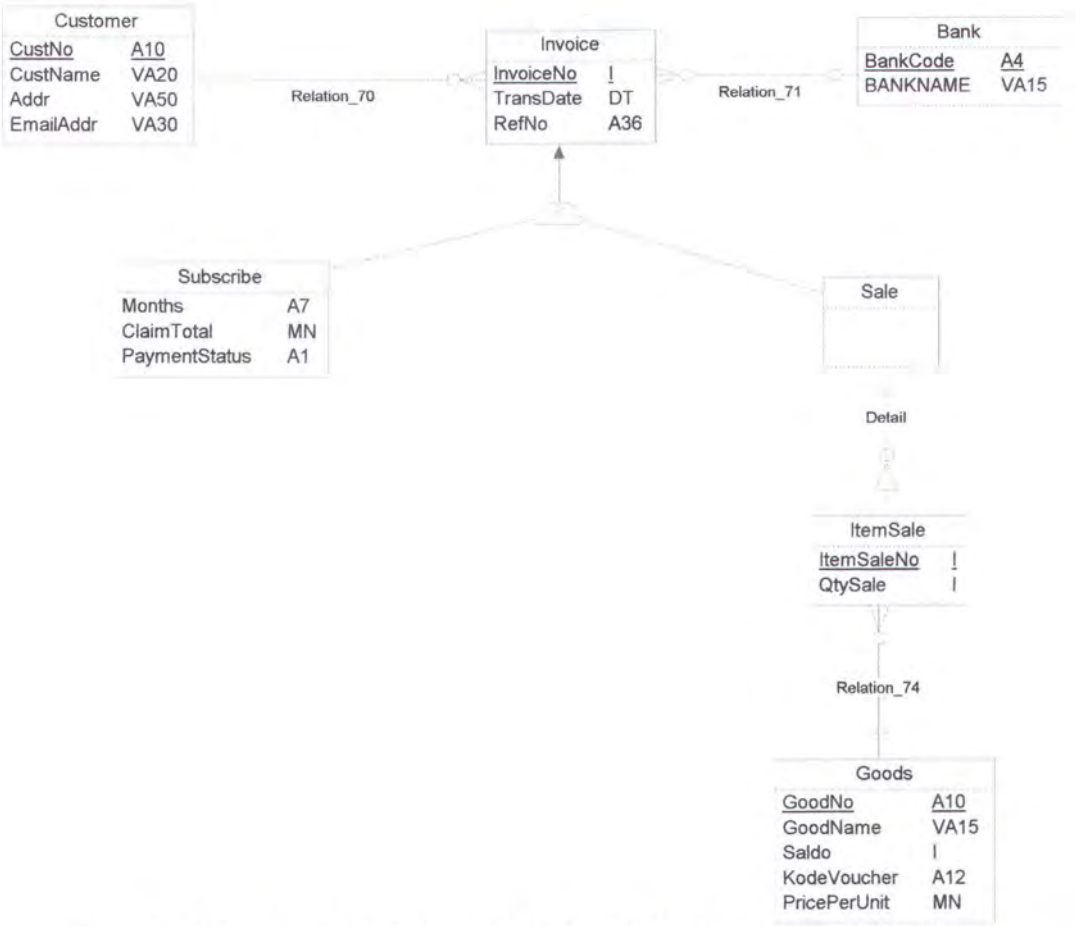
2. Transaksi, merupakan tabel yang menyimpan data transaksi nasabah,

termasuk transaksi pembayaran, pembelian, dan transfer. Sedangkan transaksi penyetoran dan penarikan tidak dibahas dalam sistem ini.

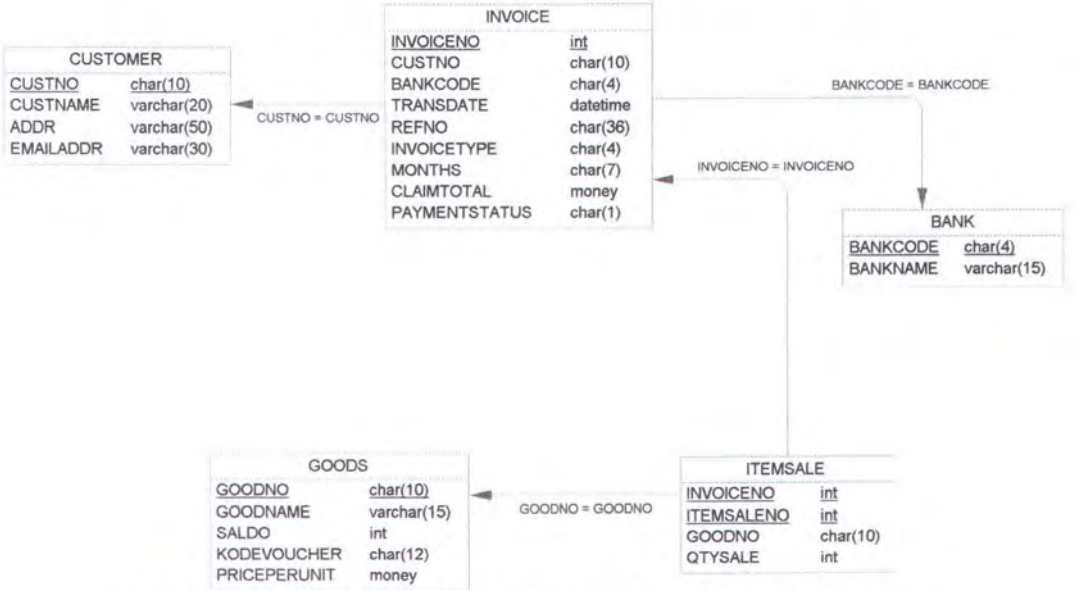
3. KodeTransaksi, merupakan tabel yang menyimpan kode transaksi, misalnya TRS untuk transaksi transfer.
4. Merchants, merupakan tabel yang menyimpan data merchant.
5. ObjectTransPembayaran, merupakan tabel yang menyimpan data object transaksi pembayaran, contohnya handphone untuk transaksi pembayaran handphone.



Gambar 4.3 : Model Data Fisik untuk Sistem Bank



Gambar 4.4 : Conceptual Data Model untuk Sistem Generic Merchant



Gambar 4.5 : Model Data Fisik untuk Sistem Generic Merchant

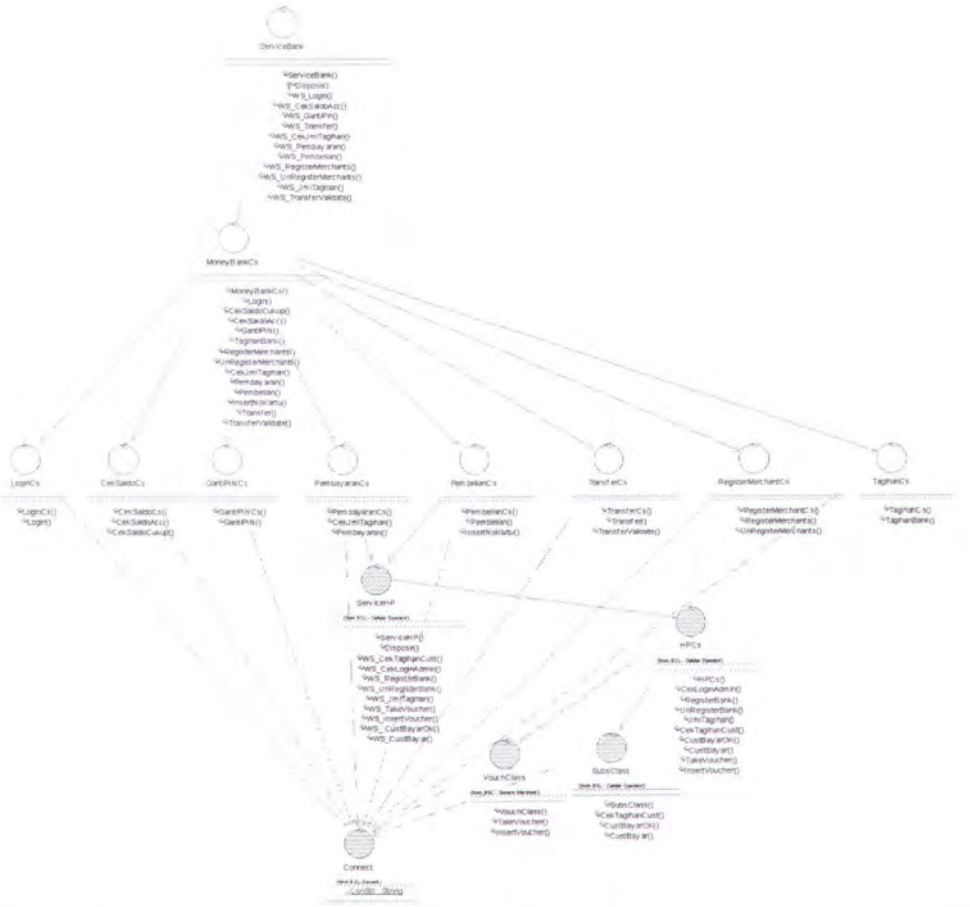
Pada database generic merchant terdapat lima buah tabel, seperti bisa dilihat pada gambar 4.5. Tabel-tabel tersebut adalah :

1. Customer, untuk menyimpan data *customer*/pelanggan/pembeli.
2. Invoice, untuk menyimpan data transaksi.
3. ItemSale, untuk menyimpan data detail transaksi pembelian.
4. Goods, untuk menyimpan data barang, dalam hal ini voucher.
5. Bank, untuk menyimpan data bank.

4.1.3 Perancangan Proses

Dalam sub bab ini dijelaskan tentang perancangan proses kerja sistem. Selain menjelaskan sistem bank sebagai alat pembayaran, juga dijelaskan dua sistem pendukung, yaitu sistem cellular operator dan sistem e-commerce, untuk memperlihatkan jalannya proses sistem bank sebagai alat pembayaran.

Sistem Bank



Gambar 4.6 : Class Diagram untuk Business Service Layer pada Sistem Bank

Seperti terlihat pada gambar 4.6, pada sistem bank terdapat sebelas class, termasuk satu Web Service, yaitu :

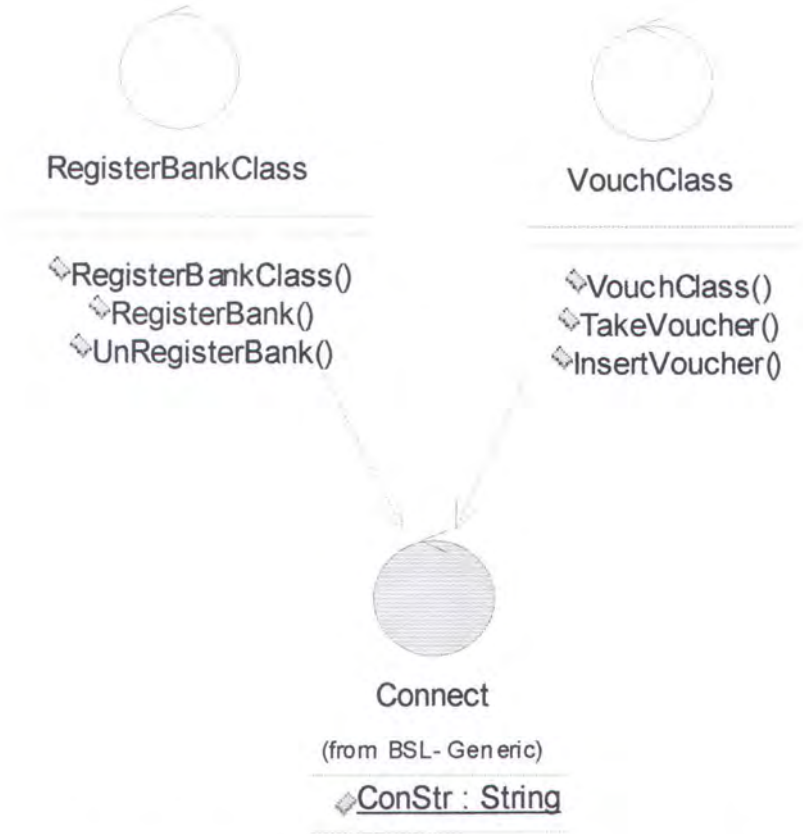
1. class Connect, merupakan class untuk koneksi ke database, dan terdapat pada semua sistem,
2. class LoginCs, merupakan class yang mengatur proses login,

3. class CekSaldoCs, merupakan class yang mengatur proses yang berhubungan dengan pengecekan saldo,
4. class GantiPINCs, merupakan class yang mengatur proses pergantian pin dari nasabah bank,
5. class PembayaranCs, merupakan class yang mengatur proses pembayaran tagihan nasabah di merchant,
6. class PembelianCs, merupakan class yang mengatur pembelian voucher,
7. class TransferCs, merupakan class yang mengatur proses yang berhubungan dengan transaksi transfer, termasuk transfer dan transfer validate,
8. class RegisterMerchantCs, merupakan class yang mengatur proses yang berkenaan dengan registerasi merchant,
9. class TagihanCs, merupakan class yang mengatur proses yang berkenaan dengan tagihan merchant kepada bank, dalam hal ini adalah pengecekan jumlahnya,
10. class MoneyBankCs, merupakan class yang merupakan kumpulan dari method-method pada sistem bank,
11. class ServiceBank, merupakan Web Service sistem bank, yang memungkinkan bank berinteraksi dengan sistem lain.

Selain itu sistem bank juga mengakses empat class lain dari sistem cellular

operator, yaitu VouchClass (class untuk pembelian voucher), SubsClass (class untuk menangani tagihan pelanggan), HPCs (class yang merupakan kumpulan dari method-method pada sistem cellular operator), dan ServiceHP (Web Service cellular operator), yang diakses oleh sistem bank melalui interaksi antara Web Service bank (ServiceBank) dengan Web Service cellular operator (ServiceHP).

Sistem Generic Merchant

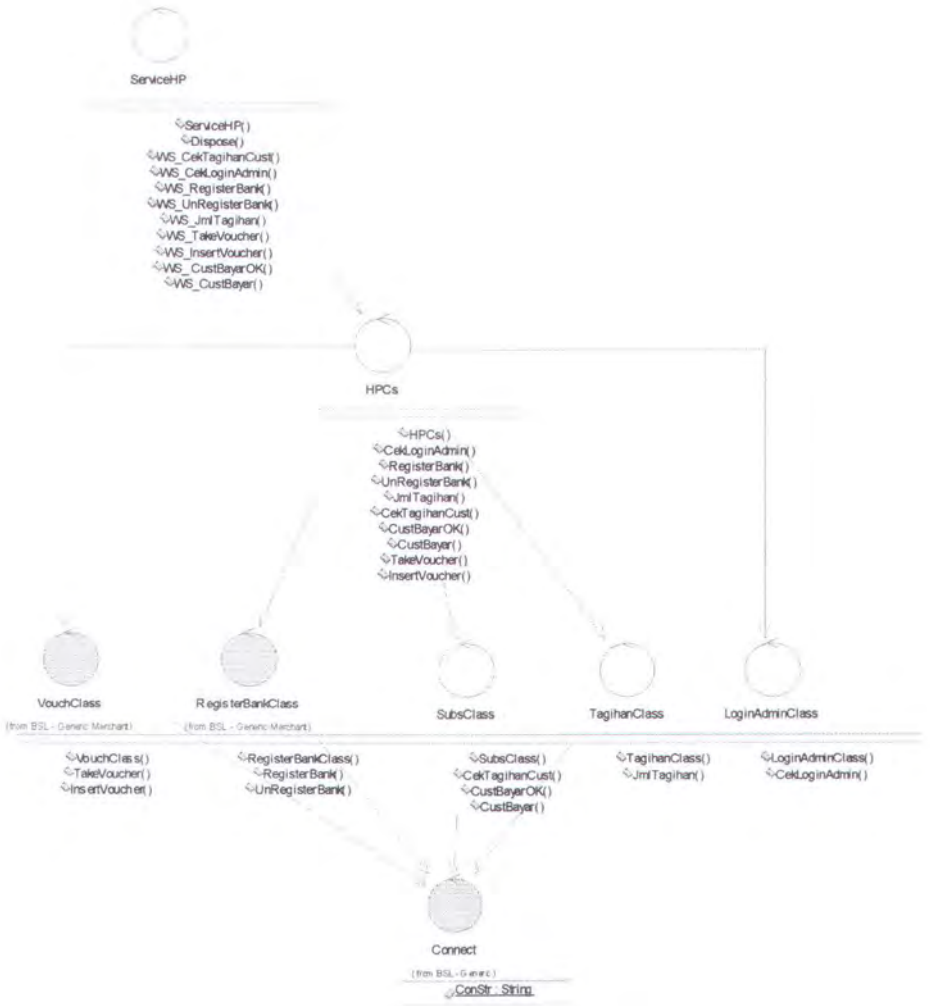


Gambar 4.7 : Class Diagram untuk Business Service Layer pada Sistem Generic Merchant

Sistem generic merchant berisikan class-class yang terdapat pada merchant pada umumnya, baik merchant subscribe seperti cellular operator, maupun pada e-commerce. Seperti terlihat pada gambar 4.7, pada sistem ini terdapat tiga class, yaitu :

1. class Connect, merupakan class untuk koneksi ke database, dan terdapat pada semua sistem,
2. class RegisterBankClass, merupakan class yang mengatur proses registrasi bank pada sistem merchant,
3. class VouchClass, merupakan class yang mengatur proses pembelian voucher, digunakan oleh cellular operator dan e-commerce TeddyBear, yang merupakan contoh e-commerce pada tugas akhir ini, karena e-commerce TeddyBear berjualan voucher handphone.

Sistem Cellular Operator

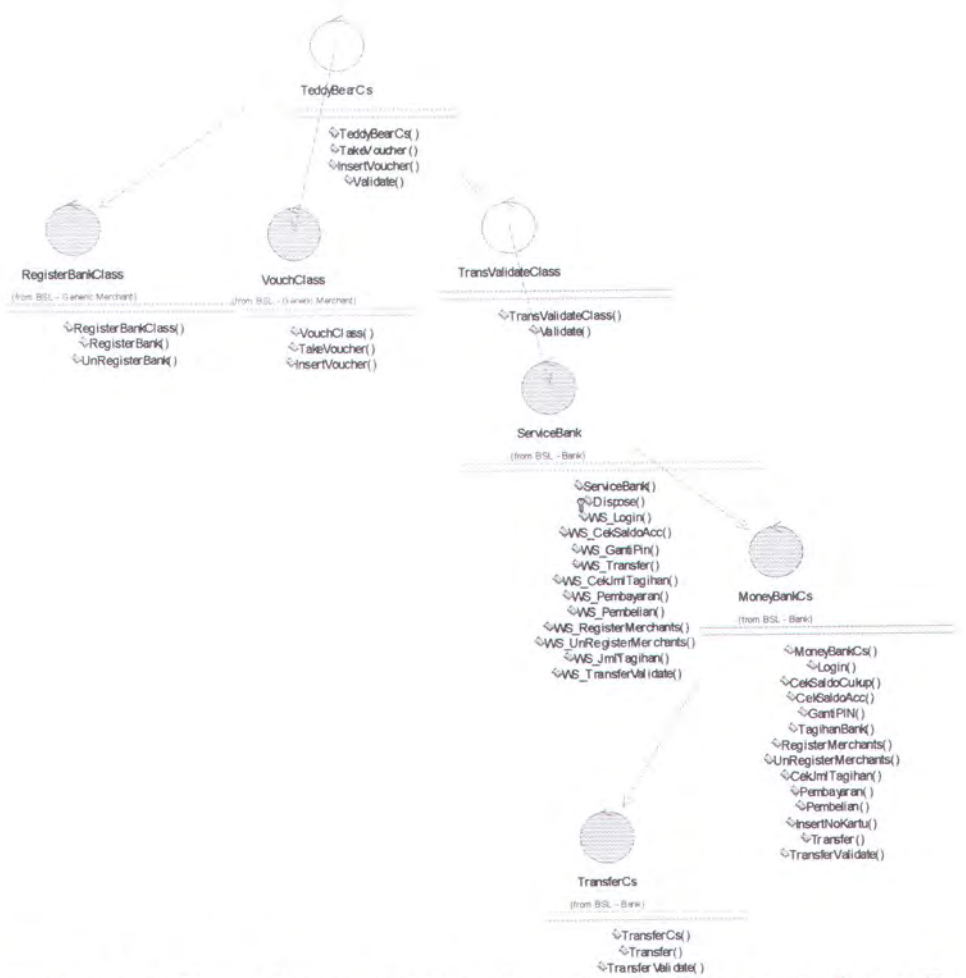


Gambar 4.8 : Class Diagram untuk Business Service Layer pada Sistem Cellular Operator

Pada gambar 4.8 diperlihatkan class-class yang terdapat pada sistem cellular operator, yaitu :

1. class Connect, merupakan class untuk koneksi ke database, dan terdapat pada semua sistem,
2. class VouchClass, dari sistem Generic Merchant,
3. class RegisterBankClass, dari sistem Generic Merchant,
4. class SubsClass, merupakan class yang mengatur proses yang berhubungan dengan tagihan pelanggan, seperti jumlah tagihan dan proses pembayaran,
5. class TagihanClass, merupakan class yang mengatur proses yang berkenaan dengan tagihan merchant kepada bank, dalam hal ini adalah pengecekan jumlahnya,
6. class LoginAdminClass, merupakan class yang memproses login untuk administrator sistem,
7. class HPCs, merupakan kumpulan method-method yang ada pada sistem ini,
8. class ServiceHP, merupakan Web Service dari sistem ini, yang mengatur proses interaksi sistem dengan sistem lain.

Sistem E-Commerce



Gambar 4.9 : Class Diagram untuk Business Service Layer pada Sistem E-Commerce

Gambar 4.9 menampilkan class-class yang terdapat pada sistem e-commerce, dalam hal ini e-commerce TeddyBear yang berjualan voucher handphone. Class-class tersebut adalah :

1. class Connect, merupakan class untuk koneksi ke database, dan terdapat

Sistem Generic, seperti terlihat pada gambar 4.10, terdiri dari satu class yaitu class Connect. Class Connect merupakan class untuk koneksi ke database. Class yang ada pada sistem generic ini terdapat pada semua sistem pada tugas akhir ini.

4.1.4 Perancangan Antarmuka

Pada sub bab ini akan dijelaskan mengenai perancangan antarmuka pada ketiga sistem yang ada. Antarmuka pada ketiga sistem menggunakan bahasa pemrograman web aspx.

Jumlah antarmuka yang terdapat pada sistem bank adalah dua belas buah, seperti bisa dilihat pada gambar 4.11. Antarmuka-antarmuka ini akan mengakses business service layer melalui Web Service ServiceBank.

Sistem Cellular Operator memiliki antarmuka sebanyak tujuh buah, dan antarmuka ini mengakses business service layer melalui Web Service ServiceHP.

Sedangkan untuk E-Commerce hanya terdapat satu aplikasi, yaitu Main.aspx. Aplikasi ini mengakses business service layer melalui class TeddyBearCs.

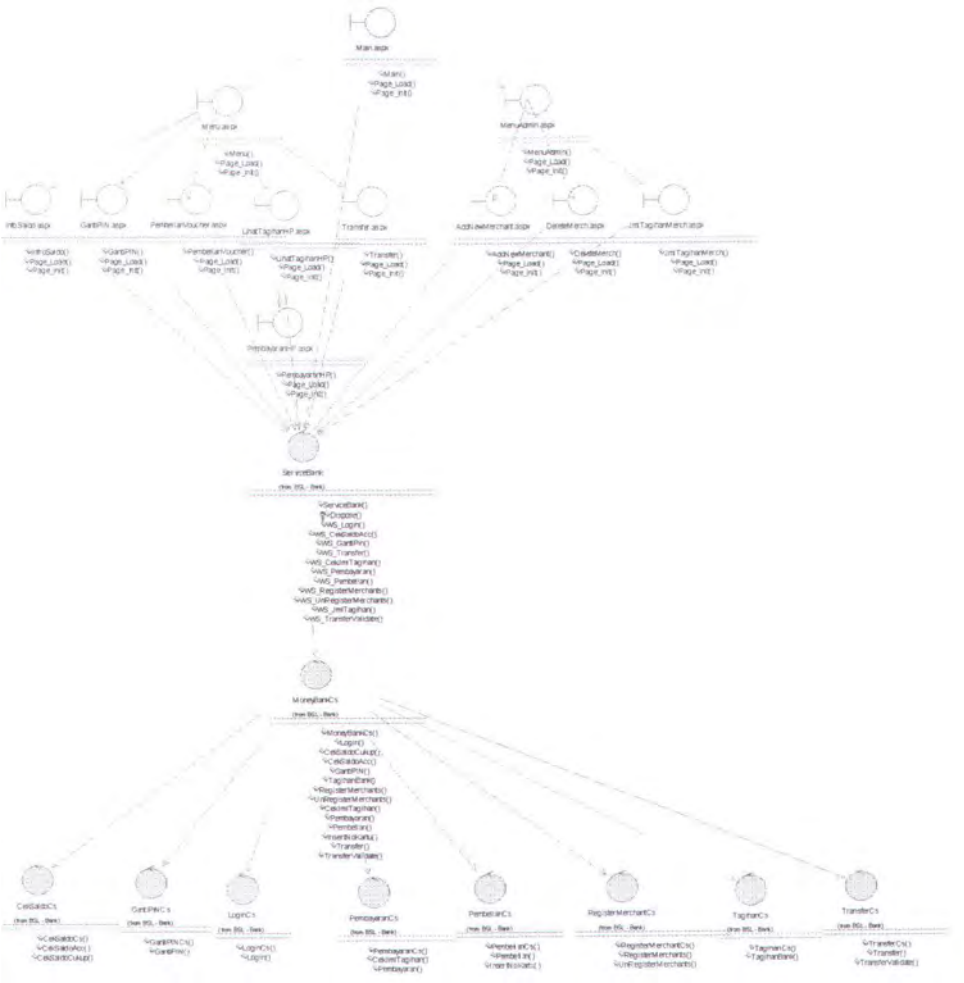
pada semua sistem,

2. class VouchClass, dari sistem Generic Merchant,
3. class RegisterBankClass, dari sistem Generic Merchant,
4. class TransValidateClass, class ini mengatur proses validasi transfer pembeli ke bank,
5. class TeddyBearCs, class ini merupakan kumpulan method-method yang ada pada sistem ini.

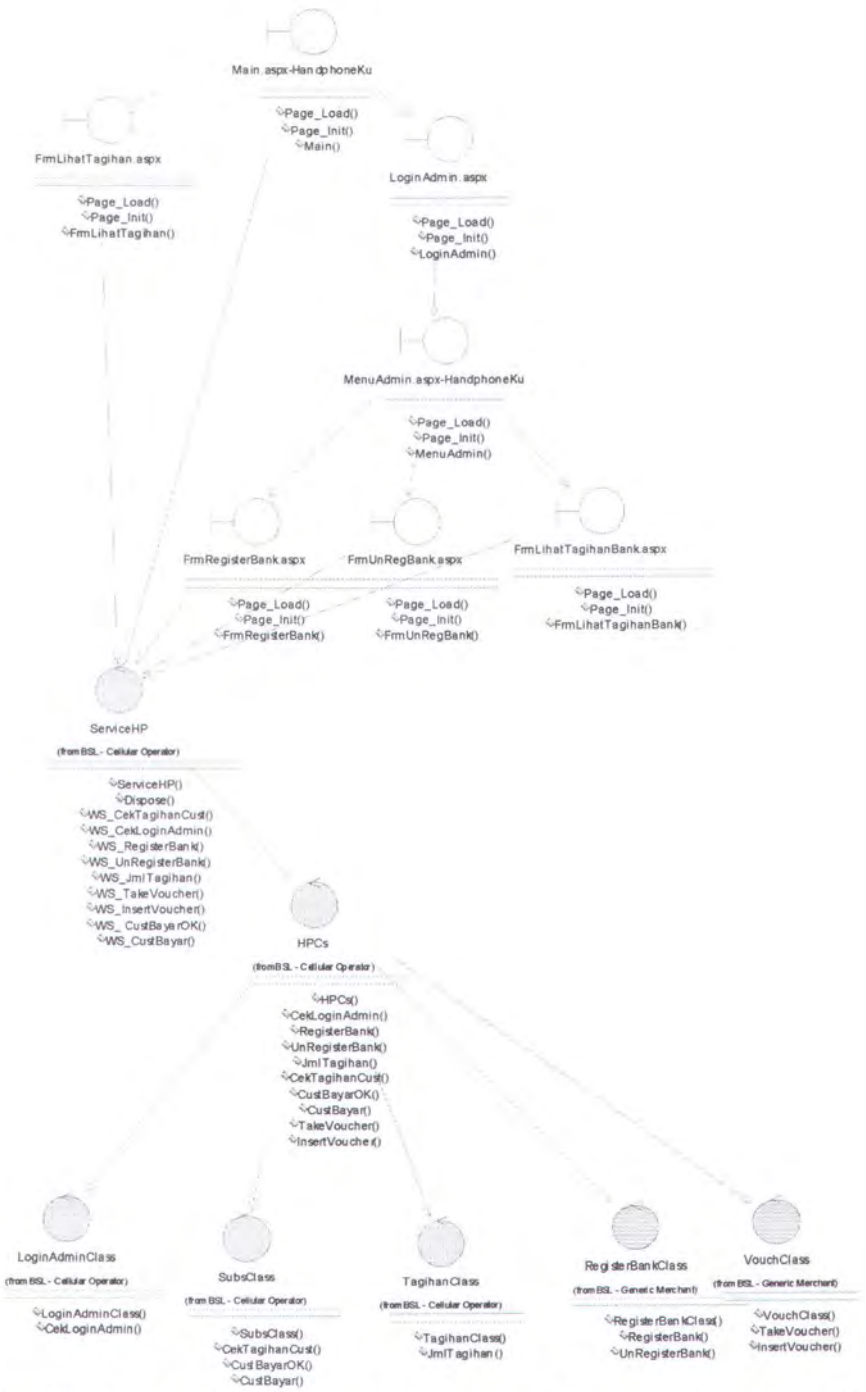
Selain itu, sistem ini juga mengakses class dari sistem bank, yaitu class TransferCs (mengatur proses yang berhubungan dengan transfer), class MoneyBankCs (kumpulan method-method dalam sistem bank), dan Web Service ServiceHP.

