# IMPLEMENTASI *JSON PARSING* PADA APLIKASI *MOBILE E-COMMERCE* Studi Kasus : CV V3 Tekno Indonesia

Bhakti Destian Wijaya[1], Fenty E.M.A[2], dan Andrew Fiade [3]

[1,2,3] Program Studi Teknik Informatika, Fakultas Sains dan Teknologi
Universitas Islam Negeri Syarif Hidayatullah Jakarta

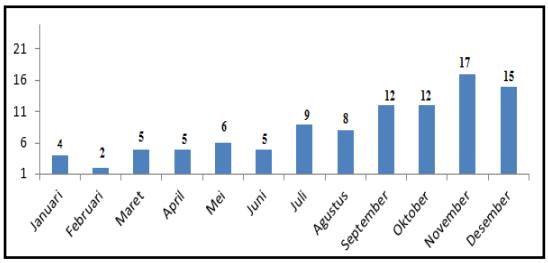
[1]destian.bhakti@gmail.com
[2]fenty.eka@uinjkt.ac.id
[3]andrew_fiade@uinjkt.ac.id

*Abstrak*: Sistem operasi Android saat ini merupakan salah satu dari sekian banyak sistem operasi pada *smartphone* yang sedang mengalami pertumbuhan pesat di dunia, khususnya di Indonesia. CV Vitiga Tekno Indonesia bergerak di bidang pemasaran *gadget* membutuhkan media *marketing* yang dapat diakses secara *online* selain *website* yaitu melalui *smartphone*. Oleh karena itu penelitian ini berfokus pada bagaimana membuat aplikasi *mobile e-commerce* pada *smartphone* Android, dengan cara mengambil data yang berasal dari *website*, dan kemudian ditampilkan ke dalam sebuah aplikasi, menggunakan metode *HTTP Connection* dan *JSON Parsing*. Pembuatan aplikasi menggunakan bahasa pemrograman *Java*, dengan bantuan *tools IDE Eclipse* dan *MySQL* untuk *database server*. JSON berkomunikasi melalui *Application Program Interface* (API) bertujuan untuk menghubungkan aplikasi *mobile* dengan *database* yang ada di dalam *server*. Pada penelitian ini API ialah kumpulan *source code* PHP, yang isinya adalah *query* untuk mengambil data dari *website*, yang hasilnya di-*encode* ke dalam bentuk JSON.
Kata Kunci: *Mobile e-commerce*, *JSON Parsing*, *Application Program Interface* (API).

***Abstract: Nowadays, Android operating* sistem *is one of the existing operating* systems *on smartphones that growing rapidly. CV Vitiga Tekno Indonesia is a company in the marketing gadget requires marketing media that can be accessed online via smartphone. Therefore, this research focuses on how to make a mobile e-commerce application on the Android smartphone, by taking data that comes from a website, and then show into a user as an application, using the HTTP Connection method through JSON Parsing. The mobile e-commerce application is built using java programming language / IDE Eclipse and MySQL tools for the database server. JSON, communicates through Application Program Interface (API), aims to connect mobile applications with the existing database on the server. In this study, API is a collection of queries that retrieve data from the website in PHP language. The results are encoded into JSON form.***
***Keywords: Mobile e-commerce, JSON parsing, Application Program Interface (API).***

## I. PENDAHULUAN

Saat ini, CV V3 Tekno Indonesia adalah sebuah perusahaan yang menjual berbagai macam *gadget* dan menggunakan *website* sebagai media *marketing* untuk menjangkau para konsumennya. Dengan bertumbuh pesatnya sistem operasi Android, CV V3 Tekno Indonesia ingin mempunyai sistem dan media *marketing* yang baru selain *website*.




Gambar 1. Peningkatan transaksi perdagangan *gadget* selama tahun 2013 melalui www.*vitigaon.com*

Berdasarkan Gambar 1, diketahui bahwa selama tahun 2013 ada peningkatan transaksi perdagangan *gadget* melalui *website*. Hal ini menunjukkan adanya minat konsumen yang cukup besar untuk bertransaksi secara *online*. Oleh karena itu, perlu dilakukan terobosan baru untuk memberi kemudahan konsumen bertransaksi melalui

*smartphone* yaitu dengan membuat aplikasi berbasis Android tentang *vitigaon.com*. Aplikasi ini dibuat dengan mengacu pada *database website vitigaon.com* sehingga data yang ditampilkan akan sama dengan yang ada di *website*.

