Payment Point Online Bank (PPOB) *Mobile Hybrid* dengan Koneksi *Billers* Berbasis HTTP dan *RESTful Services* untuk Usaha Mikro

Rinto Priambodo[1], Yaya Sudarya Triana[2]

[1,2]Fakultas Ilmu Komputer, Universitas Mercu Buana
Jl. Raya Meruya Selatan, Kembangan, Jakarta, 11650
e-mail : [1]rinto.priambodo@mercubuana.ac.id, [2]yaya.sudarya@mercubuana.ac.id

Abstrak

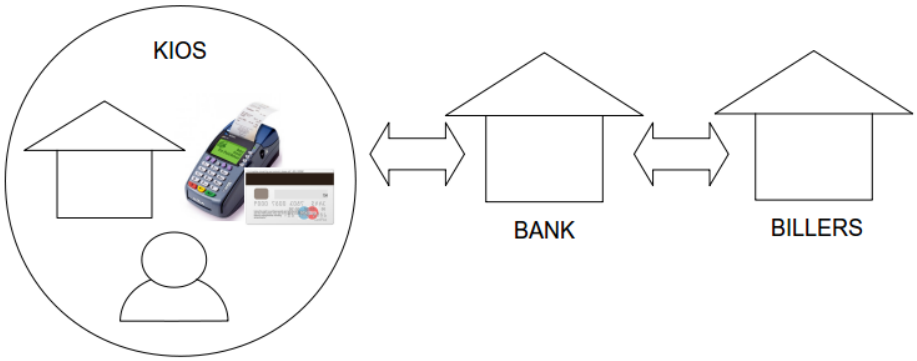
Sebuah perusahaan memiliki sejumlah agen kios yang sehari-harinya berjualan pulsa dengan perangkat Electronic Data Capture *(EDC). Perusahaan tersebut ingin meningkatkan penjualan dengan menambah variasi produk yang dapat dijual dengan cara yang sama yaitu menggunakan perangkat* mobile *secara* online*. Sayangnya untuk melakukan pembaruan terhadap aplikasi yang ada di EDC tidak mudah karena membutuhkan pemrograman secara khusus dan aplikasi tersebut harus dipasang ulang satu persatu. Sebuah perusahaan yang memiliki layanan switching penjualan pulsa, token listrik dan pembayaran tagihan (*billers*) memiliki fasilitas yang memungkinkan pelanggannya membangun aplikasi yang dapat bertransaksi secara* online *antar* server *(*Host-to-Host*/H2H). Fitur tersebut berbasis HTTP dan RESTful* services *yang memungkinkan pengembang aplikasi lebih fleksibel dengan bahasa pemrograman yang dipilihnya. Dalam penelitian ini akan dikembangkan sebuah aplikasi* mobile hybrid *yang memiliki koneksi* billers *menggunakan metode tersebut. Aplikasi* mobile hybrid *memungkinkan pengembang menambahkan fitur-fitur baru tanpa harus mengubah aplikasi yang ada di perangkat milik pengguna. Hal ini sangat menguntungkan karena produk pembayaran* online *semakin banyak sehingga penambahan fitur akan sering dilakukan. Selain itu arsitektur* mobile hybrid *dengan koneksi HTTP dan RESTful* service *juga dapat digunakan dalam kasus lain yang serupa di mana penyedia layanan menyediakan* Application Programming Interface (API) *yang berbasis HTTP dan RESTful* services.

Keyword: *Mobile Hybrid, PPOB, Pulsa, REST, Usaha Mikro*

1. Pendahuluan

PT LI di Surabaya memiliki lebih dari 100 agen kios penjualan pulsa telepon seluler di Jawa Timur yang sebelumnya berjualan dengan menggunakan perangkat *Electronic Data Capture* (EDC). PT LI ingin meningkatkan penjualan dengan memperbanyak jenis produk yang dijual. Sayangnya untuk menambahkan jenis produk yang dijual pada EDC membutuhkan usaha yang tidak mudah karena membutuhkan bahasa pemrograman dengan spesifikasi tertentu dan setiap perangkat harus diperbarui satu persatu untuk menambahkan fitur baru tersebut. PT PCT memiliki layanan penjualan pulsa telepon seluler, token listrik, dan pembayaran tagihan dari banyak perusahaan (*billers*). Layanan ini bisa diakses secara *online* dari *server* ke *server* (*Host-to-Host*/H2H). Dengan mengasumsikan bahwa setiap agen telah memiliki perangkat telepon genggam berbasis Android, PT LI menginginkan adanya sebuah aplikasi *mobile* yang dapat menyediakan layanan pembelian pulsa, token listrik atau pembayaran tagihan. Gambar 1 menunjukkan sistem berjalan sebelum penelitian dilakukan.

Saat ini banyak penyedia layanan yang menyediakan koneksi integrasi berbasis HTTP dan RESTful *service* seperti yang disediakan oleh PT PCT. Layanan seperti ini memudahkan rekan pengembang aplikasi untuk memilih bahasa pemrograman apa saja karena model seperti ini telah banyak didukung.



