Perancangan Arsitektur Aplikasi Tersebar Berbasis File Text
Architectural Design of Distributed Applications Based on Text Files

Joko Triyono[1], Erfanti Fatkhiyah[2], Muhammad Edi Wicaksono[3]
[1,2,3]Program Studi Informatika, Fakultas Teknologi Industri, Institut Sains dan Teknologi AKPRIND Yogyakarta,
Jl. Kalisahak 28 Komplek Balapan Yogyakarta, Indonesia
Email: jack@akprind.ac.id[1], erfanti@akprind.ac.id[2], muhammadediwicaksono@gmail.com[3]

ABSTRAK

Covid-19 membuat semuanya berubah. Begitu juga dalam pengembangan aplikasi-aplikasi mulai dari *bisnis online, ojek online, dropshipper, web konten*, bidang pendidikan dan pelayanan kepada masyarakat. Pelayanan teknologi informasi semakin hari semakin banyak dan komplek sehingga dibutuhkan solusi baru untuk memperbaiki mutu layanan tersebut. Beberapa jenis data dalam sebuah sistem informasi memiliki sifat yang berbeda-beda, terkait dengan usia perubahan data, maka data akan terbagi menjadi beberapa bagian usia perubahan. Sehingga akan ada data yang harus selalu online, ada data yang perubahannya hanya terjadi setiap periode tertentu. Karena semakin meningkatnya kebutuhan akan informasi maka dibutuhkan aplikasi-aplikasi yang tersebar untuk meningkatkan ketersediaan dan kemudahan informasi. Penggunaan *web service, json* dan kombinasi dengan file teks sangat dimungkinkan terciptanya aplikasi tersebar yang tidak selalu harus terhubung ke *server* atau *internet*, dengan melakukan distribusi data menjadi file text di masing-masing aplikasi mengikuti pola tertentu. Penelitian ini bertujuan untuk melakukan pemodelan dalam melakukan perancangan arsitektur aplikasi tersebar berbasis file text yang berisi data dengan format *JSON* yang diperoleh melalui *web-service* dari server induk dengan memaksimalkan teknologi informasi yang ada, sehingga akan diperoleh sebuah aplikasi tersebar yang tetap memenuhi kebutuhan informasi tetapi juga terjadi penghematan penggunaan data internet dan juga kecepatan data saat disajikan.

Kata Kunci: *JSON;* Aplikasi Tersebar; Data Tersebar; File Text; *Web-Service*

ABSTRACT

Covid-19 has changed everything. Likewise in the development of applications ranging from online business, online motorcycle taxis, dropshippers, web content, education and services to the community. Information technology services are getting more and more complex, so new solutions are needed to improve the quality of these services. Several types of data in an information system have different properties, related to the age of data change, then the data will be divided into several parts of the age of change. So there will be data that must always be online, there will be data whose changes only occur every certain period. Due to the increasing need for information, it is necessary to spread applications to increase the availability and ease of information. The use of web services, json and combinations with text files makes it possible to create distributed applications that do not always have to be connected to a server or the internet, by distributing data into text files in each application following a certain pattern. This study aims to perform modeling in designing distributed application architecture based on text files containing data in JSON format obtained via web-service from the main server by maximizing existing information technology, so that a distributed application will be obtained that still meets information needs but also there is a savings in internet data usage and data speed when served.

Keywords: *JSON; Scattered Applications; Scattered Data; Text Files; Web-Service*

1. PENDAHULUAN

Saat pandemi covid-19 terjadi lonjakan yang luar biasa dalam penggunaan teknologi informasi, yang sebelumnya belum begitu melirik atau masih setengah-setengah menggunakan teknologi informasi dengan terpaksa mau tidak mau harus menggunakan teknologi informasi. Begitu juga dalam pengembangan aplikasi-aplikasi berbasis online dalam semua bidang kegiatan menjadi marak. Mulai dari *bisnis online, ojek online, dropshipper, web konten*, bidang pendidikan dan pelayanan kepada masyarakat.