Aplikasi *e-commerce* yang diakses atau diinstal pada *smartphone* disebut juga *mobile e-commerce*. Aplikasi ini memakai teknologi *JSON Parsing* yang ada di dalam android melalui *HTTP Connection*. Sehingga aplikasi *website* yang ada saat ini dapat terintegrasi dengan aplikasi yang ada di *handset*. Hal ini menyebabkan jika terjadi perubahan data di dalam sisi aplikasi *website*, perubahan data tersebut terjadi pada aplikasi *handset*. Dibandingkan dengan hanya sekedar menggunakan *webview*, tentunya *JSON Parsing* akan lebih unggul dari segi *resource* yang dikeluarkan ketika memuat *content* yang berasal dari aplikasi *website*.

*Webview* adalah sebuah *component* yang ada di dalam sistem operasi Android yang berfungsi untuk memuat sebuah tampilan *website* ke dalam bentuk tampilan *mobile web*. *Webview* menampilkan secara serupa dengan tampilan sesungguhnya di *desktop*. *Webview* memuat sendiri semua atribut yang ada di *website* seperti *button*, *text*, *image* dan lain sebagainya dengan format *HTML*. Tentunya aktifitas ini memerlukan konsumsi data yang lebih banyak. Berbeda dengan *JSON Parsing* dalam *HTTP Connection*, tampilan yang baru dari sisi aplikasi *mobile* dapat dibuat dengan cara memilih sendiri data atau in*form*asi apa saja yang ingin ditampilkan pada aplikasi. Dengan demikian pada saat aplikasi berjalan, aplikasi hanya akan memuat jenis *content* yang sudah ditentukan dan tidak akan memuat *content* dari *website* secara keseluruhan persis seperti *website* aslinya.

Berdasarkan latar belakang tersebut diatas, maka tujuan penelitian ini adalah membuat sebuah aplikasi *mobile e-commerce* pada *smartphone* Android, dengan menerapkan JSON *parsing* untuk menampilkan data dari *website*. Pada aplikasi yang dibuat, terdapat sistem transaksi *Business-To-Business* (B2B) dan *Business-To-Consumer* (B2C). Sistem B2B terjadi antara CV V3 Tekno Indonesia dengan mitra bisnisnya (para *Reseller*) dan B2C adalah antara CV V3 Tekno Indonesia dengan pengguna akhir barang dagangan.

Sebelum melaksanakan penelitian dilakukan jajak pendapat kepada 50 orang pengguna *smartphone* yang diantaranya adalah konsumen CV V3 Tekno Indonesia, mengenai perlu dibuatnya aplikasi *mobile e-commerce* untuk memudahkan transaksi. Hasilnya adalah 64% responden menjawab perlu adanya aplikasi yang menunjang transaksi perdagangan di *smartphone*. Dengan demikian hasil penelitian ini diharapkan dapat membuktikan manfaat *JSON Parsing* pada pembuatan aplikasi *mobile e-commerce* baik bagi pemilik bisnis maupun kosumennya.

## II. LANDASAN TEORI

### 2.1. Sekilas Tentang JSON (*Java Script Object Notation*)

JSON (*Java Script Object Notation*) adalah format pertukaran data yang bersifat ringan, disusun oleh Douglas Crockford. Fokus JSON adalah pada representasi data di *website* [1][2][4][5]. JSON dirancang untuk memudahkan pertukaran data pada situs dan merupakan perluasan dari fungsi-fungsi *javascript*.

Contoh teks php dalam memformat hasil query berupa format JSON:

```
<?php
include_once("conn.php");
$sql = "SELECT * FROM penjualan";
```

```
$query = mysql_query($sql);
$data = array();
while($row=mysql_fetch_object($query))
{$data[] = $row;}
$response
array('status'=>"Ok",'item'=>$data);
echo json_encode($response)
?>
```

Luaran JSON dari kode diatas adalah sebagai berikut :

