PEMANFAATAN MIDDLEWARE LAYER UNTUK MULTI PLATFORM PADA MOBILE DAN DESKTOP

Hendro Purwoko
Informatika, Universitas Indraprasta PGRI
hendroprwk08@gmail.com

Submitted August 4, 2019; Revised October 10, 2019; Accepted October 25, 2019

Abstrak

Teknologi saat ini memiliki banyak *platform* dari segi perangkat keras dan lunak, seperti yang kita ketahui beberapa perangkat pun memiliki bahasa pemprograman yang berbeda pula. Hal ini rumit jika didapati sistem yang ada, masih menggunakan metode *Client-Server* yang hanya berdiri pada satu *platform* saja sehingga perlu mengusulkan arsitektur untuk pengembangan aplikasi *mobile* dengan menggunakan infrastruktur yang ada, Dampak dari itu muncullah kombinasi logika pemrograman dengan basis data terdistribusi yang disebut *middleware*, yang cara berkomunikasinya disepakati agar memiliki kesatuan. Kondisi *multi platform* juga memerlukan internet agar terhubung dengan pemakai aplikasi karena menggunakan protokol HTTP atau HTTPS sehingga pertukaran data dapat berjalan dengan baik. Hasil dari proses basis data menggunakan format JSON yang dinilai efektif dan telah terbukti kehandalannya dalam menghasilkan keluaran yang dapat diuraikan oleh *multi platform*. Pemanfaatan middleware menggunakan metode Rapid Application Development agar proses pembangunan perangkat lunak efektif karena penanganan kesalahan yang cepat sehingga mampu membangun perangkat lunak yang dapat melayani berbagai *platform* dan mudah untuk dikembangkan dimasa mendatang.

Kata Kunci: *Multi Platfrom*, Android, Desktop, API, RESTful API, JSON, Middleware, Rapid Application Development

Abstract

Technology currently has many platforms from hardware and software, as we know some devices also have different programming languages. This is complicated if the existing system still uses the Client-Server method that only stands on one platform, so it is necessary to propose architecture for mobile application development using the existing infrastructure, so it appears A combination of programming logic with a distributed database called Middleware, which communicates to have unity. Multi-platform conditions also require the Internet to connect with application users because it uses HTTP or HTTPS protocols so that data exchange can run. The result of database process is formed in JSON format which is reliable and effective and has been proven reliability in generating output that can be read by multi platform. Development of Middleware using Rapid Application Development method to make software development process effective and fast error handling so as to build software that can serve various platforms and easy to develop.

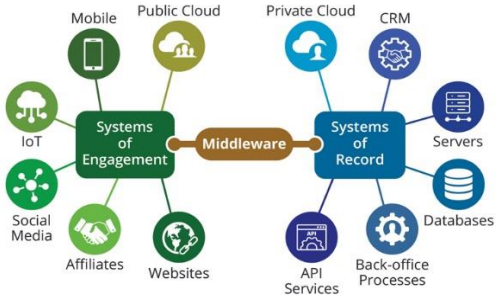
Key Words: *Multi Platfrom, Android, Desktop, API, RESTful API, JSON, Middleware, Rapid Application Development*

1. PENDAHULUAN

Guna mendukung kemajuan teknologi yang terus berkembang, berbagai institusi dan perusahaan mengembangkan sistem komputer yang dapat digunakan untuk berbagai kalangan agar memudahkan serta memanjakan *stakeholders*, hal ini rumit jika didapati sistem yang ada, masih menggunakan metode *Client-Server* yang hanya berdiri pada satu *platform* saja sehingga perlu mengusulkan arsitektur untuk pengembangan aplikasi *mobile* dengan menggunakan infrastruktur institusi yang ada [8].

Keadaan saat ini berbagai perangkat keras dan lunak baru muncul untuk memenuhi kebutuhan berbagai elemen industri, namun terkadang tak berbanding lurus dengan keadaan hingga para pembuat perangkat lunak atau *programmer* pun harus bekerja dengan berbagai Sistem Operasi, *platform* dan bahasa pemrograman berbeda menjadi aplikasi yang dapat terhubung dengan aplikasi lain melalui jaringan global [3].