Gambar 1 *Sistem Berjalan*

Dari hasil survei singkat terhadap beberapa penyedia layanan penjualan pulsa dan pembayaran tagihan yang sudah ada di pasaran yang telah memiliki fasilitas koneksi H2H, aplikasi PPOB biasanya dapat menjual produk-produk layanannya dengan karakteristik sebagai berikut:

1. Untuk pembelian, aplikasi bisa langsung mengirimkan permintaan ke *server* H2H. Begitu aplikasi mendapatkan status balasan yang menyatakan berhasil, aplikasi akan memotong deposit pengguna.
2. Untuk pembayaran tagihan membutuhkan dua langkah. Pertama, aplikasi mendapatkan lebih dulu informasi besarnya tagihan yang harus dibayar. Kedua, melakukan pembayaran. Langkah kedua tidak bisa dijalankan tanpa terlebih dahulu menjalankan langkah pertama.
3. Penyedia layanan biasanya memiliki banyak koneksi ke perusahaan pemilik tagihan (*billers*). Koneksi ini tidak selalu langsung tapi bisa juga melalui layanan perantara yang mengakibatkan lamanya waktu tunggu untuk mendapatkan status permintaan. Untuk kasus ini aplikasi harus menyediakan mekanisme *callback* untuk penyedia layanan mengirimkan status pembelian atau pembayaran yang sebenarnya beberapa waktu kemudian. Atau permintaan status diinisiasi oleh aplikasi secara periodik sampai mendapatkan status yang diinginkan.
4. Koneksi yang tersedia biasanya berbasis XMPP/Jabber atau HTTP dengan format pesan XML.

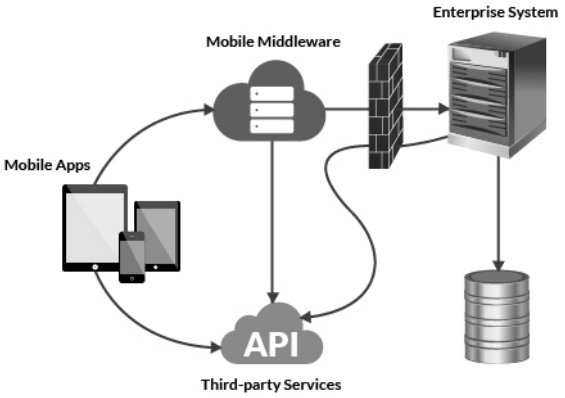
Dengan semakin banyaknya produk digital yang dapat dijual secara *online*, penambahan fitur produk menjadi lebih sering dilakukan. Aplikasi *mobile hybrid* memiliki beberapa kelebihan antara lain mudah dikembangkan menggunakan bahasa HTML dan JavaScript namun tetap dapat memiliki koneksi dengan fitur-fitur *native* dari *platform* telepon seluler (IBM, 2013) dengan demikian aplikasi tetap bisa memiliki kemampuan yang terkait dengan fitur *native* seperti mengakses mesin pencetak melalui *bluetooth* yang biasanya juga dibutuhkan dalam penjualan token PLN atau pembayaran tagihan untuk mencetak struk bukti pembayaran. Aplikasi *mobile hybrid* yang halaman *web*-nya ditempatkan di *cloud* juga memudahkan dalam pembaharuan aplikasi. Karena pengguna tidak perlu meng-*install* ulang aplikasi yang sudah terpasang dalam telepon seluler. Dengan demikian apabila ada penambahan fitur produk, pengembang dapat langsung memperbarui aplikasi yang ada di *cloud* kapan pun dan memastikan semua pengguna langsung dapat menggunakannya tanpa harus memperbarui aplikasi.

Untuk merealisasikan aplikasi ini akan dicoba dikembangkan sebuah aplikasi *native* dengan komponen *webview* untuk menampilkan aplikasi *web* yang berjalan di *cloud*. Aplikasi yang berjalan di *cloud* tersebut yang akan melakukan koneksi ke penyedia layanan untuk mengajukan permintaan pembelian pulsa dan token atau pembayaran tagihan. Sementara itu model pembayaran yang akan dibuat adalah dengan pemotongan deposit. Dengan demikian pengguna harus lebih dahulu menyetorkan sejumlah uang untuk kemudian dicatat ke dalam sistem. Sehingga ketika terjadi pembayaran maka sistem akan memotong nilai uang yang tersimpan sebagai bentuk pengurangan deposit.

2. Tinjauan Pustaka

Arsitektur aplikasi mobile

Arsitektur aplikasi mobile pada umumnya memiliki sebuah *middleware* yang berfungsi sebagai jembatan untuk mengakses database atau *resource* internal lain yang biasanya berada di belakang *firewall*. *Middleware* ini juga dapat berfungsi untuk menghubungkan aplikasi *mobile* dengan aplikasi atau layanan pihak ketiga di lokasi yang berbeda.



Gambar 2 *Arsitektur Aplikasi* Mobile *pada Umumnya* (Chan, 2016)

Aplikasi *middleware* ini tidak hanya melakukan otentikasi pengguna tapi juga mewakili pengguna untuk berinteraksi dan melakukan transaksi dengan penyedia layanan di pihak ketiga (Chan, 2016).

Gambar 2 menunjukkan hubungan antara aplikasi *mobile*, *middleware*, dan penyedia layanan. Pihak ketiga penyedia layanan harus menyediakan sebuah *Application Programming Interface* (API) sebagai sarana bagi aplikasi untuk menghubungkan diri dan melakukan transaksi.

Aplikasi mobile Hybrid

Pengembangan aplikasi *mobile* dengan pendekatan *hybrid* menggabungkan kode-kode *native* dengan teknologi *web*. Dengan menggunakan cara ini, pengembang dapat mengambil keuntungan dari teknologi *web* yang mendukung *cross-platform* sekaligus memiliki akses langsung terhadap API *native* dari *platform* perangkat *mobile*. Salah satu API *native* yang digunakan adalah mesin *render* HTML untuk menampilkan halaman berbasis HTML. Dengan demikian aplikasi yang dikembangkan tersebut tetap dapat memaksimalkan kemampuan yang tersedia dalam perangkat telepon genggam. Kode-kode halaman *web* yang bisa menggunakan skrip HTML, JavaScript, CSS beserta *resource* media yang dibutuhkan dapat ditempatkan dalam satu paket bersama dengan aplikasinya atau ditempatkan di dalam *server*. Apabila ditempatkan di *server*, pengembang dapat melakukan pembaharuan terhadap aplikasi *web* tanpa harus membaharui aplikasi *native* dan meminta pengguna untuk melakukan instalasi ulang (IBM, 2013).