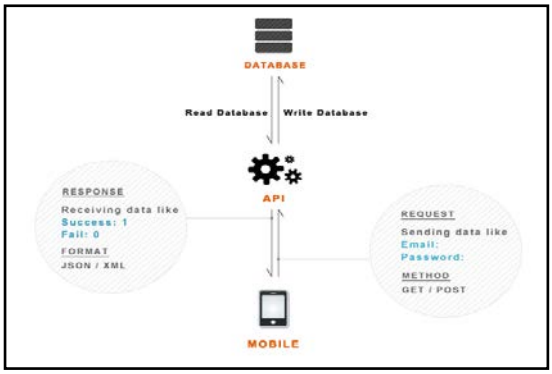
*{"status":"Ok","item":[{"id":"1","bulan":"Januari","total":"700"},{"id":"2","bulan":"Februari","total":"400"},{"id":"3","bulan":"Maret","total":"200"},{"id":"4","bulan":"April","total":"800"},{"id":"5","bulan":"Mei","total":"700"},{"id":"6","bulan":"Juni","total":"230"},{"id":"7","bulan":"Juli","total":"400"},{"id":"8","bulan":"Agustus","total":"350"},{"id":"9","bulan":"September","total":"800"},{"id":"10","bulan":"Oktober","total":"350"},{"id":"11","bulan":"November","total":"400"},{"id":"12","bulan":"Desember","total":"600"}]}*

### 2.2. JSON *parsing*

Penggunaan JSON *Parsing* dalam *HTTP Connection* ini bertujuan untuk memberi kemudahan bagi *user* ketika menggunakan aplikasi ini. Dengan adanya JSON *Parsing* dalam *HTTP Connection*, informasi yang ada didalam *website* dapat ditampilkan di dalam sebuah aplikasi *mobile*[3]. Hal ini menyebabkan aplikasi *mobile* tidak perlu menampilkan seluruh content yang ada didalam *website* seperti halnya pada *mobile* web browser. Aplikasi android akan memilih jenis konten yang ingin ditampilkan, seperti halnya gambar, *item description*, dan lain-lain.

Dalam JSON *Parsing*, kita membutuhkan API (*Application Program Interface*) yang berfungsi untuk menghubungkan antara aplikasi *mobile* dan aplikasi *website*[3].

Pada penelitian ini API ialah kumpulan *source code* berbasis PHP, yang isinya adalah Query untuk mengambil data dari *website*, yang hasilnya di-*encode* ke dalam bentuk JSON. Pada saat aplikasi *mobile* mengirimkan *request* untuk menampilkan data (SELECT * FROM) dari *website*, API akan meneruskannya sesuai dengan Query yang tepat untuk menampilkan data (SELECT * FROM). Setelah data diambil, kemudian di- EN*CODE* ke dalam bentuk JSON, dan diteruskan ke dalam aplikasi *mobile*. Di dalam aplikasi *mobile*, JSON tersebut di-*parsing* ke dalam bentuk yang kita inginkan, misalnya dalam bentuk *list*. Gambar 2 menunjukkan arsitektur JSON *parsing*.




Gambar 2. Arsitektur JSON *parsing*

## III. ANALISIS SISTEM BERJALAN

Pada Gambar 3 menunjukkan sistem yang saat ini berjalan di CV V3 Tekno Indonesia. Saat ini, perusahaan menggunakan *website* sebagai media e-commmerce untuk menjangkau pelanggannya. Calon pembeli dapat mengakses *website* perusahaan untuk melakukan transaksi maupun hanya sekedar melihat *item*. Di dalam *website* tidak tersedia fitur *review item* yang lebih lengkap sehingga calon pembeli dapat melihat *review item* melalui mesin pencari Google. Setelah melakukan pemesanan, pembeli melakukan konfirmasi pembayaran dengan cara menelpon langsung ke *Customer Service* dari CV V3 Tekno Indonesia,

ataupun melalui *e-mail* yang berisikan bukti pembayaran yang di *upload*. Setelah itu CV V3 Tekno Indonesia meresponnya dengan cara mencocokannya dengan data di *database*.


Gambar 3. *Rich Picture* Sistem Berjalan

Berdasarkan *rich picture* sistem yang berjalan pada saat ini, penulis melakukan sebuah analisa terkait permasalahan yang ada.