Salah satu solusi agar berbagai *platform* tersebut menjadi satu kesatuan adalah dengan memanfaatkan *Web Service* API yang dikombinasikan menggunakan REST atau disebut pula dengan RESTful API [1] kemudian menggunakan HTTP atau HTTPS sebagai protokol dan menghasilkan keluaran berupa notasi JSON agar mudah diurai serta merupakan bahasa pertukaran data yang ideal [5] agar dapat digunakan oleh berbagai *platform*.



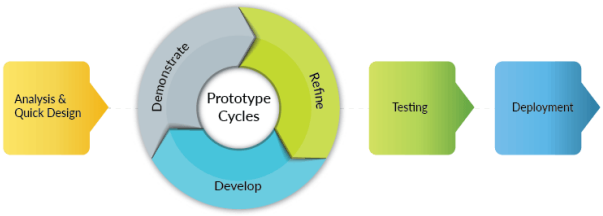
Gambar 1. Konsep Middleware (Corebitts, 2019)

Kombinasi tersebut merupakan alur hubungan antara logika pemrograman dengan basis data terdistribusi yang disebut *middleware*. Menurut Oracle, *middleware* adalah lapisan perangkat lunak yang terletak di antara sistem operasi dan aplikasi di setiap sisi jaringan komputer terdistribusi [7], sedangkan menurut technopedia *middleware* mirip dengan sistem operasi karena dapat mendukung program aplikasi lain, menyediakan interaksi terkontrol, mencegah gangguan antara komputasi dan memfasilitasi interaksi antara komputasi pada komputer yang berbeda melalui jaringan Layanan Komunikasi [9].

Dalam bahasan ini akan membangun perangkat lunak menggunakan RESTful API yang dapat mengakses basis data dan diolah menjadi keluaran berupa JSON sebagai lapisan *middleware* yang akan berinteraksi dan dapat dimanfaatkan oleh bahasa pemprograman C# berbasis Desktop dengan Sistem Operasi Windows dan Sistem Operasi Android dengan bahasa pemprograman Java sehingga kedua *platform* tersebut menjadi satu kesatuan yang utuh dalam mengolah data.

2. METODE PENELITIAN

Metode RAD (Rapid Application Development) merupakan metode pengembangan dari Waterfall atau metode tradisional pada SDLC yang digunakan untuk membangun simulasi *middleware***.**



Gambar 1. RAD

Perbedaan RAD dengan Waterfall berada pada tahap pembangunan perangkat lunak yang dilakukan secara terus menerus dan responsif, jika terjadi kesalahan maka langsung dilakukan perbaikan sehingga saat ingin memasuki tahap pengujian minim terjadi kesalahan. Adapun penerapan dari metode tersebut dijelaskan pada tabel dibawah ini:

Tabel 1.Alur Proses Pembangunan Perangkat Lunak

No.	Aktifitas	Hasil	Keterangan
1	Menganalisa kebutuhan sistem.	struktur basis data, bahasa pemprograman yang dibutuhkan	Bahasa pemrograman yang digunakan adalah Java dan C#
2	Kebutuhan integrasi dengan melakukan perbandingan host yang akan digunakan	Menggunakan host dengan domain	Bahasa pemprograman pada server menggunakan PHP dengan basis data MySQL
3	Perancangan basis data	Pembentukan tabel, view dan prosedur operasi: menambah, menghapus dan mengubah data	
4	Perancangan antar-muka	Pada kedua platform : Desktop dan Mobile	
5	Pengkodean Desktop	Menggunakan Microsoft Visual studio dan bahasa pemprograman C#	Jika masih terdapat *bugs* maka perlu dilakukan perbaikan
6	Pengkodean Android	Dengan bahasa pemprograman Java	Jika masih terdapat *bugs* maka perlu dilakukan perbaikan
7	Pengujian seluruh perangkat lunak	Android, Desktop dan Web	Bila terjadi kesalahan maka perlu melakukan revisi pada bagian yang mengalami *bugs* saja

3. HASIL DAN PEMBAHASAN

Tahap awal dalam membangun simulasi *middleware* adalah dengan mempersiapkan *host* dan basis data, mengingat bahwa platform akan menggunakan protokol HTTP, maka salah satu cara adalah menggunakan PHP secara native untuk berkomunikasi dengan basis data pada lapisan middleware, merespon *request* dari *client* dan sebagai penyedia keluaran berformat JSON, sebagai salah satu pendukung keamanan, protokol yang digunakan adalah HTTPS yang mengenkripsi setiap request yang dikirim dari *client*. Bentuk penerapan *request* akan dikirimkan melalui alamat URL yang selanjutnya akan di proses server. Berikut cara pengiriman perintah yang akan diproses server:

Memasukkan data

```
https://www.domainanda.com/peserta.php
?action=1&nama=sebastian&kampus=ABC&wa
=081522889966&phone=0218852369&email=s
ebastian@live.com
```

Memperbarui data

```
https://www.domainanda.com/peserta.php
?action=2&id=2&nama=sebastian&kampus=U
nindra&wa=081522889977&phone=021885236
9&email=sebastian@gmail.com
```

Menampilkan seluruh data

```
https://www.domainanda.com/peserta.php
?action=4
```

Menghapus data

```
https://www.domainanda.com/peserta.php
?action=3&id=2
```

Pencarian data

```
https://www.domainanda.com/peserta.php
?action=5&find=andre
```

Berdasarkan kode tersebut seluruh platform cukup mengikuti URL yang telah diberikan dan tak perlu melakukan pengaturan basis data, selanjutnya pemrogram perangkat lunak cukup melakukan permintaan dan penerimaan data saja. Adapun kode PHP yang digunakan sebagai peladen *client* terbagi menjadi dua kode, yang pertama kode berupa Class yang mengoperasikan basis data seperti yang diuraikan dibawah.

```
<?php
class Database{
    var $conn;
    var $host = "host";
    var $db = "nama_database";
    var $user = "nama_pengguna";
    var $pass = "kata_kunci";

    public function open(){
        $this->conn = mysqli_connect
($this->host,
        $this->user,
```

```
        $this->pass,
        $this->db)
                or
die(mysqli_error());
    }

    public function execute($sql){
        mysqli_query($this->conn,
$sql)
            or die
(mysqli_error($this->conn));
    }

    public function getAll($sql){
        $datas = Array();

        $query = mysqli_query($this-
>conn, $sql) or die
(mysqli_error($this->conn));

        while($row =
mysqli_fetch_assoc($query)){
            $datas[] = $row;
        }
        return $datas;
    }
}
?>
```

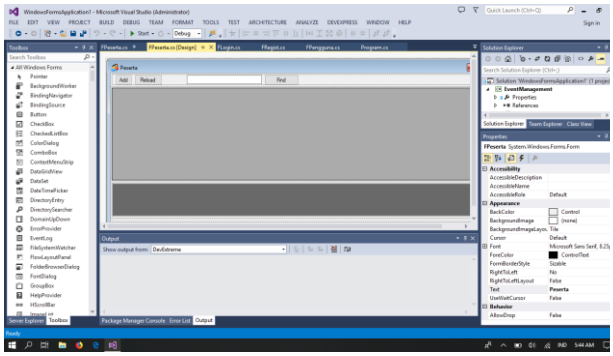
Kemudian yang kedua adalah kode yang berkerja sebagai peladen *client*.

```
<?php
include "database.php";

$action = null;

if (!isset($_REQUEST['action'])){
   $action = null;
}else{
   $action = $_REQUEST['action'];
}

$d = new Database();
$d->open();

$result = array();

if ($action == null){
    $myarray[] = array(
        "response" => "0"
    );

    $result =
array("result"=>$myarray);
    print json_encode($result);
}elseif($action == "1"){
    $sql = "insert into peserta (nama,
kampus, wa, phone, email) "
            . "values ('".
$_REQUEST['nama']."', '".
$_REQUEST['kampus']."', '".
$_REQUEST['wa']."', "
            . "'".
$_REQUEST['phone']."', '".
$_REQUEST['email']."')";
    $d->execute($sql);

    $myarray[] = array(
        "response" =>
$_REQUEST['nama']." inserted"
    );

    $result =
array("result"=>$myarray);
    print json_encode($result);
}elseif($action == "2"){
    $sql = "update peserta set nama =
'". $_REQUEST['nama']."', "
            . "kampus = '".
$_REQUEST['kampus']."', "
            . "wa = '".
$_REQUEST['wa']."', "
            . "phone = '".
$_REQUEST['phone']."', "
            . "email = '".
$_REQUEST['email']."' "
            . "where id = ".
$_REQUEST['id'];
    $d->execute($sql);

    $myarray[] = array(
        "response" =>
$_REQUEST['nama']." updated"
    );

    $result =
array("result"=>$myarray);
    print json_encode($result);
}elseif($action == "3"){
    $sql = "delete from peserta where
id = ". $_REQUEST['id'];
    $d->execute($sql);

    $myarray[] = array(
        "response" =>
$_REQUEST['id']." deleted"
    );

    $result =
array("result"=>$myarray);
    print json_encode($result);
}elseif($action == "4"){
    $sql = "select ID, NAMA, KAMPUS,
WA as WHATSAPP, PHONE, EMAIL, INPUT
from peserta";
    $result = array("result"=>$d-
>getAll($sql));
    print json_encode($result);
}elseif($action == "5"){
    $sql = "select ID, NAMA, KAMPUS,
WA as WHATSAPP, PHONE, EMAIL, INPUT
```

```
from peserta where nama like '%".
$_REQUEST['find'] ."%'"
            ." or kampus like '%".
$_REQUEST['find'] ."%'";
    $result = array("result"=>$d-
>getAll($sql));
    print json_encode($result);
}
?>
```