Sistem Generic

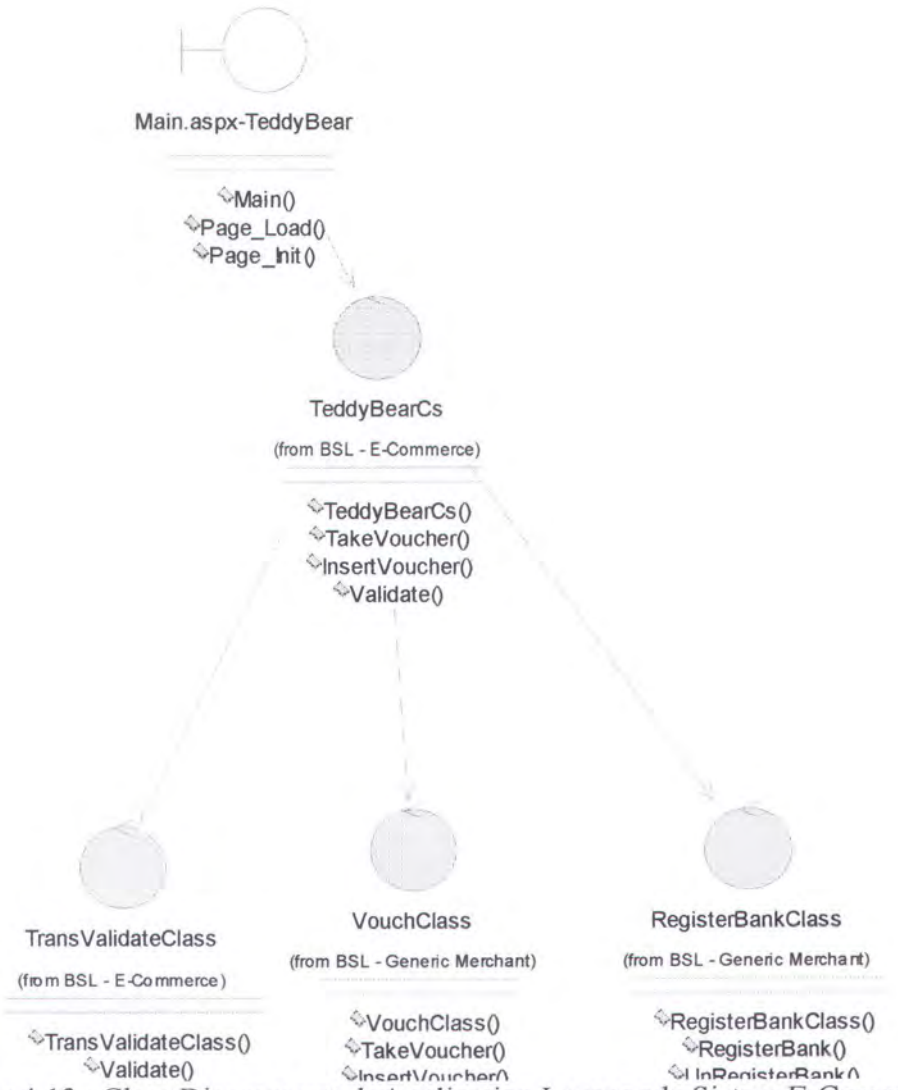
Gambar 4.10 : Class Diagram untuk Business Service Layer pada Sistem Generic/Keseluruhan



Gambar 4.11 : Class Diagram untuk Application Layer pada Sistem Bank



Gambar 4.12 : Class Diagram untuk Application Layer pada Sistem Cellular Operator

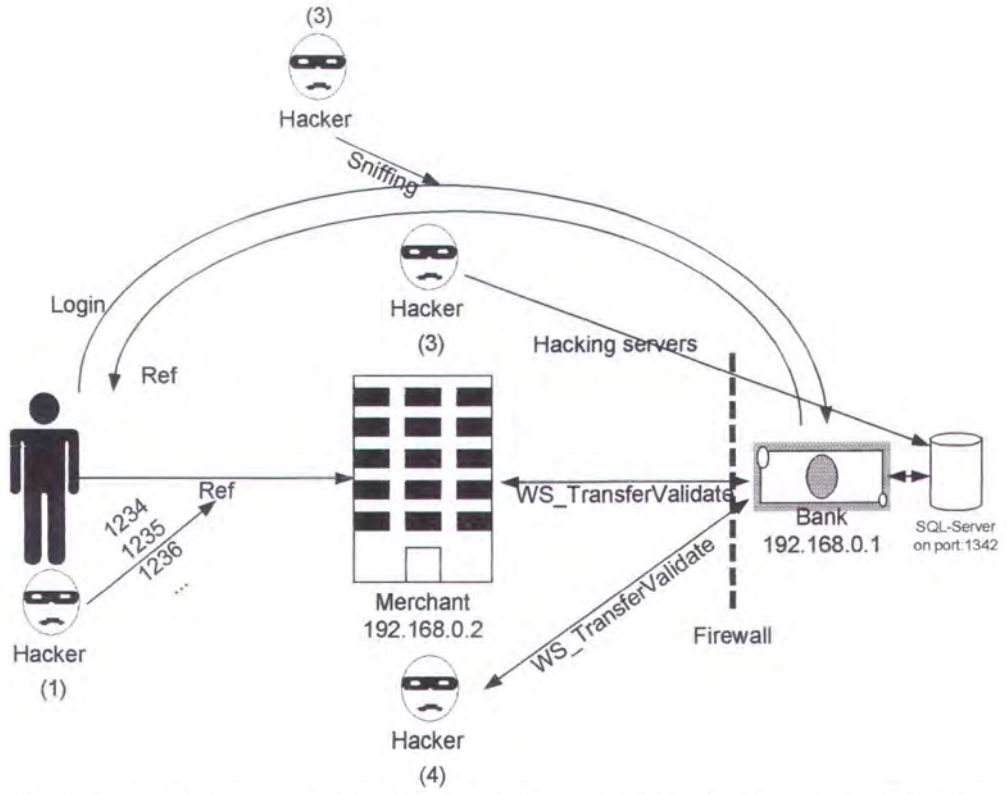


Gambar 4.13 : Class Diagram untuk Application Layer pada Sistem E-Commerce

4.1.5 Keamanan

Internet adalah jaringan publik oleh karena itu melakukan komunikasi data dengan menggunakan media internet rentan terhadap berbagai ancaman keamanan

data, terlebih dalam sistem ini berpusat pada bank yang berhubungan dengan data-data keuangan. Oleh karena itu dalam perancangan sistem ini diperhatikan pula aspek keamanannya.



Gambar 4.14 : Keamanan Sistem dan Kemungkinan Penyerangan Sistem

Kemungkinan serangan terhadap sistem, seperti bisa dilihat pada gambar 4.14 :

(1) Pemalsuan Nomer Referensi Transfer, hacker mencoba memalsukan nomer referensi transfer dengan metoda brute-force/trial and error. Nomer referensi yang digunakan di-*generate* secara random, yang terdiri dari tiga puluh dua digit, yang berupa kombinasi huruf dan

angka sehingga didapatkan kombinasi sebanyak 32^36 = 1,5.10^54. Dan setelah nomer referensi ini divalidasi, maka nomer referensi ini tidak bisa digunakan lagi, karena nomer referensi ini sudah diberi tanda telah divalidasi. Selain itu nomer referensi yang didapat hanya bisa digunakan maksimal dua hari sejak nasabah mendapatkannya.

(2) Penyadapan data (Sniffing)

Hacker berusaha menyadap data informasi login, transfer reference dengan cara memasang program penyadap/sniffer di salah satu jalur komunikasi yang dilewati (digunakan). Hal ini dicegah dengan melakukan enkripsi semua data yang akan dikirim, atau alternatif yang lebih transparan adalah menggunakan protokol SSL diatas protokol HTTP (HTTPS).

(3) Hacking Server

Hacker berusaha mencuri data pada sistem dengan cara mendapatkan akses previlege admin pada server. Pertama-tama hacker akan melakukan deteksi, aplikasi server apa saja yang berjalan pada server seperti IIS, SQL-Server. Kemudian menggunakan program yang dapat mengeksploitasi aplikasi server-server tersebut. Hal ini dapat ditanggulangi dengan selalu melakukan update terhadap aplikasi server yang digunakan dan memasang firewall untuk membatasi akses ke sistem dan DMZ (demiliterize zone) atau komputer-komputer di belakang firewall.

(4) Filter WebService

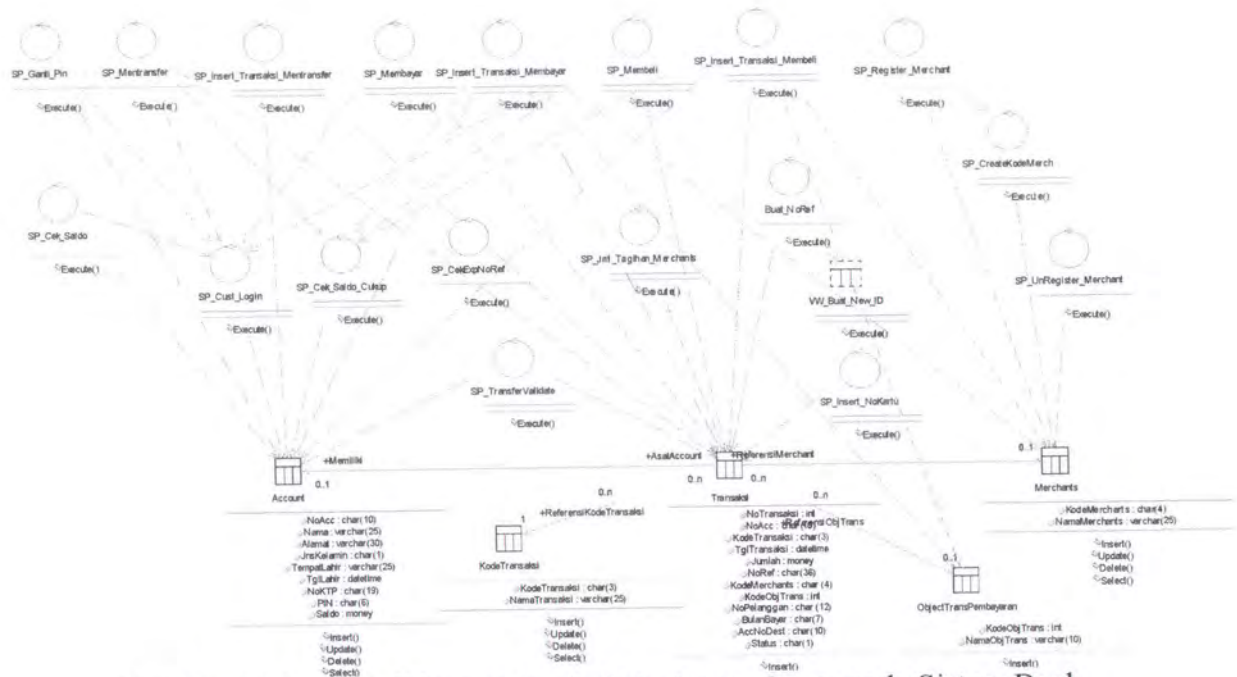
Agar lebih aman, pihak Bank melakukan pembatasan akses WebService hanya kepada alamat-alamat IP dari merchant yang telah terdaftar sebelumnya. Pembatasan akses ini dilakukan dengan menggunakan firewall.

4.2 Implementasi

4.2.1 Implementasi Data

Untuk implementasi data digunakan Microsoft SQL Server 2000.

Skema DataBase untuk Sistem Bank



Gambar 4.15 : Class Diagram untuk Data Access Layer pada Sistem Bank

Trigger untuk Database Bank

Sistem bank menggunakan satu trigger, yaitu ti_Transaksi untuk lebih menjamin integritas data pada database. Trigger ti_Transaksi adalah trigger pada tabel Transaksi dan dieksekusi setiap kali ada penambahan data pada tabel Transaksi. Trigger ini akan mengupdate tabel transaksi dan mengeset NoRef dengan mengeksekusi *function* Buat_NoRef pada data yang baru ditambahkan.

```
CREATE     TRIGGER [ti_Transaksi] ON dbo.TRANSAKSI
FOR INSERT
AS
--                Update Transaksi->masukkan No.Ref
                  Update Transaksi
                  Set NoRef=dbo.Buat_NoRef()
                  Where NoTransaksi=@@IDENTITY

GO
```

Gambar 4.16 : Script MS SQL Server 2000 untuk Pembuatan Trigger Untuk Database Bank

Stored Procedure/Function untuk Database Bank

Pada Database Bank terdapat tujuh belas stored procedure, yaitu :

1. SP_Cust_Login, digunakan untuk mengecek apakah nomer account (login) dan pin yang diinputkan sesuai dengan data yang ada pada tabel Account.
2. SP_Cek_Saldo_Cukup, digunakan untuk mengecek apakah jumlah saldo rekening cukup untuk melakukan transaksi sejumlah inputan.

3. SP_Cek_Saldo, digunakan untuk mengecek jumlah saldo rekening nasabah pada saat itu.

4. SP_CreateKodeMerch, digunakan untuk membuat kode merchant.

5. SP_GantiPIN, digunakan untuk memproses pergantian pin dengan mengupdate data pin tersebut di tabel Account.

6. SP_InsertNoKartu, digunakan untuk memasukkan nomer kartu ke *field* NoPelanggan pada tabel Transaksi.

```
--Procedure utk Menyimpan No.Kartu-->NoPelanggan di Tabel
CREATE    PROCEDURE [dbo].[SP_Insert_NoKartu]
        (@NoKartu       char(12),
         @NoRef char(36),
         @ReplyStatus  int     OUTPUT,
         @ReplyMsg     VARCHAR(255)    OUTPUT
        )

AS
        set @ReplyStatus=0
        set @ReplyMsg='Sukses'

        if exists(Select NoRef From Transaksi Where NoRef = @NoRef)
                Update Transaksi
                Set NoPelanggan  = @NoKartu
                Where NoRef = @NoRef
        else
        begin
                set @ReplyStatus = 11
                set @ReplyMsg = 'No. Ref yang anda masukkan salah! Transaksi Gagal'
        end
GO
```


Gambar 4.17 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Insert_NoKartu Untuk Database Bank

7. SP_Insert_Transaksi_Membayar, digunakan untuk memasukkan data transaksi pembayaran ke tabel Transaksi.

8. SP_Insert_Transaksi_Membeli, digunakan untuk memasukkan data transaksi pembelian ke tabel Transaksi.

```
CREATE PROCEDURE SP_Insert_Transaksi_Membayar
        (@NoAcc char(10), @Jumlah money, @KodeMerchants char(4),
         @KodeObjTrans int, @NoPelanggan char(12), @BulanBayar char(7),
         @Status int output, @Msg varchar(255) output
        )
AS
        --Cek NoAcc
        if not exists(select NoAcc From Account Where NoAcc=@NoAcc)
        begin
                set @Status=1
                set @Msg='NoAcc salah'
                return
        end
        --Cek Saldo
        execute dbo.SP_Cek_Saldo_Cukup @NoAcc,@Jumlah,
                @ReplyStatus=@Status output,@REplyMsg=@Msg output

        if (@Status<>0)
        begin
                return
        end
        --Status oke
        else
        begin
                        --Cek Kode Merchants
                        if not exists (select KodeMerchants from Merchants   where
KodeMerchants=@KodeMerchants)
                                begin
                                        set @Status=5
                                        set @Msg='Kode Merchants Tidak Ada'
                                        return
                                end
                        --Cek KodeObjTrans
                        if     not     exists     (select     KodeObjTrans     from
ObjectTransPEmbayaran where KodeObjTrans=@KodeObjTrans)
                        begin
                                set @Status=6
                                set @Msg='KodeObjTrans Tidak Ada'
                                return
                        end

                        if (@Status<>0)
                        begin
                                return
                        end

                        --Kurangi Saldo Asal
                        Update Account
                        Set Saldo=Saldo-@Jumlah
                        Where NoAcc=@NoAcc --and PIN=@PIN


--                      Insert Transaksi
                        Insert
Transaksi(NoAcc,KodeTransaksi,NoPelanggan,BulanBayar,KodeMerchants,KodeObjTrans,Ju
mlah)
                        Values
(@NoAcc,'PBR',@NoPelanggan,@Bulanbayar,@KodeMErchants,@KodeObjTrans,@Jumlah)
        end
GO
```

Gambar 4.18 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Insert_Transaksi_Membayar Untuk Database Bank

```
CREATE PROCEDURE SP_Insert_Transaksi_Membeli
        (@NoAcc char(10),
         @Jumlah        money,
         @KodeMerchants char(4),
         @KodeObjTrans int,
         @Status int output,
         @Msg   varchar(255) output
        )
AS
        --Cek NoAcc
if not exists(select NoAcc From Account Where NoAcc=@NoAcc)
        begin
                set @Status=1
                set @Msg='NoAcc salah'
                return
        end
        --Cek Saldo cukup
        execute dbo.SP_Cek_Saldo_Cukup @NoAcc,@Jumlah,
                @ReplyStatus=@Status output,@REplyMsg=@Msg output
        if (@Status<>0)
        begin
                return
        end
        --Status oke
        else
        begin
                --PEMBAYARAN
                        --Cek Kode Merchants
                        if not exists (select KodeMerchants from Merchants  where
KodeMerchants=@KodeMerchants)
                        begin
                                set @Status=5
                                set @Msg='Kode Merchants Tidak Ada'
                                return
                        end
                        --Cek KodeObjTrans
                        if     not     exists     (select     KodeObjTrans     from
ObjectTransPEmbayaran where KodeObjTrans=@KodeObjTrans)
                        begin
                                set @Status=6
                                set @Msg='KodeObjTrans Tidak Ada'
                                return
                        end
                        if (@Status<>0)
                        begin
                                return
                        end
                        --Kurangi Saldo Asal
                        Update Account
                        Set Saldo=Saldo-@Jumlah
                        Where NoAcc=@NoAcc --and PIN=@PIN
--                      Insert Transaksi
                        Insert
Transaksi(NoAcc,KodeTransaksi,KodeMerchants,KodeObjTrans,Jumlah)
                        Values (@NoAcc,'PBR',@KodeMErchants,@KodeObjTrans,@Jumlah)
        end
GO
```

Gambar 4.19 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Insert_Transaksi_Membeli Untuk Database Bank

9. SP_Insert_Transaksi_Mentransfer, digunakan untuk memasukkan

data transaksi transfer dana ke tabel Transaksi.

```
CREATE PROCEDURE SP_Insert_Transaksi_Mentransfer
        (@NoAcc char(10),
         @Jumlah        money,
         @AccNoDest     char(10),
         @Status int output,
         @Msg   varchar(255) output
        )
AS
        --Cek NoAcc
        if not exists(select NoAcc From Account Where NoAcc=@NoAcc)
        begin
                set @Status=1
                set @Msg='NoAcc salah'
                return
        end

        if (@Status<>0)
        begin
                return
        end
        --Status oke
        else
        begin
        --TRANSFER

                if (@Status<>0)
                begin
                        return
                end

                --Kurangi Saldo Asal
                Update Account
                Set Saldo=Saldo-@Jumlah
                Where NoAcc=@NoAcc --and PIN=@PIN
                --Tambahkan Saldo Tujuan
                Update Account
                Set Saldo=Saldo+@Jumlah
                Where NoAcc=@NoAcc

--                      Insert Transaksi
                        Insert
Transaksi(NoAcc,KodeTransaksi,AccNoDest,Jumlah,Status)
                        Values (@NoAcc,'TRS',@AccNoDest,@Jumlah,'N')

        end
GO
```

Gambar 4.20 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Insert_Transaksi_Mentransfer Untuk Database Bank

10. SP_Jml_Tagihan_Merchants, digunakan untuk menghitung berapa jumlah utang bank ke merchant pada bulan tertentu untuk transaksi yang dilakukan oleh pelanggan merchant melalui bank.

11. SP_Membayar, tabel ini memproses transaksi pembayaran.

```
--Procedure utk Memproses Pembayaran
CREATE     PROCEDURE [dbo].[SP_Membayar]
        (@NoAcc char(10),
         @PIN          char(6),
         @NoPelanggan char(12),
         @BulanBayar   char(7),
         @KodeMerchants char(4),
         @KodeObjTrans int,
         @Jumlah       money,
         @TglTrans     datetime output,
         @NoRef char(36) output,
         @ReplyStatus  int     OUTPUT,
         @ReplyMsg     VARCHAR(255)   OUTPUT
        )

AS

        --Cek Apakah No Acc dan PIN benar
        EXEC    dbo.SP_Cust_Login    @NoAcc,@Pin,@ReplyStatus=@ReplyStatus    output,
@ReplyMsg=@ReplyMsg output


        if @ReplyStatus=0
        begin
        --insert ke tabel transaksi
                exec SP_Insert_Transaksi_Membayar @NoAcc = @NoAcc,
                                @Jumlah = @Jumlah,
                                @KodeMerchants = @KodeMerchants,
                                @KodeObjTrans = @KodeObjTrans,
                                @NoPelanggan   = @NoPelanggan,
                                @BulanBayar = @BulanBayar,
                                @Status = @ReplyStatus output,
                                @Msg   = @ReplyMsg output

--              Ambil tgltrans dan noref
                Select  @NoRef=NoRef,  @TglTrans=TglTransaksi  From  Transaksi  Where
NoTransaksi=@@IDENTITY
        end
        else
        begin
                return
        end
GO
```

Gambar 4.21 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure

SP_Membayar Untuk Database Bank

12. SP_Membeli, tabel ini memproses transaksi pembelian.

```
--Procedure utk Memproses Pembelian
CREATE    PROCEDURE [dbo].[SP_Membeli]
        (@NoAcc char(10),
         @PIN          char(6),
         @KodeMerchants char(4),
         @KodeObjTrans int,
         @Jumlah       money,
         @TglTrans     datetime output,
         @NoRef char(36) output,
         @ReplyStatus  int     OUTPUT,
         @ReplyMsg     VARCHAR(255)   OUTPUT
        )

AS

        --Cek Apakah No Acc dan PIN benar
        EXEC    dbo.SP_Cust_Login    @NoAcc,@Pin,@ReplyStatus=@ReplyStatus    output,
@ReplyMsg=@ReplyMsg output

        if @ReplyStatus=0
        begin
        --Insert Transaksi Pembelian Voucher
                exec SP_Insert_Transaksi_Membeli @NoAcc = @NoAcc,
                                @Jumlah = @Jumlah,
                                @KodeMerchants = @KodeMerchants,
                                @KodeObjTrans = @KodeObjTrans,
                                @Status = @ReplyStatus output,
                                @Msg   = @ReplyMsg output


--      Ambil noref dan tgltrans
                Select  @NoRef=NoRef,  @TglTrans=TglTransaksi  From  Transaksi  Where
NoTransaksi=@@IDENTITY

        end
        else
        begin
                return
        end
GO
```

Gambar 4.22 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Membeli Untuk Database Bank

13. SP_MenTransfer, tabel ini memproses transaksi transfer dana.

```
--Prosedur untuk memproses Transfer dari Customer
CREATE       PROCEDURE [dbo].[SP_MenTransfer]
        (@NoAcc char(10),
         @PIN          char(6),
         @AccNoDest    char(10),
         @Jumlah       money,
         @TglTrans     datetime       output,
         @NoRef char(36) output,
         @ReplyStatus  int     OUTPUT,
         @ReplyMsg     VARCHAR(255)   OUTPUT
        )
AS
Declare @tmpNoTransaksi char(6)

        --Cek Apakah No Acc dan PIN benar
        EXEC    dbo.SP_Cust_Login    @NoAcc,@Pin,@ReplyStatus=@ReplyStatus    output,
@ReplyMsg=@ReplyMsg output

        --Cek Apakah No Acc Tujuan Ada
        if not exists (select NoAcc From Account Where NoAcc=@AccNoDest)
        begin
                set @ReplyStatus=2
                set @ReplyMsg='No. Account Tujuan tidak Ada'
        end

        --Cek Apakah Saldo mencukupi
        exec  dbo.SP_Cek_Saldo_Cukup  @NoAcc,  @Jumlah,  @ReplyStatus=@ReplyStatus
output,@ReplyMsg=@ReplyMsg output

        if (@ReplyStatus=0)
        begin
                --Insert Transaksi
                exec SP_Insert_Transaksi_Mentransfer
                        @NoAcc = @NoAcc,
                        @Jumlah= @Jumlah,
                        @AccNoDest = @AccNoDest,
                        @Status = @ReplyStatus  output,
                        @Msg   = @ReplyMsg output

                Select  @NoRef=NoRef,  @TglTrans=TglTransaksi  From  Transaksi  Where
NoTransaksi=@@IDENTITY
        end
        else
        begin
                return
        end
GO
```

Gambar 4.23 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure

SP_Mentransfer Untuk Database Bank

14. SP_RegisterMerchant, digunakan untuk memproses registrasi merchant dengan memasukkannya ke tabel Merchants.

15. SP_TransferValidate, digunakan untuk memproses validasi transfer. Validasi transfer gagal apabila data nomer referensi, nomer rekening tujuan dan atau jumlah tidak sesuai atau tidak ditemukan, atau nomer referensi sudah kadaluarsa atau sudah pernah divalidasi sebelumnya.

```
CREATE PROCEDURE [dbo].[SP_TransferValidate]
      (@AccNoDest char(10),
       @NoRef     char(36),
       @Jumlah    money ,
       @ReplyStatus     int OUTPUT,
       @ReplyMsg  varchar(255) OUTPUT
      )
AS
      Set @ReplyStatus=0
      Set @ReplyMsg='Sukses'

      --Cek Apakah betul ada Transfer ke AccNoDest dgn NoRef tsb dan Apakah Sdh
Dicek/Blm
      IF    NOT    EXISTS(Select    AccNoDest,NoRef    From    Transaksi    Where
AccNoDest=@AccNoDest and NoRef=@NoRef and Status='N'
                  and Jumlah = @Jumlah)
      BEGIN
            Set @ReplyStatus=7
            Set @ReplyMsg='Transfer Validate Gagal'
            Return
      END
      ELSE
      BEGIN
            Update  Transaksi  Set  Status='C'  Where  AccNoDest=@AccNoDest  and
NoRef=@NoRef and Status='N'
      END
GO
```

Gambar 4.24 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_TransferValidate Untuk Database Bank

16. SP_UnRegister_Merchant, digunakan untuk memproses penghapusan data merchant dari tabel Merchants.

17. SP_CekExpNoRef, digunakan untuk memproses pengecekan masa berlakunya nomer referensi.

Pada database bank terdapat satu *function*, yaitu Buat_NoRef yang berguna untuk membuat nomer referensi yang digunakan untuk transaksi.

```
CREATE    FUNCTION [dbo].[Buat_NoRef]()
RETURNS  char(36)  AS
BEGIN
      Declare @tmp varchar(36),
            @Baru int


      set @Baru=0
      While (@Baru=0)
      begin
            select @tmp=NEWID from VW_Buat_NewID
            --CEK APAKAH NOREF SDH ADA PADA TABEL TRANSAKSI
            if ((Select NoRef From Transaksi Where NoRef=@tmp)is null)
            begin
                  set @Baru=1
            end
      end
      return CAST(@tmp as char(36))
END
```

Gambar 4.25 : Script MS SQL Server 2000 untuk Pembuatan Function Untuk Database Bank

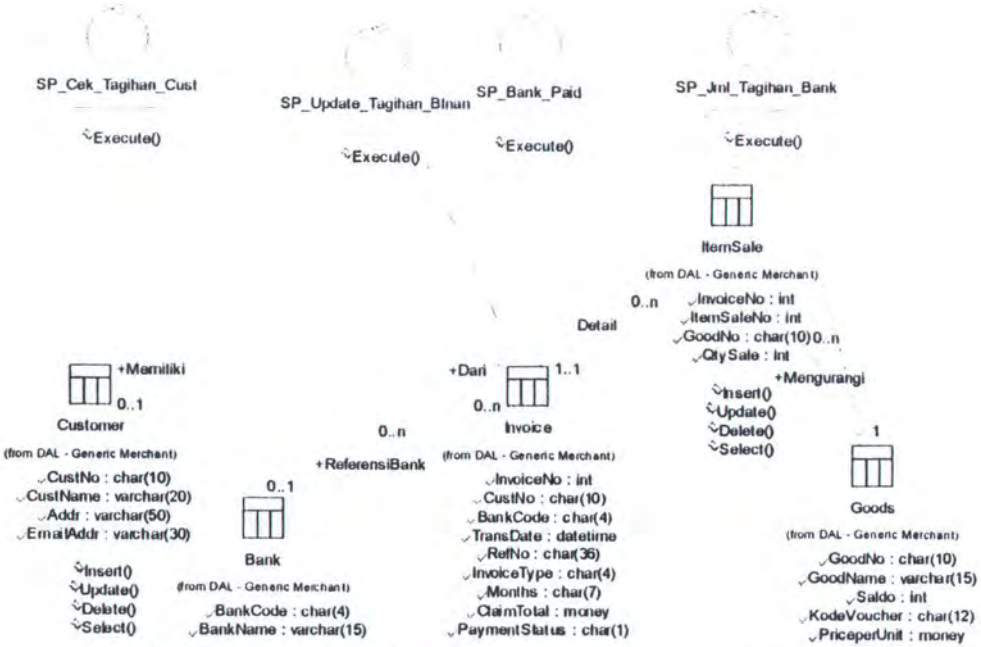
View untuk Database Bank

Pada database bank terdapat satu view, yaitu VW_Buat_NEWID, untuk membuat nomer secara random sebanyak 32 digit yang terdiri dari huruf dan angka.

```
CREATE VIEW dbo.VW_Buat_NEWID
AS
SELECT     NEWID() AS NewID
```