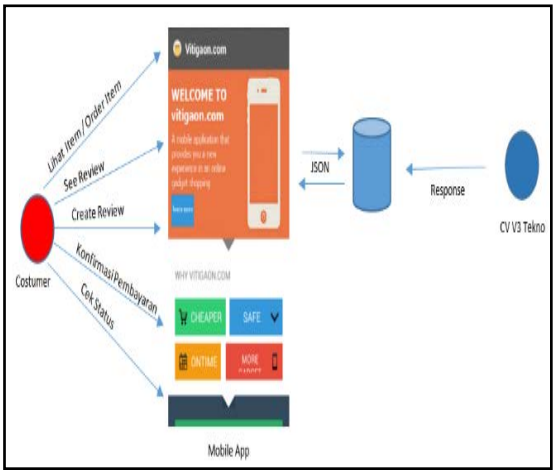
1. Dengan tampilan *website* yang statis seperti pada saat ini, para calon pembeli akan kesulitan ketika mengakses *website*nya lewat *mobile* phone, khususnya Android. Sehingga penulis berkesimpulan, perlu dibangun sebuah aplikasi berbasis Android untuk menjangkau calon pembeli yang mengaksesnya lewat *mobile phone*.
2. Dengan *website* yang ada pada saat ini, hanya terdapat fitur pemesanan saja. Sehingga kegiatan lain yang termasuk di dalam *e-commerce*, seperti halnya konfirmasi pembayaran, dan permintaan status transaksi, tidak dapat dilakukan di dalam satu aplikasi (aplikasi). Sehingga penulis berkesimpulan perlu di buat sejumlah fitur seperti konfirmasi pembayaran, permintaan status, dan *review item* yang akan ada di dalam satu aplikasi (*mobile*).

## IV. EKSPERIMEN

### 4.1 Gambaran Sistem Yang Diusulkan

Berdasarkan analisis yang dilakukan, penulis mengusulkan sebuah sistem yang baru dengan penjelasan sebagai berikut (Gambar 4):

1. Semua kegiatan yang ada di dalam *e-commerce* (*order*, konfirmasi pembayaran, permintaan status) dapat dilakukan di dalam satu aplikasi (*mobile*).
2. Data yang ditampilkan di dalam aplikasi berasal dari *database website vitigaon.com*. sehingga data yang ada di dalam *website* sama dengan data yang ditampilkan di aplikasi *mobile*.
3. Data *exchange* menggunakan format JSON dengan metode *Parsing* dalam *HTTP Connection*
4. Pembayaran dilakukan dengan cara transfer.
5. Konfirmasi pembayaran dilakukan dengan cara mengisi *form* didalam aplikasi
6. Permintaan status transaksi dilakukan di dalam aplikasi dengan mengirimkan nomor *invoice*-nya.




Gambar 4. *Rich Picture* Sistem Usulan

### 4.2 Metode Pembayaran

Pada saat pembeli telah selesai melakukan transaksi, pembeli akan mendapatkan email konfirmasi dari *vitigaon.com* berupa detail pesanan dan jumlah harga beserta nomor *Invoice*-nya. Kemudian pembayaran dilakukan dengan cara transfer bank sesuai yang tertera di *invoice*. Pada saat transfer, pembeli diharuskan mengirim sejumlah uang sesuai dengan total transaksi yang

dilakukan ditambah dengan nomor *invoice* 3 digit dibelakangnya. Contoh penulisan jumlah yang harus ditransfer oleh pembeli dapat dilihat pada Tabel 1.

Tabel 1. Contoh Jumlah Pembayaran

| Jumlah transaksi | *Invoice* number | Jumlah yang harus di transfer |
|---|---|---|
| Rp. 4.000.000 | 111 | Rp. 4.000.111 |

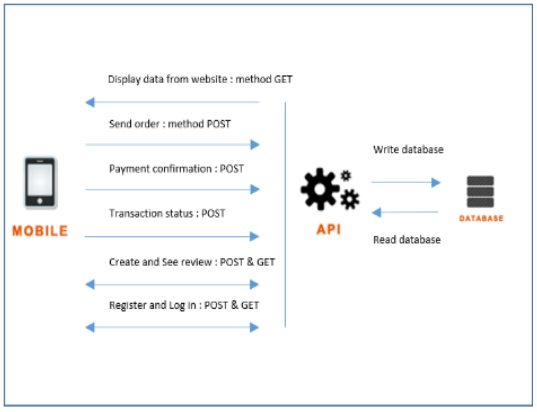
Hal ini bertujuan untuk memudahkan *vitigaon.com* dalam melacak pembeli yang sudah melakukan transfer. Selanjutnya *vitigaon.com* akan mengecek apakah sesuai dengan *form* konfirmasi yang telah di isi oleh pembeli jika sudah melakukan pembayaran.