Dengan tingkat kepadatan transaksi yang terjadi akan menyebabkan masalah baru antara lain *respon server* yang melambat, *traffic* jaringan yang tinggi sehingga jalur komunikasi menjadi lambat juga. Padahal di sisi lain kebutuhan masyarakat akan pelayanan teknologi informasi semakin hari semakin banyak dan komplek sehingga dibutuhkan solusi baru untuk memperbaiki mutu layanan tersebut.

Beberapa jenis data dalam sebuah sistem informasi memiliki sifat yang berbeda-beda, terkait dengan usia perubahan data, maka data akan terbagi menjadi beberapa bagian usia perubahan. Sehingga akan ada data yang harus selalu online, ada data yang perubahannya hanya terjadi setiap periode tertentu.

Banyak arsitektur teknologi informasi yang bisa diterapkan untuk mengatasi masalah-masalah diatas, mulai dari penggunaan *cloud computing, distributed data, distributed aplikasi*. Karena semakin meningkatnya kebutuhan akan informasi maka dibutuhkan aplikasi-aplikasi yang tersebar untuk meningkatkan ketersediaan dan kemudahan informasi.

Dengan memanfaatkan web *service, json* dan kombinasi dengan file teks sangat dimungkinkan terciptanya aplikasi tersebar yang tidak selalu harus terhubung ke *server* atau *internet*, dengan melakukan distribusi data menjadi file text di masing-masing aplikasi mengikuti pola tertentu, sedangkan untuk mendapatkan data yang terbaru atau *update* data hanya membutuhkan koneksi di waktu-waktu tertentu sehingga akan mengurangi kebutuhan dan atau penggunaan *bandwidth internet*.

Penelitian ini bertujuan untuk melakukan pemodelan dalam melakukan perancangan arsitektur aplikasi tersebar berbasis file text yang berisi data dengan format *JSON* yang diperoleh melalui *web-service* dari *server* induk dengan memaksimalkan teknologi informasi yang ada, sehingga akan diperoleh sebuah aplikasi tersebar yang tetap memenuhi kebutuhan informasi tetapi juga terjadi penghematan penggunaan data *internet* dan juga kecepatan data saat disajikan.

2. TINJAUAN PUSTAKA

Dalam sebuah penelitian oleh (Triyono & Haryani, Membangun Aplikasi Front End Web Dosen Berbasis Web APPML (*Application Modeling Language*), 2017) disimpulkan bahwa walaupun sistem informasi telah online selama 24jam dan bisa diakses dari mana saja selama ada koneksi internet tetapi akan menjadi tidak efektif dan efisien jika harus mengunjungi masing-masing sistem informasi yang ada untuk mendapatkan informasi yang dibutuhkan. APPML (*Application Modeling Language*) sebagai sebuah metode atau teknik pemrograman berbasis HTML5 memiliki kemampuan untuk mengambil data dari sumber data lain berupa data *JSON* statis maupun *JSON* yang diperoleh dari sebuah *service* dinamis aplikasi *server side*. Penelitian ini menghasilkan metode atau teknik dalam membangun *front end web* pribadi dosen dengan melakukan koleksi data dari berbagai sumber sistem informasi yang relevan, dengan menanamkan *service server* dan *service client* sebagai penghubung antar sistem. Sehingga diperoleh sebuah penyajian informasi pada sebuah web pribadi dosen tanpa harus mengunjungi sistem informasi yang ada.