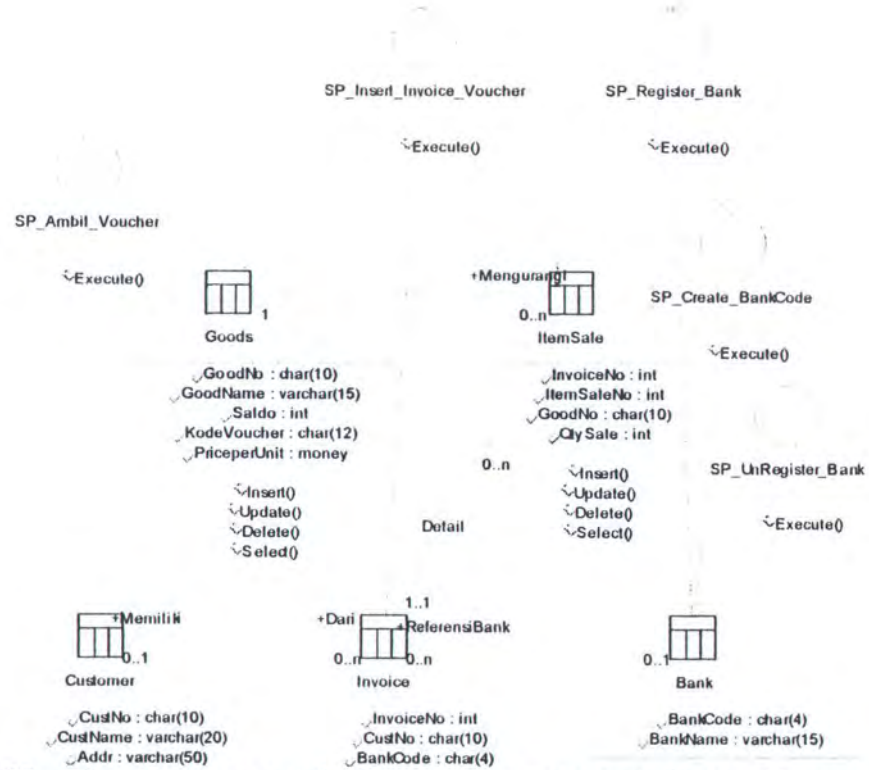
Gambar 4.26 : Script MS SQL Server 2000 untuk Pembuatan View untuk Database Bank

Skema Database untuk Sistem Cellular Operator



Gambar 4.27 : Class Diagram untuk Data Access Layer pada Sistem Cellular Operator

Skema Database untuk Sistem Generic Merchant



Gambar 4.28 : Class Diagram untuk Data Access Layer pada Sistem Generic Merchant

Stored Procedure untuk Database Cellular Operator

Pada database cellular operator terdapat sembilan stored procedure, yaitu :

1. SP_Ambil_Voucher, stored procedure ini mengecek apakah persediaan voucher masih ada.
2. SP_Bank_Paid, stored procedure ini menandai bahwa pembayaran dilakukan melalui bank yang diinputkan dengan meng-update bankcode dan paymentstatus pada tabel Invoice.

```
CREATE PROCEDURE SP_Bank_Paid
      (
       @CustNo     char(12),
       @BankCode   char(4),
       @Months     char(7),
       @Status     int   output,
       @Msg        varchar(255)      output
      )
 AS
declare @PAymentStatus  char(1)
      set @PaymentStatus='C'
      set @Status=0
      set @Msg='Sukses'
      --Cek customer
      IF NOT EXISTS (Select CustNo From Customer Where CustNo=@CustNo)
      BEGIN
            Set @Status=1
            Set @Msg='Data Customer Tidak Ditemukan'
            return
      END
      --Cek TAgihan Bln Ini Ada.tdk
      if not exists (Select CustNo, Months From Invoice Where (CustNo=@CustNo) and
(Months=@Months))
      begin
            set @STatus=3
            set @Msg='Tagihan Bln Ini Blm Ada'
      end
      if (@Status=0)
      begin
            --Update
            Update dbo.Invoice
            Set BankCode=@BankCode, PaymentStatus= @PaymentStatus
            Where (CustNo=@CustNo) and (Months=@Months)
      end
GO
```

Gambar 4.29 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Bank_Paid Untuk Database Cellular Operator

3. SP_Cek_Tagihan_Cust, stored procedure ini digunakan untuk mengecek jumlah tagihan pelanggan pada bulan tertentu dan status pembayarannya.
4. SP_Create_BankCode, stored procedure ini digunakan untuk membuat BankCode.
5. SP_Insert_Invoice_Voucher, digunakan untuk memasukkan data transaksi pembelian voucher ke tabel Invoice.
6. SP_Jml_Tagihan_Bank, digunakan untuk menghitung jumlah tagihan kepada bank dari transaksi pada bulan tertentu melalui bank.
7. SP_Register_Bank, digunakan untuk memasukkan data bank ke tabel Bank.
8. SP_Unregister_Bank, digunakan untuk menghapus data bank dari tabel bank.
9. SP_Update_Tagihan_Blnan, stored procedure ini digunakan untuk mengupdate tagihan pada tabel invoice, bila pelanggan membayarnya.

```
CREATE PROCEDURE dbo.SP_Update_Tagihan_Blnan
      (@CustNo    char(12),
       @BankCode  char(4),
       @TransDate datetime,
       @RefNo     char(36),
       @Months    char(7),
       @Status    int   output,
       @Msg       varchar(255)      output
      )
AS
declare @PAymentStatus  char(1)
      set @PaymentStatus='P'
      set @Status=0
      set @Msg='Sukses'
      --Cek Customer
      IF NOT EXISTS (Select CustNo From Customer Where CustNo=@CustNo)
      BEGIN
            Set @Status=1
            Set @Msg='Data Customer Tidak Ditemukan'
            return
      END
      --Cek TAgihan Bln Ini Ada.tdk
      if not exists (Select CustNo, Months From Invoice Where (CustNo=@CustNo) and
(Months=@Months))
      begin
            set @STatus=3
            set @Msg='Tagihan Bln Ini Blm Ada'
      end
      if (@Status=0)
      begin
            if exists (Select * From Invoice Where PaymentStatus='C' and
BankCode=@BankCode)
            begin
                  Update dbo.Invoice
                  Set    PaymentStatus=    @PaymentStatus,    TransDate=@TransDate,
refNo=@RefNo
                  Where    (CustNo=@CustNo)    and    (Months    =    @Months)    and
(BankCode=@BankCode)
            end
            else
            begin
                  if exists(Select * From Invoice Where PaymentStatus='N')
                  begin
                        Update dbo.Invoice
                        Set  PaymentStatus=  @PaymentStatus,  TransDate=@TransDate,
refNo=@RefNo
                        Where    (CustNo=@CustNo)    and    (Months=@Months)    and
BankCode=@BankCode
                  end
                  else
                  begin
                        set @Status=7
                        set @Msg='Tagihan Bulan Ini Sudah Dibayar'
                  end
            end
      end
GO
```

Gambar 4.30 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Update_Tagihan_Blnan Untuk Database Cellular Operator

Stored Procedure untuk Database E-Commerce

Pada database e-commerce terdapat lima stored procedure, yaitu :

1. SP_Ambil_Voucher, stored procedure ini mengecek apakah persediaan voucher masih ada.

```
-- cek persediaan voucher
CREATE PROCEDURE dbo.SP_Ambil_Voucher
      (@GoodName  varchar(15),
       @PriceperUnit    money,
       @Status int output,
       @Msg varchar(255) output
      )
AS

      set @Status=0
      set @Msg='Sukses'

      if exists(Select GoodNo From Goods Where (Saldo=1 and GoodName=@GoodName and
PriceperUnit = @PriceperUnit))
      begin
            set @Status=0
            set @Msg='Sukses'
      end
      else
      begin
            set @Status=3
            set @Msg='Persediaan Kosong'
      end
GO
```

Gambar 4.31 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Ambil_Voucher Untuk Database E-Commerce

2. SP_Create_BankCode, stored procedure ini digunakan untuk membuat BankCode.

3. SP_Insert_Invoice_Voucher, digunakan untuk memasukkan data transaksi pembelian voucher ke tabel Invoice.

```
CREATE PROCEDURE SP_Insert_Invoice_Voucher
      (@BankCode  char(4),
       @RefNo     char(36),
       @GoodName  varchar(15),
       @priceperunit money,
       @TransDate       datetime    output,
       @InvoiceNo int   output,
       @GoodNo    char(10) output,
       @KodeVoucher char(12) output,
       @Status    int output,
       @Msg       varchar(255) output
      )
AS
--declare @priceperunit money
      set @Status=0
      set @Msg='Sukses'
      set @TransDate = GetDate()
      exec SP_Ambil_Voucher @GoodName=@GoodName , @PriceperUnit = @PriceperUnit  ,
@Status=@Status,@Msg=@Msg

      if (@Status=0)
      begin
            Set @GoodNo = (Select Min(GoodNo)
            From Goods
            Where ((Saldo = 1) and (GoodName = @GoodName) and Priceperunit =
@priceperunit)
            )
            Select  @KodeVoucher = KodeVoucher
            From Goods
            Where GoodNo = @GoodNo

            Update Goods
            Set Saldo = 0
            Where GoodNo=@GoodNo

            Insert into Invoice(BankCode , TransDate , REfNo , INVOICETYPE)
            VALUES(@BankCode , @TransDate , @RefNo , 'ECOM')
            set @InvoiceNo=@@identity

            Insert into ItemSale(InvoiceNo,GoodNo,QtySale)
            Values (@InvoiceNo,@GoodNo,1)
      end
GO
```

Gambar 4.32 : Script MS SQL Server 2000 untuk Pembuatan Stored Procedure SP_Insert_Invoice_Voucher Untuk Database E-Commerce

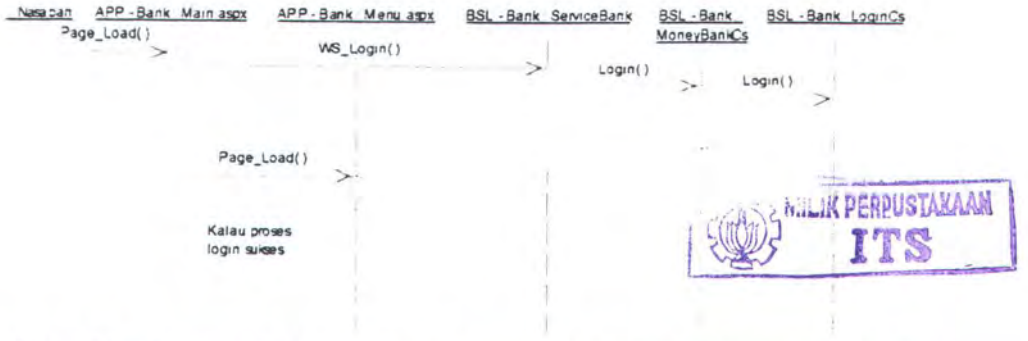
4. SP_Register_Bank, digunakan untuk memasukkan data bank ke tabel Bank.
5. SP_Unregister_Bank, digunakan untuk menghapus data bank dari tabel bank.

4.2.2 Implementasi Proses

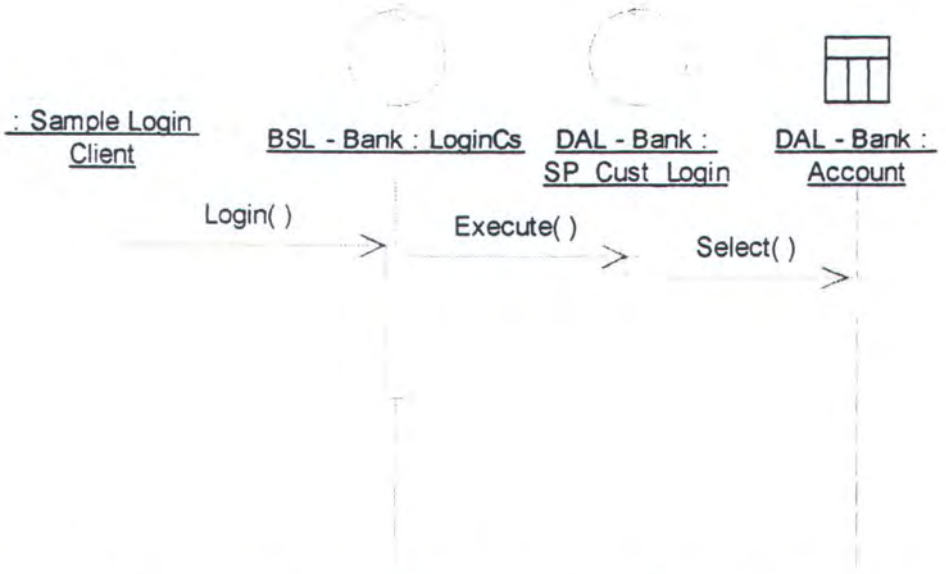
Implementasi proses adalah pengembangan dari perancangan proses. Penjelasan untuk implementasi proses ini menggunakan sequence diagram dari masing-masing proses.

1. Login :

   Setiap nasabah yang ingin melakukan transaksi melalui website bank harus melakukan login terlebih dahulu. Bila login sesuai maka nasabah bisa melakukan transaksi, jika tidak maka nasabah tidak bisa melakukan transaksi.



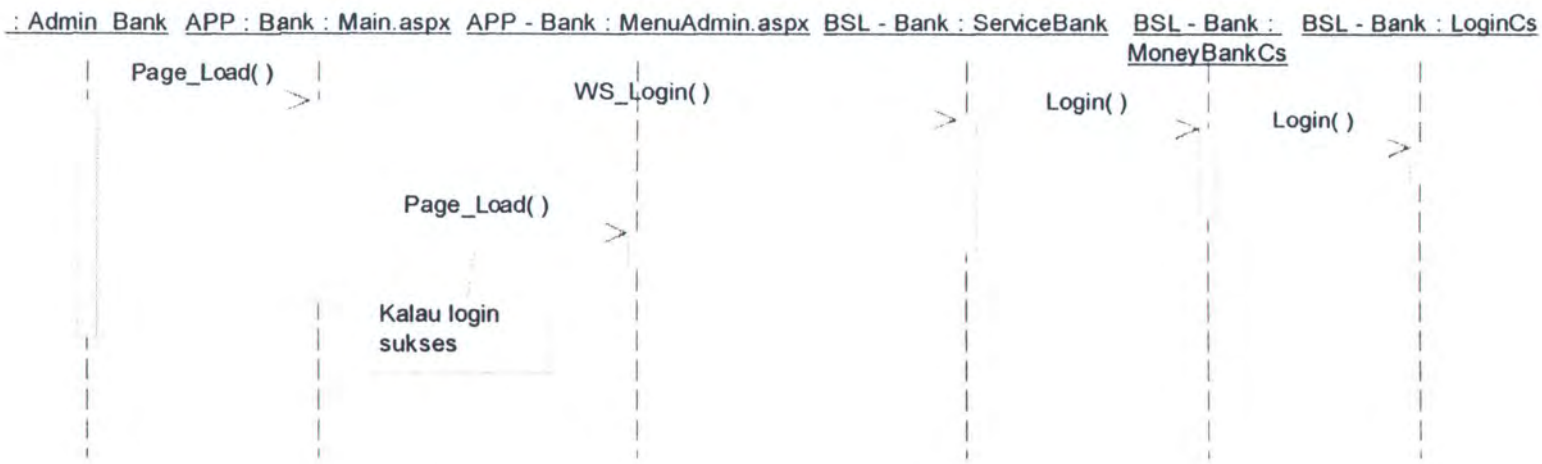
Gambar 4.33 : Sequence diagram untuk proses login nasabah sampai dengan Business Service Layer



Gambar 4.34 : Sequence diagram untuk proses login nasabah untuk Data Access Layer

2. Login admin bank :

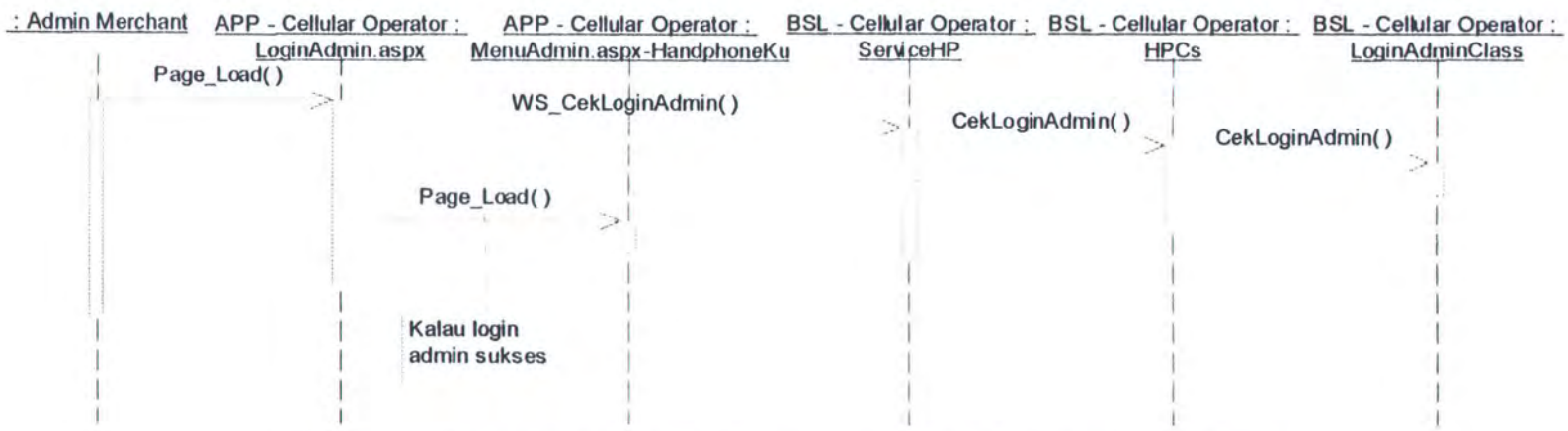
Untuk masuk ke menu administrator dan mengakses layanan untuk administrator, administrator bank harus melakukan login terlebih dahulu pada halaman Main.aspx. Bila login sesuai maka administrator bisa masuk dan mengakses layanan tersebut.



Gambar 4.35 : Sequence diagram untuk proses login administrator bank

3. Login admin merchant :

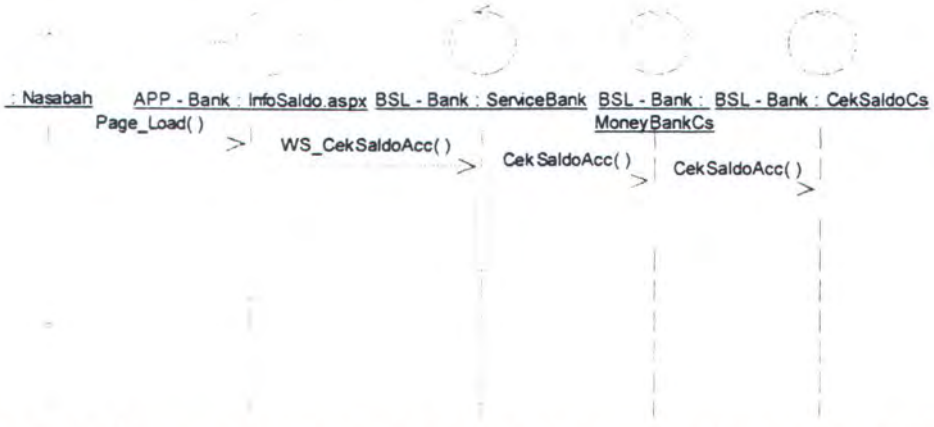
Untuk masuk ke menu administrator dan mengakses layanan untuk administrator, administrator merchant harus melakukan login terlebih dahulu pada halaman Main.aspx. Bila login sesuai maka administrator bisa masuk dan mengakses layanan tersebut.



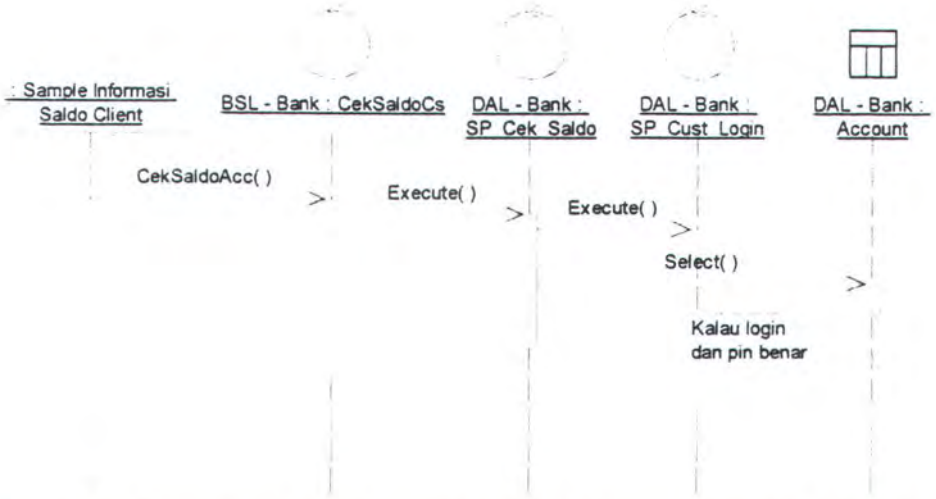
Gambar 4.36 : Sequence diagram untuk proses login administrator merchant

4. Informasi saldo :

Untuk melihat informasi saldo, nasabah cukup memilih menu Informasi Saldo pada halaman Menu dan nasabah akan melihat saldo rekeningnya pada halaman InfoSaldo.aspx.



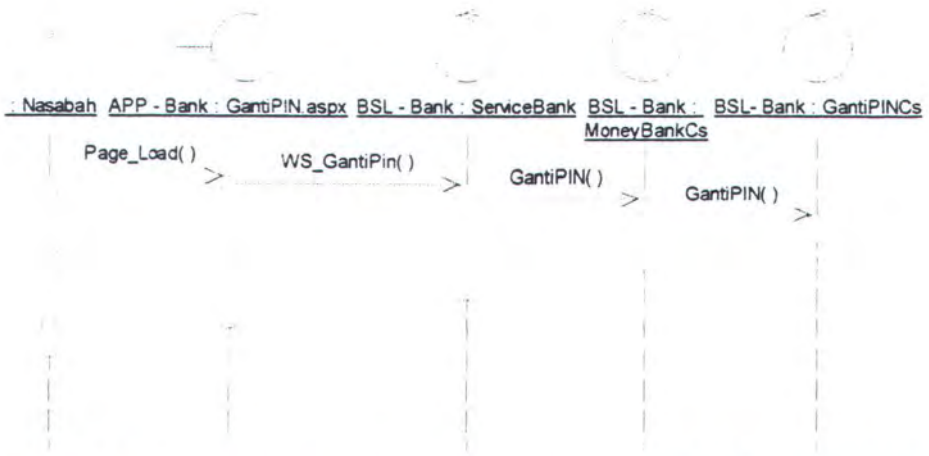
Gambar 4.37 : Sequence diagram untuk melihat informasi saldo rekening nasabah sampai dengan business service layer



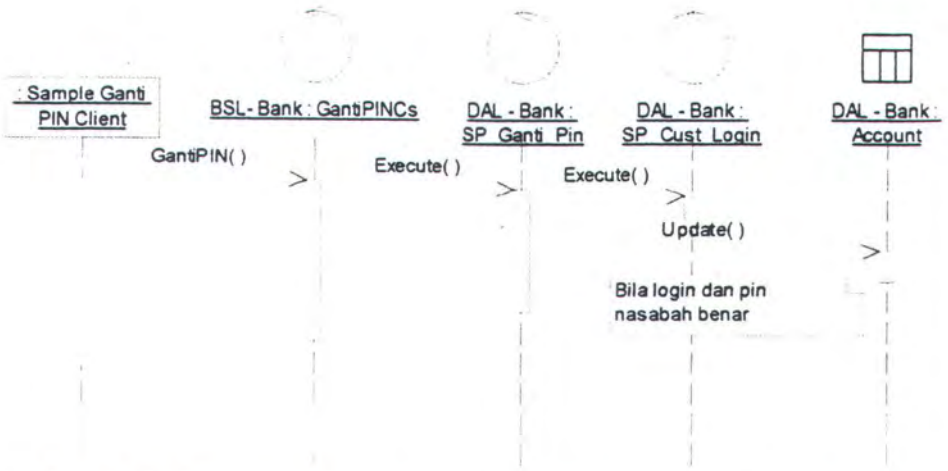
Gambar 4.38 : Sequence diagram untuk melihat informasi rekening nasabah untuk Data Access Layer

5. Ganti PIN :

Untuk mengganti PIN-nya, seorang nasabah harus memasukkan pin lama dan juga pin baru sebanyak dua kali ke halaman GantiPIN.aspx untuk kemudian dilakukan update ke tabel Account di data access layer melalui business service layer.



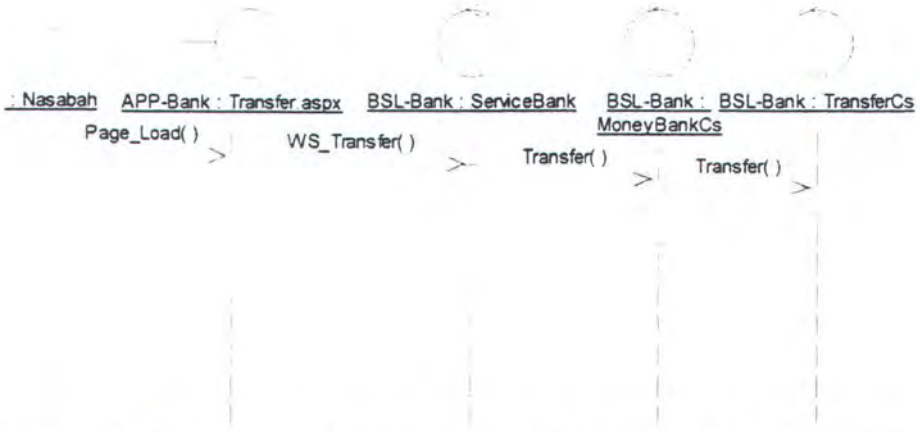
Gambar 4.39 : Sequence diagram untuk proses mengganti PIN untuk nasabah bank sampai dengan business service layer



Gambar 4.40 : Sequence diagram untuk proses mengganti PIN untuk nasabah bank untuk Data Access Layer

6. Mentransfer :

Untuk melakukan transfer, nasabah harus memasukkan nomer rekening tujuan dan jumlah dana yang ditransfer. Bila saldonya mencukupi maka transaksi sukses dan nasabah mendapatkan nomer referensi.



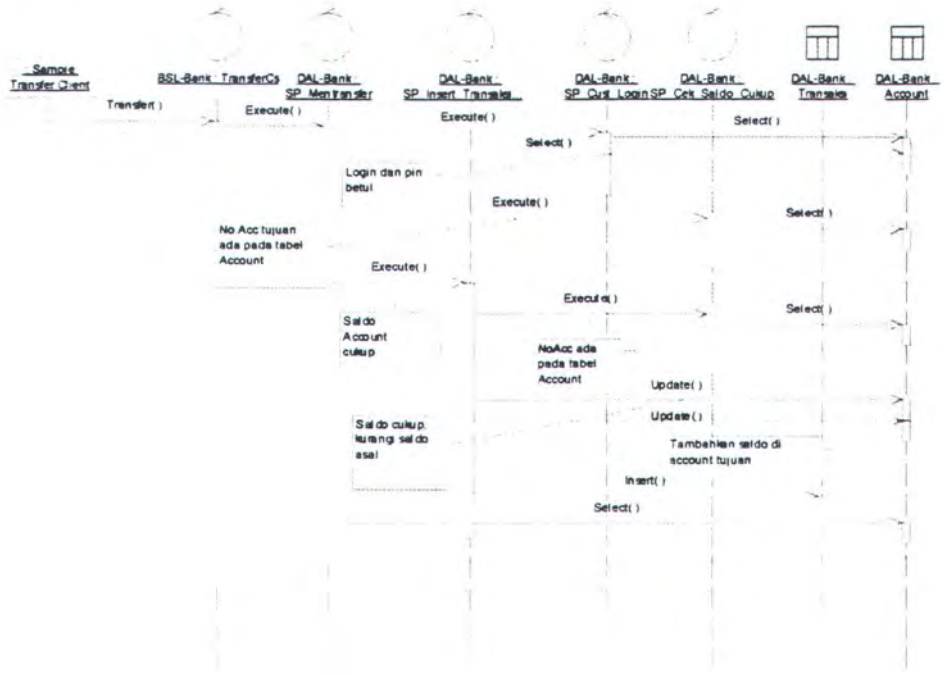
Gambar 4.41 : Sequence diagram untuk proses mentransfer sampai dengan business service layer

Nomer referensi yang didapat ini bisa digunakan sebagai bukti pembayaran. Untuk menjaga keamanan, terutama kalau transaksi terputus sebelum menyelesaikan pembelian, uang nasabah tidak boleh hilang begitu saja. Transaksi pendebetan account dan pembayaran merchant mestinya berada dalam satu atom. Untuk mengembalikan uang tersebut harus disediakan mekanisme rollbacknya, dalam hal ini memakai cara expiration atau pengkadaluarsaan transaksi.

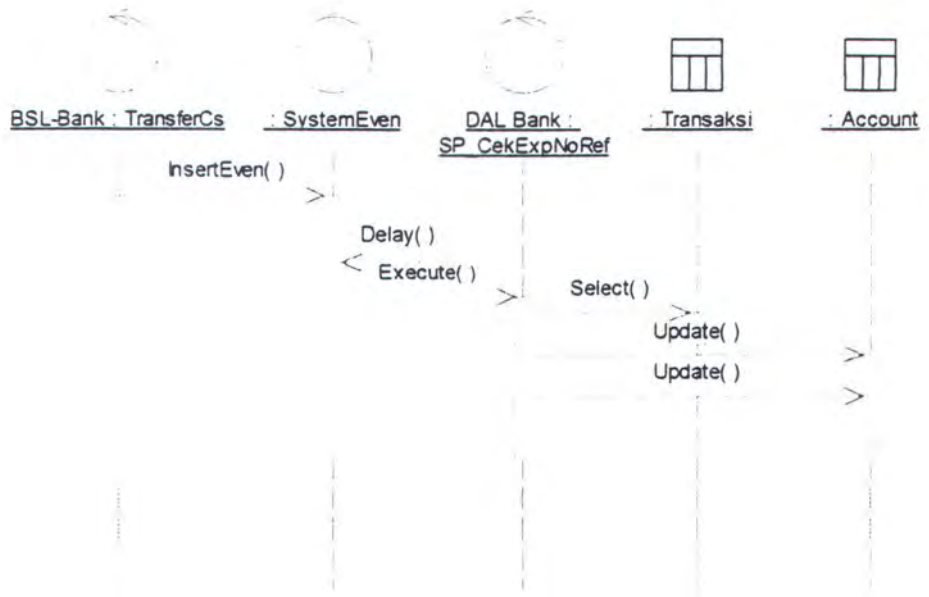
Nomer referensi yang didapat sebagai bukti transfer hanya bisa digunakan untuk kurun waktu tertentu, misalnya dua hari untuk transfer antar rekening, dua jam untuk transaksi e-commerce. Perlu adanya komponen *service* yang mengatur pengkadaluarsaan tersebut.