Hypertext Transfer Protocol (HTTP)

HTTP adalah protokol standar yang digunakan oleh *browser* untuk mengambil atau mengirimkan data dari atau ke dalam server. Protokol ini memiliki mekanisme Request dan Response, yang artinya untuk setiap permintaan dari aplikasi klien akan diberikan respon atau balasan dari aplikasi *server* berupa data status pengiriman atau data yang hendak ditampilkan kepada pengguna. Adanya mekanisme request dan response ini memastikan data yang dikirim ke server dalam setiap sesi pengiriman dapat dipastikan status keberhasilannya. Selain itu pula, karena protokol HTTP adalah protokol standar yang telah lama digunakan oleh *browser*, maka telah banyak bahasa pemrograman yang mendukung implementasi protokol ini. Contohnya *library* cURL di PHP, *class* HttpUrlConnection di Java, atau class HttpClient di .NET.

Klien HTTP adalah aplikasi yang membuat koneksi ke *server* untuk mengirimkan satu atau lebih request HTTP. Sementara itu server HTTP adalah aplikasi yang menerima koneksi untuk melayani request HTTP dengan mengirim balik *response* HTTP. Klien mengirimkan *request* HTTP ke *server* berupa sebuah pesan yang berisi metode, lokasi *resource* yang diminta, versi protokol, sekumpulan metadata dan badan pesan jika ada. Server membalas dengan mengirim satu atau lebih pesan *response* yang berisi baris-baris status, versi protokol, kode kesalahan, metadata, dan lain-lain termasuk di dalamnya badan pesan berisi data yang diminta oleh klien. *User Agent* (UA) atau aplikasi klien yang mengirim *request* dapat berupa *browser*, aplikasi *mobile*, aplikasi *command-line*, atau aplikasi lainnya. Begitu juga dengan aplikasi *server* dapat berupa perangkat *home automation*, komponen jaringan, mesin-mesin, kamera lalu-lintas, dan lain-lain. Istilah "*user agent*" tidak selalu berarti ada manusia yang mengoperasikan aplikasi tersebut, tapi dapat juga berupa aplikasi otomatis yang berjalan sendiri, contohnya *web crawler* (Fielding & Reschke, 2014).

API Billers

PT PCT memiliki API untuk pembelian dan pembayaran online berupa *web services* yang dapat diakses melalui internet. Spesifikasi penggunaan API ini dijelaskan dalam dokumen spesifikasi API yang disediakan oleh PT PCT.

API yang dimiliki oleh PT PCT memiliki sejumlah fungsi, antara lain:

- Pembelian produk top up prabayar
- Cek status transaksi
- *Balance*
- *Echo*
- Inquiry pascabayar
- Pembayaran bil*l*ing/tagihan pascabayar
- Pembelian tiket *travel*/*shuttle*
- Pembelian tiket pesawat terbang

Dokumen Teknikal PT PCT juga dilengkapi dengan mekanisme pertukaran data, format data dan kode transaksi yang dibutuhkan untuk membedakan tiap transaksi yang dilakukan. Dalam penelitian ini API tersebut digunakan untuk menghubungkan aplikasi dengan layanan yang disediakan oleh *billers*.

Penelitian Terkait

Sejumlah penelitian telah banyak dilakukan untuk membuat sebuah rancangan maupun aplikasi yang dapat memfasilitasi pembelian pulsa maupun pembayaran tagihan melalui aplikasi mobile berbasis Android. Aplikasi tersebut dirancang untuk dapat dijalankan di beberapa versi Android (Rachman, 2017) dan ada juga yang media komunikasinya dengan penyedia layanan menggunakan *Short Message Service* (SMS) (Mulyasari, 2014).

3. Analisis Kebutuhan

Proses Bisnis Berjalan

Dalam proses pembelian maupun pembayaran, pihak-pihak yang terlibat dalam proses transaksi adalah sebagai berikut:

1. Agen

Agen merupakan pengguna utama yang melakukan transaksi dalam sistem. Agen menerima permintaan pembelian atau pembayaran tagihan dari pelanggan kemudian meneruskan permintaan tersebut ke sistem melalui perangkat EDC. Setelah EDC mendapatkan konfirmasi dari billers, agen akan mendapatkan struk bukti pembelian/pembayaran. Pembayaran dilakukan menggunakan deposit yang dimiliki oleh agen di bank dengan otorisasi menggunakan kartu debit yang digesekkan di perangkat EDC.

2. Bank

Dalam sistem ini bank memiliki dua peran, yaitu sebagai sistem *switching* yang meneruskan permintaan pembelian/pembayaran ke *billers* yang sesuai. Kedua, sebagai pihak yang meneruskan pembayaran ke *billers* menggunakan deposit yang dimiliki oleh agen. Bank melakukan pemotongan saldo tabungan agen sesuai otorisasi dari agen yang menggunakan kartu debit melalui mesin EDC.

3. *Billers*

Billers merupakan pihak yang menyediakan produk digital untuk dibeli seperti pulsa telepon seluler prabayar dan token listrik prabayar. *Billers* juga menyediakan layanan pembayaran tagihan untuk layanan-layanan prabayar seperti telepon seluler dan listrik PLN. *Billers* bekerja sama dengan bank untuk menyediakan layanannya melalui komunikasi *Host-to-Host* (H2) dan memiliki kesepakatan tersendiri untuk pembayarannya.

