MODUL 7

REpresentational State Transfer (REST)

7.1 Pengertian

REST adalah singkatan dari *REpresentational State Transfer*. *REST* merupakan gaya arsitektur untuk menyediakan standar antara sistem komputer di *web*, sehingga memudahkan sistem untuk berkomunikasi satu sama lain.

Menurut *Code Academy,* sistem yang sesuai dengan *REST*, sering disebut sistem *RESTful*, dicirikan oleh sifatnya yang *stateless* dan dapat memisahkan masalah *client* dan *server*.

Stateless artinya *server* tidak perlu mengetahui apa pun tentang status *client* dan sebaliknya. Sehingga, baik *server* maupun *client* dapat memahami pesan apapun yang diterima, bahkan tanpa melihat pesan sebelumnya.

Dalam gaya arsitektur *REST*, implementasi *client* dan implementasi *server* dapat dilakukan secara independen tanpa saling mengetahui satu sama lain. Ini berarti kode di sisi *client* dapat diubah kapan saja tanpa memengaruhi pengoperasian *server*, dan kode di sisi *server* dapat diubah tanpa memengaruhi operasi *client*.

REST menggunakan serangkaian metode permintaan standar termasuk *GET, POST, PUT, DELETE,* dan kemampuan *HTTP* lain yang ada.

Pada intinya, *REST* terdiri dari empat hal berikut

- Sumber daya yang direferensikan oleh pengenal global seperti *URI*.
- Representasi sumber daya, yaitu dokumen yang berisi informasi sumber daya yang dibutuhkan.
- Komponen yang berkomunikasi satu sama lain antara *client* dan *server*.
- *Interface* standar yang mengidentifikasi sumber daya, memungkinkan manipulasi sumber daya melalui representasi, termasuk pesan descripttif mandiri yang menjelaskan cara memproses permintaan dan respons, dan hypermedia yang diidentifikasi secara dinamis.

Menurut *REST API,* terdapat enam prinsip dasar dari *REST*.

1. *Client-server*

Prinsip dasar *REST* yang pertama adalah *client-server*, yaitu dengan memisahkan masalah user *interface* dari masalah penyimpanan data. Dengan memisahkan masalah ini, kita dapat meningkatkan portabilitas antarmuka pengguna di berbagai platform dan meningkatkan skalabilitas dengan menyederhanakan komponen *server*.

2. *Stateless*

Prinsip dasar *REST* yang kedua adalah *stateless*. Prinsip ini memudahkan akses informasi dari *client* ke *server* maupun sebaliknya. Setiap permintaan dari *client* ke *server* harus berisi semua informasi yang diperlukan untuk memahami permintaan tersebut, dan tidak dapat memanfaatkan konteks yang disimpan di *server*. Maka dari itu, status sesi disimpan sepenuhnya pada *client*.

3. *Cacheable*

Batasan *cache* mengharuskan data dalam respons terhadap permintaan diberi label secara implisit atau eksplisit sebagai dapat disimpan dalam *cache* atau tidak dapat disimpan dalam *cache*. Jika respons tersebut dapat disimpan dalam *cache*, *cache client* diberikan hak untuk menggunakan kembali data respons tersebut untuk permintaan yang setara nanti.

4. *Interface* yang seragam

Interface yang seragam dibuat dengan menerapkan prinsip umum rekayasa perangkat lunak ke antarmuka komponen, arsitektur sistem secara keseluruhan disederhanakan dan visibilitas interaksi ditingkatkan. Untuk mendapatkan *interface* yang seragam, beberapa batasan arsitektur diperlukan untuk memandu perilaku komponen.

REST didefinisikan oleh empat batasan *interface*.

- Identifikasi sumber daya
- Manipulasi sumber daya melalui representasi
- Pesan descripttif diri
- Hypermedia sebagai mesin status aplikasi

5. Sistem yang berlapis

Prinsip dasar *REST* selanjutnyya adalah sistem yang berlapis. Gaya sistem berlapis memungkinkan arsitektur untuk terdiri dari lapisan hierarki dengan membatasi perilaku komponen sedemikian rupa. Sehingga, setiap komponen tidak dapat “melihat” di luar lapisan langsung yang berinteraksi dengan mereka.