Dalam mekanisme pengkadaluarsaan, sistem bank akan melakukan Insert Event ke system event. Setelah melakukan delay untuk kurun waktu tertentu, system even men-execute stored procedure SP_CekExpNoRef. Stored procedure ini mengecek apakah ada nomer referensi yang lebih dari dua hari dan belum dicek, bila ada maka uang pengirim dikembalikan sehingga saldonya bertambah sedangkan uang penerima dikurangi sebanyak jumlah transfer sehingga saldonya berkurang dan status transaksi akan diset = 'R', yang berarti transaksi transfer ini dibatalkan dan uang transfer dikembalikan ke pengirim. Karena status transaksi ='R', maka nomer referensi ini sudah tidak valid lagi dan tidak bisa digunakan sebagai bukti transfer.

Service yang mengatur proses pengkadaluarsaan terletak pada class TransferCs. Class TransferCs ini terletak pada *business service layer* service bank.



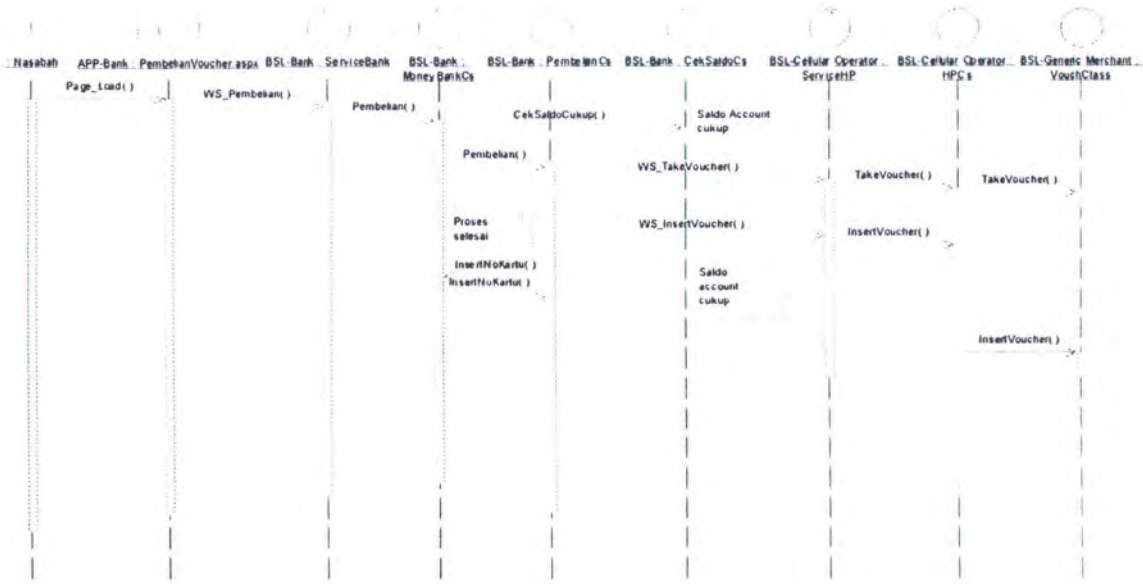
Gambar 4.42 : Sequence diagram untuk proses mentransfer untuk Data Access Layer



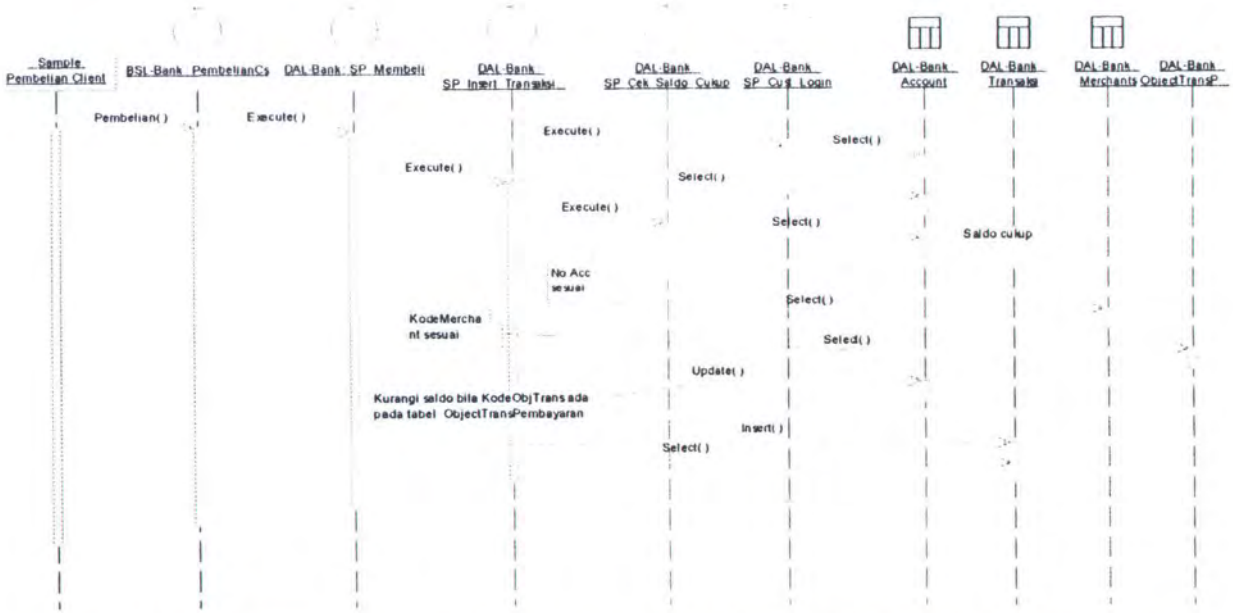
Gambar 4.43 : Sequence diagram untuk proses mengecek masa berlaku nomer referensi

7. Membeli voucher HP:

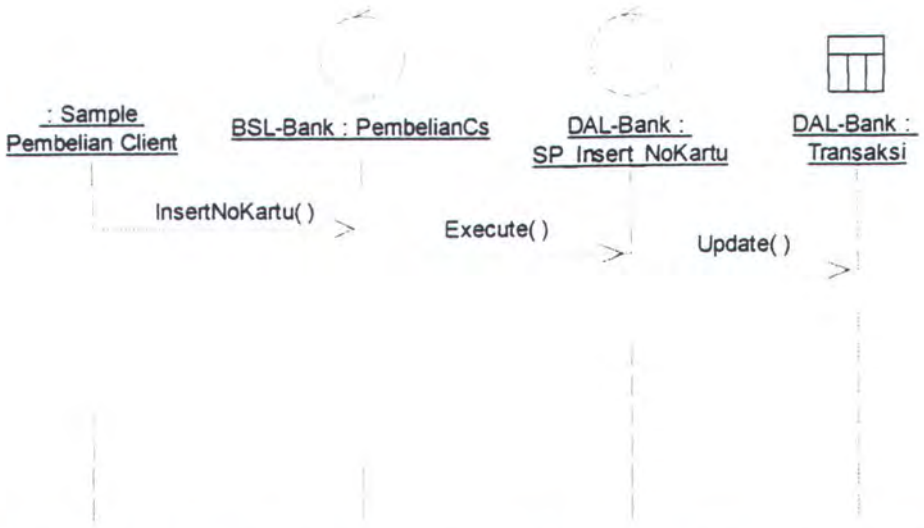
Nasabah bisa membeli voucher melalui bank dengan memotong saldo rekeningnya. Bila persediaan voucher pada merchant masih ada dan saldo pada rekeningnya cukup maka transaksi pembelian sukses dan nasabah mendapatkan nomer kartu dan kode voucher handphone.



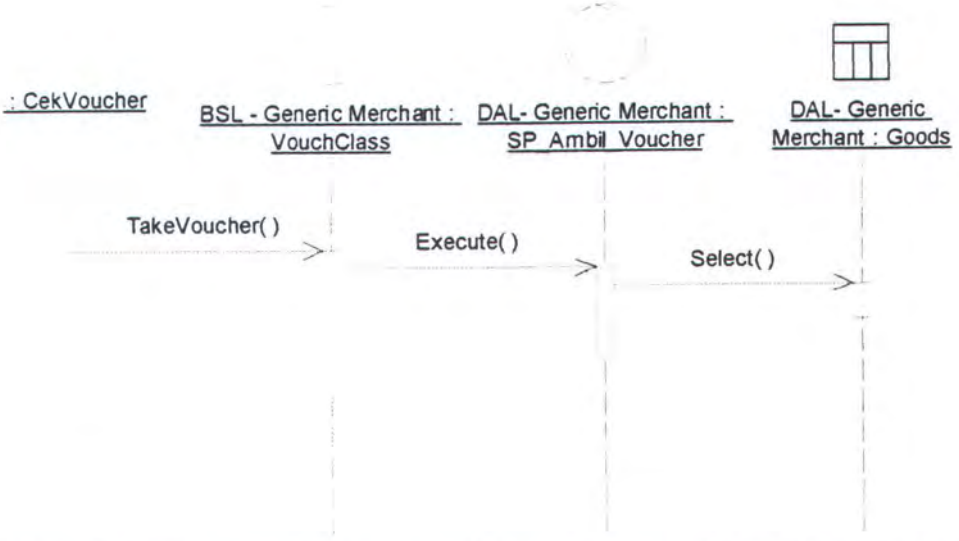
Gambar 4.44 : Sequence diagram untuk proses membeli voucher sampai dengan business service layer



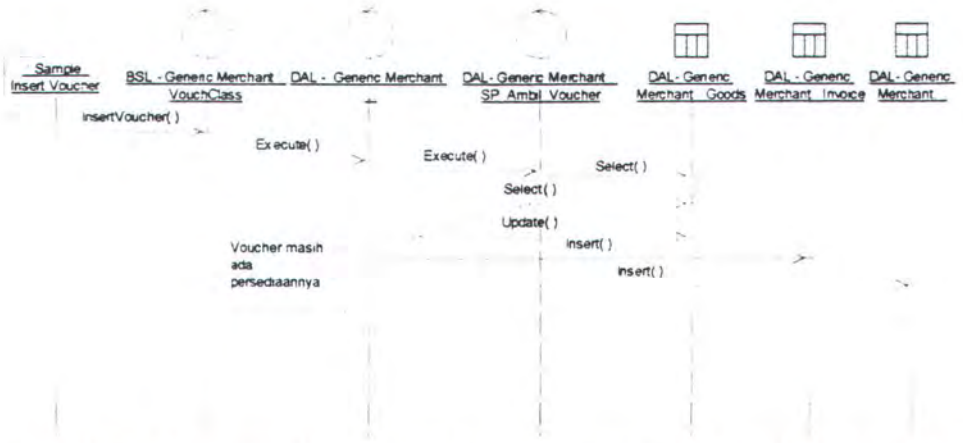
Gambar 4.45 : Sequence diagram untuk proses membeli voucher untuk Data Access Layer



Gambar 4.46 : Sequence diagram untuk proses menginsert nomer kartu ke tabel Transaksi untuk Data Access Layer



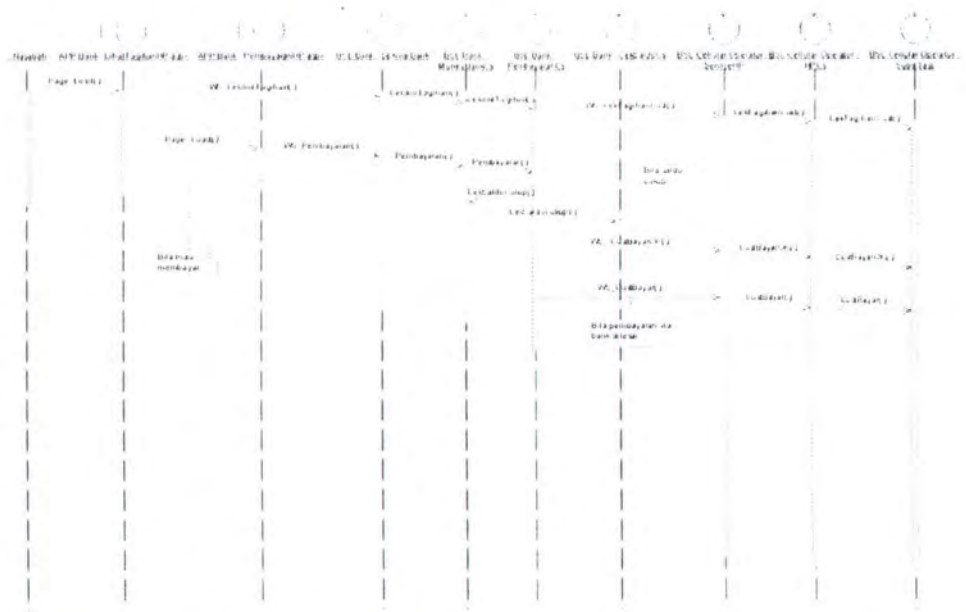
Gambar 4.47 : Sequence diagram untuk proses mengecek persediaan voucher untuk Data Access Layer Merchant



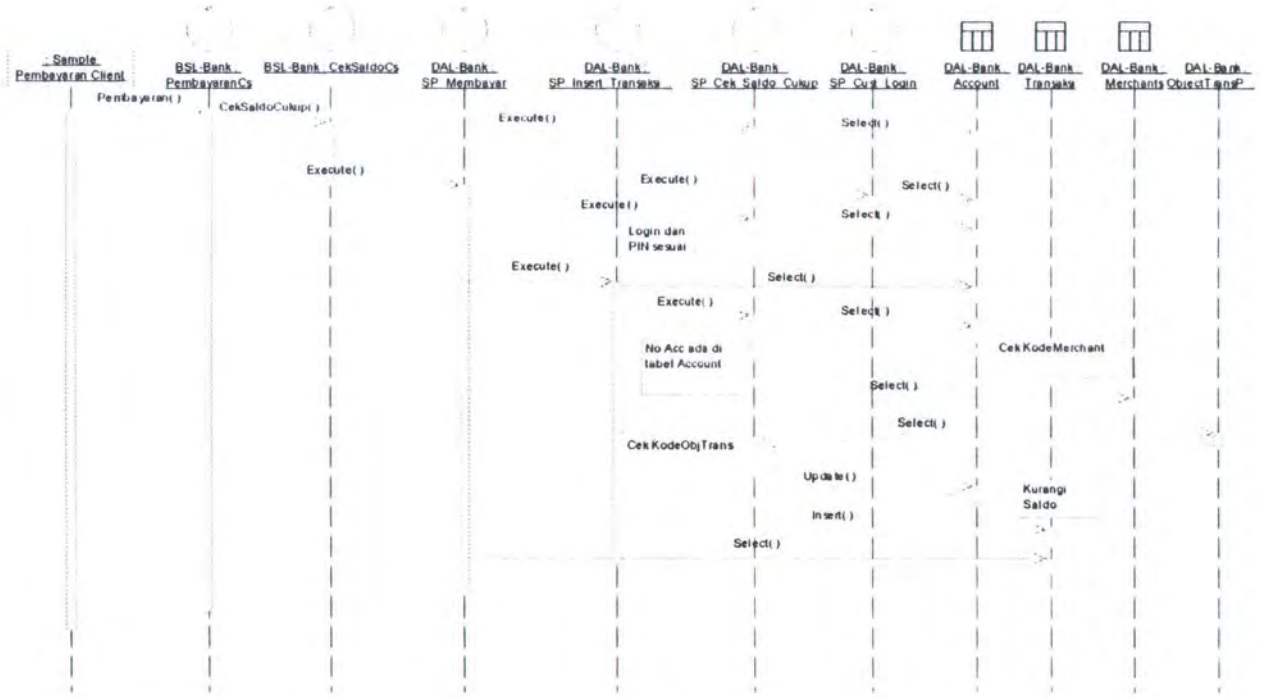
Gambar 4.48 : Sequence diagram untuk proses menginsert pembelian voucher ke tabel Invoice untuk Data Access Layer Merchant

8. Membayar Tagihan HP:

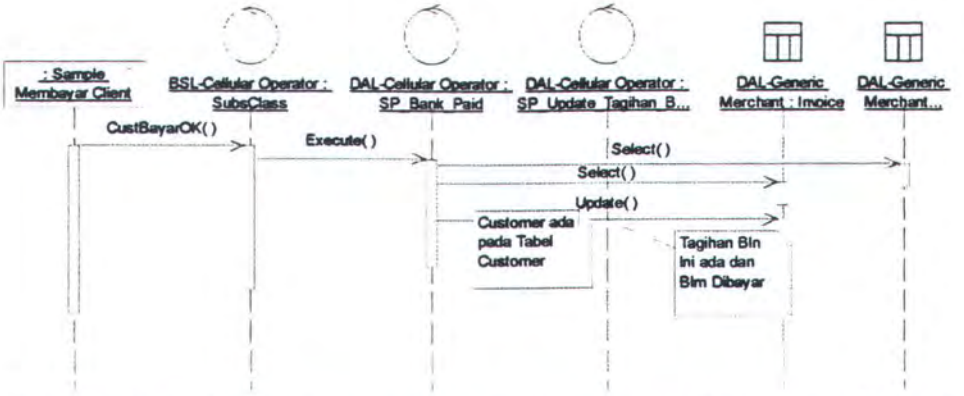
Nasabah dapat membayar tagihannya di merchant melalui website bank dengan memasukkan nomer langganannya dan bulan bayar. Bila tagihan belum terbayar, nasabah dapat membayarnya dengan memotong saldo rekeningnya bila saldonya cukup.



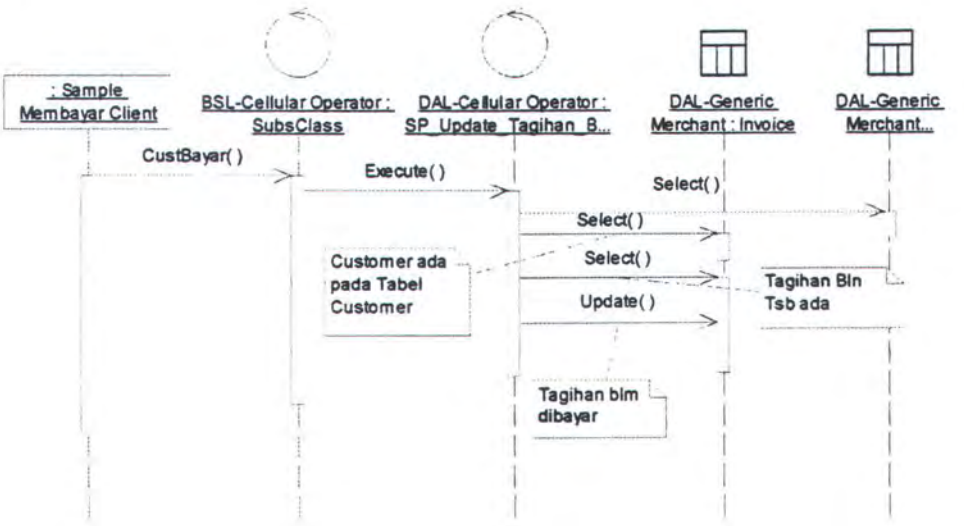
Gambar 4.49 : Sequence diagram untuk proses membayar tagihan pelanggan merchant melalui bank sampai dengan business service layer



Gambar 4.50 : Sequence diagram untuk proses membayar tagihan di merchant untuk Data Access Layer



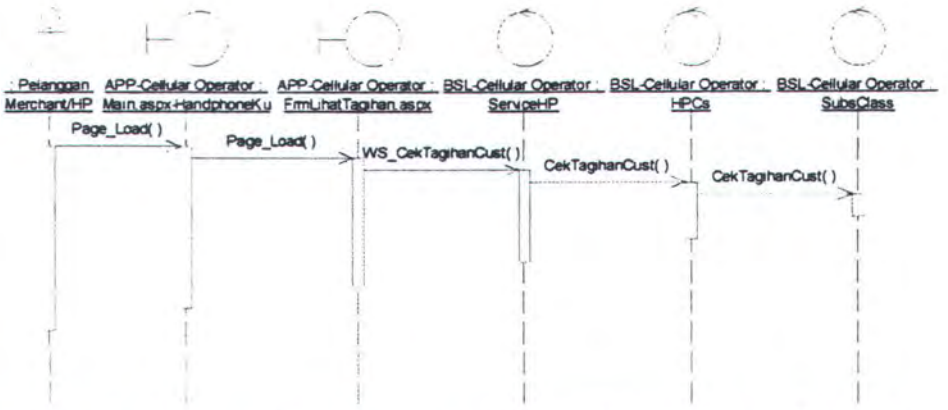
Gambar 4.51 : Sequence diagram untuk proses menandai bahwa tagihan akan dibayar melalui bank tertentu untuk Data Access Layer Merchant



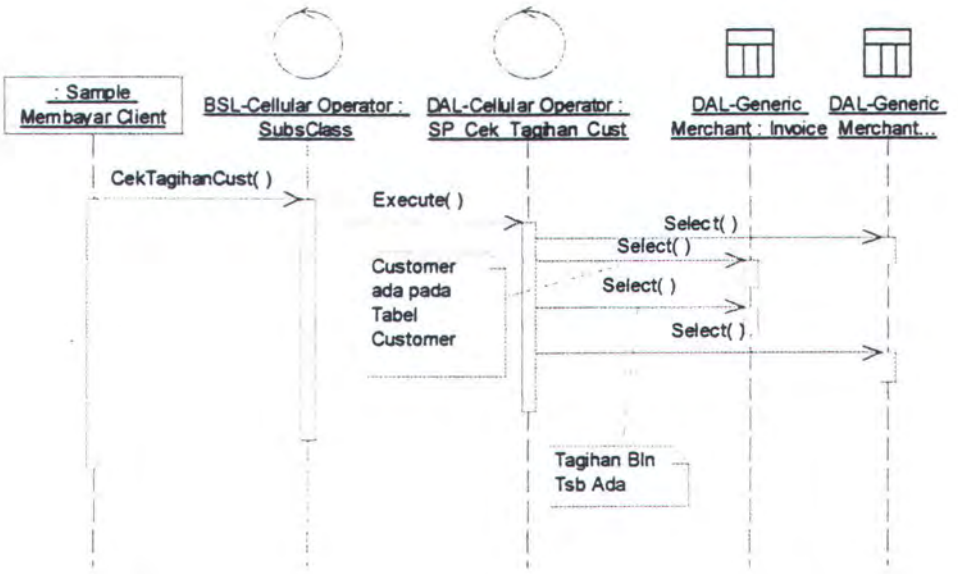
Gambar 4.52 : Sequence diagram untuk proses proses membayar tagihan pelanggan merchant melalui bank untuk Data Access Layer Merchant

9. Melihat jumlah tagihan bulanan :

Pelanggan dapat melihat jumlah tagihannya dan status pembayarannya di merchant melalui website merchant atau ketika akan membayar tagihan melalui website bank.



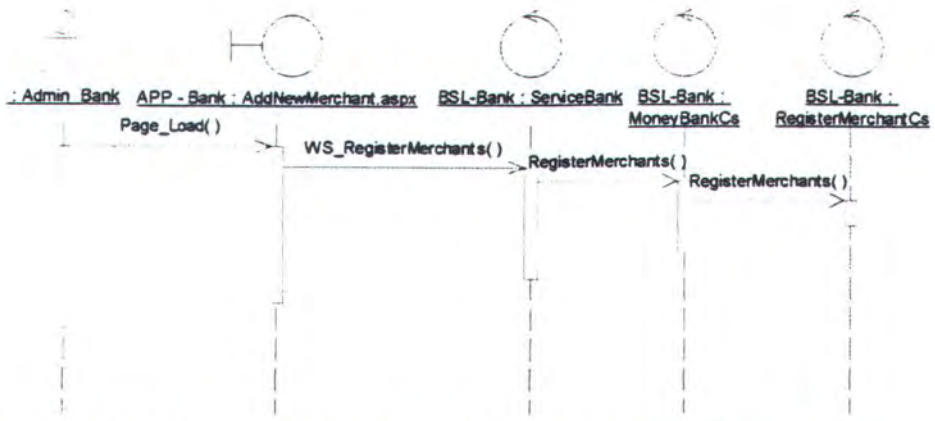
Gambar 4.53 : Sequence diagram untuk proses melihat jumlah tagihan pelanggan di merchant sampai dengan business service layer



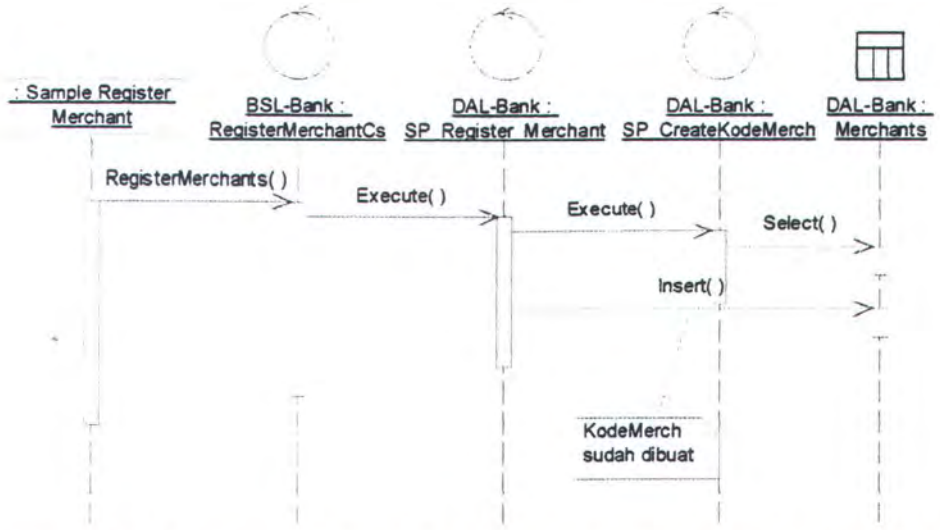
Gambar 4.54 : Sequence diagram untuk proses melihat jumlah tagihan pelanggan di merchant untuk Data Access Layer

10. Register merchant :

Proses registrasi merchant ke database bank hanya bisa dilakukan oleh administrator bank dengan memasukkan nama merchant lewat halaman AddNewMerchant.aspx.



Gambar 4.55 : Sequence diagram untuk proses meregister merchant sampai dengan business service layer

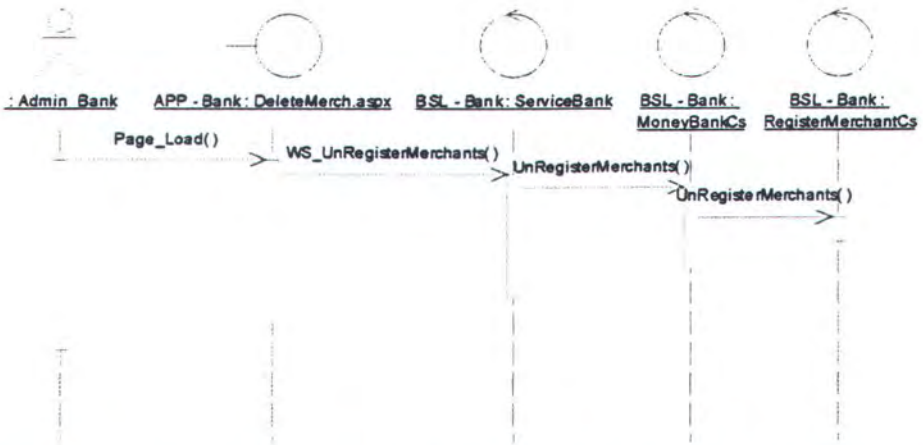


Gambar 4.56 : Sequence diagram untuk proses meregister merchant untuk Data Access Layer

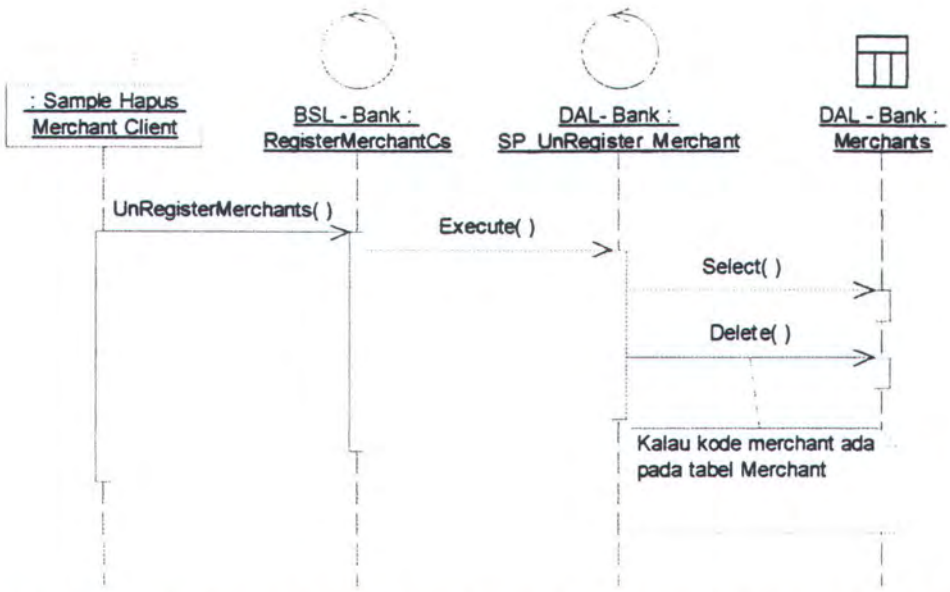
11. Hapus Merchant :

Penghapusan merchant dari database bank hanya bisa dilakukan oleh

administrator bank dengan memilih nama merchant yang akan dihapus pada halaman DeleteMerch.aspx.