Produk yang dijual dapat dibagi menjadi dua macam kategori, yaitu:

1. Produk pembelian

Merupakan produk digital yang dapat diperoleh setelah dilakukan pembayaran. Produk ini memiliki pilihan nominal tertentu dan memiliki keterkaitan dengan nomor identitas pelanggan dari layanan yang terkait dengan produk tersebut. Dengan demikian pembeli harus terlebih dahulu memiliki nomor pelanggan dan memberikan nomor tersebut saat melakukan pembelian. Contoh dari produk pembelian ini adalah sebagai berikut:

- Pulsa telepon seluler prabayar
- Token listrik prabayar
- TV prabayar
- Paket data seluler
- *Voucher game*

2. Produk pembayaran tagihan

Merupakan produk layanan pembayaran tagihan dari sebuah akun berlangganan pada layanan tertentu. Untuk menggunakan produk ini pelanggan harus terlebih dahulu mengecek jumlah tagihan dengan memberikan nomor pelanggan yang dimiliki. Setelah mengetahui besar tagihan maka pelanggan baru bisa melakukan pembayaran. Contoh dari produk pembayaran tagihan ini adalah sebagai berikut:

- Pembayaran tagihan telepon seluler pascabayar
- Pembayaran tagihan listrik PLN
- Pembayaran tagihan PDAM
- Pembayaran cicilan kredit
- Pembayaran asuransi
- Pembayaran kartu kredit

Kebutuhan Fungsional

Dari proses bisnis yang berjalan tersebut, maka sistem ini diharapkan memiliki fungsi-fungsi sebagai berikut:

1. Transaksi
    a. Pembelian
    Merupakan fungsi yang disediakan untuk melayani produk pembelian. Di bagian ini harus tersedia fungsi untuk pemilihan operator pemilik produk dan penyedia layanan terkait, pengisian nomor identitas pelanggan, dan konfirmasi harga produk yang akan dibeli. Fungsi ini harus menyediakan juga informasi mengenai status pembelian. Produk digital seperti pulsa telepon seluler prabayar akan dikirimkan langsung melalui sistem yang dimiliki oleh operator ke perangkat yang dimiliki oleh pelanggan. Karena proses tersebut tidak melalui sistem ini maka sistem ini harus menyediakan informasi yang jelas mengenai status pengiriman tersebut. Apakah statusnya berhasil, gagal atau tertunda. Informasi status tersebut juga harus menampilkan kode referensi dari operator yang dapat digunakan untuk melakukan pengecekan langsung ke operator.
    b. Pembayaran Tagihan
    Merupakan fungsi yang disediakan untuk melayani produk pembayaran tagihan. Produk pembayaran tagihan memiliki perbedaan dengan produk pembelian karena sebelum melakukan pembayaran harus terlebih dahulu dilakukan inquiry atau pengecekan jumlah tagihan. Setelah itu baru dapat dilakukan pembayaran. Untuk dapat dilakukan *inquiry* pengecekan jumlah tagihan dibutuhkan nomor identitas pelanggan dan pilihan operator penyedia layanan berlangganan. Setelah mendapatkan jumlah tagihan baru dapat dilakukan pembayaran sesuai dengan jumlah tagihan tersebut.
    c. Histori transaksi
    Fungsi histori transaksi dibutuhkan untuk menyimpan catatan transaksi yang pernah dilakukan. Fungsi ini dibutuhkan untuk melakukan pelacakan informasi dari status transaksi apabila terjadi perselisihan dan harus dilakukan audit.
2. Administrasi dan Manajemen
    a. Manajemen Pengguna
    Fungsi ini disediakan bagi pengguna dan administrator sistem untuk mengelola informasi pengguna sistem. Pengguna sistem dapat melakukan pendaftaran melalui aplikasi mobile dengan memasukkan nomor telepon seluler sebagai identitas unik dan password dengan kombinasi alfanumerik. Dalam proses pendaftaran, aplikasi mobile harus menyertakan nomor IMEI dari perangkat yang digunakan oleh pelanggan. Setelah melakukan pendaftaran pengguna dapat melengkapi detil informasi pengguna dan melakukan perubahan atau update. Sementara itu untuk administrator sistem harus disediakan juga aplikasi berbasis web untuk memudahkan pengelolaan informasi pengguna.
    b. Deposit
    Fungsi administrasi deposit dibutuhkan untuk mengetahui saldo deposit, menambahkan jumlah saldo, memberikan konfirmasi penambahan (top-up) saldo dan melihat histori penggunaan/ pengurangan atau penambahan jumlah saldo. Deposit merupakan sejumlah uang yang dititipkan oleh pengguna sistem kepada penyedia sistem untuk digunakan dalam pembelian atau pembayaran tagihan. Setiap transaksi yang berhasil akan memotong saldo deposit pengguna. Pengguna dapat menambahkan saldo dengan memberikan sejumlah uang kepada penyedia sistem. Pemberian uang ini dilakukan di luar sistem. Sehingga setelah penyedia

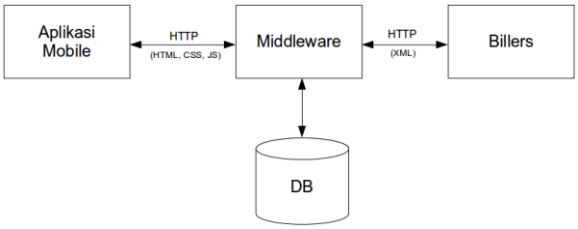
sistem menerima uang, penyedia sistem harus melakukan penambahan jumlah saldo milik pengguna. Pemberian uang ke penyedia layanan dapat melalui transfer bank. Setelah pengguna melakukan transfer, pengguna dapat memberikan konfirmasi melalui form yang disediakan di dalam aplikasi mobile. Administrator sistem kemudian akan melakukan pengecekan penerimaan uang sesuai konfirmasi yang diberikan dan kemudian melakukan penambahan saldo deposit melalui aplikasi berbasis web. Baik pengguna maupun administrator sistem dapat melihat histori mutasi deposit untuk melakukan pengecekan dan pelacakan.