6. *Code on demand* yang opsional

REST memungkinkan fungsionalitas *client* diperluas dengan mengunduh dan menjalankan kode dalam bentuk *applet* atau *script*. Ini menyederhanakan *client* dengan mengurangi jumlah fitur yang diperlukan untuk diterapkan sebelumnya.

7.2 Aplikasi

Aplikasi Toko menggunakan *RESTful* format *XML* - *CRUD* sederhana, dimana **data akan dikirim dari *Client* (*Windows* dengan *IP Address* 192.168.56.1) ke *Server* (*Debian Server* dengan *IP Address* 192.168.56.xx) dan data hanya disimpan pada *database Debian Server***. *Database* tetap menggunakan *database* "toko" dan tabel "barang" pada *Debian Server*. Langkah selanjutnya adalah sebagai berikut:

1. Buat *folder* ***/opt/lampp/htdocs/restful-xml-toko/server/*** pada *Debian Server* lalu buat *file* ***Database.php*** dan ***server.php***.
2. Buat juga *folder* ***xampp\htdocs\restful-xml-toko\client*** pada *Windows* lalu buat *file* ***Client.php, proses.php*** dan ***index.php***.
3. Akses *phpinfo* pada *Server (Debian Server)* serta pastikan *PHP extension* ***xml, PDO*** dan ***pdo_mysql*** sudah aktif.
4. Akses *phpinfo* pada *Client (Windows)* serta pastikan *PHP extension* ***xml*** sudah aktif.

7.2.1 *Source code Database.php* di *Debian Server*

Database.php

```php
<?php
class Database
{   private $host="localhost";
    private $dbname="toko";
    private $user="root";
    private $password="root";
    private $port="3306";
    private $conn;

    // function yang pertama kali di-load saat class dipanggil
    public function __construct()
    {   // koneksi database
        try
        {   $this->conn = new PDO("mysql:host=$this->host;port=$this->port;dbname=$this->dbname;charset=utf8",$this->user,$this->password);

        } catch (PDOException $e)
        {   echo "Koneksi gagal";
        }
    }

    public function tampil_semua_data()
    {   $query = $this->conn->prepare("select id_barang, nama_barang from barang order by id_barang");
        $query->execute();
        // mengambil banyak data dengan fetchAll
        $data = $query->fetchAll(PDO::FETCH_ASSOC);
        // mengembalikan data
        return $data;
        // hapus variable dari memory
        $query->closeCursor();
        unset($data);
    }

    public function tampil_data($id_barang)
    {   $query = $this->conn->prepare("select id_barang,nama_barang from barang where id_barang=?");
        $query->execute(array($id_barang));
        // mengambil satu data dengan fetch
        $data = $query->fetch(PDO::FETCH_ASSOC);
        return $data;
        $query->closeCursor();
        unset($id_barang,$data);
    }

    public function tambah_data($data)
    {   $query = $this->conn->prepare("insert ignore into barang (id_barang,nama_barang) values (?,?)");
        $query->execute(array($data['id_barang'],$data['nama_barang']));
        $query->closeCursor();
        unset($data);
    }

    public function ubah_data($data)
    {   $query = $this->conn->prepare("update barang set nama_barang=? where id_barang=?");
        $query->execute(array($data['nama_barang'],$data['id_barang']));
        $query->closeCursor();
        unset($data);
    }

    public function hapus_data($id_barang)
    {   $query = $this->conn->prepare("delete from barang where id_barang=?");
        $query->execute(array($id_barang));
        $query->closeCursor();
        unset($id_barang);
    }
}
?>
```