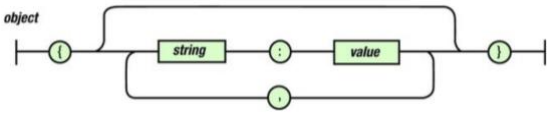
Juga penelitian tentang aplikasi *web front end* dengan teknik *JSON* (Triyono, Haryani, & Farid AS, Membangun Aplikasi *Web Front End* dengan Teknik *JSON (Java Script Object Notation*), 2018) disimpulkan bahwa JSON (*Java Script Object Notation*) merupakan salah satu teknik pengelolaan data yang akan merubah paradigma DBMS (*Database Management Systems*) dalam hal perancangan dan implementasi. Begitu juga dalam hal kecepatan proses, *JSON* menawarkan kecepatan yang sangat menggiurkan, dikombinasikan dengan local storage pada *HTML5*. Tentunya perkembangan tersebut akan juga berpengaruh terhadap kebutuhan akan informasi sehingga penambahan modul dan atau fasilitas-fasilitas dalam sistem informasi tentunya harus bisa dilakukan tanpa

mengganggu keberadaan sistem informasi yang saat ini running. Sistem modular menjadi hal yang perlu diperhatikan dalam pengembangan sistem informasi, dengan sistem modular menjadikan pengembang bisa bekerja secara paralel ditempat yang berbeda-beda pada saat yang bersamaan tanpa harus saling menunggu, pemakaian *database* secara bersama-sama sudah bukan masalah untuk saat ini. Hasil dari penelitian ini adalah sebuah teknik dalam membangun web front end dengan menggunakan teknik *JSON* sebagai penyedia data dan diimplementasikan bersama *HTML5* dan PHP, dengan menanamkan pada *server side with database dan Server side on Client* sebagai penghubung antar sistem. Sehingga diperoleh sebuah penyajian informasi pada sebuah aplikasi web di bagian client tanpa melakukan koneksi ke *database* secara langsung. *Client* dibangun selain menggunakan yang berbasis PHP/Web juga bisa menggunakan aplikasi berbasis Android

Dalam penelitian yang berbeda (Triyono 2015) di peroleh hasil tentang pengembangan sebuah *prototype* sistem informasi yang di kombinasikan dengan jejaring sosial *twitter*, dimana jejaring *twitter* digunakan petani untuk melaporkan semua kegiatannya ke sistem informasi, dengan menggunakan fasilitas *APIs* (*Aplication Programming Language*) maka informasi yang masuk akan di kirimkan ke sistem informasi dengan menggunakan *account* dari *twitter* pengirim. Dengan metode ini secara teknologi dan biaya petani tidak mengalami kesulitan dalam melaporkan kegiatannya, sedangkan dari sisi investor akan bisa melihat perkembangan investasinya. Tentang kepuasan pengguna *website* (Haryani 2016) disimpulkan bahwa dari hasil uji hipotesis menunjukkan bahwa dimensi *citizen support, content and appreance of information, reability, functionality of the interaction environment, trust, dan dimensi ease of use* berpengaruh terhadap kualitas layanan *e-government*. Kualitas layanan *e-government* berpengaruh terhadap kepuasan pengguna *website*, kualitas layanan *e-government* berpengaruh terhadap intensitas pengguna *website* dan kepuasan pengguna *website* berpengaruh terhadap intensitas penggunaan *website* Pemerintah Kota Yogyakarta. Dari beberapa penelitian yang telah ada di dapat disimpulkan bahwa metode pengiriman informasi tidak hanya menggunakan sebuah *WebSite*, tetapi bisa menggunakan berbagai aplikasi. *WebSite* akan digunakan sebagai tempat menampung informasi utama, sedangkan aplikasi lain digunakan sebagai tempat menyampaikan atau *notifikasi* ke user terdaftar maupun broadcast ke calon user.

Dalam sebuah buku dengan judul *JSON at Work* (Marrs 2017) *JavaScript Object Notation (JSON)* telah menjadi standar *de facto* untuk antarmuka *RESTful*, tetapi kelompok yang tidak banyak diketahui yaitu standar, alat, dan teknologi yang tersedia dapat mulai digunakan oleh perancang dan pengembang saat ini untuk membangun aplikasi yang dirancang dengan baik. *JSON* lebih dari sekadar pengganti sederhana untuk XML ketika anda melakukan panggilan *AJAX*. Ini menjadi tulang punggung dari pertukaran data yang serius melalui *internet*. Standar yang utama dan praktik terbaik dapat digunakan untuk memanfaatkan kemampuan dan keinginan menggunakan *JSON* untuk membangun aplikasi yang benar-benar elegan, berguna, dan efisien.