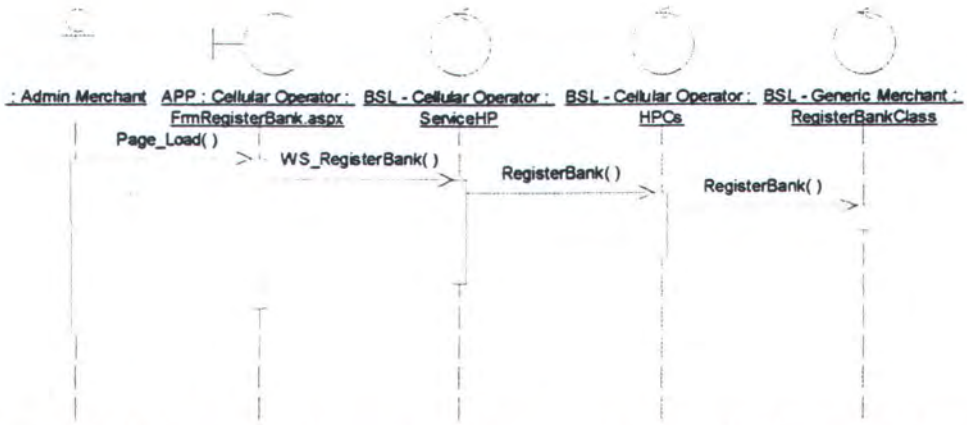
Gambar 4.57 : Sequence diagram untuk proses menghapus merchant dari database bank sampai dengan business service layer



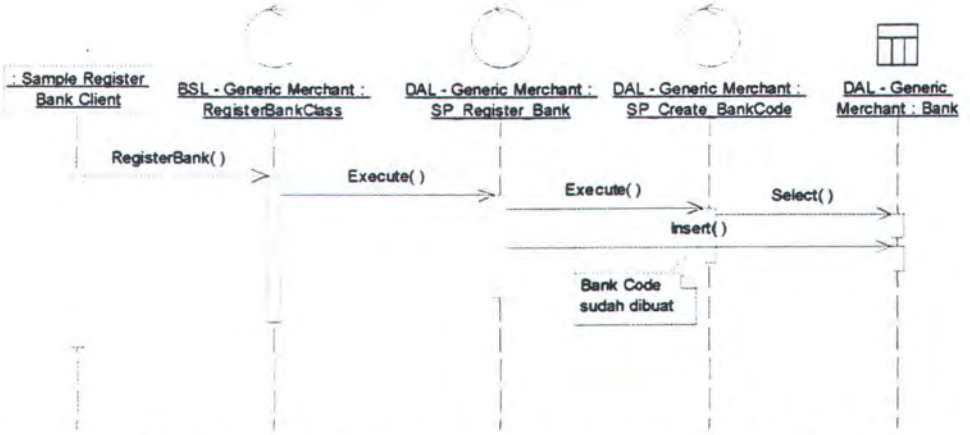
Gambar 4.58 : Sequence diagram untuk proses menghapus merchant dari database bank untuk Data Access Layer

12. Register Bank :

Registrasi bank hanya bisa dilakukan oleh administrator merchant dengan menginputkan nama bank.



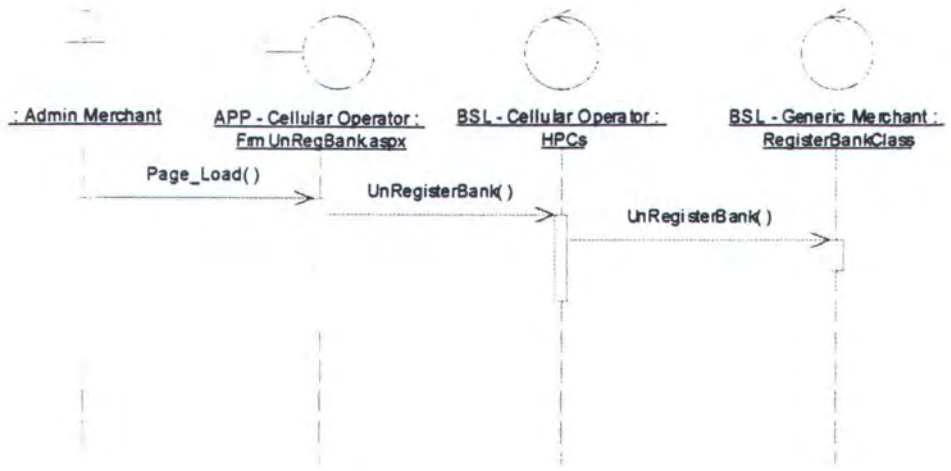
Gambar 4.59 : Sequence diagram untuk proses meregister bank sampai dengan business service layer



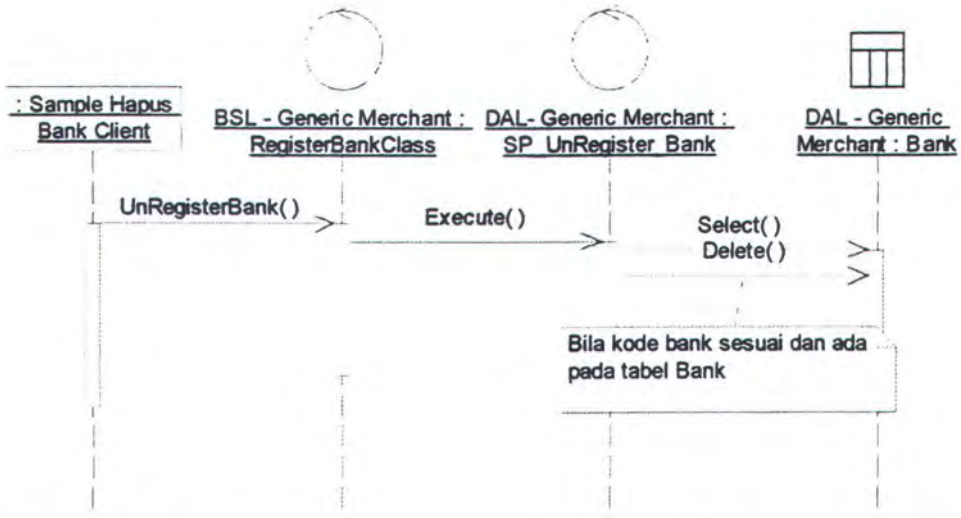
Gambar 4.60 : Sequence diagram untuk proses meregister bank untuk Data Access Layer

13. Hapus Bank :

Penghapusan bank dari database merchant hanya bisa dilakukan oleh administrator merchant dengan memilih nama bank yang akan dihapus pada halaman FrmUnRegBank.aspx.



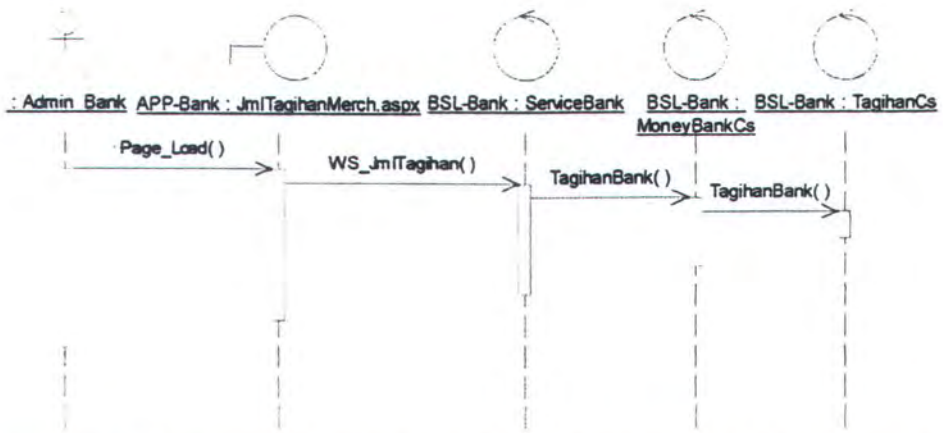
Gambar 4.61 : Sequence diagram untuk proses menghapus bank dari database merchant sampai dengan business service layer



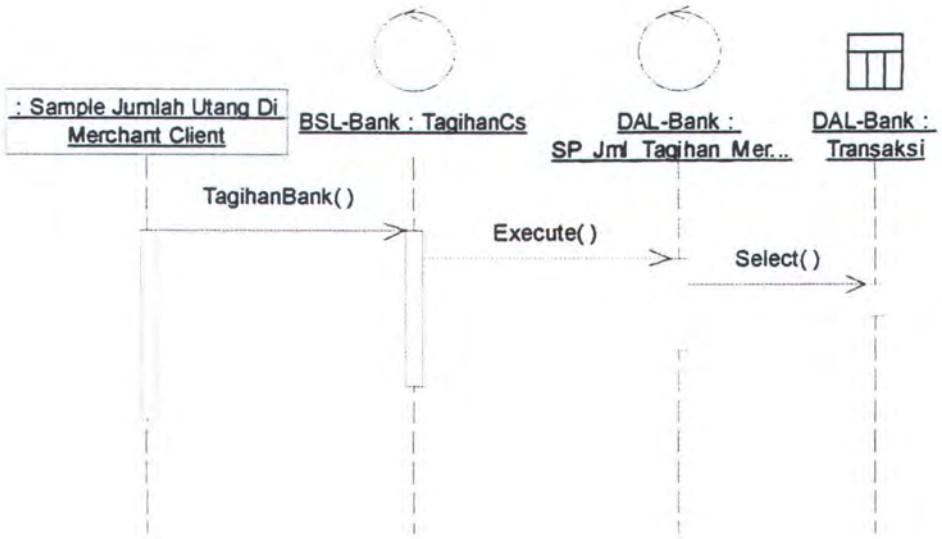
Gambar 4.62 : Sequence diagram untuk proses menghapus bank dari database merchant untuk Data Access Layer

14.Lihat jumlah utang bank ke merchant :

Untuk melihat jumlah utang bank kepada merchant dari hasil transaksi pembayaran tagihan pelanggan merchant dan atau pembelian voucher merchant melalui bank maka administrator harus menginputkan bulan bayar dan memilih nama merchant pada halaman JmlTagihanMerch.aspx.



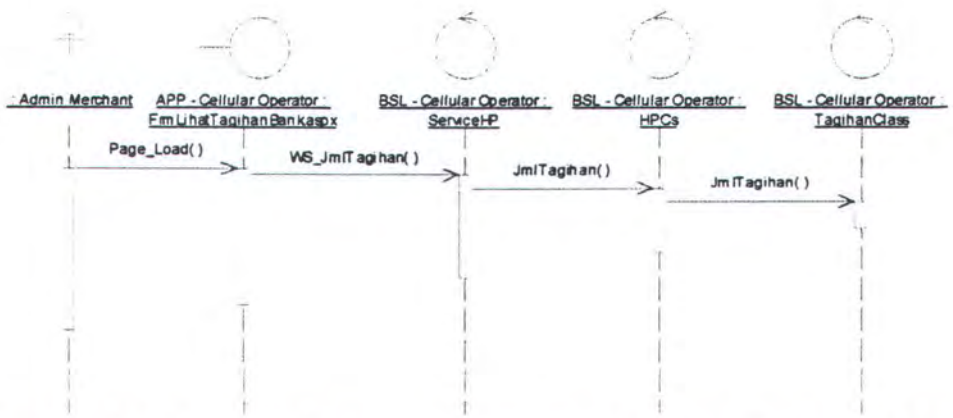
Gambar 4.63 : Sequence diagram untuk proses melihat jumlah utang bank ke merchant sampai dengan business service layer



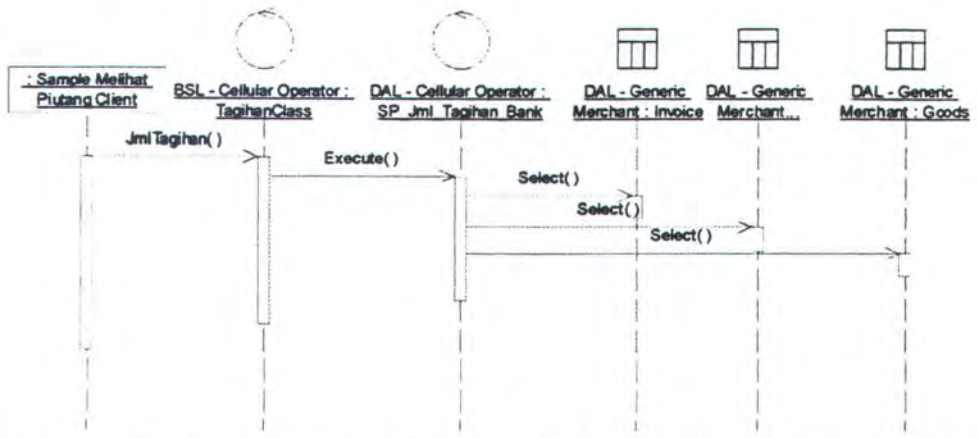
Gambar 4.64 : Sequence diagram untuk proses melihat jumlah utang bank ke merchant untuk Data Access Layer

15.Lihat jumlah piutang merchant di bank:

Untuk melihat jumlah piutang di bank dari hasil transaksi pembayaran tagihan pelanggan merchant dan atau pembelian voucher merchant melalui bank maka administrator harus menginputkan bulan bayar dan memilih nama merchant pada halaman FrmLihatTagihanBank.aspx.



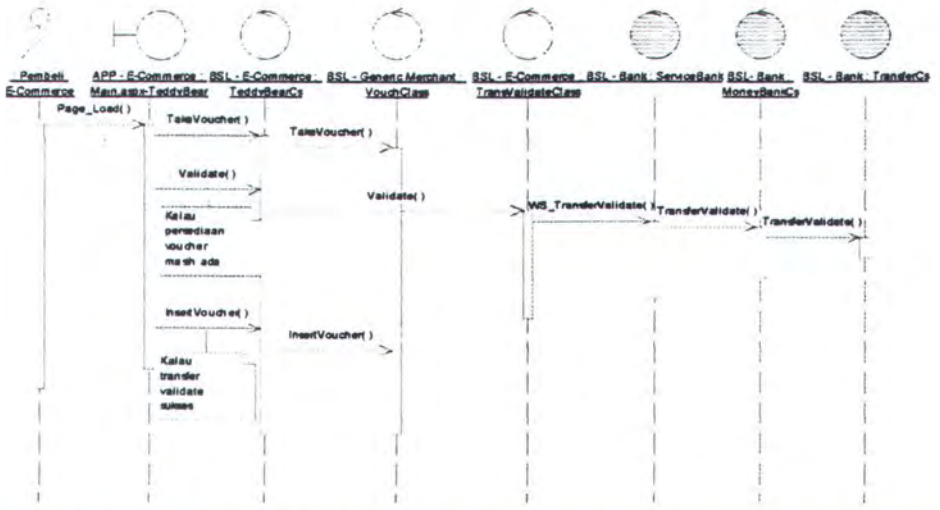
Gambar 4.65 : Sequence diagram untuk proses melihat jumlah piutang di bank sampai dengan business service layer



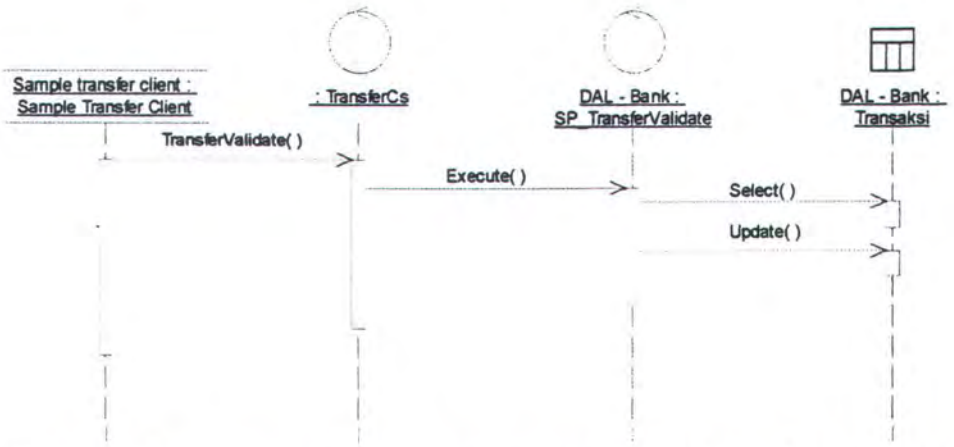
Gambar 4.66 : Sequence diagram untuk proses melihat jumlah piutang di bank untuk Data Access Layer

16. Beli barang:

Untuk membeli barang di e-commerce TeddyBear, pembeli harus memilih voucher dan mentransfer dana ke rekening e-commerce sejumlah total transaksi. Setelah itu pembeli harus memasukkan No. Referensi yang didapat dari transaksi transfer tersebut untuk kemudian divalidasi oleh sistem e-commerce ke sistem bank. Bila sukses maka pembeli mendapatkan nomer kartu dan kode voucher handphone tersebut. No. Referensi ini hanya bisa digunakan maksimal dua hari sejak nasabah mendapatkannya.



Gambar 4.67 : Sequence diagram untuk proses pembelian barang untuk transaksi e-commerce sampai dengan business service layer



Gambar 4.68 : Sequence diagram untuk proses validasi transfer untuk Data Access Layer

Sistem ini menggunakan dua webservice, yaitu webservice bank, ServiceBank, dan webservice cellular operator, ServiceHP.

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Web;
using System.Web.Services;

namespace MoneyBank
{
      public class ServiceBank : System.Web.Services.WebService
      {
            public ServiceBank()
            {
                  InitializeComponent();
            }

            #region Component Designer generated code
            private void InitializeComponent()
            {
            }
            #endregion
            protected override void Dispose( bool disposing )
            {
            }

            [WebMethod(Description="CUST ===> Mentransfer")]
            public Transf WS_Transfer(string NoAcc, string Pin, string AccNoDest,
decimal Jml)
            {
                  MoneyBank.MoneyBankCs  x = new MoneyBank.MoneyBankCs();
                  return(x.Transfer(NoAcc,Pin,AccNoDest,Jml));
            }


            [WebMethod(Description="CUST ===> Membayar")]
            public  Bayar  WS_Pembayaran(string  NoAcc,  string  Pin,  string
NoPelanggan, string BulanBayar, string KodeMerchants, int KodeObjTrans)//, decimal
Jumlah)
            {
                  MoneyBank.MoneyBankCs  x = new MoneyBank.MoneyBankCs();

      return(x.Pembayaran(NoAcc,Pin,NoPelanggan,BulanBayar,KodeMerchants,KodeObjTr
ans));//,Jumlah));
            }

            [WebMethod(Description="CUST ===> Membeli")]
            public  Beli  WS_Pembelian(string  NoAcc,  string  Pin,  string
KodeMerchants, int KodeObjTrans, decimal Jumlah)
            {
                  MoneyBank.MoneyBankCs  x = new MoneyBank.MoneyBankCs();