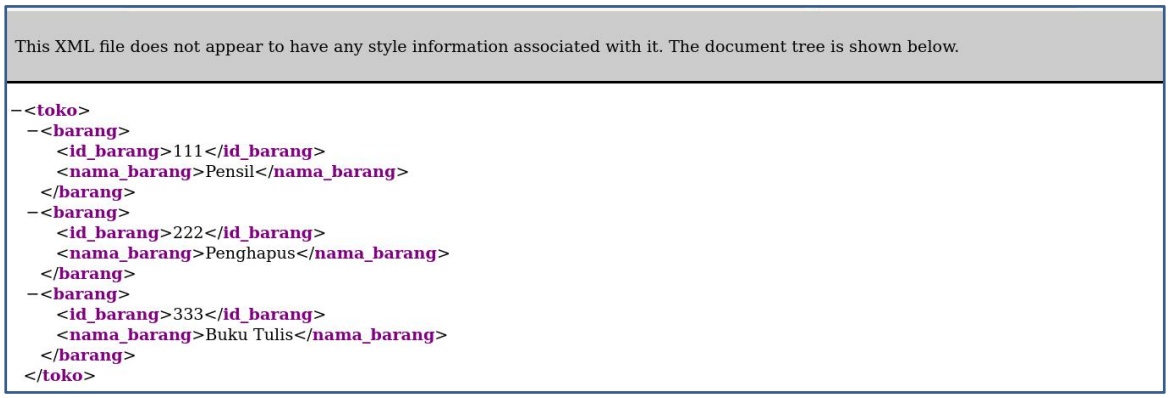
Gambar 1. *Source code Database.php* di *Debian Server*

7.2.2 *Source code* ***server.php*** di *Debian Server*

```php
<?php
error_reporting(1); // error ditampilkan
header('Content-Type: text/xml; charset=UTF-8');
include "Database.php";
// buat objek baru dari class Database
$abc = new Database();

// function untuk menghapus selain huruf dan angka
function filter($data)
{   $data = preg_replace('/[^a-zA-Z0-9]/', '', $data);
    return $data;
    unset($data);
}

if ($_SERVER['REQUEST_METHOD'] == 'POST')
{   $input = file_get_contents("php://input");
    $data = simplexml_load_string($input);
    $aksi = $data->barang->aksi;
    $id_barang = $data->barang->id_barang;
    $nama_barang = $data->barang->nama_barang;

    if ($aksi == 'tambah')
    {   $data2=array('id_barang' => $id_barang,
                     'nama_barang' => $nama_barang
                    );
        $abc->tambah_data($data2);
    } elseif ($aksi == 'ubah')
    {   $data2=array('id_barang' => $id_barang,
                     'nama_barang' => $nama_barang
                    );
        $abc->ubah_data($data2);
    } elseif ($aksi == 'hapus')
    {   $abc->hapus_data($id_barang);
    }
    // hapus variable dari memory
    unset($input,$data,$data2,$id_barang,$nama_barang,$aksi,$abc);
} elseif ($_SERVER['REQUEST_METHOD'] == 'GET')
{   if ( ($_GET['aksi']=='tampil') and (isset($_GET['id_barang'])) )
    {   $id_barang = filter($_GET['id_barang']);
        $data=$abc->tampil_data($id_barang);
        $xml = "<toko>";
            $xml .= "<barang>";
                $xml .= "<id_barang>".$data['id_barang']."</id_barang>";
                $xml .= "<nama_barang>".$data['nama_barang']."</nama_barang>";
            $xml.="</barang>";
        $xml.="</toko>";
        echo $xml;
    } else  //menampilkan semua data
    {   $data = $abc->tampil_semua_data();
        $xml = "<toko>";
        foreach($data as $a)
        {   $xml .= "<barang>";
            foreach($a as $kolom => $value)
            {   $xml .= "<$kolom>$value</$kolom>";
                // atau menggunakan
                // $xml .= "<$kolom><![CDATA[$value]]></$kolom>";
            }
            $xml.="</barang>";
        }
        $xml.="</toko>";
        echo $xml;
    }
    unset($id_barang,$data,$xml);
}
?>
```

Gambar 2. *Source code server.php* di *Debian Server*

Akses *RESTful* menggunakan *web browser* http://192.168.56.xx/restful-xml-toko/server/server.php



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
−<toko>
  −<barang>
     <id_barang>111</id_barang>
     <nama_barang>Pensil</nama_barang>
   </barang>
  −<barang>
     <id_barang>222</id_barang>
     <nama_barang>Penghapus</nama_barang>
   </barang>
  −<barang>
     <id_barang>333</id_barang>
     <nama_barang>Buku Tulis</nama_barang>
   </barang>
 </toko>
```

Gambar 3. *RESTful format xml* di *Debian Server*

7.2.3 *Source code Client.php* di *Windows*