Pembangunan Perangkat Lunak Berbasis Desktop

Menggunakan Microsoft Visual Studio, dengan bahasa pemrograman C#. Walaupun Microsoft Visual Studio mampu membangun berbagai basis *platform*, namun kali ini digunakan hanya untuk desktop saja. Adapun bentuk antar-muka Microfost Visual Studio seperti tergambar dibawah.



Gambar 2.Aantar Muka Microsoft Visual Studio

Untuk menjalankan URL yang telah ditentukan pada bahasan diatas, maka proses akan dijalankan dalam beberapa kondisi:

1. Memuat Form dengan memanggil URL yang berfungsi menampilkan data pada Data grid:

```
private void FPeserta_Load(object
sender, EventArgs e){

showGrid("http://www.domainanda.com/pe
serta.php?action=4");
}
```

2. Menyimpan dan memperbarui data

```
if (dataGridView2.NewRowIndex == -1){

post("http://www.domainanda.com/pesert
a.php?action=2" +
            "&id=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[0].Value.ToString()) +
            "&nama=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[1].Value.ToString()) +
            "&kampus=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[2].Value.ToString()) +
            "&wa=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[3].Value.ToString()) +
            "&phone=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[4].Value.ToString()) +
            "&email=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[5].Value.ToString()));
} else {

post("http://www.domainanda.com/pesert
a.php?action=1" +
            "&nama=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[1].Value.ToString()) +
            "&kampus=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[2].Value.ToString()) +
            "&wa=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[3].Value.ToString()) +
            "&phone=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[4].Value.ToString()) +
            "&email=" +
HttpUtility.UrlEncode(dataGridView2.Ro
ws[i].Cells[5].Value.ToString()));
}
```

3. Menghapus data

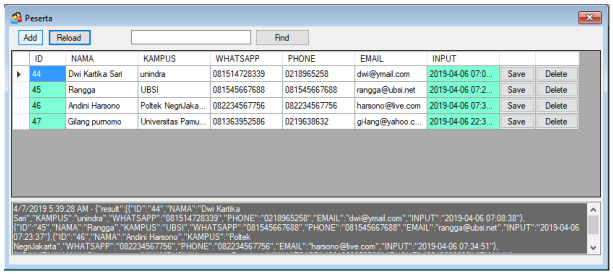
```
post("http://www.domainanda.com/pesert
a.php?action=3&id=" +
dataGridView2.Rows[i].Cells[0].Value.T
oString());
```

4. Mencari data

```
private void button1_Click(object
sender, EventArgs e){

showGrid("http://www.domainanda.com/pe
serta.php?action=5&find=" +
textBox3.Text);
}
```

Bentuk antar muka dalam menerapkan pola URL diatas digambarkan seperti

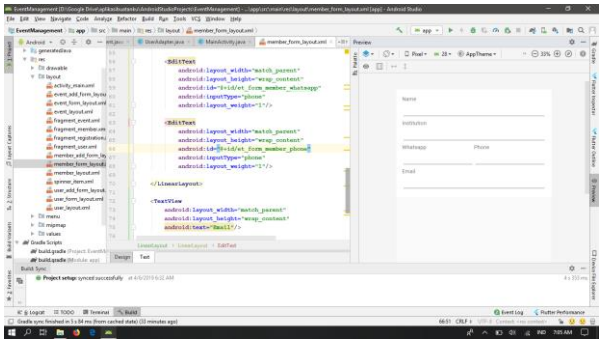
dibawah ini. Proses tambah data, simpan, hapus dan pencarian menjadi lebih sederhana dikarenakan tak adanya pengaturan basis data.