      return(x.Pembelian(NoAcc,Pin,KodeMerchants,KodeObjTrans,Jumlah));
            }
```

Gambar 4.69 : Webservice Bank

4.2.3 Implementasi Antarmuka

Implementasi antarmuka dibangun berdasarkan rancangan antarmuka yang sudah ditetapkan. Implementasi antarmuka ini dibangun dengan berbasis web dengan menggunakan bahasa pemrograman web aspx.

MoneyBank

- Halaman Utama

  Main.aspx adalah halaman utama pada MoneyBank. Untuk melakukan login dilakukan pada halaman utama ini.

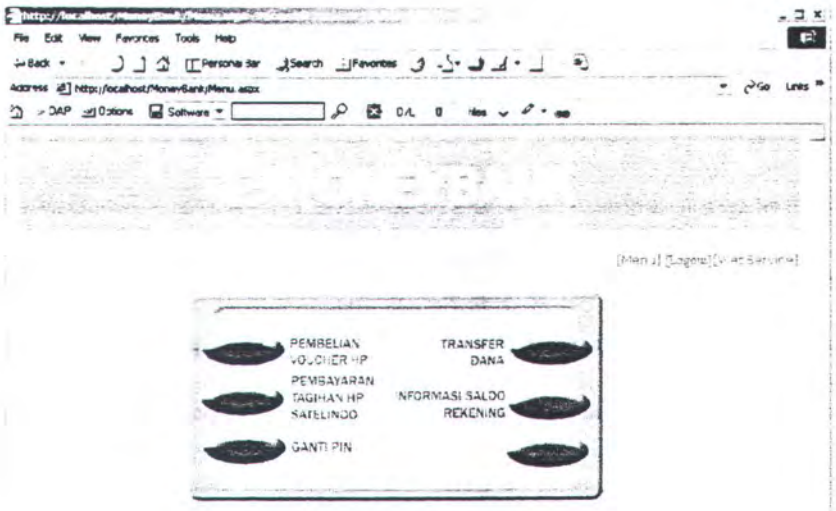
Gambar 4.70 : Main.aspx

- Halaman Menu Untuk Nasabah Bank

  Menu.aspx adalah halaman yang menampilkan menu untuk nasabah bank.

Terdapat lima pilihan menu, yaitu :

1. pembelian voucher hp,
2. pembayaran tagihan hp satelindo (merchant subscribe yang digunakan sebagai sistem bantu untuk tugas akhir ini),
3. ganti pin,
4. transfer dana,
5. informasi saldo rekening.

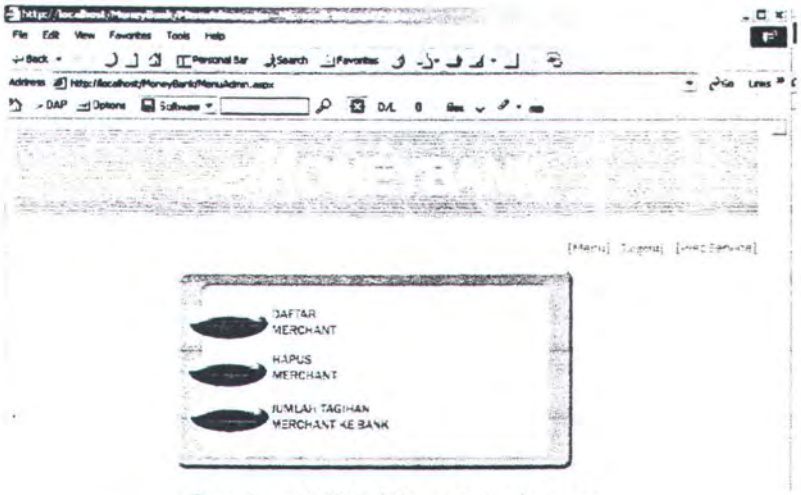


Gambar 4.71 : Menu.aspx

- Halaman Menu untuk Administrator Bank

MenuAdmin.aspx adalah halaman yang menampilan menu untuk adminstrator bank. Menu yang tersedia adalah :

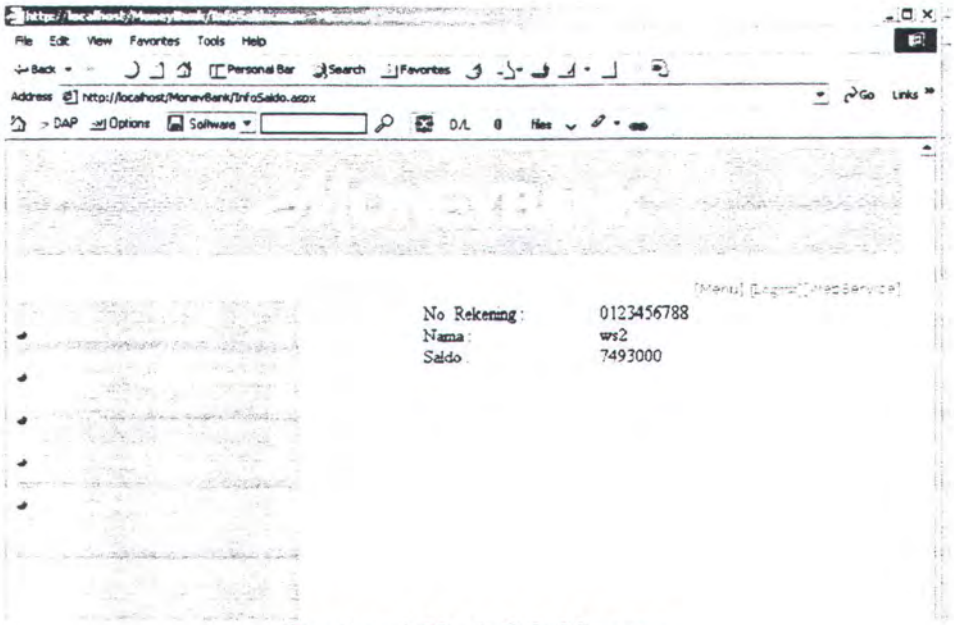
1. daftar merchant,
2. hapus merchant,
3. jumlah tagihan merchant ke bank.



Gambar 4.72 : MenuAdmin.aspx

- Halaman Informasi Saldo

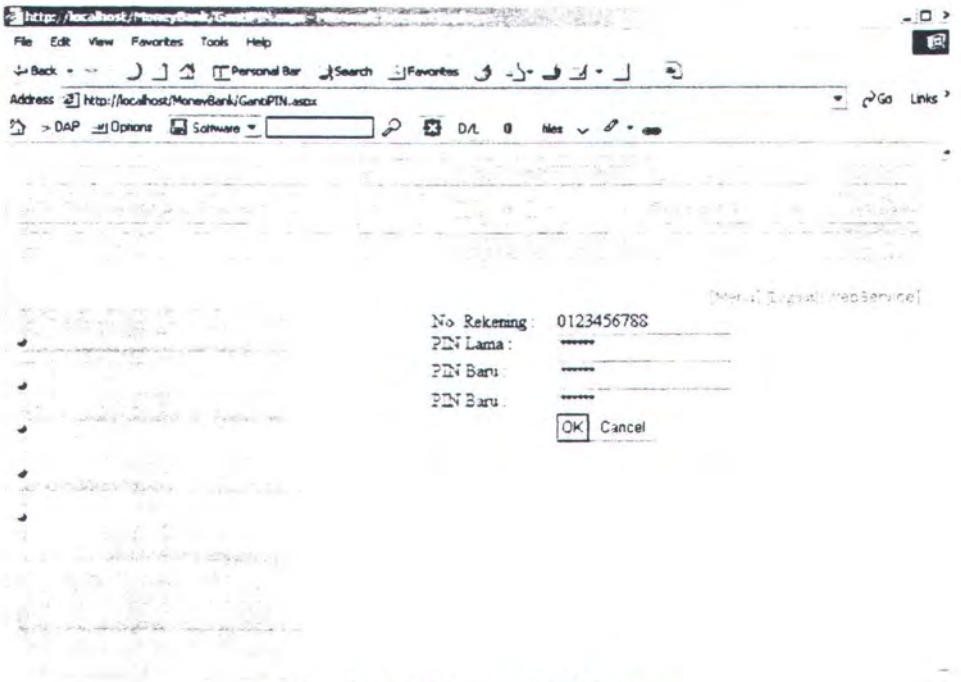
  InfoSaldo.aspx adalah halaman yang menampilkan informasi saldo rekening nasabah.



Gambar 4.73 : InfoSaldo.aspx

- Halaman Ganti PIN

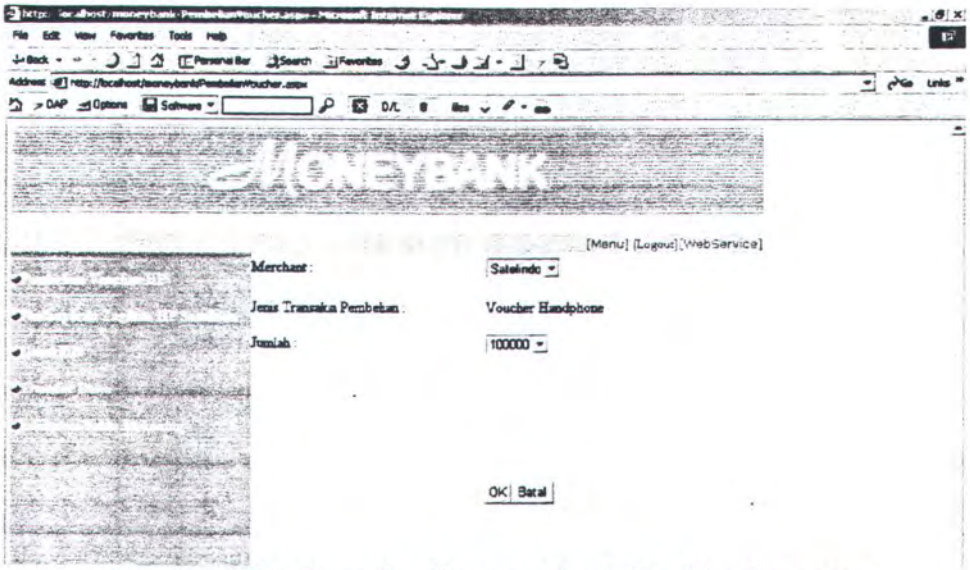
GantiPIN.aspx adalah halaman untuk mengganti pin.



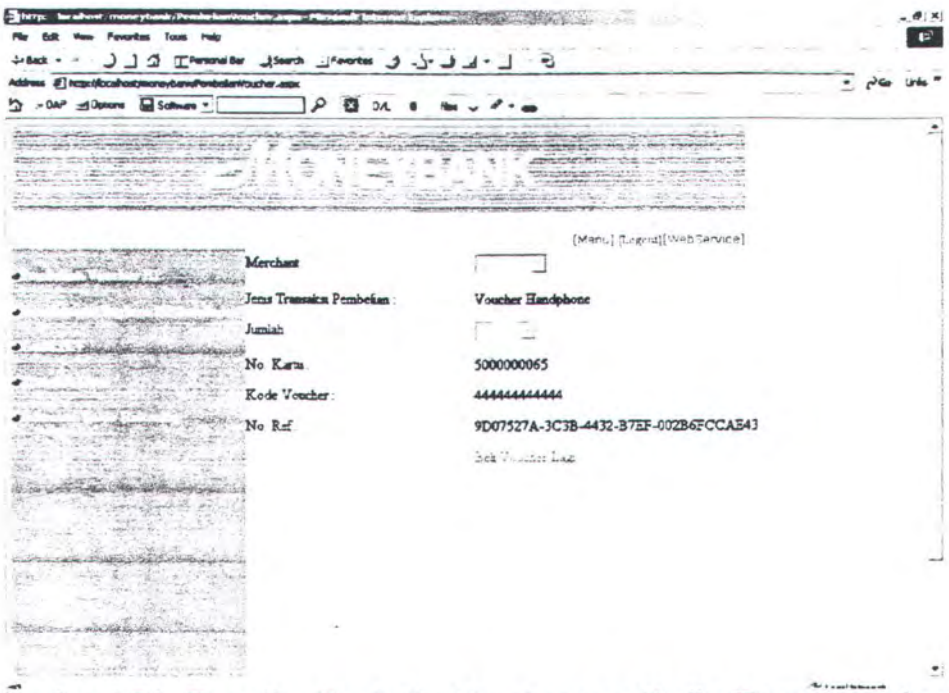
Gambar 4.74 : GantiPIN.aspx

- Halaman Pembelian Voucher HP

PembelianVoucher.aspx adalah halaman untuk pembelian voucher.



Gambar 4.75 : Tampilan Pertama PembelianVoucher.aspx

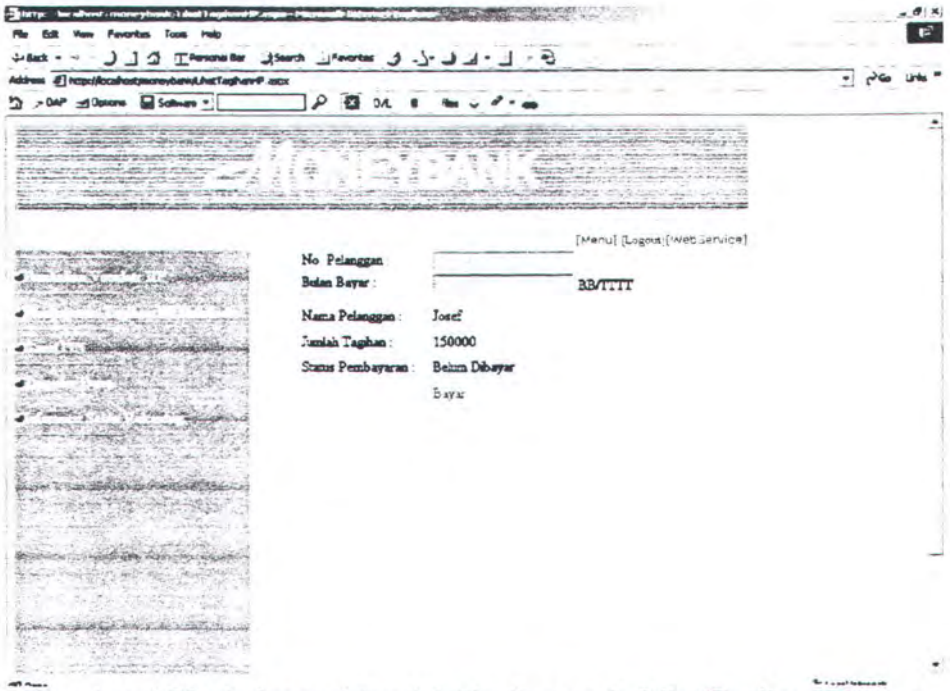


Gambar 4.76 : Tampilan PembelianVoucher.aspx Ketika Transaksi Sukses

- Halaman Lihat Jumlah Tagihan Pelanggan Untuk Pembayaran Tagihan HP Satelindo

  Ketika nasabah memilih menu Pembayaran Tagihan HP Satelindo, maka halaman yang akan terlihat adalah halaman untuk mengecek jumlah tagihan, yaitu halaman LihatTagihanHP.aspx. Bila tagihan belum terbayar dan nasabah ingin membayar, baru masuk ke halaman PembayaranHP.aspx untuk melakukan transaksi pembayaran.

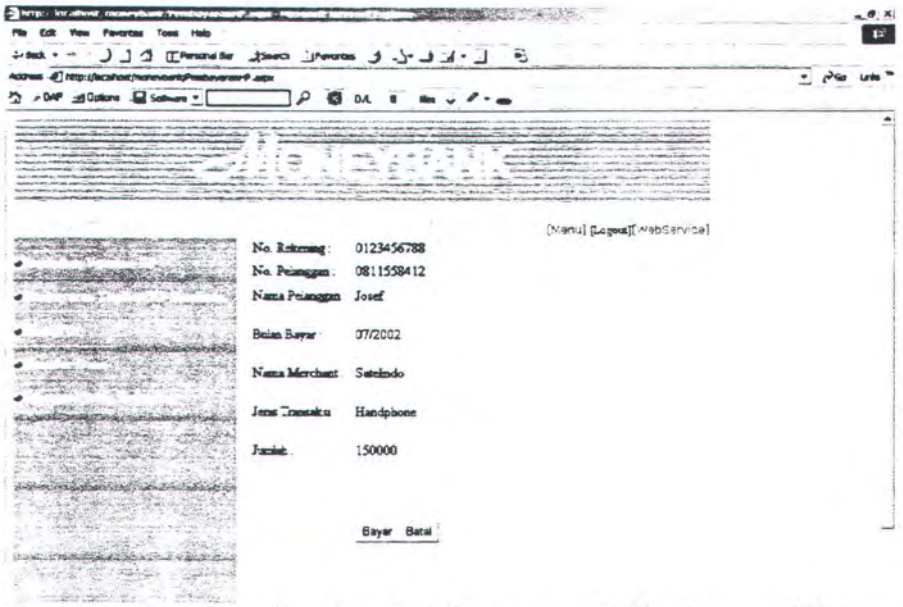
Gambar 4.77 : Tampilan Pertama LihatTagihanHP.aspx



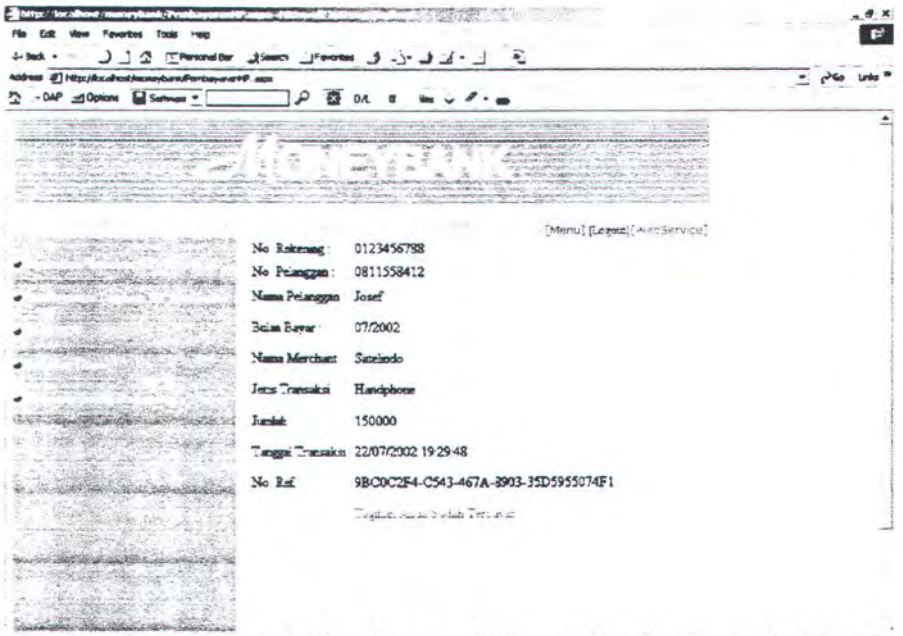
Gambar 4.78 : Informasi Jumlah Tagihan pada LihatTagihanHP.aspx

- Halaman Pembayaran Tagihan HP Satelindo

  Bila nasabah mau membayar tagihannya, maka ia harus meng-klik hyperlink Bayar pada LihatTagihanHP.aspx dan ia akan masuk ke halaman PembayaranHP.aspx .



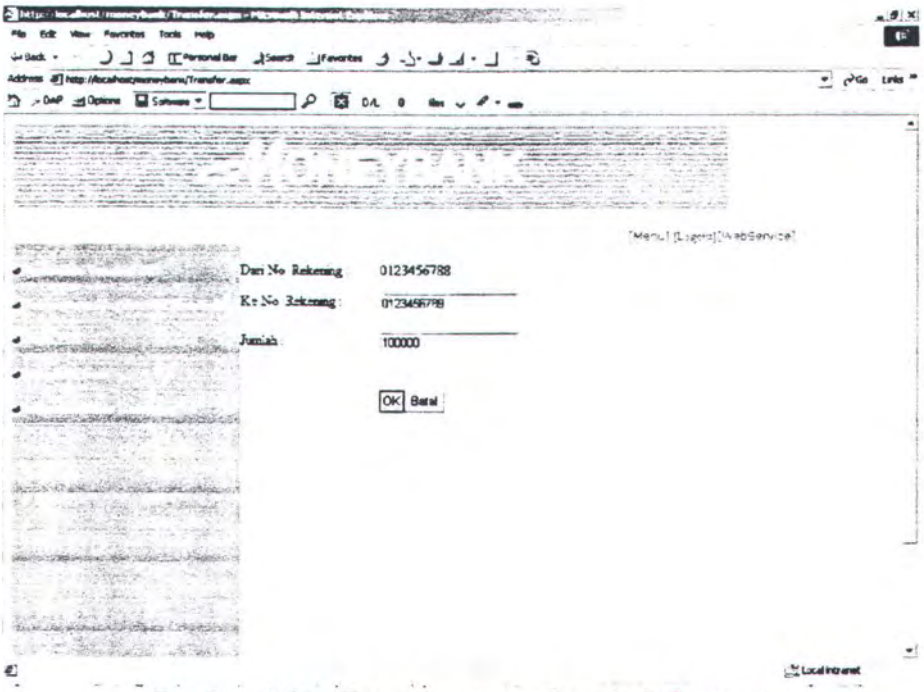
Gambar 4.79 : Tampilan Langkah Pertama pada PembayaranHP.aspx



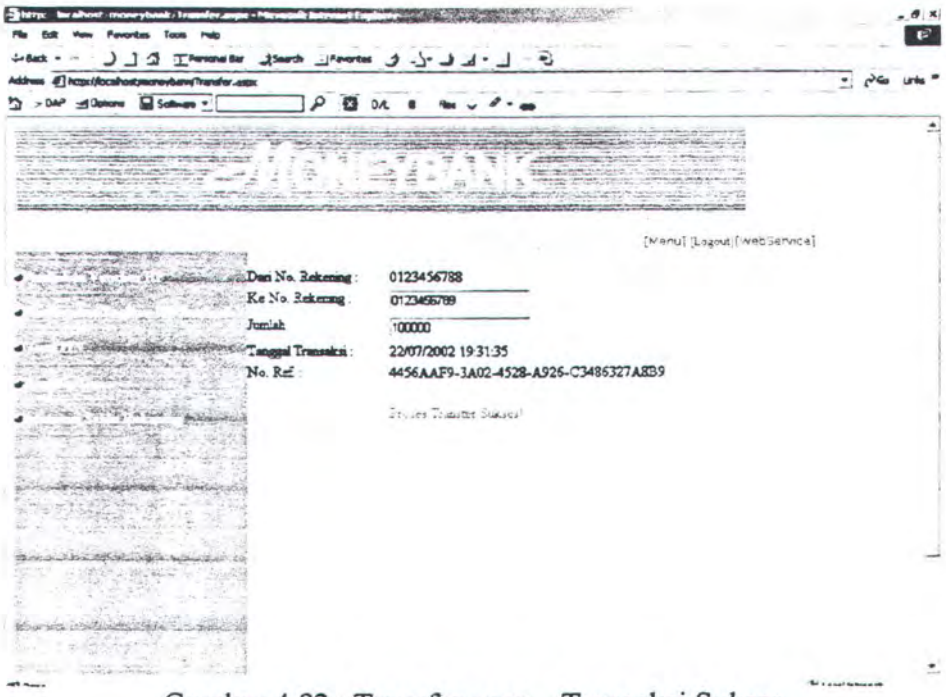
Gambar 4.80 : Tampilan pada PembayaranHP.aspx Ketika Transaksi Pembayaran Sukses

- Halaman Untuk Mentransfer

Transfer.aspx adalah halaman untuk melakukan transfer.



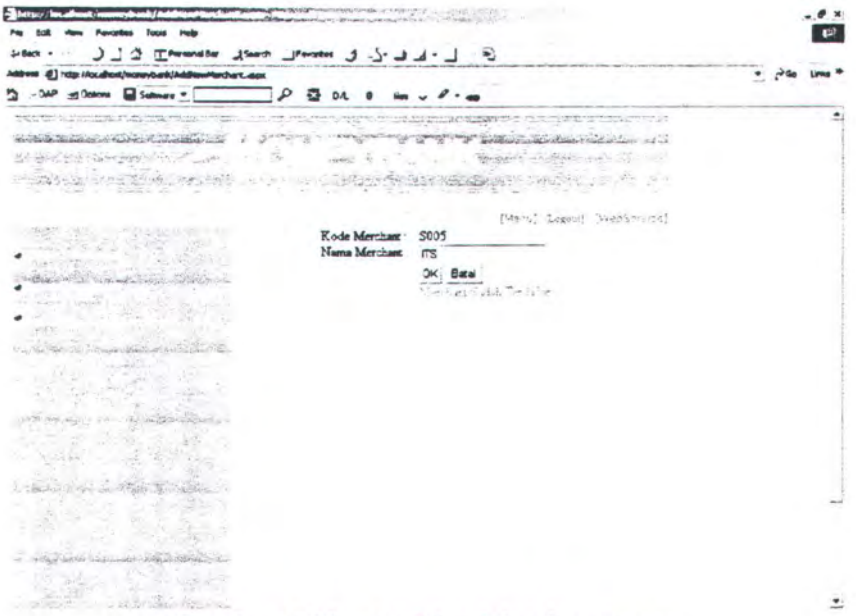
Gambar 4.81 : Transfer.aspx – Langkah Pertama



Gambar 4.82 : Transfer.aspx – Transaksi Sukses

- Halaman Daftar Merchant

  AddNewMerchant.aspx adalah halaman untuk mendaftar/meregistrasi merchant.



Gambar 4.83 : AddNewMerchant.aspx

- Halaman untuk Hapus Merchant

  DeleteMerch.aspx adalah halaman untuk menghapus merchant dari tabel Merchants.

Gambar 4.84 : DeleteMerch.aspx

- Halaman Lihat Jumlah Tagihan Merchant Ke Bank

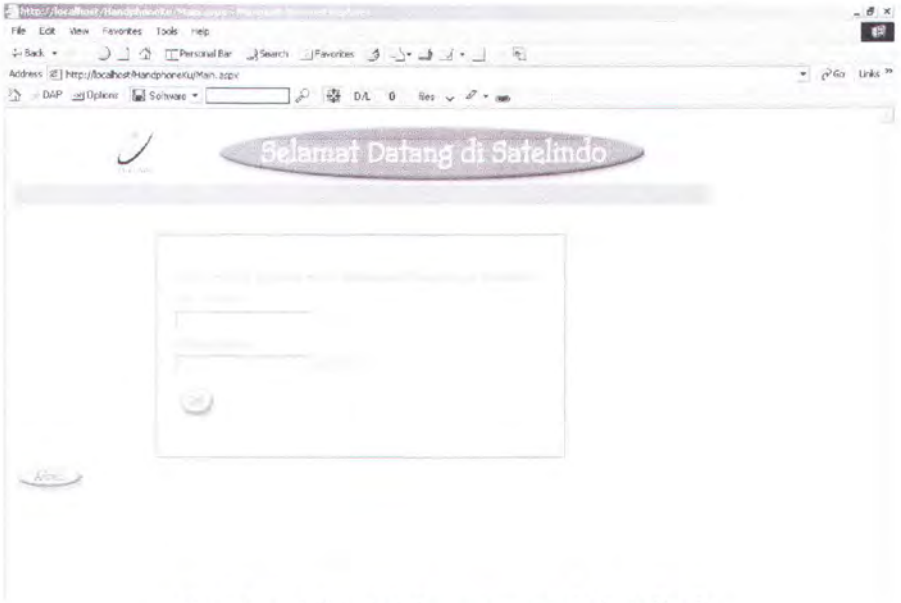
  JmlTagihanMerch.aspx adalah halaman untuk melihat jumlah tagihan merchant kepada bank untuk transaksi yang dilakukan melalui bank.

Gambar 4.85 : JmlTagihanMerch.aspx

Cellular Operator-Satelindo

- Halaman Utama

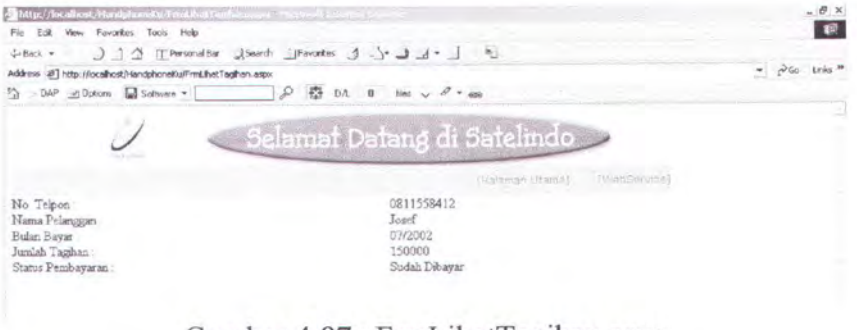
  Main.aspx adalah halaman utama pada Satelindo. Pada halaman ini pelanggan bisa mengecek jumlah tagihan, sedangkan untuk administrator merchant harus meng-klik hyperlink Admin untuk masuk ke halaman login admin.



Gambar 4.86 : Main.aspx untuk Satelindo

- Halaman Lihat Tagihan PascaBayar Handphone Satelindo

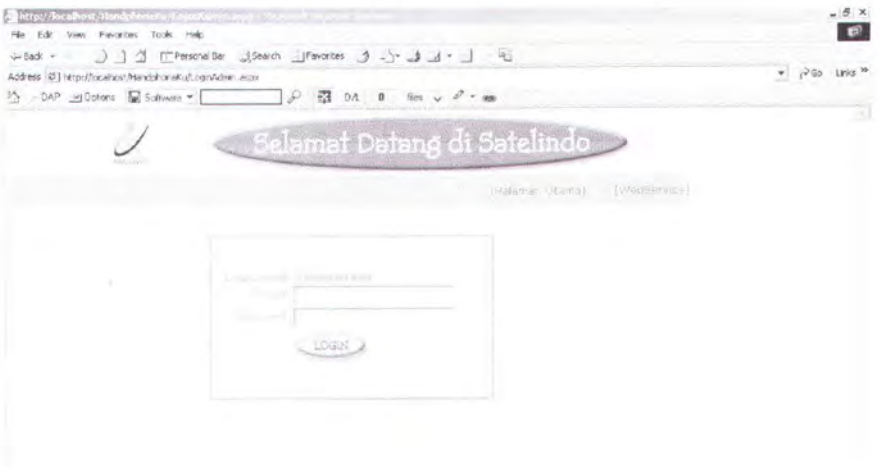
  FrmLihatTagihan.aspx menampilkan jumlah tagihan pelanggan.



Gambar 4.87 : FrmLihatTagihan.aspx

- Halaman Login Untuk Administrator Merchant Satelindo

  LoginAdmin.aspx adalah halaman untuk login bagi administrator merchant.

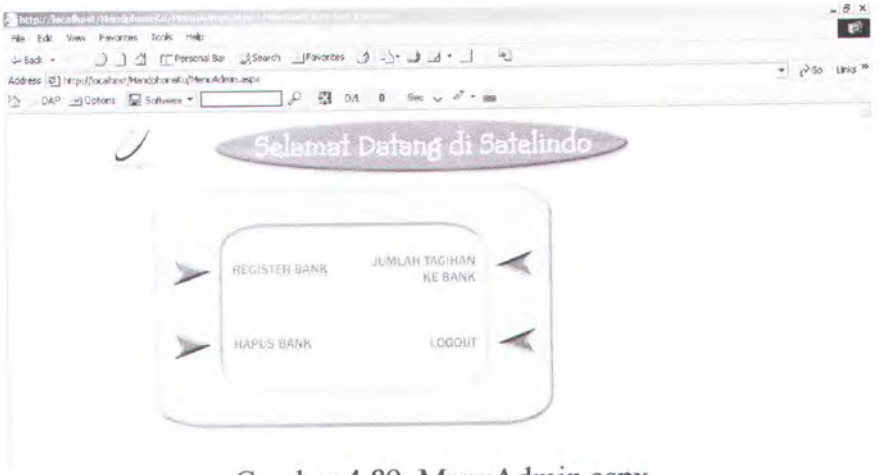


Gambar 4.88 : LoginAdmin.aspx

- Halaman Menu Untuk Administrator Merchant Satelindo

  MenuAdmin.aspx adalah halaman yang menampilkan menu untuk administrator merchant. Menu layanan yang ada adalah :

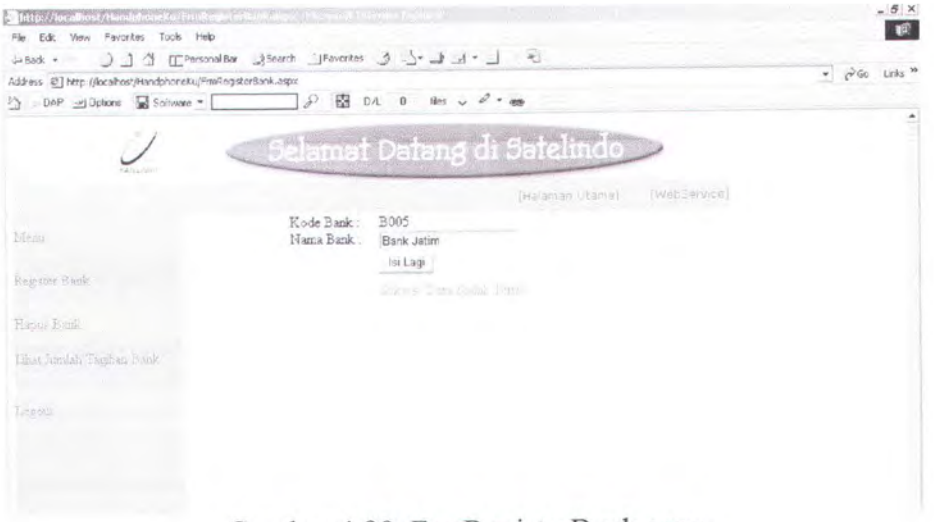
  1. register bank,
  2. hapus bank,
  3. jumlah tagihan ke bank,
  4. logout.



Gambar 4.89: MenuAdmin.aspx

- Halaman Register Bank

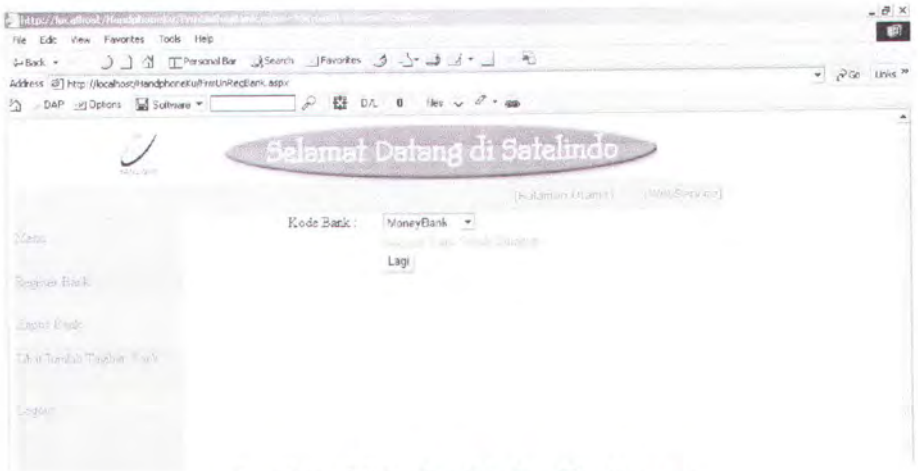
  FrmRegisterBank adalah halaman untuk registrasi bank yang dilakukan oleh admininstrator merchant.



Gambar 4.90: FrmRegisterBank.aspx

- Halaman Hapus Bank

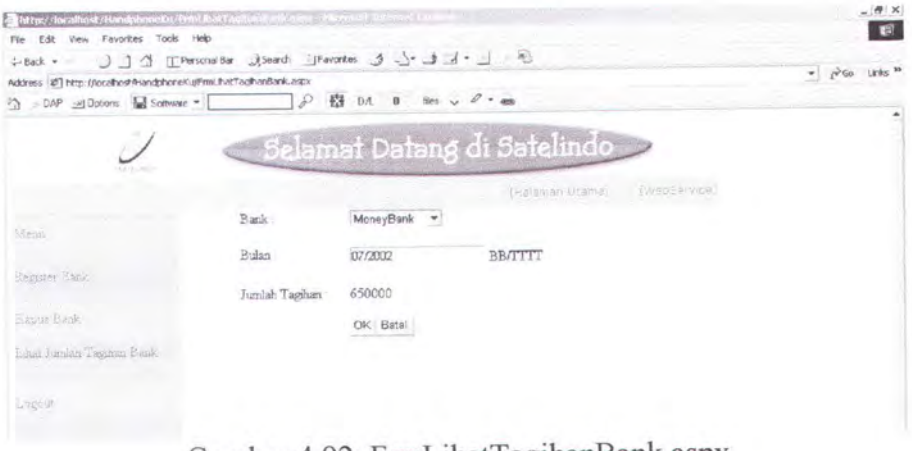
FrmUnRegBank.aspx adalah halaman untuk menghapus bank dari tabel bank yang dilakukan oleh administrator merchant.



Gambar 4.91 : FrmUnRegBank.aspx

- Halaman Lihat Jumlah Tagihan Merchant Ke Bank

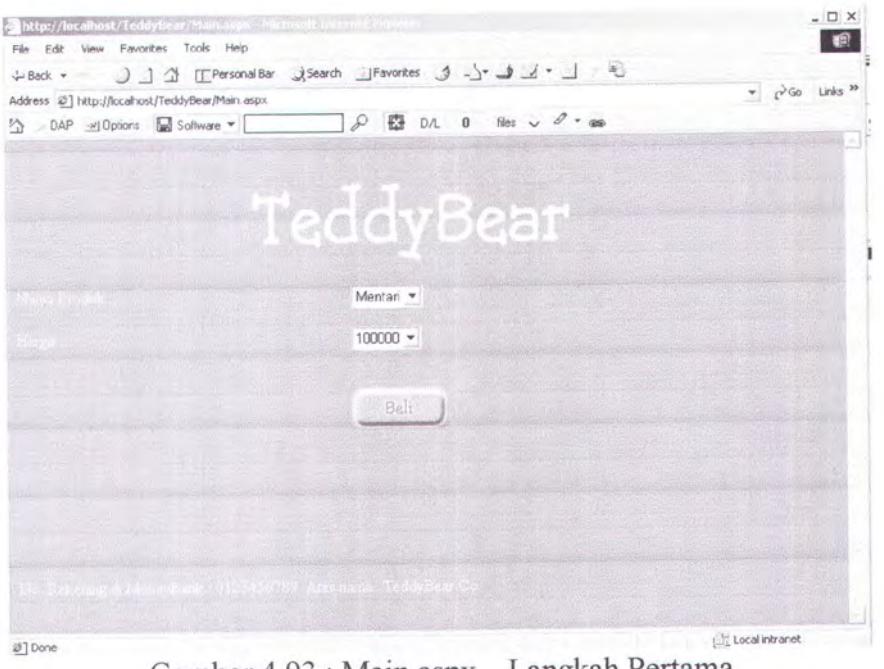
FrmLihatTagihanBank.aspx adalah halaman untuk melihat jumlah tagihan merchant kepada bank untuk transaksi yang dilakukan melalui bank.



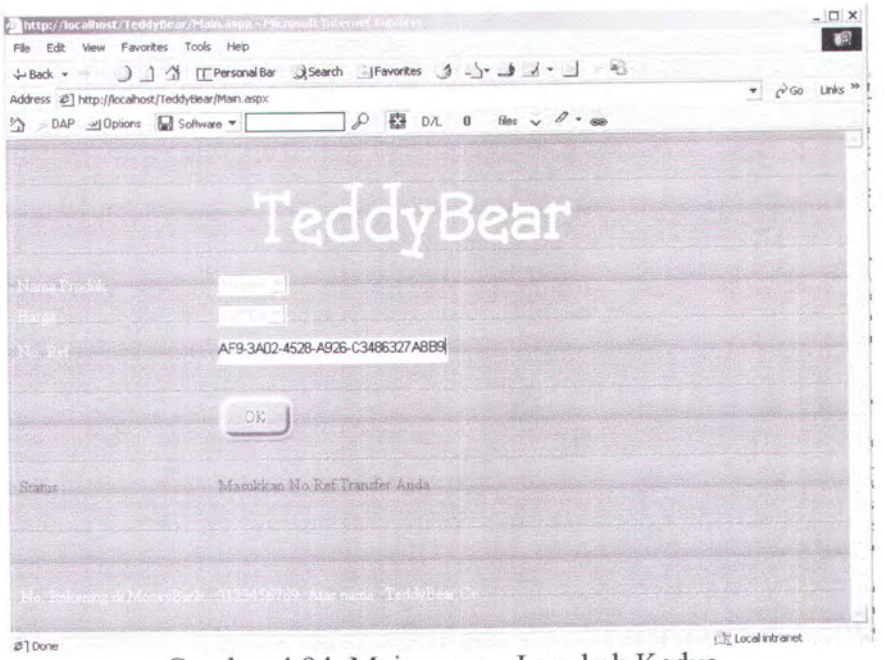
Gambar 4.92: FrmLihatTagihanBank.aspx

E-Commerce - TeddyBear

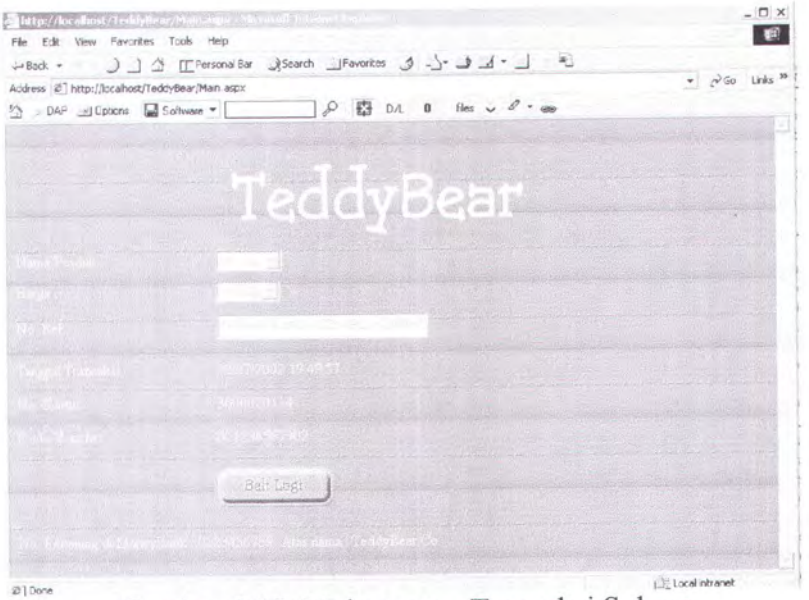
Aplikasi yang ada pada TeddyBear hanya satu, yaitu Main.aspx. Pada halaman ini pembeli bisa memilih jenis kartu dan melakukan transaksi pembelian.



Gambar 4.93 : Main.aspx – Langkah Pertama



Gambar 4.94: Main.aspx – Langkah Kedua



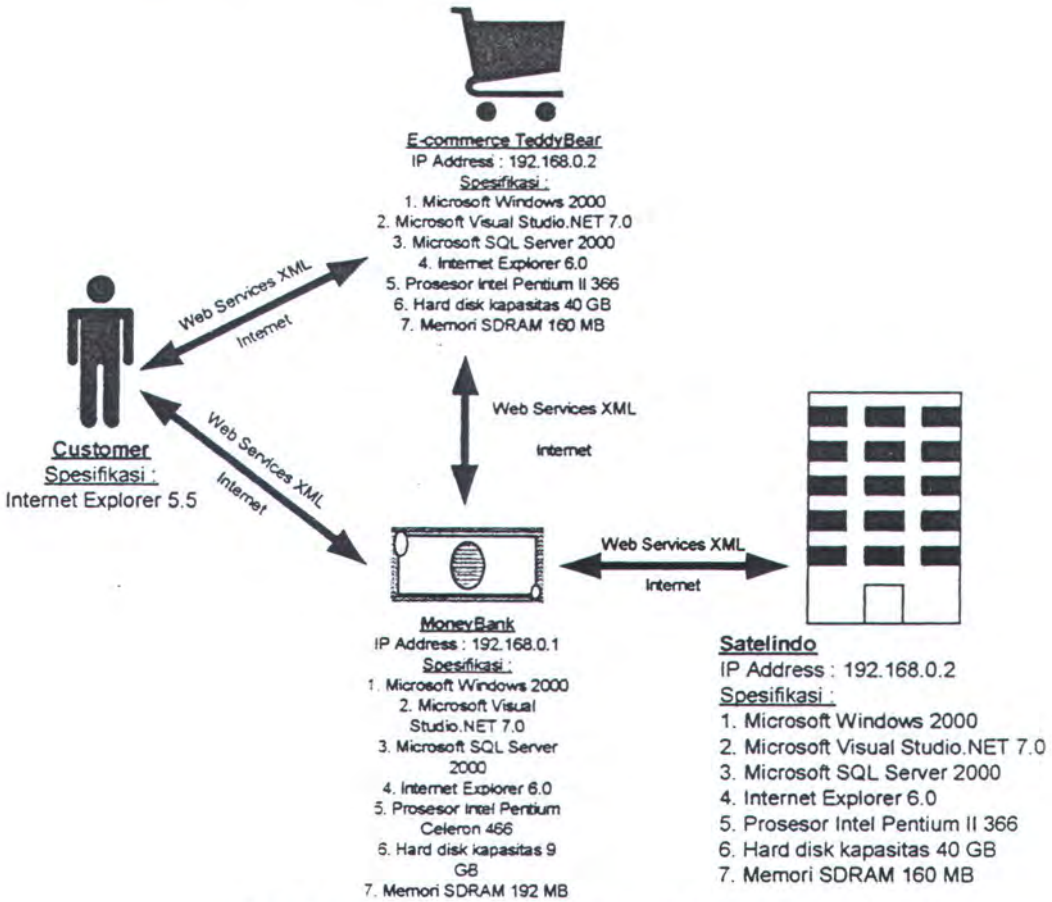
Gambar 4.95: Main.aspx – Transaksi Sukses

BAB V

UJI COBA DAN EVALUASI

BAB V
UJICOBA DAN EVALUASI

5.1 Lingkungan Uji Coba



Gambar 5.1 Diagram Konfigurasi Uji Coba

Untuk tugas akhir ini digunakan tiga sistem. Satu sistem utama, dua sistem pendukung uji coba ini. Sistem utama adalah sistem bank sedangkan sistem cellular operator, Satelindo, dan sistem e-commerce, TeddyBear, adalah sistem pendukung.

1. Perangkat keras :
    - Sistem bank :
        - Prosesor Intel Pentium Celeron 466
        - Hard disk kapasitas 9 GB
        - Memori SDRAM 192 MB
        - IP Address : 192.168.0.1
    - Sistem cellular operator dan e-commerce :
        - Prosesor Intel Pentium II 366
        - Hard disk kapasitas 40 GB
        - Memori SDRAM 160 MB
        - IP Address : 192.168.0.2
2. Sistem operasi dan perangkat lunak :
    - Windows 2000 Advanced Server

- Microsoft Visual Studio.NET 7.0
- Microsoft SQL Server 2000
- Internet Explorer versi 6.0

5.2 Skenario

Uji coba ini akan menguji aspek :

1. Functionality
2. Security
3. Error Handling

5.3 Pelaksanaan Skenario

5.3.1 Functionality

5.3.1.1 Uji Coba Login Nasabah

Untuk bisa melakukan transaksi pada website bank ini, nasabah harus melakukan login terlebih dahulu pada halaman utama, yaitu Main.apsx. Login untuk nasabah adalah nomer rekening tabungannya. Bila proses login sukses maka nasabah akan masuk ke halaman menu, yaitu Menu.aspx dan nasabah bisa

mengakses layanan perbankan yang disediakan bank lewat internet.

5.3.1.2 Uji Coba Pembelian Voucher Handphone

Nasabah dapat membeli voucher handphone dari merchant Satelindo melalui website bank. Pembayaran dengan melakukan debet pada rekening nasabah sejumlah harga voucher.

Sebelum melakukan transaksi ini, nasabah harus melakukan login dulu pada halaman utama, Main.aspx. Dan selanjutnya memilih menu Pembelian Voucher HP pada halaman Menu.aspx.

Pada menu PembelianVoucher.aspx, nasabah memilih merchant dan jumlah/harga voucher. Ada dua pilihan jumlah/harga voucher, yaitu 100.000 dan 150.000. Bila sukses maka saldo nasabah akan di-debet dan nasabah mendapatkan kode voucher dan nomer kartu dan jumlah saldo rekening didebet sebanyak harga voucher.

Uji coba dilakukan dengan melakukan pembelian voucher. Hasil uji coba memperlihatkan proses pembelian berhasil. Nasabah mendapatkan nomer kartu dan kode voucher dan setelah diperiksa jumlah saldo rekening nasabah berkurang sesuai dengan harga voucher. Selain itu jumlah utang bank ke merchant pun, setelah diperiksa pada sistem bank dan merchant, jumlahnya bertambah sesuai

dengan jumlah tagihan.

5.3.1.3 Uji Coba Pembayaran Tagihan Bulanan HP di Satelindo

Nasabah dapat membayar tagihan bulanan handphonenya di Satelindo melalui MoneyBank. Bila tagihan belum dibayar dan saldo nasabah cukup maka akan dilakukan proses pendebetan saldo rekening nasabah dan pembayaran tagihan sukses.

5.3.1.3.1 Lihat Jumlah Tagihan

Mula-mula akan dilihat jumlah tagihan, dengan memasukkan nomer pelanggan dan bulan bayar pada halaman LihatTagihanHP.aspx. Bila tagihan belum terbayar dan nasabah ingin membayarnya maka nasabah harus men-klik hyperlink Bayar pada bagian bawah halaman LihatTagihanHP.aspx. Uji coba dilakukan dengan memasukkan nomer pelanggan yang berupa nomer handphone dan bulan bayar. Karena tagihan belum dibayar, maka dilanjutkan dengan membayarnya.

5.3.1.3.2 Membayar

Bila nasabah akan membayar setelah masuk ke menu PembayaranHP.aspx maka nasabah men-klik *button* Bayar. Bila sukses maka tagihan terbayar dan

saldo rekening nasabah didebet sejumlah tagihan. Uji coba dilakukan dengan membayar tagihan yang sudah diperiksa belum terbayar. Uji coba berhasil dilakukan dengan sukses dan setelah saldo rekening nasabah diperiksa, saldonya sudah berkurang sesuai dengan jumlah tagihan. Jumlah utang bank ke merchant pun, setelah diperiksa pada sistem bank dan merchant, jumlahnya bertambah sesuai dengan jumlah tagihan.

5.3.1.4 Uji Coba Pembelian Voucher di E-Commerce TeddyBear dengan Membayar Melalui Transfer Ke Rekening E-Commerce di MoneyBank

Pembeli dapat membayar transaksi pembelian e-commerce-nya dengan mentransfer sejumlah transaksinya ke rekening e-commerce, TeddyBear. Bila transfer sukses, saldo rekening pembeli didebet dan saldo rekening penjual dikredit sejumlah dana yang ditransfer dan pembeli mendapatkan nomer referensi. Setelah mendapatkan nomer referensinya, nomer referensi ini dimasukkan ke website TeddyBear. Nomer referensi ini kemudian divalidasi oleh sistem TeddyBear ke sistem MoneyBank dengan mengirimkan nomer referensi ini, jumlah dana yang harus ditransfer oleh pembeli dan nomer rekening TeddyBear di MoneyBank. Bila validasi transfer ini sukses maka pembeli akan mendapatkan nomer kartu dan kode voucher handphone dari website TeddyBear.

Uji coba dilakukan dengan melakukan transfer ke rekening penjual dan memasukkan nomer referensi yang didapat ke website TeddyBear. Karena proses validasi transfer sukses, maka didapatkan nomer kartu dan kode voucher. Setelah melakukan transfer, saldo rekening pembeli dan penjual dicek, dan saldo pembeli sudah berkurang sesuai dengan jumlah transfer demikian pula dengan saldo rekening penjual yang bertambah sesuai dengan jumlah transfer.

5.3.1.5 Uji Coba Melihat Saldo Rekening

Untuk melihat saldo rekeningnya pada saat itu, nasabah harus memilih Informasi Saldo Rekening pada menu yang ada. Maka akan ditampilkan data saldo rekening dan nomer rekening serta nama pemilik rekening. Uji coba juga dilakukan setelah dilakukan transaksi pembayaran, pembelian, dan transfer.

5.3.1.6 Uji Coba Ganti PIN

Untuk mengganti pin yang lama dengan yang baru, nasabah harus memasukkan pin lama dan pin baru dua kali pada halaman GantiPIN.aspx. Bila pin yang lama yang dimasukkan benar dan pin baru yang dimasukkan pertama kali sama dengan pin baru yang dimasukkan kedua kali, maka proses pergantian

pin sukses dan muncul keterangan bahwa pin sudah diganti. Uji coba ganti pin ini dilakukan dengan mencoba mengganti pin yang lama dengan pin yang baru. Hasil uji coba memperlihatkan bahwa pin lama terganti dengan pin yang baru ketika kondisi persyaratan, yaitu pin lama sesuai dan pin baru yang dimasukkan dua kali harus sama, terpenuhi.

5.3.1.7 Uji Coba Login Administrator Bank

Untuk dapat mengakses menu khusus administrator bank, administrator bank harus login terlebih dahulu pada halaman Main.aspx. Bila sukses maka administrator akan masuk ke halaman MenuAdmin.aspx yang menampilkan menu-menu untuk administrator. Dari halaman ini administator bisa memilih jenis menu yang akan dilakukannya.

5.3.1.8 Uji Coba Meregister Merchant

Untuk meregister merchant, administrator memasukkan nama merchant ke halaman AddNewMerchant.aspx, kemudian men-klik button OK. Maka administrator akan mendapatkan kode merchant dan nama merchant tersimpan ke database bank. Hasil uji coba yang melakukan registrasi merchant, memperlihatkan setelah dilakukan registrasi, data merchant tersebut langsung

tersimpan ke database bank.

5.3.1.9 Uji Coba Menghapus Merchant

Untuk menghapus merchant dari database bank, administrator memilih nama merchant yang datanya akan dihapus. Selanjutnya administrator men-klik button OK. Bila penghapusan data sukses, maka akan muncul keterangan sukses yang menyatakan data sudah terhapus.

5.3.1.10 Uji Coba Melihat Jumlah Utang Bank Kepada Merchant Oleh Administrator Bank

Untuk melihat jumlah utang bank kepada merchant dari hasil transaksi pembayaran dan atau pembelian dari merchant yang dilakukan melalui bank, administrator harus memilih nama merchant dan memasukkan bulan bayar, kemudian men-klik button OK pada halaman JmlTagihanMerch.aspx. Bila pada bulan tersebut ada tagihan maka akan diperlihatkan jumlah tagihan. Jumlah tagihan yang diperhitungkan oleh sistem bank harus sama dengan hasil perhitungan pada sistem merchant. Hasil uji coba memperlihatkan bahwa hal ini sudah berjalan dengan baik, dimana jumlah tagihan yang ditampilkan pada aplikasi merchant sama dengan jumlah tagihan yang ditampilkan pada aplikasi

bank.

5.3.1.11 Uji Coba Login Administrator Merchant

Untuk dapat mengakses menu khusus administrator merchant, administrator bank harus login terlebih dahulu pada halaman LoginAdmin.aspx. Halaman ini bisa dibuka dari halaman Main.aspx dengan men-klik hyperlink Admin pada pojok kiri bawah halaman. Bila proses login sukses maka administrator akan masuk ke halaman MenuAdmin.aspx yang menampilkan menu-menu untuk administrator. Dari halaman ini administator bisa memilih jenis menu yang akan dilakukannya.

5.3.1.12 Uji Coba Meregister Bank

Untuk meregister bank, administrator memasukkan nama bank ke *text box* pada FrmRegisterBank.aspx. Lalu men-klik button OK. Bila proses register sukses, maka akan muncul keterangan pengisian data sukses. Bila admin mau mengisi data lagi, admin bisa men-klik button Isi Lagi. Uji coba dilakukan dengan melakukan registrasi bank ke aplikasi ini, dan uji coba mendapatkan hasil baik dimana data bank tersimpan di database merchant setelah diregistrasi oleh administrator merchant.

5.3.1.13 Uji Coba Menghapus Bank

Administrator merchant bisa menghapus data bank yang ada pada database merchant. Pertama-tama administrator memilih nama bank yang akan dihapus pada *combo box* pada halaman FrmUnRegBank.aspx. Kemudian administrator men-klik button OK. Bila proses sukses, maka akan muncul keterangan sukses. Untuk menghapus lagi data yang lain, administrator bisa men-klik button Lagi.

5.3.1.14 Uji Coba Melihat Jumlah Utang Bank Kepada Merchant Oleh Administrator Merchant

Untuk melihat jumlah utang bank kepada merchant dari hasil transaksi pembayaran tagihan merchant dan atau pembelian voucher handphone milik merchant melalui bank, administrator merchant harus memilih nama bank dan memasukkan bulan bayar pada FrmLihatTagihanBank.aspx. Setelah itu, ia men-klik button OK. Bila pada bulan itu ada transaksi pembayaran dan atau pembelian melalui bank tersebut, maka akan ditampilkan jumlahnya. Hasil pada sistem bank sama dengan hasil pada sistem merchant. Uji coba dilakukan dengan melihat jumlah tagihan di sistem bank dan di sistem merchant. Hasil yang didapat dari keduanya adalah sama.

5.3.1.15 Uji Coba Melihat Tagihan Bulanan HP Melalui WebSite Satelindo

Pelanggan kartu pasca bayar Satelindo bisa melihat jumlah tagihannya pada website Satelindo. Untuk melihatnya pelanggan harus memasukkan nomer nomer telpon dan bulan bayar pada halaman Main.aspx pada website Satelindo. Lalu pelanggan men-klik button OK. Maka pelanggan bisa melihat jumlah tagihan dan status pembayarannya pada FrmLihatTagihan.aspx. Uji coba dilakukan dengan melakukan transaksi pembayaran untuk tagihan handphone pasca bayar Satelindo. Transaksi pembayaran ini berlangsung dengan sukses.

5.3.2 Security

Uji Coba Pembelian Voucher di E-Commerce TeddyBear

Untuk uji coba ini dicoba memasukkan nomer referensi ke website TeddyBear secara trial and error. Namun karena nomer referensi ini tidak ada maka validasi selalu gagal, karena nomer referensi yang ada digenerate secara random dan terdiri dari 32 digit yang terdiri dari huruf dan angka, sehingga memiliki kombinasi yang banyak. Selain itu dicoba juga memasukkan nomer referensi yang sudah divalidasi, namun validasi ini juga gagal, karena nomer

referensi sudah diset pernah dicek dan tidak bisa digunakan lagi.

5.3.3 Error Handling

5.3.1.1 Uji Coba Login Nasabah

Ketika nasabah memasukkan login/nomer rekening yang tidak terdapat pada database bank atau nasabah memasukkan PIN yang salah, maka akan muncul pesan kesalahan : "Gagal!NoAccount/PIN salah".

5.3.1.2 Uji Coba Pembelian Voucher Handphone

Ketika nasabah berusaha membeli voucher namun jumlah saldo pada rekeningnya tidak cukup, maka akan muncul pesan kesalahan : "Saldo Tidak Cukup".

5.3.1.3 Uji Coba Pembayaran Tagihan Bulanan HP di Satelindo

5.3.1.3.1 Lihat Jumlah Tagihan

Ketika nasabah memasukkan bulan bayar yang belum ada data tagihannya, maka akan muncul pesan kesalahan : "Tagihan Bulan Ini Belum Ada". Sedangkan ketika nasabah memasukkan nomer pelanggan yang tidak ada pada database

Satelindo maka akan muncul pesan kesalahan : "Data Customer Tidak Ditemukan".

5.3.1.3.2 Membayar

Ketika nasabah berusaha membayar namun jumlah saldo pada rekeningnya tidak cukup, maka akan muncul pesan kesalahan : "Saldo Tidak Cukup".

5.3.1.4 Uji Coba Pembelian Voucher di E-Commerce TeddyBear dengan Membayar Melalui Transfer Ke Rekening E-Commerce di MoneyBank

Ketika nasabah berusaha membeli voucher namun salah memasukkan nomer referensi atau jumlah dana yang dikirimkan tidak sesuai sehingga ketika dicek ke webservice bank proses validasi gagal, maka akan muncul pesan kesalahan : "Transfer Validate Gagal".

5.3.1.5 Uji Coba Ganti PIN

Ketika nasabah memasukkan PIN lama yang salah, maka akan muncul pesan kesalahan : "Gagal!NoAccount/PIN salah". Sedangkan ketika nasabah memasukkan data PIN baru yang pertama tidak sama dengan PIN baru yang kedua, maka akan muncul pesan kesalahan : "Gagal! PIN Baru".

5.3.1.6 Uji Coba Login Administrator Bank

Ketika administrator bank memasukkan login/nomer rekening atau PIN yang salah, maka akan muncul pesan kesalahan : "Gagal!NoAccount/PIN salah".

5.3.1.7 Uji Coba Login Administrator Merchant

Ketika administrator merchant Satelindo memasukkan login/nomer rekening atau PIN yang salah, maka akan muncul pesan kesalahan : "Gagal!NoAccount/PIN salah".

5.3.1.8 Uji Coba Melihat Tagihan Bulanan HP Melalui WebSite Satelindo

Ketika nasabah memasukkan bulan bayar yang belum ada data tagihannya, maka akan muncul pesan kesalahan : "Tagihan Bulan Ini Belum Ada". Sedangkan ketika nasabah memasukkan nomer pelanggan yang tidak ada pada database Satelindo maka akan muncul pesan kesalahan : "Data Customer Tidak Ditemukan".

5.4 Evaluasi

Berbagai uji coba pada *multiple server environtment* memperlihatkan bahwa proses pembayaran dan pembelian serta pembelian dengan menggunakan nomer referensi dari hasil transfer untuk divalidasi sudah berjalan dengan baik. Namun karena nomer referensi tidak melihat nomer rekening pengirim, maka ada kemungkinan nomer referensi yang dimasukkan salah tapi karena jumlah dan nomer rekening tujuannya benar serta statusnya belum dicek, maka validasi transfer sukses. Walaupun hal ini kemungkinan terjadinya kecil sekali karena nomer referensi merupakan kombinasi random dari tiga puluh dua huruf dan angka.

BAB VI

KESIMPULAN DAN SARAN

BAB VI
KESIMPULAN DAN SARAN

Bab terakhir ini menjelaskan kesimpulan yang didapat dari pelaksanaan tugas akhir ini dan saran-saran yang perlu diperhatikan untuk pengembangan perangkat lunak selanjutnya.

6.1 Kesimpulan

Kesimpulan dari keseluruhan pembuatan tugas akhir ini adalah :

1. Dengan menggunakan nomer referensi sebagai bukti pembayaran transfer dengan memasukkannya ke website merchant maka customer dapat membayar secara on-line tanpa perlu meninggalkan data credit card di merchant. Dan nomer referensi ini tidak bisa digunakan lebih dari sekali untuk mencegah digunakannya nomer referensi yang sama untuk dua transaksi pembayaran yang berbeda.
2. Sistem yang dibuat mampu membuat merchant tahu dengan cepat dan yakin tentang status pembayaran cutomernya dengan cara sistem merchant memvalidasi transfer dengan mengirimkan nomer referensi, yang diinputkan customer, jumlah kiriman seharusnya ke rekeningnya, dan nomer rekeningnya ke sistem bank. Bila transfer dengan nomer referensi tersebut benar ada untuk

nomer rekeningnya dan jumlahnya benar serta belum pernah digunakan untuk bukti pembayaran yang lain, maka validasi transfer sukses dan merchant bisa memberikan barang ke customer.

3. Sistem bank yang dibuat mampu menyediakan layanan perbankan khususnya pembayaran berbasis internet ke nasabah dengan menyediakan layanan pembayaran tagihan handphone, layanan pembelian voucher handphone, layanan validasi transfer, informasi saldo, dan ganti pin.
4. Sistem bank dan sistem merchant yang dibuat, terintegrasi secara on-line dengan menggunakan webservice dengan menggunakan standard-standard protocol yang terbuka, yaitu SOAP dan XML.
5. Otorisasi pembayaran antara bank dengan merchant dapat dilakukan secara langsung dengan menggunakan webservice.

6.2 Saran

Saran-saran untuk pengembangan perangkat lunak ini lebih lanjut :

1. Data yang dikirim di-encrpyt terlebih dahulu.
2. Memasang firewall untuk membatasi akses ke sistem.
3. Pemberitahuan transaksi lewat sms ke handphone nasabah langsung setelah

transaksi dilakukan.

4. Transaksi pembayaran, pembelian voucher, transfer, ganti pin, dan informasi saldo bisa dilakukan melalui handphone nasabah.
5. Pemberitahuan transaksi ke e-mail nasabah langsung setelah transaksi dilakukan oleh sistem bank.
6. Setiap bulan nasabah mendapatkan rekening koran yang dikirim ke e-mailnya

DAFTAR PUSTAKA

DAFTAR PUSTAKA

[1]. "Visual Basic and Visual C# Concepts : XML Technology Backgrounder". In MSDN Library Visual Studio.NET release, Beta 2, Microsoft Corporation. 2001.

[2]. "C# Programmer's Reference : XML Documentation". In MSDN Library Visual Studio.NET release, Beta 2, Microsoft Corporation. 2001.

[3]. ".NET Framework Developer's Guide : Web Services Overview". In MSDN Library Visual Studio.NET release, Beta 2.

[4]. ".NET Framework Developer's Guide : Web Service Scenarios". In MSDN Library Visual Studio.NET release, Beta 2.

[5]. ".NET Framework Developer's Guide : Simple Services". In MSDN Library Visual Studio.NET release, Beta 2.

[6]. ".NET Framework Developer's Guide : Application Integration". In MSDN Library Visual Studio.NET release, Beta 2.

[7]. ".NET Framework Developer's Guide : Workflow Solutions". In MSDN Library Visual Studio.NET release, Beta 2.

[8]. ".NET Framework Developer's Guide : Web Services Infrastructure". In

MSDN Library Visual Studio.NET release, Beta 2.

[9]. ".NET Framework Developer's Guide : Web Services Description". In MSDN Library Visual Studio.NET release, Beta 2.

[10]. ".NET Framework Developer's Guide : Web Services Wire Formats". In MSDN Library Visual Studio.NET release, Beta 2.

[11]. Christoper Laur. ".NET Overview". www.techmetrix.com, 5 Maret 2001.

LAMPIRAN

LAMPIRAN

STORED PROCEDURE UNTUK DATABASE BANK