Dalam sebuah buku dengan judul *Beginning JSON Learn and Preferred Data Format of The Web* (Smith 2015) tentang Struktur Komposit JSON di katakan bahwa karena asal-usul JSON berasal dari standardisasi *ECMAScript*, implementasi dari dua struktur diwakili dalam bentuk objek dan larik. *Crockford* menguraikan dua representasi struktural *JSON* melalui serangkaian diagram sintaks. Karena saya yakin Anda akan setuju, diagram ini menyerupai rel kereta api dari pandangan mata burung dan dengan demikian juga disebut sebagai diagram rel kereta api. Gambar 1 mengilustrasikan representasi gramatikal untuk koleksi pasangan string / nilai.



Gambar 1. Sintak diagram dari koleksi pasangan string / nilai

Gambar 1 tentang Sintak diagram menguraikan, koleksi dimulai dengan penggunaan brace pembukaan (**{**), dan diakhiri dengan penggunaan penjepit penutup (**}**). Konten koleksi dapat terdiri dari salah satu dari tiga kemungkinan jalur yang ditentukan berikut:

- Jalur atas menggambarkan bahwa koleksi dapat tetap tanpa pasangan string / nilai apa pun.
- Jalan tengah menggambarkan bahwa koleksi kami dapat berupa pasangan string / nilai tunggal.
- Jalur bawah mengilustrasikan bahwa setelah sepasang string / nilai tunggal diberikan, koleksi tidak perlu berakhir, melainkan, memungkinkan untuk sejumlah string / pasangan nilai, sebelum mencapai akhir. Setiap pasangan string / nilai yang dimiliki oleh koleksi harus dipisahkan atau dipisahkan satu sama lain dengan cara koma (,).

Model mendeskripsikan aplikasi, Model tidak memperhatikan presentasi atau *user interface* (*UI*), model diimplementasikan dengan *JSON*.

```
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```

JSON dalam PHP

PHP memiliki beberapa fungsi built-in untuk menangani JSON. Objek dalam PHP dapat diubah menjadi JSON dengan menggunakan fungsi PHP *json_encode ()*:

```
<?php
$myObj->name =  "John";
$myObj->age = 30;
$myObj->city= "New York";
$myJSON=json_encode($myObj);
echo $myJSON;
?>
```

Script diatas akan menghasilkan output sebagai berikut:

```
{"name":"John","age":30,"city":"New York"}
```

Database PHP

PHP adalah bahasa pemrograman sisi server, dan harus digunakan untuk operasi yang hanya dapat dilakukan oleh server, seperti mengakses database. Bayangkan Anda memiliki database di server, yang berisi pelanggan, produk, dan pemasok. Anda ingin mengajukan permintaan ke server tempat Anda meminta 10 rekaman pertama di tabel "pelanggan":

Script PHP View

```
<!DOCTYPE html>
<html> <body>
<h2>Get data as JSON from a PHP file on the
server.</h2>
<p>The JSON received from the PHP file:</p>
<p id="demo"></p> <script>
var obj, dbParam, xmlhttp;
obj = { "table":"customers", "limit":10 };
dbParam = JSON.stringify(obj);
xmlhttp = new XMLHttpRequest();
xmlhttp.onreadystatechange = function() {
   if (this.readyState == 4 && this.status ==
200) {

document.getElementById("demo").innerHTML
= this.responseText;
   } };
xmlhttp.open("GET", "json_demo_db.php?x=" +
dbParam, true);
xmlhttp.send();
</script> </body> </html>
```

Script PHP Database