4.3 Perancangan entitas

Pada tahap ini dirancang dan ditentukan entitas-entitas yang terdapat di dalam sistem. Entitas-entitas tersebut adalah :

1. *Seller* (*vitigaon.com*)
2. *User* untuk *Reseller*
3. *User* untuk grosir
4. Kategori *item*
5. *Object item*
6. *Cart*
7. Tutorial
8. *Review*

4.4 Perancangan Arsitektur Aplikasi




Gambar 5. Arsitektur aplikasi

Berdasarkan Gambar 5 mengenai arsitektur aplikasi, secara umum gambaran sistem sebagai berikut :

- Aplikasi menampilkan data dari *website* dengan metode GET
- Aplikasi mengirim data pesanan (order), konfirmasi pembayaran dan status order dengan metode POST
- Aktivitas *create* dan *see review* menggunakan metode POST dan GET secara bergantian.
- Proses pendaftaran (*register*) dan masuk ke aplikasi (*login*) menggunakan metode POST dan GET.

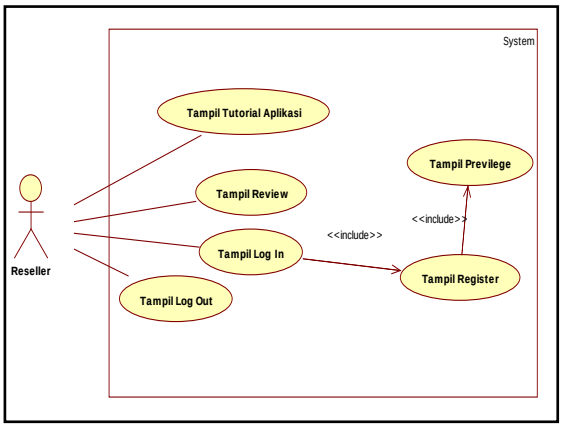
4.5 Perancangan sistem

4.5.1 Menentukan Aktor dan Membuat *Use Case*

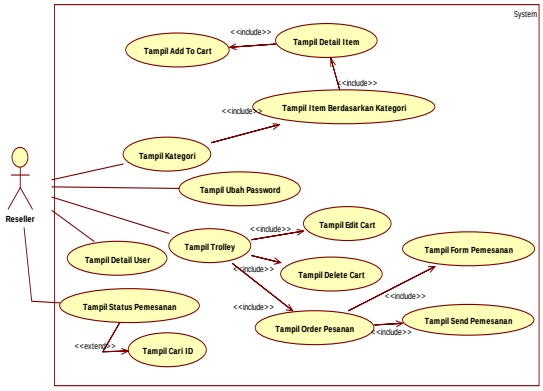
Tabel 2. Identifikasi Aktor

| No | Aktor | Deskripsi |
|---|---|---|
| 1 | *Buyer* | Pengguna aplikasi yang berperan sebagai pembeli yang membeli product dalam satuan eceran. (B2C) |
| 2 | *Reseller* | Pengguna aplikasi yang berperan sebagai pembeli yang membeli product dalam satuan grosir. (B2B) |

*Reseller* dapat melihat tutorial penggunaan aplikasi, *review* dari *gadget* yang ada didalam aplikasi, dan juga dapat *login* ke dalam aplikasi atau daftar sebagai *user* jika belum terdaftar. *Reseller* juga dapat melihat previlege bila menjadi *user* dengan level *Reseller* (Gambar 6).




Gambar 6. *Usecase Diagram Reseller* Bagian 1




Gambar 7. *Usecase Diagram* untuk *Reseller* bagian 2

Setelah *login*, proses yang terjadi sebagai berikut (Gambar 7):

- *Reseller* dapat melihat bermacam-macam kategori untuk memulai transaksi. Setelah sesuai dengan keinginan, *reseller* dapat menambahkan *item* yang dipilihnya ke dalam sebuah *cart*. Ketika sudah selesai dalam memilih *item* untuk dibeli, *reseller* dapat melanjutkan ke proses berikutnya yaitu tampil *trolley*.
- Di dalam proses *trolley* ini, *reseller* dapat mengedit dan menghapus *cart* nya jika ingin membatalkan transaksi. Setelah selesai, *reseller* dapat memesan barang dengan cara mengisi *form* pemesanan yang disediakan oleh aplikasi.
- Kemudian data dari *form* pemesanan tersebut akan masuk ke dalam *database* aplikasi. Selain itu *reseller* juga dapat mengedit *user* profilnya seperti mengganti password yang baru untuk proses log in. *Reseller* juga dapat melihat status pengiriman atau pemesanan dengan cara memasukan ID transaksi.