```
CREATE  PROCEDURE [dbo].[SP_Cust_Login]
        (@NoAcc char(10),
         @PIN           char(6),
         @ReplyStatus   int     OUTPUT,
         @ReplyMsg      varchar(255)    OUTPUT
        )

AS

        Set @ReplyStatus=0
        Set @ReplyMsg='Sukses'

        if not exists (Select NoAcc From Account Where NoAcc=@NoAcc and PIN=@PIN)
        begin
                Set @ReplyStatus=1
                Set @ReplyMsg='Gagal!NoAccount/PIN salah'
                return
        end
GO

--Cek Apakah Saldo mencukupi
CREATE PROCEDURE [dbo].[SP_Cek_Saldo_Cukup]
        (@NoAcc char(10),
         @Jumlah money,
         @ReplyStatus int output,
         @ReplyMsg varchar(255) output
        )

 AS
        Set @ReplyStatus=0
        Set @ReplyMsg='Sukses'

        if (@Jumlah>(Select Saldo From Account Where NoAcc=@NoAcc))
        begin
                set @ReplyStatus=3
                set @ReplyMsg='Saldo Tidak Cukup'
        end

 GO

 CREATE PROCEDURE [dbo].[SP_Cek_Saldo]
        (@NoAcc char(10),
         @PIN char(6),
         @Nama varchar (25) OUTPUT,
         @Saldo money OUTPUT,
         @ReplyStatus int OUTPUT,
         @ReplyMessage varchar(255) OUTPUT
```

```
        )
AS

        exec        dbo.SP_Cust_Login        @NoAcc,@Pin,@ReplyStatus=@ReplyStatus
output,@ReplyMsg=@ReplyMessage output

        if (@ReplyStatus<>0)
                return
        else
        begin
                Select @Nama=Nama,@Saldo=Saldo
                from Account
                where NoAcc=@NoAcc and PIN=@PIN
        end
GO



CREATE PROCEDURE SP_CreateKodeMerch
        (@KodeMerchants        char(4) output)
 AS
declare @Max char(4), @tmp char(3), @tmpmax int

        Select @Max=Max(KodeMerchants) From Merchants

        set @tmp = right(@max,3)
        set @tmpmax=(cast(@tmp as int))+1

        if (@tmpMax<10)
        begin
                set @KodeMerchants='S00' + (cast (@tmpMax as char(1)))
        end
        else if (@tmpMax<100)
        begin
                set @KodeMerchants='S0' + (cast (@tmpMax as char(2)))
        end
        else
        begin
                set @KodeMerchants='S' + (cast (@tmpMax as char(3)))
        end


GO


CREATE PROCEDURE [dbo].[SP_Ganti_PIN] (
        @NoAcc char(10),
        @Pin char(6),
        @PinBaru char(6),
         @REplyStatus  int     output,
         @ReplyMsg     varchar(255)   output
        )
AS

        set @ReplyStatus=0
        set @ReplyMsg='Sukses'

        exec           sp_cust_login           @NoAcc,@pin,@ReplyStatus=@REplyStatus
output,@ReplyMsg=@replymsg output
        if (@ReplyStatus=0)
```

```
        begin
                UPDATE Account
                SET PIN=@PINBARU
                WHERE NoAcc=@NoAcc and PIN=@PIN
        end
GO

--Procedure utk Menyimpan No.Kartu-->NoPelanggan di Tabel
CREATE    PROCEDURE [dbo].[SP_Insert_NoKartu]
        (@NoKartu       char(12),
         @NoRef char(36),
         @ReplyStatus  int     OUTPUT,
         @ReplyMsg     VARCHAR(255)    OUTPUT
        )

AS
        set @ReplyStatus=0
        set @ReplyMsg='Sukses'

        if exists(Select NoRef From Transaksi Where NoRef = @NoRef)
                Update Transaksi
                Set NoPelanggan  = @NoKartu
                Where NoRef = @NoRef
        else
        begin
                set @ReplyStatus = 11
                set @ReplyMsg = 'No. Ref yang anda masukkan salah! Transaksi Gagal'
        end
GO


CREATE PROCEDURE SP_Insert_Transaksi_Membayar
        (@NoAcc char(10),
         @Jumlah        money,
         @KodeMerchants char(4),
         @KodeObjTrans int,
         @NoPelanggan          char(12),
         @BulanBayar    char(7),
         @Status int output,
         @Msg   varchar(255) output
        )
AS
--Declare @KodeTransaksi char(3)