Gambar 3. Antar Muka Berbasis Desktop

Pembangunan Perangkat Lunak Berbasis Android

Dibangun menggunakan perangkat lunak Android Studio yang mendukung bahasa pemrograman Kotlin, Java dan Dart. Namun pada bahasan ini menggunakan Java dengan XML sebagai *layout.*



Gambar 4. Antar Muka Android Studio

Teknis pemanfaatan HTTP pada bahasan ini menggunakan *library* Volley untuk mengirim URL dan mengurai JSON sebagai hasil keluaran. Teknis pengoperasian URL tersebut akan diuraikan dibawah ini:

1. Memuat Form dengan memanggil URL yang berfungsi menampilkan data:

```
RequestQueue queue =
Volley.newRequestQueue(getContext());
String url =
"http://www.domainanda.com/peserta.php
?action=4";
```

```
JsonObjectRequest jsObjRequest = new
JsonObjectRequest(
   Request.Method.GET,
   url,
   null,
   new Response.Listener<JSONObject>()
{
      @Override
      public void
onResponse(JSONObject response) {
         Log.d("Events: ",
response.toString());
         }, new
Response.ErrorListener() {
            @Override
            public void
onErrorResponse(VolleyError error) {
              Log.d("Events: ",
error.toString());


Toast.makeText(getContext(),
                  error.toString(),

Toast.LENGTH_SHORT).show();
            }
        });
        queue.add(jsObjRequest);
}
```

2. Menyimpan data

```
try {
   url =
"http://www.domainanda.com/peserta.php
?action=1" +
         "&nama=" +
URLEncoder.encode(d_tv_name.getText().
toString(), "utf-8") +
         "&kampus=" +
URLEncoder.encode(d_tv_institution.get
Text().toString(), "utf-8") +
         "&wa=" +
URLEncoder.encode(d_tv_whatsapp.getTex
t().toString(), "utf-8") +
         "&phone=" +
URLEncoder.encode(d_tv_phone.getText()
.toString(), "utf-8") +
         "&email=" +
URLEncoder.encode(d_tv_email.getText()
.toString(), "utf-8");

   RequestQueue queue =
Volley.newRequestQueue(context);
   JsonObjectRequest jsObjRequest =
new JsonObjectRequest(

Request.Method.GET, url, null, new
Response.Listener<JSONObject>() {

      @Override
```

```
      public void
onResponse(JSONObject response) {
         Log.d("Save ",
response.toString());
      }, new Response.ErrorListener()
{
         @Override
         public void
onErrorResponse(VolleyError error) {
            Log.d("Events: ",
error.toString());
         }
     });

     queue.add(jsObjRequest);
```

3. Memperbarui data

```
try {
url =
"http://www.domainanda.com/peserta.php
?action=2" +
      "&id=" + id +
      "&nama=" +
URLEncoder.encode(d_tv_name.getText().
toString(), "utf-8") +
      "&kampus=" +
URLEncoder.encode(d_tv_institution.get
Text().toString(), "utf-8") +
      "&wa=" +
URLEncoder.encode(d_tv_whatsapp.getTex
t().toString(), "utf-8") +
      "&phone=" +
URLEncoder.encode(d_tv_phone.getText()
.toString(), "utf-8") +
      "&email=" +
URLEncoder.encode(d_tv_email.getText()
.toString(), "utf-8");

   RequestQueue queue =
Volley.newRequestQueue(context);
   JsonObjectRequest jsObjRequest =
new JsonObjectRequest(

Request.Method.GET, url, null, new
Response.Listener<JSONObject>() {

      @Override
      public void
onResponse(JSONObject response) {
         Log.d("Save ",
response.toString());
      }, new Response.ErrorListener()
{
         @Override
         public void
onErrorResponse(VolleyError error) {
            Log.d("Events: ",
error.toString());
         }
     });

     queue.add(jsObjRequest);
```