4. Perancangan

Dari analisis kebutuhan di atas dibuat perancangan sistem yang digambarkan secara garis besar dalam diagram arsitektur sistem pada Gambar 3.

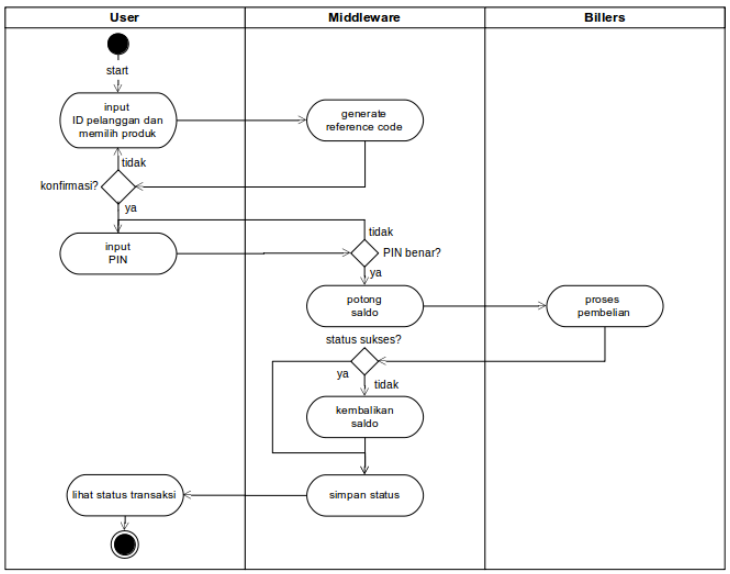
Gambar 3 tersebut menggambarkan arsitektur sistem yang menggambarkan hubungan antara aplikasi *mobile* dan *middleware* yang merupakan satu kesatuan aplikasi dengan sebuah entiti eksternal sebagai pihak ketiga penyedia layanan yang disebut dengan *billers*. Aplikasi *middleware* menggunakan sebuah basis data internal yang digunakan untuk menyimpan data-data berikut:

- Data pengguna sistem untuk kebutuhan identifikasi dan otorisasi penggunaan sistem.
- Daftar produk yang merupakan produk-produk yang disediakan oleh *billers* lengkap dengan harga beli dan biaya administrasi jika ada.
- Rekaman transaksi pembelian dan pembayaran lengkap dengan produk yang diperjualbelikan, pengguna yang terlibat jual beli dan harga serta biaya yang timbul dalam transaksi.
- Deposit pelanggan yang akan berkurang ketika terjadi transaksi dan bertambah kembali ketika dilakukan *top-up* atau *refund* dari sistem.

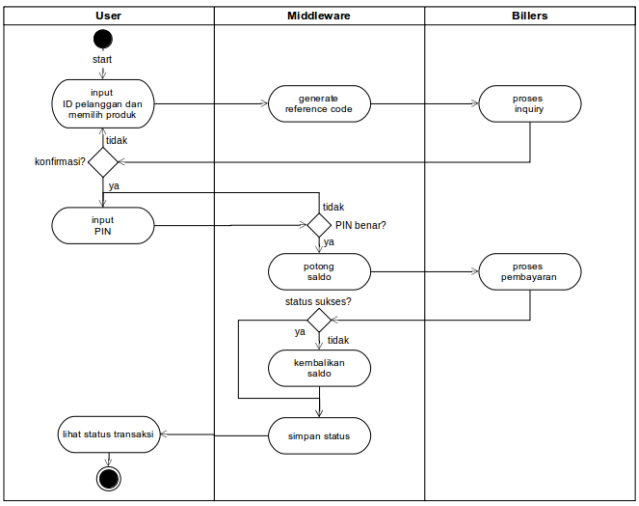


Gambar 3 *Arsitektur Sistem*

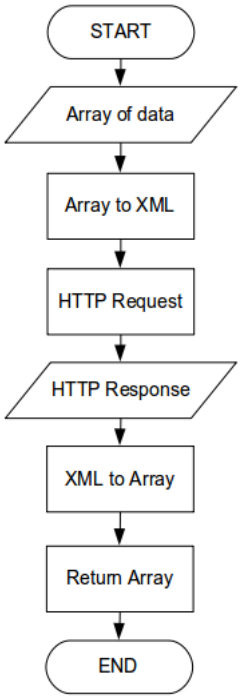
Sementara itu hubungan dengan entitas eksternal antara *middleware* dan *biller* mengikuti spesifikasi yang disediakan oleh penyedia layanan [4]. Gambar 4 dan Gambar 5 menunjukkan *activity diagram* yang menunjukkan aliran aktifitas mulai dari pengguna sampai *billers*.



Gambar 4 *Activity Diagram Proses Pembelian*



Gambar 5 *Activity Diagram Proses Pembayaran Tagihan*



Gambar 6 *Diagram Alir Proses Konversi Tipe Data dalam Proses Pengiriman dan Penerimaan Data*

Aplikasi yang dibangun harus dapat mengirimkan data dalam format XML melalui HTTP *Request* dan menerima data *Response* juga dalam format XML. Gambar 6 menunjukkan aliran proses pengubahan data dari tipe *array* menjadi XML sebelum dikirimkan melalui HTTP *Request* dan menerimanya kembali juga dalam format XML.