```php
<?php
error_reporting(1); // error ditampilkan
class Client
{   private $url;

    // function yang pertama kali di-load saat class dipanggil
    public function __construct($url)
    {   $this->url=$url;
        unset($url);
    }

    // function untuk menghapus selain huruf dan angka
    public function filter($data)
    {   $data = preg_replace('/[^a-zA-Z0-9]/','',$data);
        return $data;
        unset($data);
    }

    public function tampil_semua_data()
    {   $client = curl_init($this->url);
        curl_setopt($client,CURLOPT_RETURNTRANSFER,1);
        $response = curl_exec($client);
        curl_close($client);
        $data = simplexml_load_string($response);
        // mengembalikan data
        return $data;
        // menghapus variabel dari memory
        unset($data,$client,$response);
    }

    public function tampil_data($id_barang)
    {   $id_barang = $this->filter($id_barang);
        $client = curl_init($this->url."?aksi=tampil&id_barang=".$id_barang);
        curl_setopt($client,CURLOPT_RETURNTRANSFER,1);
        $response = curl_exec($client);
        curl_close($client);
        $data = simplexml_load_string($response);
        return $data;
        unset($id_barang,$client,$response,$data);
    }

    public function tambah_data($data)
    {   $data="<toko>
                  <barang>
                    <id_barang>".$data['id_barang']."</id_barang>
                    <nama_barang>".$data['nama_barang']."</nama_barang>
                    <aksi>".$data['aksi']."</aksi>
                  </barang>
                </toko>";
        $c = curl_init();
        curl_setopt($c,CURLOPT_URL,$this->url);
        curl_setopt($c,CURLOPT_RETURNTRANSFER,true);
        curl_setopt($c,CURLOPT_POST,true);
        curl_setopt($c,CURLOPT_POSTFIELDS,$data);
        $response = curl_exec($c);
        curl_close($c);
        unset($data,$c,$response);
    }

    public function ubah_data($data)
    {   $data="<toko>
                  <barang>
                    <id_barang>".$data['id_barang']."</id_barang>
                    <nama_barang>".$data['nama_barang']."</nama_barang>
                    <aksi>".$data['aksi']."</aksi>
                  </barang>
                </toko>";
        $c = curl_init();
        curl_setopt($c,CURLOPT_URL,$this->url);
        curl_setopt($c,CURLOPT_RETURNTRANSFER,true);
        curl_setopt($c,CURLOPT_POST,true);
        curl_setopt($c,CURLOPT_POSTFIELDS,$data);
        $response = curl_exec($c);
        curl_close($c);
```

```
75          unset($data,$c,$response);
76      }
77
78      public function hapus_data($id_barang)
79      {   $id_barang = $this->filter($id_barang);
80          $data = "<toko>
81                      <barang>
82                          <id_barang>".$id_barang."</id_barang>
83                          <aksi>hapus</aksi>
84                      </barang>
85                  </toko>";
86          $c = curl_init();
87          curl_setopt($c,CURLOPT_URL,$this->url);
88          curl_setopt($c,CURLOPT_RETURNTRANSFER,true);
89          curl_setopt($c,CURLOPT_POST,true);
90          curl_setopt($c,CURLOPT_POSTFIELDS,$data);
91          $response = curl_exec($c);
92          curl_close($c);
93          unset($id_barang,$data,$c,$response);
94      }
95
96      // function yang terakhir kali di-load saat class dipanggil
97      public function __destruct()
98      {   // hapus variable dari memory
99          unset($this->options,$this->api);
100     }
101 }
102
103 $url = 'http://192.168.56.2/restful-xml-toko/server/server.php';
104 // buat objek baru dari class Client
105 $abc = new Client($url);
106 ?>
```

Gambar 4. *Source code Client.php* di *Windows*

7.2.4 *Source code proses.php* di *Windows*