        --Cek Kode Transaksi
/*      exec dbo.SP_Cek_Kode_Transaksi
                @KodeTransaksi,@ReplyStatus=@Status output,@ReplyMsg=@Msg output
        */
        --Cek NoAcc
        if not exists(select NoAcc From Account Where NoAcc=@NoAcc)
        begin
                set @Status=1
                set @Msg='NoAcc salah'
                return
        end
        --Cek Saldo
        execute dbo.SP_Cek_Saldo_Cukup @NoAcc,@Jumlah,
                @ReplyStatus=@Status output,@REplyMsg=@Msg output

        if (@Status<>0)
        begin
```

```
            return
        end
        --Status oke
        else
        begin
                        --Cek Kode Merchants
                        if not exists (select KodeMerchants from Merchants   where
KodeMerchants=@KodeMerchants)
                                begin
                                        set @Status=5
                                        set @Msg='Kode Merchants Tidak Ada'
                                        return
                                end
                        --Cek KodeObjTrans
                        if     not     exists     (select     KodeObjTrans      from
ObjectTransPEmbayaran where KodeObjTrans=@KodeObjTrans)
                        begin
                                set @Status=6
                                set @Msg='KodeObjTrans Tidak Ada'
                                return
                        end

                        if (@Status<>0)
                        begin
                                return
--                              raiserror(@Msg,16,1)
                --              rollback transaction
                        end

                        --Kurangi Saldo Asal
                        Update Account
                        Set Saldo=Saldo-@Jumlah
                        Where NoAcc=@NoAcc --and PIN=@PIN


--                      Insert Transaksi
                        Insert
Transaksi(NoAcc,KodeTransaksi,NoPelanggan,BulanBayar,KodeMerchants,KodeObjTrans,Ju
mlah)
                        Values
(@NoAcc,'PBR',@NoPelanggan,@Bulanbayar,@KodeMErchants,@KodeObjTrans,@Jumlah)



        end
GO


CREATE PROCEDURE SP_Insert_Transaksi_Membeli
        (@NoAcc char(10),
         @Jumlah        money,
         @KodeMerchants char(4),
         @KodeObjTrans int,
         @Status int output,
         @Msg   varchar(255) output
        )
AS
        --Cek NoAcc
```

```
if not exists(select NoAcc From Account Where NoAcc=@NoAcc)
	begin
		set @Status=1
		set @Msg='NoAcc salah'
		return
	end
	--Cek Saldo cukup
	execute dbo.SP_Cek_Saldo_Cukup @NoAcc,@Jumlah,
		@ReplyStatus=@Status output,@REplyMsg=@Msg output

	if (@Status<>0)
	begin
		return
	end
	--Status oke
	else
	begin
		--PEMBAYARAN
			--Cek Kode Merchants
			if not exists (select KodeMerchants from Merchants where
KodeMerchants=@KodeMerchants)
			begin
				set @Status=5
				set @Msg='Kode Merchants Tidak Ada'
				return
			end
			--Cek KodeObjTrans
			if not exists (select KodeObjTrans from
ObjectTransPEmbayaran where KodeObjTrans=@KodeObjTrans)
			begin
				set @Status=6
				set @Msg='KodeObjTrans Tidak Ada'
				return
			end

			if (@Status<>0)
			begin
				return
	--			raiserror(@Msg,16,1)
		--			rollback transaction
			end

			--Kurangi Saldo Asal
			Update Account
			Set Saldo=Saldo-@Jumlah
			Where NoAcc=@NoAcc --and PIN=@PIN

--			Insert Transaksi
			Insert
Transaksi(NoAcc,KodeTransaksi,KodeMerchants,KodeObjTrans,Jumlah)
			Values (@NoAcc,'PBR',@KodeMErchants,@KodeObjTrans,@Jumlah)

	end
GO

CREATE PROCEDURE SP_Insert_Transaksi_Mentransfer
	(@NoAcc char(10),
	 @Jumlah	money,
	 @AccNoDest	char(10),
	 @Status int output,
	 @Msg	varchar(255) output
```

```
        )
AS
        --Cek NoAcc
        if not exists(select NoAcc From Account Where NoAcc=@NoAcc)
        begin
                set @Status=1
                set @Msg='NoAcc salah'
                return
        end

        if (@Status<>0)
        begin
                return
        end
        --Status oke
        else
        begin
        --TRANSFER

                if (@Status<>0)
                begin
                        return
                end

                --Kurangi Saldo Asal
                Update Account
                Set Saldo=Saldo-@Jumlah
                Where NoAcc=@NoAcc --and PIN=@PIN
                --Tambahkan Saldo Tujuan
                Update Account
                Set Saldo=Saldo+@Jumlah
                Where NoAcc=@NoAcc

--                      Insert Transaksi
                        Insert
Transaksi(NoAcc,KodeTransaksi,AccNoDest,Jumlah,Status)
                        Values (@NoAcc,'TRS',@AccNoDest,@Jumlah,'N')

        end
GO


CREATE   PROCEDURE [dbo].[SP_Jml_Tagihan_Merchants]
        (@KodeMerchants         char(4),
         @BulanBayar            char(7),
         @JmlTagihan            money OUTPUT,
         @ReplyStatus           int     OUTPUT,
         @ReplyMsg              varchar(255)    OUTPUT
        )
AS
Declare @TmpBln1        int,--char(2),
        @TmpBln2        char(2),
        @TmpThn1        int,--char(4),
        @TmpThn2        char(4)


        Set @ReplyStatus=0
        Set @ReplyMsg='Sukses'

        --Ambil Bln dan Thn dari BlnBayar
        Set @TmpBln1=Cast((LEFT(@BulanBayar,2))as int)
```

```
        Set @TmpThn1=Cast((RIGHT(@BulanBayar,4))as int)

        --Cek Apakah Pada BulanBayar itu ada Tagihan Merchants ke bank
        IF NOT EXISTS
        (Select KodeMerchants,BulanBayar From Transaksi
         Where
         (KodeMerchants=@KodeMerchants and
          (--BulanBayar=@BulanBayar
          --or
          ((month(cast(TglTransaksi as char))=@TmpBln1 )
               and
           (year(cast(TglTransaksi as char))=@TmpThn1)
          )--or
          )--bln
         )--kode
        )--select
        BEGIN
               Set @ReplyStatus=8
               Set @ReplyMsg='Tidak Ada Pembayaran Bln ini mll Bank utk Merchants
ini'
               return
        END
        ELSE
        BEGIN
               --TOTAL JML TAGIHAN
               Set @JmlTagihan=(Select Sum(Jumlah) From Transaksi
               Where
                (KodeMerchants=@KodeMerchants and
                 (--BulanBayar=@BulanBayar
                 --or
                 ((month(cast(TglTransaksi as char))=@TmpBln1 )
                      and
                  (year(cast(TglTransaksi as char))=@TmpThn1)
                 )--or
                 )--bln
                )--kode
               )--select
               --Where KodeMerchants=@KodeMerchants and BulanBayar=@BulanBayar)
--             Group By KodeMerchants
--             Order By KodeMerchants
        END

GO


--Procedure utk Memproses Pembayaran
CREATE     PROCEDURE [dbo].[SP_Membayar]
        (@NoAcc char(10),
         @PIN          char(6),
         @NoPelanggan char(12),
         @BulanBayar   char(7),
         @KodeMerchants char(4),
         @KodeObjTrans int,
         @Jumlah       money,
         @TglTrans     datetime output,
         @NoRef char(36) output,
         @ReplyStatus  int    OUTPUT,
         @ReplyMsg     VARCHAR(255)   OUTPUT
        )

AS
```

```
	--Cek Apakah No Acc dan PIN benar
	EXEC	dbo.SP_Cust_Login	@NoAcc,@Pin,@ReplyStatus=@ReplyStatus	output,
@ReplyMsg=@ReplyMsg output


	if @ReplyStatus=0
	begin
	--insert ke tabel transaksi
		exec SP_Insert_Transaksi_Membayar @NoAcc = @NoAcc,
				@Jumlah = @Jumlah,
				@KodeMerchants = @KodeMerchants,
				@KodeObjTrans = @KodeObjTrans,
				@NoPelanggan	= @NoPelanggan,
				@BulanBayar = @BulanBayar,
				@Status = @ReplyStatus output,
				@Msg	= @ReplyMsg output

--		Ambil tgltrans dan noref
		Select	@NoRef=NoRef,	@TglTrans=TglTransaksi	From	Transaksi	Where
NoTransaksi=@@IDENTITY
	end
	else
	begin
		return
	end
GO


--Procedure utk Memproses Pembelian
CREATE	PROCEDURE [dbo].[SP_Membeli]
	(@NoAcc char(10),
	 @PIN		char(6),
	 @KodeMerchants char(4),
	 @KodeObjTrans int,
	 @Jumlah	money,
	 @TglTrans	datetime output,
	 @NoRef char(36) output,
	 @ReplyStatus	int	OUTPUT,
	 @ReplyMsg	VARCHAR(255)	OUTPUT
	)

AS

	--Cek Apakah No Acc dan PIN benar
	EXEC	dbo.SP_Cust_Login	@NoAcc,@Pin,@ReplyStatus=@ReplyStatus	output,
@ReplyMsg=@ReplyMsg output

	if @ReplyStatus=0
	begin
	--Insert Transaksi Pembelian Voucher
		exec SP_Insert_Transaksi_Membeli @NoAcc = @NoAcc,
				@Jumlah = @Jumlah,
				@KodeMerchants = @KodeMerchants,
				@KodeObjTrans = @KodeObjTrans,
				@Status = @ReplyStatus output,
				@Msg	= @ReplyMsg output


--	Ambil noref dan tgltrans
```

```
            Select @NoRef=NoRef, @TglTrans=TglTransaksi From Transaksi Where
NoTransaksi=@@IDENTITY

        end
        else
        begin
                return
        end
GO


--Prosedur untuk memproses Transfer dari Customer
CREATE      PROCEDURE [dbo].[SP_MenTransfer]
        (@NoAcc char(10),
         @PIN          char(6),
         @AccNoDest    char(10),
         @Jumlah       money,
         @TglTrans     datetime       output,
         @NoRef char(36) output,
         @ReplyStatus  int    OUTPUT,
         @ReplyMsg     VARCHAR(255)   OUTPUT
        )
AS
Declare @tmpNoTransaksi char(6)

        --Cek Apakah No Acc dan PIN benar
        EXEC   dbo.SP_Cust_Login   @NoAcc,@Pin,@ReplyStatus=@ReplyStatus   output,
@ReplyMsg=@ReplyMsg output

        --Cek Apakah No Acc Tujuan Ada
        if not exists (select NoAcc From Account Where NoAcc=@AccNoDest)
        begin
                set @ReplyStatus=2
                set @ReplyMsg='No. Account Tujuan tidak Ada'
        end

        --Cek Apakah Saldo mencukupi
        exec  dbo.SP_Cek_Saldo_Cukup  @NoAcc,  @Jumlah,  @ReplyStatus=@ReplyStatus
output,@ReplyMsg=@ReplyMsg output

        if (@ReplyStatus=0)
        begin
                --Insert Transaksi
                exec SP_Insert_Transaksi_Mentransfer
                        @NoAcc = @NoAcc,
                        @Jumlah= @Jumlah,
                        @AccNoDest = @AccNoDest,
                        @Status = @ReplyStatus  output,
                        @Msg   = @ReplyMsg output

                Select  @NoRef=NoRef,  @TglTrans=TglTransaksi  From  Transaksi  Where
NoTransaksi=@@IDENTITY
        end
        else
        begin
                return
        end
GO


CREATE PROCEDURE [dbo].[SP_Register_Merchant]
```

```
        (@Nama varchar(25),
         @KodeMerchants        char(4) output,
         @ReplyStatus  int     output,
         @ReplyMsg varchar(255) output
        )
AS

        set @ReplyStatus=0
        set @ReplyMsg='Sukses'

        exec sp_createkodemerch @KodeMErchants=@KodeMerchants output

        INSERT INTO MERCHANTS
                (KODEMERCHANTS,NAMAMERCHANTS)
               VALUES
                (@KodeMerchants, @Nama)
        set @ReplyStatus=0
        set @ReplyMsg='Sukses'
GO


CREATE PROCEDURE [dbo].[SP_TransferValidate]
        (@AccNoDest    char(10),
         @NoRef char(36),
         @Jumlah       money ,
         @ReplyStatus  int OUTPUT,
         @ReplyMsg     varchar(255) OUTPUT
        )
AS
Declare @TglExp datetime, @TmpTgl datetime

        Set @ReplyStatus=0
        Set @ReplyMsg='Sukses'

        --Sudah no.ref kadaluarsa/blm
        Set @TglExp = (Select day(cast(TglTransaksi as char))+2 From Transaksi
Where NoRef = @NoRef and AccNoDest = @AccNoDest)
        Set @TmpTgl = getdate()

        IF (@TmpTgl>@TglExp)
        Begin
                Update Transaksi Set Status='C' Where AccNoDest=@AccNoDest and
NoRef=@NoRef
        End

        --Cek Apakah betul ada Transfer ke AccNoDest dgn NoRef tsb dan Apakah Sdh
Dicek/Blm
        IF    NOT    EXISTS(Select    AccNoDest,NoRef    From    Transaksi    Where
AccNoDest=@AccNoDest and NoRef=@NoRef and Status='N'
                        and Jumlah = @Jumlah)
        BEGIN
                Set @ReplyStatus=7
                Set @ReplyMsg='Transfer Validate Gagal'
                Return
        END
        ELSE
        BEGIN
                Update Transaksi Set Status='C' Where AccNoDest=@AccNoDest and
NoRef=@NoRef and Status='N'
        END
GO
```

```
CREATE PROCEDURE [dbo].[SP_UnRegister_Merchant]
        (@KodeMerchants         char(4),
         @ReplyStatus           int     output,
         @ReplyMsg      varchar(255)    output
        )
AS

        set @ReplyStatus=0
        set @ReplyMsg='Sukses'

        if not exists (select KodeMerchants from Merchants where
KodeMerchants=@KodeMerchants)
        begin
                set @ReplyStatus=5
                set @ReplyMsg='Kode Merchants Tidak Ada'
                return
        end
        if (@ReplyStatus=0)
        begin
                DELETE FROM Merchants
                        WHERE KODEMERCHANTS=@KodeMerchants
        end
GO
```

STORED PROCEDURE UNTUK DATABASE MERCHANT

```
-- cek persediaan voucher
CREATE PROCEDURE dbo.SP_Ambil_Voucher
       (@GoodName     varchar(15),
        @PriceperUnit money,
        @Status int output,
        @Msg varchar(255) output
       )
AS

       set @Status=0
       set @Msg='Sukses'

       if exists(Select GoodNo From Goods Where (Saldo=1 and GoodName=@GoodName
and PriceperUnit = @PriceperUnit))
       begin
              set @Status=0
              set @Msg='Sukses'
       end
       else
       begin
              set @Status=3
              set @Msg='Persediaan Kosong'
       end
GO

CREATE PROCEDURE SP_Bank_Paid
       (
        @CustNo       char(12),
        @BankCode     char(4),
        @Months       char(7),
        @Status       int    output,
        @Msg          varchar(255)   output
       )
 AS
declare @PAymentStatus char(1)

       set @PaymentStatus='C'
       set @Status=0
       set @Msg='Sukses'

       --Cek customer
       IF NOT EXISTS (Select CustNo From Customer Where CustNo=@CustNo)
       BEGIN
              Set @Status=1
              Set @Msg='Data Customer Tidak Ditemukan'
              return
       END

       --Cek TAgihan Bln Ini Ada.tdk
       if not exists (Select CustNo, Months From Invoice Where (CustNo=@CustNo)
 and (Months=@Months))
       begin
              set @STatus=3
              set @Msg='Tagihan Bln Ini Blm Ada'
       end
```

```
        if (@Status=0)
        begin
                --Update
                Update dbo.Invoice
                Set BankCode=@BankCode, PaymentStatus= @PaymentStatus
                Where (CustNo=@CustNo) and (Months=@Months)
        end
GO

--SUBSCRIBE
--BY USER/CUSTOMER
CREATE PROCEDURE dbo.SP_Cek_Tagihan_Cust
        (@CustNo        char(12),
         @Months        char(7),
         @CustName      varchar(20) output,
         @ClaimTotal    money   output,
         @PaymentStatus         char(1) output,
         @Status        int     output,
         @Msg           varchar(255)    output
        )
AS
        set @Status= 0
        set @Msg = 'Sukses'

        --Cek CustNo ->Ada/tdk
        IF NOT EXISTS (Select CustNo From Customer Where CustNo=@CustNo)
        BEGIN
                Set @Status=1
                Set @Msg='Data Customer Tidak Ditemukan'
                return
        END

        --Cek TAgihan Bln Ini Ada.tdk
        if not exists (Select CustNo, Months From Invoice Where (CustNo=@CustNo)
 and (Months=@Months))
        begin
                set @STatus=3
                set @Msg='Tagihan Bln Ini Blm Ada'
        end

        if (@Status=0)
        begin
                --Ambil nama customer
                Select @CustName=CustName
                        From Customer
                        Where CustNo=@CustNo
                --Ambil data Claimtotal dan PaymentStatus
                Select @ClaimTotal=ClaimTotal, @PaymentStatus= PaymentStatus
                From Invoice
                Where (CustNo= @CustNo) and (Months=@Months)
        end
 GO

 CREATE PROCEDURE dbo.SP_Create_BankCode
        (@BankCode      char(4) output)
 AS
 declare @Max char(4),@tmp char(3),@tmpmax int

 Select @Max=Max(BankCode) From Bank

 set @tmp= right(@max,3)
```

```
set @tmpMax= (cast (@tmp as int) ) + 1

if (@tmpMax<10)
begin
	set @BankCode='B00'+ (cast (@tmpMax as char(1)))
end
else if(@tmpMax<100)
begin
	set @BankCode='B0' + (cast (@tmpMax as char(2)))
end
else
begin
	set @BankCode='B' + (cast (@tmpMax as char(3)))
end

GO


CREATE PROCEDURE SP_Insert_Invoice_Voucher
	(@BankCode	char(4),
	 @RefNo char(36),
	 @GoodName	varchar(15),
	 @priceperunit money,
	 @TransDate	datetime	output,
	 @InvoiceNo	int	output,
	 @GoodNo	char(10) output,
	 @KodeVoucher char(12) output,
	 @Status	int output,
	 @Msg		varchar(255) output
	)
AS
--declare @priceperunit money

	set @Status=0
	set @Msg='Sukses'
	set @TransDate = GetDate()

	exec SP_Ambil_Voucher @GoodName=@GoodName , @PriceperUnit = @PriceperUnit
, @Status=@Status,@Msg=@Msg

	if (@Status=0)
	begin
		Set @GoodNo = (Select Min(GoodNo)
		From Goods
		Where ((Saldo = 1) and (GoodName = @GoodName) and Priceperunit =
@priceperunit)
		)


		Select  @KodeVoucher = KodeVoucher
		From Goods
		Where GoodNo = @GoodNo

		Update Goods
		Set Saldo = 0
		Where GoodNo=@GoodNo

		Insert into Invoice(BankCode , TransDate , REfNo , INVOICETYPE)
		VALUES(@BankCode , @TransDate , @RefNo , 'ECOM')
		set @InvoiceNo=@@identity
```

```
                Insert into ItemSale(InvoiceNo,GoodNo,QtySale)
                Values (@InvoiceNo,@GoodNo,1)
        end
GO


--SUBSCRIBE
CREATE        PROCEDURE [dbo].[SP_Jml_Tagihan_Bank]
        (@BankCode              char(4),
         @Months                char(7),
         @JmlTagihan            money OUTPUT,
         @ReplyStatus           int     OUTPUT,
         @ReplyMsg              varchar(255)    OUTPUT
        )
AS
Declare @TmpBln1        int,--char(2),
        @TmpBln2        char(2),
        @TmpThn1        int,--char(4),
        @TmpThn2        char(4),
        @PaymentStatus char(1),
        @TMP    money,
        @TMP2   money


        Set @ReplyStatus=0
        Set @ReplyMsg='Sukses'
        Set @PaymentStatus = 'P' --Paid=Sdh Bayar

        --Ambil Bln dan Thn dari Months/BlnBayar
        Set @TmpBln1=Cast((LEFT(@Months,2))as int)
        Set @TmpThn1=Cast((RIGHT(@Months,4))as int)

        --Cek Apakah Pada BulanBayar itu ada Tagihan Merchants ke bank
        IF NOT EXISTS
        (Select BankCode , Months From Invoice
         Where
          ((InvoiceType='ECOM' and BankCode=@BankCode)
        or
         (BankCode=@BankCode and PaymentStatus=@PaymentStatus and
               InvoiceType = 'SUBS' and
          (--Months = @Months
          --or
          ((month(cast(TransDate as char))=@TmpBln1 )
               and
           (year(cast(TransDate as char))=@TmpThn1)
          )--or
           )--bln
         )--kode
         )--type
        )--select
        BEGIN
               Set @ReplyStatus=5
               Set @ReplyMsg='Tidak Ada Pembayaran Bln ini mll Bank Ini '
               return
        END
        ELSE
        BEGIN
               --TOTAL JML TAGIHAN
               Set @Tmp2 = (Select (Sum(Invoice.ClaimTotal)) From Invoice
               Where
```

```
                (Invoice.BankCode=@BankCode                                    and
Invoice.PaymentStatus=@PaymentStatus and
                      Invoice.InvoiceType='SUBS' and
                 (--Months=@Months
                 --or
                 ((month(cast(Invoice.TransDate as char))=@TmpBlnl )
                      and
                  (year(cast(Invoice.TransDate as char))=@TmpThnl)
                 )--or
                  )--bln
                )--kode

               )--select

               if exists(select @tmp2 where @tmp2=null)
               begin
                      set @tmp2=0
               end

               if exists(select InvoiceNo From Invoice where
                             ((month(cast(Invoice.TransDate as char))=@TmpBlnl )
                                    and
                               (year(cast(Invoice.TransDate as char))=@TmpThnl)
                             ))
               begin
                      (Select @Tmp=(Sum(Goods.PriceperUnit))
                             From ItemSale,Invoice,Goods
                             Where (ItemSale.InvoiceNo=Invoice.InvoiceNo and
                                    ItemSale.GoodNo=Goods.GoodNo and
                                    Invoice.InvoiceType='ECOM'                 and
Invoice.BAnkCode=@BankCode and
                                    (month(cast(Invoice.TransDate              as
char))=@TmpBlnl )
                                           and
                                      (year(cast(Invoice.TransDate             as
char))=@TmpThnl)

                             )--where
                      )--select
                end
                else
                begin
                      set @tmp=0
                end
        set @JmlTagihan=@TMp+@TMP2


        END
GO

CREATE PROCEDURE dbo.SP_Register_Bank
        (@BankName      varchar(15),
         @BankCode      char(4) output,
         @Status        int     output,
         @Msg           varchar(255) output
        )
AS
--declare @BankCode char(4)

        set @Status=0
        set @Msg='Sukses'
```

```
        exec SP_Create_BankCode @BankCode=@BankCode output

        insert into Bank (BankCode, BankName)
        values (@BankCode, @BankName)


        set @Status=0
        set @Msg='Sukses'
GO


CREATE PROCEDURE dbo.SP_Unregister_Bank
        (@BankCode      char(4),
         @Status        int     output,
         @Msg           varchar(255) output
        )
AS
        set @Status = 0
        set @Msg = 'Sukses'

        --Cek bank
        IF NOT EXISTS (Select BankCode From Bank Where BankCode=@BankCode)
        BEGIN
                Set @Status=4
                Set @Msg='Data Tidak Ditemukan'
                return
        END

        if (@Status=0)
        begin
                delete from dbo.Bank Where BankCode=@BankCode
        end
GO

--SUBSCRIBE
--Update Tagihan Blnan

CREATE PROCEDURE dbo.SP_Update_Tagihan_Blnan
        (@CustNo        char(12),
         @BankCode      char(4),
         @TransDate     datetime,
         @RefNo char(36),
         @Months        char(7),
         @Status        int     output,
         @Msg           varchar(255)    output
        )

AS
declare @PAymentStatus char(1)

        set @PaymentStatus='P'
        set @Status=0
        set @Msg='Sukses'

        --Cek Customer
        IF NOT EXISTS (Select CustNo From Customer Where CustNo=@CustNo)
        BEGIN
                Set @Status=1
                Set @Msg='Data Customer Tidak Ditemukan'
                return
```

```
	END

	--Cek TAgihan Bln Ini Ada.tdk
	if not exists (Select CustNo, Months From Invoice Where (CustNo=@CustNo)
and (Months=@Months))
	begin
		set @STatus=3
		set @Msg='Tagihan Bln Ini Blm Ada'
	end

	if (@Status=0)
	begin
		if exists (Select * From Invoice Where PaymentStatus='C' and
BankCode=@BankCode)
		begin
			--Update
			Update dbo.Invoice
			Set PaymentStatus= @PaymentStatus, TransDate=@TransDate,
refNo=@RefNo
			Where (CustNo=@CustNo) and (Months = @Months) and
(BankCode=@BankCode)
		end
		else
		begin
			if exists(Select * From Invoice Where PaymentStatus='N')
			begin
				--Update
				Update dbo.Invoice
				Set PaymentStatus= @PaymentStatus,
TransDate=@TransDate, refNo=@RefNo
				Where (CustNo=@CustNo) and (Months=@Months) and
BankCode=@BankCode
			end
			else
			begin
				set @Status=7
				set @Msg='Tagihan Bulan Ini Sudah Dibayar'
			end
		end

	end
GO
```

STORED PROCEDURE UNTUK DATABASE E-COMMERCE

```
-- cek persediaan voucher
CREATE PROCEDURE dbo.SP_Ambil_Voucher
        (@GoodName      varchar(15),
         @PriceperUnit money,
         @Status int output,
         @Msg varchar(255) output
        )
AS

        set @Status=0
        set @Msg='Sukses'

        if exists(Select GoodNo From Goods Where (Saldo=1 and GoodName=@GoodName
and PriceperUnit = @PriceperUnit))
        begin
                set @Status=0
                set @Msg='Sukses'
        end
        else
        begin
                set @Status=3
                set @Msg='Persediaan Kosong'
        end
GO

CREATE PROCEDURE dbo.SP_Create_BankCode
        (@BankCode      char(4) output)
AS
declare @Max char(4),@tmp char(3),@tmpmax int

Select @Max=Max(BankCode) From Bank

set @tmp= right(@max,3)
set @tmpMax= (cast (@tmp as int) ) + 1

if (@tmpMax<10)
begin
        set @BankCode='B00'+ (cast (@tmpMax as char(1)))
end
else if(@tmpMax<100)
begin
        set @BankCode='B0' + (cast (@tmpMax as char(2)))
end
else
begin
        set @BankCode='B' + (cast (@tmpMax as char(3)))
end

GO


CREATE PROCEDURE SP_Insert_Invoice_Voucher
        (@BankCode      char(4),
         @RefNo char(36),
         @GoodName      varchar(15),
         @priceperunit money,
```

```
        @TransDate    datetime       output,
        @InvoiceNo    int    output,
        @GoodNo       char(10) output,
        @KodeVoucher char(12) output,
        @Status       int output,
        @Msg          varchar(255) output
       )
AS
--declare @priceperunit money

       set @Status=0
       set @Msg='Sukses'
       set @TransDate = GetDate()

       exec SP_Ambil_Voucher @GoodName=@GoodName , @PriceperUnit = @PriceperUnit
, @Status=@Status,@Msg=@Msg

       if (@Status=0)
       begin
              Set @GoodNo = (Select Min(GoodNo)
              From Goods
              Where ((Saldo = 1) and (GoodName = @GoodName) and Priceperunit =
@priceperunit)
              )


              Select  @KodeVoucher = KodeVoucher
              From Goods
              Where GoodNo = @GoodNo

              Update Goods
              Set Saldo = 0
              Where GoodNo=@GoodNo

              Insert into Invoice(BankCode , TransDate , REfNo , INVOICETYPE)
              VALUES(@BankCode , @TransDate , @RefNo , 'ECOM')
              set @InvoiceNo=@@identity

              Insert into ItemSale(InvoiceNo,GoodNo,QtySale)
              Values (@InvoiceNo,@GoodNo,1)
       end
GO

CREATE PROCEDURE dbo.SP_Register_Bank
       (@BankName     varchar(15),
        @BankCode     char(4) output,
        @Status       int    output,
        @Msg          varchar(255) output
       )
AS
--declare @BankCode char(4)

       set @Status=0
       set @Msg='Sukses'

       exec SP_Create_BankCode @BankCode=@BankCode output

       insert into Bank (BankCode, BankName)
       values  (@BankCode, @BankName)
```

```
      set @Status=0
      set @Msg='Sukses'
GO

CREATE PROCEDURE dbo.SP_Unregister_Bank
      (@BankCode  char(4),
       @Status    int   output,
       @Msg       varchar(255) output
      )
AS
      set @Status = 0
      set @Msg = 'Sukses'

      --Cek bank
      IF NOT EXISTS (Select BankCode From Bank Where BankCode=@BankCode)
      BEGIN
            Set @Status=4
            Set @Msg='Data Tidak Ditemukan'
            return
      END

      if (@Status=0)
      begin
            delete from dbo.Bank Where BankCode=@BankCode
      end
GO


CREATE PROCEDURE SP_CekExpNoRef
AS
declare @asal char(10)
declare @tujuan char(10)
declare @notrans int
declare @jumlah money
declare cekval cursor FOR
   SELECT notransaksi,noacc,accnodest,jumlah
   FROM transaksi
   where (getdate()>tgltransaksi+2) and (status='N')

open cekval

      FETCH NEXT FROM cekval INTO @notrans,@asal,@tujuan,@jumlah
      WHILE @@FETCH_STATUS=0
      BEGIN
            update transaksi set status='R' where notransaksi=@notrans
            update  account set saldo=saldo+@jumlah where noacc = @asal
            update  account set saldo=saldo-@jumlah where noacc = @tujuan

            FETCH NEXT FROM pekok INTO @notrans,@asal,@tujuan,@jumlah
      END

close cekval
deallocate cekval

GO
```