Contoh *Request* Prabayar dalam format XML adalah sebagai berikut:

```
<transaction>
   <transactioncode>
         TRANSACTION_CODE
   </transactioncode>
```

```
    <productcode>PRODUCT_CODE</productcode>
    <accnumber>081234567890</accnumber>
    <userid>USER_ID</userid>
    <userpin>USER_PIN</userpin>
    <signature>1234567890abcdef</signature>
    <referencecode>
          KODE_REFERENSI_CA
    </referencecode>
</transaction>
```

Contoh *Response* Prabayar dalam format XML adalah sebagai berikut:

```
<transaction-status>
    <tracecode>1234567890123456</tracecode>
    <transactionstatus>000</transactionstatus>
    <transactiontime>
          YYMMddHHmmss
    </transactiontime>
    <serialnumber>
          11223344556677889900
    </serialnumber>
    <harga>110000</harga>
    <balance>999999</balance>
    <productcode>PRODUCT_CODE</productcode>
    <accnumber>081234567890</accnumber>
    <userid>USER_ID</userid>
    <info>Informasi untuk CA</info>
    <referencecode>
          KODE_REFERENSI_CA
    </referencecode>
</transaction-status>
```

Contoh Request Inquiry Pascabayar adalah sebagai berikut:

```
<transaction>
    <transactioncode>
          TRANSACTION_CODE
    </transactioncode>
    <productcode>PRODUCT_CODE</productcode>
    <accnumber>081234567890</accnumber>
    <userid>USER_ID</userid>
    <userpin>USER_PIN</userpin>
    <signature>1234567890abcdef</signature>
    <referencecode>
          KODE_REFERENSI_CA
    </referencecode>
</transaction>
```

Arsitektur *mobile hybrid* yang digunakan dibangun di atas *platform* Android dengan menempatkan sebuah elemen WebView yang dapat menampilkan halaman *web* standar yang menggunakan skrip HTML, CSS dan JavaScript. Sebuah objek yang diturunkan dari kelas WebView diinisiasikan dengan mengaktifkan fitur pendukung JavaScript dan melekatkannya ke sebuah elemen *layout* bertipe WebView. Elemen WebView dimuat pertama kali dengan memanggil URL web yang berisi halaman *login*. Sementara itu akses internet dari aplikasi dibuka melalui tag *permission* yang ditambahkan dalam *file* AndroidManifest.xml seperti berikut:

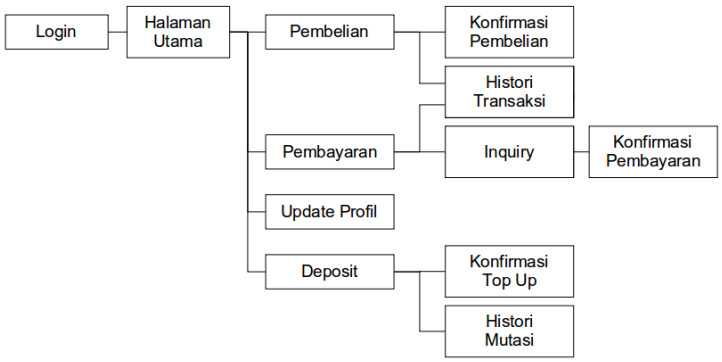
```
<uses-permission android:name="android.permission.INTERNET" />
```

Selanjutnya komponen WebView tersebut akan melakukan pengambilan *resource* dari *server* menggunakan protokol HTTP untuk mendapatkan halaman web yang dibutuhkan.

Tata letak halaman *web* menggunakan framework jQuery Mobile yang memiliki pustaka CSS dan JavaScript dengan spesifikasi khusus perangkat *mobile* yang memiliki ukuran layar kecil dan input menggunakan layar sentuh. Sehingga diperoleh antarmuka yang ramah pengguna.

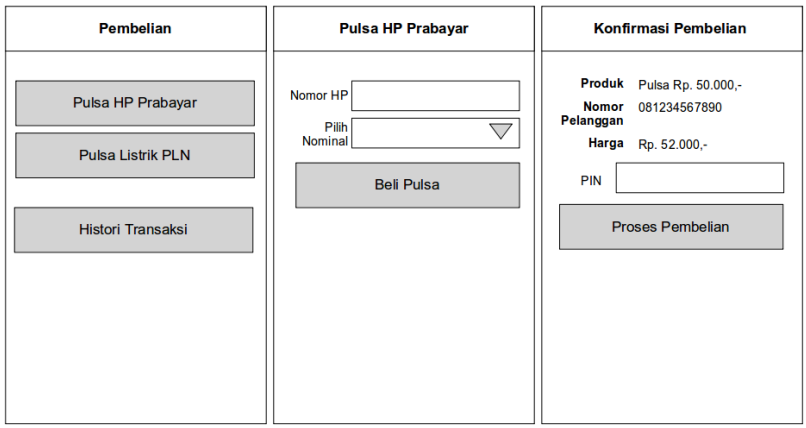
Desain navigasi dari aplikasi digambarkan dengan hirarki menu seperti pada Gambar 7. Setelah melewati halaman *login*, pengguna akan memasuki Halaman Utama yang memiliki empat pilihan menu utama, yaitu:

1. Menu Pembelian
   Berisi pilihan produk pembelian seperti pulsa HP prabayar dan token listrik PLN. Setelah memilih produk, pengguna akan diminta untuk mengisi detil pembelian dan memberikan konfirmasi sebelum melakukan pembayaran. Pengguna juga dapat memilih untuk melihat histori transaksi.