4.5.2 Perancangan *database*

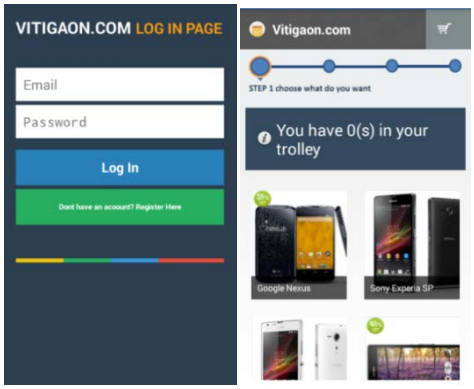
*Database* yang digunakan adalah *database* yang ada pada *server* aplikasi *website vitigaon.com*, yang telah ditambahkan sesuai dengan kebutuhan untuk fitur tambahan pada aplikasi *mobile*. Dengan demikian, data dan in*form*asi yang ditampilkan di dalam aplikasi yang dibuat, sama dengan yang ditampilkan di aplikasi *website*. Sehingga transaksi yang sedang berlangsung di aplikasi web maupun *mobile* dapat berjalan langsung secara *real time* dan tersimpan di dalam satu *database*.

Berikut adalah rincian *database* yang penulis gunakan :

Nama *database* : vitigaDB
Lokasi *database* : www.*vitigaon.com*
Jumlah tabel : 8 tabel

4.5.4 Pembuatan API (*Application Program Interface*)

*Application Program Interface* (API) bertujuan untuk menghubungkan aplikasi *mobile* dengan *database* yang ada di dalam *server*. API ini nantinya untuk memenuhi seluruh jenis fungsi yang dibutuhkan di dalam aplikasi. Seperti fungsi tampil *item*, fungsi edit *item*, fungsi POST transaksi, dan lain sebagainya[5]. API ini pada akhirnya akan di-*hosting* secara *online* sehingga dapat diakses secara *online* (Gambar 8).




Gambar 8. Rancangan *Interface*

4.5.5 Implementasi JSON *Parsing*

1. Aplikasi *mobile* ingin menampilkan data produk dari *database website*. Yang pertama kali di lakukan, adalah membuat API. API berisi Query SELECT*FROM. Seperti di bawah ini :

```php
// get all products from products table
$result = mysql_query("SELECT *FROM products") or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    // looping through all results
    // products node
    $response["products"] = array();

    while ($row = mysql_fetch_array($result)) {
        // temp user array
        $product = array();
        $product["pid"] = $row["pid"];
        $product["name"] = $row["name"];
        $product["price"] = $row["price"];
        $product["created_at"] = $row["created_at"];
        $product["updated_at"] = $row["updated_at"];

        // push single product into final response array
        array_push($response["products"], $product);
    }
    // success
    $response["success"] = 1;

    // echoing JSON response
    echo json_encode($response);
} else {
    // no products found
    $response["success"] = 0;
    $response["message"] = "No products found";

    // echo no users JSON
    echo json_encode($response);
}
```

*Source code* tersebut berbasis PHP, berisikan Query SELECT, dan jika data di*item*ukan, data di "echo" ke dalam bentuk JSON ("echo json_en*code*").

Jika *source code* tersebut dijalankan, maka akan tampil JSON *code* seperti di bawah ini :

```json
{
    "products": [
        {
            "pid": "1",
            "name": "iPhone 4S",
            "price": "300.00",
            "created_at": "2012-04-29 02:04:02",
            "updated_at": "0000-00-00 00:00:00"
        },
        {
            "pid": "2",
            "name": "Macbook Pro",
            "price": "600.00",
            "created_at": "2012-04-29 02:04:51",
            "updated_at": "0000-00-00 00:00:00"
        },
        {
            "pid": "3",
            "name": "Macbook Air",
            "price": "800.00",
            "created_at": "2012-04-29 02:05:57",
            "updated_at": "0000-00-00 00:00:00"
        },
        {
            "pid": "4",
            "name": "OS X Lion",
            "price": "100.00",
            "created_at": "2012-04-29 02:07:14",
            "updated_at": "0000-00-00 00:00:00"
        }
    ],
    "success": 1
}
```

Ini adalah bentuk JSON dari data hasil *Query* dari API yang di buat. Selanjutnya, JSON *code* ini akan dilanjutkan ke dalam aplikasi *mobile* untuk di-*parse*. Caranya sebagai berikut: dari sisi aplikasi android, yang pertama di lakukan adalah menentukan tampilan antarmuka untuk menampilkan JSON yang akan di-*parsing* nantinya.

all_products.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <!-- Main ListView
         Always give id value as list(@android:id/list)
    -->
    <ListView
        android:id="@android:id/list"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"/>