4. Menghapus data

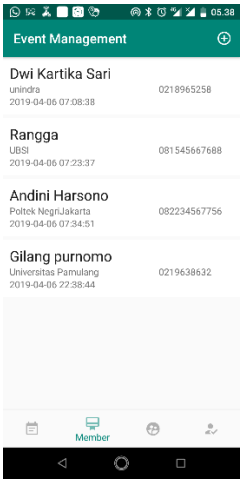
```
url =
"http://www.domainanda.com/peserta.php
?action=3&id=" + id;
RequestQueue queue =
Volley.newRequestQueue(context);
JsonObjectRequest jsObjRequest = new
JsonObjectRequest(

Request.Method.GET, url, null, new
Response.Listener<JSONObject>() {

   @Override
   public void onResponse(JSONObject
response) {
      Log.d("Delete: ",
response.toString());
   }, new Response.ErrorListener() {
      @Override
      public void
onErrorResponse(VolleyError error) {
         Log.d("Events: ",
error.toString());
      }
   });

   queue.add(jsObjRequest);
```

5. Mencari data

```
RequestQueue queue =
Volley.newRequestQueue(getContext());
String url =
"http://www.domainanda.com/peserta.php
?action=5&find=" +
ed_cari.getText().toString();

JsonObjectRequest jsObjRequest = new
JsonObjectRequest(
   Request.Method.GET,
   url,
   null,
   new Response.Listener<JSONObject>()
{
      @Override
      public void
onResponse(JSONObject response) {
         Log.d("Events: ",
response.toString());
         }, new
Response.ErrorListener() {
            @Override
            public void
onErrorResponse(VolleyError error) {
              Log.d("Events: ",
error.toString());


Toast.makeText(getContext(),
```

```
                    error.toString(),

Toast.LENGTH_SHORT).show();
            }
        });
        queue.add(jsObjRequest);
}
```

Dari rangkuman kode diatas terbentuklah antar muka seperti tergambar dibawah. Tanda tambah disisi atas berguna untuk menambahkan data, sedangkan untuk mengubah dan menghapus data cukup tekan data yang ingin diubah.



Gambar 5. Antar Muka pada Platform Android

4. SIMPULAN

Lapisan middleware yang berisi logika pemprograman dan operasi basis data mampu melayani kebutuhan platform Desktop berbahasa pemrograman C# dan platform bersistem operasi Android berbahasa pemprograman Java. Keduanya memiliki respon terhadap data secara langsung atau *real time* dengan perbedaan cara saat mengurai hasil keluaran berupa JSON.

Guna menambah kehandalan pada sisi middleware perlu melakukan perubahan kode PHP Native dengan menggunakan Framework Laravel atau NodeJS yang memiliki performa cepat dan keamanan yang baik.

DAFTAR PUSTAKA

[1] Amin Rulloh, D. E. (2017). Implementasi REST API pada Aplikasi Panduan Kepaskibraan. *Teknikom*, 85-89.

[2] Corebitts. (2019). *Middleware Services*. Diambil kembali dari Corebitts: https://www.corebitts.com/middleware-services/

[3] Deitel, P. D. (2014). *Android How To Program*. New Jersey: Prentice Hall.

[4] Dody Agung S, H. D. (2015). *Penerapan RESTful Web Service dan JSON pada Application Programming Interface (API) Sistem Informasi Perkembangan Ayam Broiler Berbasis Kemitraan.* Salatiga: Universitas Kristen Satya Wacana.

[5] Json. (2019). *Pengenalan JSON*. Diambil kembali dari Json.org: https://www.json.org/

[6] Magableh, B. (2019). Context Oriented Software Middleware. *arXiv*.

[7] Oracle. (2019). *Overview of Oracle Fusion Middleware*. Diambil kembali dari Oracle Help Center: https://docs.oracle.com/cd/E28280_01/core.1111/e10103/intro.htm

[8] Shilpi Taneja, A. G. (2015). A Mobile App Architecture for Student Information System. *International Journal Web Applications*, 56-63.

[9] techopedia. (2019). *techopedia*. Diambil kembali dari Middleware: https://www.techopedia.com/definition/450/middleware