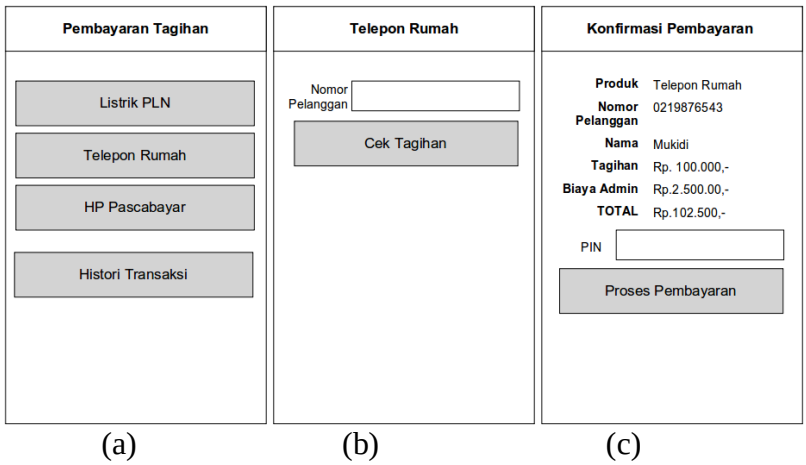


Gambar 7 *Hirarki Menu*

Gambar 8 menunjukkan desain antarmuka pembelian mulai dari pilihan produk, pengisian detil pembelian, sampai konfirmasi pembelian dengan melakukan pengisian PIN untuk otorisasi pemotongan saldo deposit.



Gambar 8. *Desain Antarmuka (a) Pembelian (b) Detil Pembelian (c) Konfirmasi Pembelian*



Gambar 9. *Desain antarmuka (a) Pembayaran Tagihan (b) Detil Pembayaran (c) Konfirmasi Pembayaran*

2. Pembayaran *Hirarki Menu*
   Berisi pilihan produk pembayaran tagihan seperti tagihan telepon seluler pascabayar dan tagihan listrik. Setelah memilih produk, pengguna akan diminta untuk mengisi detil akun pelanggan dan memberikan konfirmasi tagihan sebelum melakukan pembayaran. Pengguna juga dapat memilih untuk melihat histori transaksi.
   Gambar 9 menunjukkan desain antarmuka pembayaran tagihan mulai dari pilihan produk, pengisian detil akun pelanggan, sampai konfirmasi pembelian dengan melakukan pengisian PIN untuk otorisasi pemotongan saldo deposit.
3. Update Profil
   Berisi form untuk memperbarui detil profil pengguna
4. Deposit
   Berisi form konfirmasi top-up dan informasi histori mutasi saldo deposit.

5. Implementasi dan Pengujian

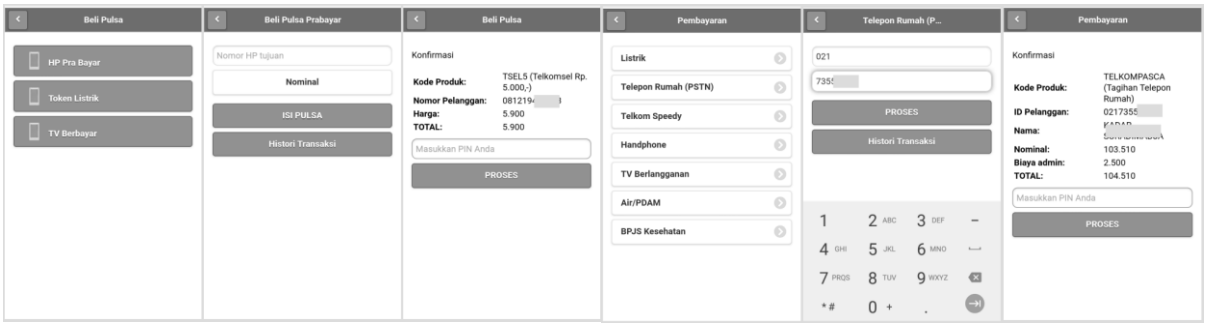
Sistem ini diimplementasikan ke dalam dua *platform*, yaitu Android untuk aplikasi *mobile hybrid* dan untuk aplikasi *middleware* juga aplikasi berbasis *web* dijalankan di *server* Linux dengan *web server* Apache dan *database* MySQL. Aplikasi di server menggunakan bahasa pemrograman PHP dengan *framework* CakePHP versi 2.7.8 yang sudah menggunakan pola *Model View Controller* (MVC)

Aplikasi di *server* menggunakan *framework* CakePHP diimplementasikan dalam struktur file sebagai berikut:

- Config
  - database.php digunakan untuk menyimmpan konfigurasi koneksi ke database
  - core.php digunakan untuk menyimpan konstanta konfirmasi dan konstanta kode status yang digunakan dalam sistem
  - routes.php digunakan untuk menyimpan pengaturan URL dari aplikasi.
- Controller
  - AppController.php berisi fungsi-fungsi yang bisa diakses dari controller mana saja. Misalnya fungsi untuk cek saldo.
  - PrepaidController.php berisi fungsi-fungsi yang digunakan dalam transaksi pembelian. Misalnya pembelian pulsa seluler prabayar dan token listrik prabaya.
  - PostpaidController.php berisi fungsi-fungsi yang digunakan dalam transaksi pembayaran tagihan. Misalnya pembayaran tagihan telepon seluler pascabayar dan tagihan listrik.
  - UsersController.php berisi fungsi-fungsi manajemen pengguna dan otorisasi serta otentikasi.
  - DepositsController.php berisi fungsi-fungsi pengelolaan deposit, pengurangan dan penambahan saldo.
  - Component
    Sebuah *class* yang diturunkan dari *class* Component juga dibuat untuk mengimplementasikan *method-method* pemanggilan API *billers*. Pemanggilan API tersebut menggunakan *library* HttpSocket yang merupakan bagian dari *library* Network/Http milik CakePHP. Selain itu digunakan juga *library* Xml yang merupakan bagian dari *library* Utility milik CakePHP yang digunakan untuk konversi antara tipe data *array* dan XML sesuai spesifikasi yang diminta oleh penyedia layanan.
- Model
  Tidak dibutuhkan model khusus untuk aplikasi ini. Cukup menggunakan konvensi CakePHP untuk merelasikan tabel dan *class* yang disediakan oleh CakePHP.
- View
  Di bagian ini dilakukan pemisahan antara bagian *head* dan *body*. Bagian *head* memuat skrip dan CSS yang dibutuhkan untuk penggunaan jQuery Mobile. Bagian *body* memuat konten sesuai halaman yang ditampilkkan.

Aplikasi *mobile hybrid* berbasis Android diimplementasikan dengan komponen WebView untuk menampilkan halaman HTML yang dibuat di *server*.

Gambar 10 menunjukkan implementasi antarmuka pengguna sesuai desain yang telah dibuat.



a) (b) (c) (d) (e) (f)

Gambar 10. . *Implementasi Antarmuka (a) Pembelian (b) Detil pembelian (c) Konfirmasi Pembelian (d) Pembayaran tagihan (e) Detil pembayaran (f) Konfirmasi pembayaran*

Pengujian aplikasi dilakukan dengan ujicoba pembelian pulsa ke sebuah nomor pelanggan telepon seluler prabayar dan pembayaran tagihan telepon rumah. Kesuksesan pengujian ditentukan dengan keberhasilan mendapatkan produk yang dibeli, tagihan yang terbayarkan, dan saldo deposit yang berhasil terpotong. Ringkasan hasil pengujiannya dapat dilihat dalam Tabel 1.

Tabel 1. Hasil Pengujian

Transaksi	Nilai Transaksi	Hasil Pengujian
Pembelian pulsa HP nomor 08121940xxxx dengan nominal Rp. 5.000,-	Rp. 5.900,-	• Pulsa berhasil terisi • Saldo terpotong sebesar Rp. 5.900,-
Pembelian token listrik prabayar ID pelanggan nomor 23211107xxxx dengan nominal Rp. 20.000,-	Rp. 22.750,-	• Token listrik diperoleh • Saldo terpotong sebesar Rp. 22.750,-
Pembayaran telepon rumah nomor 021735xxxx dengan jumlah tagihan Rp. 103.510,-	Rp. 104.510,-	• Tagihan berhasil dibayarkan • Saldo terpotong sebesar Rp. 104.510,-
Pembayaran telepon seluler pascabayar nomor 0811137xxxx dengan jumlah tagihan Rp. 197.995,-	Rp. 200.495,-	• Tagihan berhasil dibayarkan • Saldo terpotong sebesar Rp. 200.495,-
Pembayaran tagihan listrik pascabayar ID pelanggan nomor 54360056xxxx dengan jumlah tagihan Rp.613.518,-	Rp. 616.018,-	• Tagihan berhasil dibayarkan • Saldo terpotong sebesar Rp. 616.,-

6. Kesimpulan

Kesimpulan dari penelitian ini adalah:

1. Arsitektur *mobile hybrid* dapat digunakan untuk membangun aplikasi *mobile* untuk pembelian pulsa dan pembayaran tagihan yang disebut dengan istilah PPOB.
2. Model integrasi berbasis HTTP dan RESTful *service* dapat digunakan untuk menghubungkan sebuah penyedia layanan penjualan dan pembayaran tagihan dengan aplikasi berbasis *mobile hybrid*.

Ucapan Terima Kasih

Penulis mengucapkan terima kasih kepada semua pihak yang mendukung penelitian ini khususnya kepada Pusat Penelitian Universitas Mercu Buana atas bantuan dana yang diberikan.

Daftar Pustaka

CakePHP *Cookbook* 2.0. Diambil pada tanggal 27 Mei 2018, dari https://book.cakephp.org/2.0/_downloads/en/CakePHPCookbook.pdf

Chan, L. ”*Mobile Middleware: Service-Oriented-Architecture vs. Mobility-as-a-Service”*. Diambil pada tanggal 2 November 2017, dari https://prolifics.com/blog/mobile-middleware-service-oriented-architecture-vs-mobility-service, 2016.

Dokumen Teknikal PASTI v4.1, PT PCT. Tidak dipublikasikan, 2016.

Fielding, R., Reschke, J. ”RFC 7230: *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing”*. Internet Engineering Task Force (IETF). Diambil pada tanggal 2 November 2017, dari https://tools.ietf.org/html/rfc7230, 2014.

IBM Worklight V6.0 Technology Overview. IBM Corporation, 2013.

Mulyasari, Yuli. ”Perancangan Aplikasi Mobile Penjualan Pulsa Elektronik Putri Cell Berbasis Android”. Jakarta, 2014 [online], https://repository.mercubuana.ac.id/11621/ diambil pada tanggal 27 Mei 2018.

Rachman, Fathur. ”Aplikasi Transaksi Pulsa Multi Operator Berbasis Android”, Jakarta, 2017 [online], https://repository.mercubuana.ac.id/33819/ diambil pada tanggal 27 Mei 2018.