</LinearLayout>
```

list_item.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >

    <!-- Product id (pid) - will be HIDDEN - used to pass to other activity -->
    <TextView
        android:id="@+id/pid"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:visibility="gone" />

    <!-- Name Label -->
    <TextView
        android:id="@+id/name"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingTop="6dip"
        android:paddingLeft="6dip"
        android:textSize="17dip"
        android:textStyle="bold" />

</LinearLayout>
```

Kemudian ketika dibangun lagi sebuah *Class* yang akan digunakan untuk proses *parsing*

```java
public class AllProductsActivity extends ListActivity {

    // Progress Dialog
    private ProgressDialog pDialog;

    // Creating JSON Parser object
    JSONParser jParser = new JSONParser();

    ArrayList<HashMap<String, String>> productsList;

    // url to get all products list
    private static String url_all_products = "http://api.androidhive.info/androi

    // JSON Node names
    private static final String TAG_SUCCESS = "success";
    private static final String TAG_PRODUCTS = "products";
    private static final String TAG_PID = "pid";
    private static final String TAG_NAME = "name";

    // products JSONArray
    JSONArray products = null;
```

maka terbentuk *class* dengan METHOD JSON parser, dan dengan proses inisialisasi dari API dalam bentuk URL.

```java
/**
 * Background Async Task to Load all product by making HTTP Request
 * */
class LoadAllProducts extends AsyncTask<String, String, String> {


    @Override
    protected void onPreExecute()


    protected String doInBackground(String... args)


    protected void onPostExecute(String file_url)
```

*Class* tersebut berisikan 3 *method* :

- *onPreExecute* : adalah metode yang akan di proses sebelum proses koneksi dilakukan.

```
@Override
protected void onPreExecute() {
    super.onPreExecute();
    pDialog = new ProgressDialog(AllProductsActivity.this);
    pDialog.setMessage("Loading products. Please wait...");
    pDialog.setIndeterminate(false);
    pDialog.setCancelable(false);
    pDialog.show();
}
```

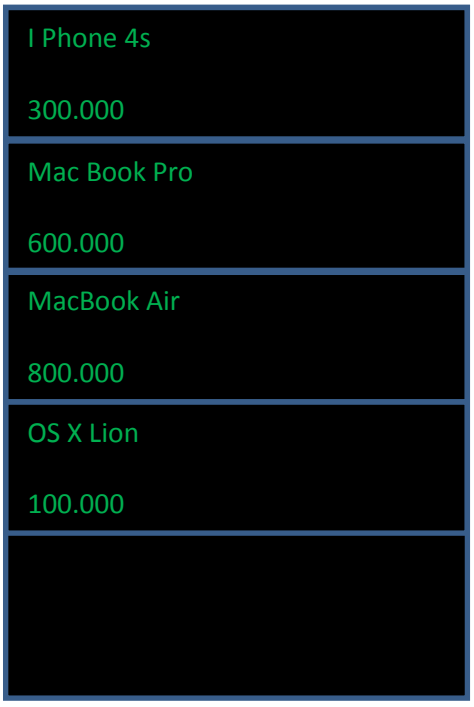
- *doInBackGround* : adalah metode yang akan di proses di dalam *background* aplikasi. Disinilah proses *PARSING* terjadi.

```
protected String doInBackground(String... args) {
    // Building Parameters
    List<NameValuePair> params = new ArrayList<NameValuePair>();
    // getting JSON string from URL
    JSONObject json = jParser.makeHttpRequest(url_all_products, "GET",

    // Check your log cat for JSON reponse
    Log.d("All Products: ", json.toString());

    try {
        // Checking for SUCCESS TAG
        int success = json.getInt(TAG_SUCCESS);

        if (success == 1) {
            // products found
            // Getting Array of Products
            products = json.getJSONArray(TAG_PRODUCTS);

            // looping through All Products
            for (int i = 0; i < products.length(); i++) {
                JSONObject c = products.getJSONObject(i);

                // Storing each json item in variable
                String id = c.getString(TAG_PID);
                String name = c.getString(TAG_NAME);

                // creating new HashMap
                HashMap<String, String> map = new HashMap<String, Stri

                // adding each child node to HashMap key => value
                map.put(TAG_PID, id);
                map.put(TAG_NAME, name);

                // adding HashList to ArrayList
                productsList.add(map);
            }
```