```
1  <?php
2  include "Client.php";
3
4  if ($_POST['aksi']=='tambah')
5  {   $data = array("id_barang"=>$_POST['id_barang'],
6                    "nama_barang"=>$_POST['nama_barang'],
7                    "aksi"=>$_POST['aksi']);
8      $abc->tambah_data($data);
9      header('location:index.php?page=daftar-data');
10 } else if ($_POST['aksi']=='ubah')
11 {   $data = array("id_barang"=>$_POST['id_barang'],
12                   "nama_barang"=>$_POST['nama_barang'],
13                   "aksi"=>$_POST['aksi']);
14     $abc->ubah_data($data);
15     header('location:index.php?page=daftar-data');
16 } else if ($_GET['aksi']=='hapus')
17 {   $abc->hapus_data($_GET['id_barang']);
18     header('location:index.php?page=daftar-data');
19 }
20 unset($abc,$data);
21 ?>
```

Gambar 5. *Source code proses.php* di *Windows*

7.2.5 *Source code index.php* di *Windows*

```php
<?php
include "Client.php";
?>
<!doctype html>
<html>
<head>
    <title></title>
</head>
<body>
<a href="?page=home">Home</a> | <a href="?page=tambah">Tambah Data</a> | <a href="?page=daftar-data">Data Server</a>
<br/><br/>

<fieldset>
<? if ($_GET['page']=='tambah') { ?>
<legend>Tambah Data</legend>
    <form name="form" method="POST" action="proses.php">
        <input type="hidden" name="aksi" value="tambah"/>
        <label>ID Barang</label>
        <input type="text" name="id_barang"/>
        <br/>
        <label>Nama Barang</label>
        <input type="text" name="nama_barang"/>
        <br/>
        <button type="submit" name="simpan">Simpan</button>
    </form>

<? } elseif ($_GET['page']=='ubah') {
    $r = $abc->tampil_data($_GET['id_barang']);
?>
<legend>Ubah Data</legend>
    <form name="form" method="post" action="proses.php">
        <input type="hidden" name="aksi" value="ubah"/>
        <input type="hidden" name="id_barang" value="<?=$r->barang->id_barang?>" />
        <label>ID Barang</label>
        <input type="text" value="<?=$r->barang->id_barang?>" disabled>
        <br/>
        <label>Nama Barang</label>
        <input type="text" name="nama_barang" value="<?=$r->barang->nama_barang?>">
        <br/>
        <button type="submit" name="ubah">Ubah</button>
    </form>

<?  unset($r);
    } else if ($_GET['page']=='daftar-data') {
?>
<legend>Daftar Data Server</legend>
    <table border="1">
    <tr><th width='5%'>No</th>
        <th width='10%'>ID Barang</th>
        <th width='75%'>Nama</th>
        <th width='5%' colspan="2">Aksi</th>
    </tr>
    <?  $no = 1;
        $data_array = $abc->tampil_semua_data();
        foreach ($data_array as $r) {
    ?>  <tr><td><?=$no?></td>
            <td><?=$r->id_barang?></td>
            <td><?=$r->nama_barang?></td>
            <td><a href="?page=ubah&id_barang=<?=$r->id_barang?>">Ubah</a></td>
            <td><a href="proses.php?aksi=hapus&id_barang=<?=$r->id_barang?>" onclick="return confirm('Apakah Anda ingin menghapus data ini?')">Hapus</a></td>
        </tr>
    <?  $no++;
        }
        unset($data_array,$r,$no);
    ?>
    </table>

<? } else { ?>
<legend>Home</legend>
    Aplikasi sederhana ini menggunakan RESTful dengan format data XML (Extensible Markup Language).
</fieldset>
<? } ?>
</body>
</html>
```

Gambar 6. *Source code index.php* di *Windows*

Akses melalui *web browser* ***http://localhost/restful-xml-toko/client/index.php*** di *Windows*. Coba jalankan aplikasi sederhana ini dengan menambah, menampilkan, mengubah dan menghapus datanya. Data akan dikirim dari *Windows* ke *Debian Server* dan data hanya disimpan di *database Debian Server*. Data yang tersimpan tersebut akan ditampilkan ke *Client di Windows.*

Gambar 7. Laman *Client*