- *onPostExecute* : adalah metode yang akan di proses setelah *Parsing* dilakukan.

```
/**
 * After completing background task Dismiss the progress dialog
 * **/
protected void onPostExecute(String file_url) {
    // dismiss the dialog after getting all products
    pDialog.dismiss();
    // updating UI from Background Thread
    runOnUiThread(new Runnable() {
        public void run() {
            /**
             * Updating parsed JSON data into ListView
             * */
            ListAdapter adapter = new SimpleAdapter(
                    AllProductsActivity.this, productsList,
                    R.layout.list_item, new String[] { TAG_PID,
                            TAG_NAME},
                    new int[] { R.id.pid, R.id.name });
            // updating listview
            setListAdapter(adapter);
        }
    });
```

Gambar 9 adalah hasil akhir tampilan *mobile application*.




Gambar 9. Tampilan *Mobile Application*

4.6 Implementasi

4.6.1 Instalasi API ke Internet

Tahap ini, API (*Application Program Interface*) yang dibuat diinstal ke dalam *hosting* internet. Proses ini dilakukan dengan tujuan API dapat berjalan secara *online*. API di-*hosting* ke dalam *server website* dari *vitigaon.com*. Sehingga untuk mengakses nya dapat melalui URL : *vitigaon.com*/API .

4.6.2 Instalasi Aplikasi ke dalam *Handset*

Proses ini dilakukan untuk mengetahui apakah aplikasi yang dibuat dapat berjalan di perangkat sesungguhnya. Berikut adalah proses instalasi aplikasi ke dalam *handset*.


Gambar 10. Instalasi di *Handset*

4.6.3 Pengujian *Blacbox*

Pengujian *Black Box* dilakukan untuk mengetahui apakah fungsi dan alur aplikasi dapat berjalan sesuai dengan harapan.

1. Fungsi *register account* dapat berjalan sesuai dengan harapan.
2. Proses *log in* untuk masing-masing *user* dapat berjalan sesuai dengan harapan.
3. Proses menampilkan halaman *Reseller* dan halaman buyer masing-masing dapat berjalan sesuai dengan harapan.
4. Proses transaksi *buyer* dan transaksi *Reseller* dapat berjalan sesuai dengan harapan, di antaranya :
   a) Proses tampil kategori dapat berjalan sesuai dengan harapan
   b) Proses memilih *item* dapat berjalan sesuai dengan harapan.
   c) Proses edit trolley dapat berjalan sesuai dengan harapan.
   d) Proses mengirim pesanan dapat berjalan sesuai dengan harapan
   e) Proses menerima *email* konfirmasi dapat berjalan sesuai dengan harapan.
   f) Proses konfirmasi pembayaran dapat berjalan sesuai dengan harapan
   g) Proses meminta status transaksi dapat berjalan sesuai dengan harapan.

## V. KESIMPULAN DAN SARAN

### 5.1 Kesimpulan

Berdasarkan hasil penelitian diperoleh kesimpulan:

1. Aplikasi *mobile e-commerce* yang dikembangkan oleh CV V3 Tekno menggunakan data yang berasal dari satu *database* yang sama, yaitu yang digunakan oleh *website*.
2. Manipulasi data yang berasal dari *database vitigaon.com* dilakukan dengan data *exchange* dalam format JSON, yang kemudian di-*parse* ke dalam aplikasi Android.

### 5.2 Saran

1. Dalam aplikasi ini, diharapkan adanya fitur dimana *user* / *buyer* dapat melihat sendiri status transaksinya sehingga tidak memerlukan fitur kirim SMS untuk melakukannya.
2. Diharapkan dalam aplikasi ini terdapat fitur *upload* gambar ketika ingin melakukan konfirmasi pembayaran.

## REFERENSI

[1] Chasseur, Craig., Li, Y., dan Patel, Jm. *Enabling JSON Document Stores in Relational Systems*. Sixteenth International Workshop on the Web and Databases (WebDB 2013). 2013.

[2] Google. *Package Summary*. HTTP://developer.android.com/reference/android/app/package-summary.html di akses pada 10 September 2014. 2014.

[3] Peng, Dunlu., Cao, Lidong., dan Xu, Wenjie. *Using JSON for Data Exchanging in Web Service Application.* Journal of Computational Information Sistem, volume 16, page 5883-5890. 2011.

[4] Safaat, Nazruddin. *Android (Pemograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android).* Informatika, Bandung. 2011.

[5] W3Schools, *JSON Tutorial*, 2013, *HTTP://www.w3schools.com/json/*. 2013..