```
<?php
header("Content-Type: application/json;
charset=UTF-8");
$obj = json_decode($_GET["x"], false);
$conn = new mysqli("myServer", "myUser",
"myPassword", "Northwind");
$result = $conn->query("SELECT name FROM
".$obj->table." LIMIT ".$obj->limit);
$outp = array();
$outp = $result->fetch_all(MYSQLI_ASSOC);
echo json_encode($outp);
?>
```

Output dari aplikasi diatas

```
Get data as JSON from a PHP file on the
server.
The JSON received from the PHP file:
[{"name":"Alfreds Futterkiste"},{"name":"Ana
Trujillo Emparedados y
helados"},{"name":"Antonio Moreno
Taqueria"},{"name":"Around the
Horn"},{"name":"Berglunds
snabbkop"},{"name":"Blauer See
Delikatessen"},{"name":"Blondel pere et
fils"},{"name":"Bolido Comidas
preparadas"},{"name":"Bon
app'"},{"name":"Bottom-Dollar Marketse"}]
```

File Text

File Text adalah sebuah file ASCII yang dalam penelitian ini akan digunakan untuk menampung data dalam format JSON yang diperoleh dari service dan disimpan di local site aplikasi, sehingga saat akan menggunakan data

tersebut maka aplikasi cukup mengambil data tersebut dengan akses local.

Membuat File Text

Membuat file text dalam PHP MANUAL dideskripsikan tentang *fwrite*() (*PHP:fwrite-manual, 2021*) adalah menulis/merekam konten *string* ke aliran *file* yang ditunjukkan oleh pegangan.

fwrite (resource $handle , string $string , int $length = ?) : int

Parameter:

Handle : Sumber daya penunjuk sistem *file* yang biasanya dibuat menggunakan *fopen* ().

String : String yang akan ditulis.

Length : Jika argumen panjang diberikan, penulisan akan berhenti setelah *byte* panjang ditulis atau akhir *string* tercapai, mana saja yang lebih dulu.

Perhatikan bahwa jika argumen panjang diberikan, maka opsi konfigurasi magic_quotes_runtime akan diabaikan dan tidak ada garis miring yang akan dilucuti dari string.

Return Values :

fwrite () mengembalikan jumlah *byte* yang ditulis, atau salah pada kesalahan.

Note : Menulis ke aliran jaringan mungkin berakhir sebelum seluruh string ditulis. Nilai kembali dari *fwrite* () dapat diperiksa:

Contoh:

```
<?php
$fp = fopen('data.txt', 'w');
fwrite($fp, '1');
fwrite($fp, '23');
fclose($fp);

// the content of 'data.txt' is now 123 and not 23!
?>
```

Contoh diatas akan menghasilkan file ‘data.txt’ dengan ini adalah 123

Membaca File Text

Membaca file text dalam PHP MANUAL dideskripsikan tentang *fread*() membaca hingga *byte* panjang dari penunjuk file yang direferensikan oleh aliran.

Pembacaan berhenti segera setelah salah satu kondisi berikut terpenuhi:

- *byte* panjang telah dibaca
- *EOF* (akhir file) tercapai
- paket menjadi tersedia atau terjadi batas waktu soket (untuk aliran jaringan)
- jika aliran dibaca *buffered* dan tidak mewakili file biasa, paling banyak satu pembacaan hingga sejumlah *byte* yang sama dengan ukuran potongan (biasanya 8192) dibuat; tergantung pada data yang di-*buffer* sebelumnya, ukuran data yang dikembalikan mungkin lebih besar dari ukuran potongan.

fread (resource $stream , int $length) : string|false

Parameter:

Stream : Sumber daya penunjuk sistem *file* yang biasanya dibuat menggunakan *fopen* ().

Length : Hingga panjang jumlah *byte* yang dibaca.

Return Values : Mengembalikan string baca atau false jika gagal.

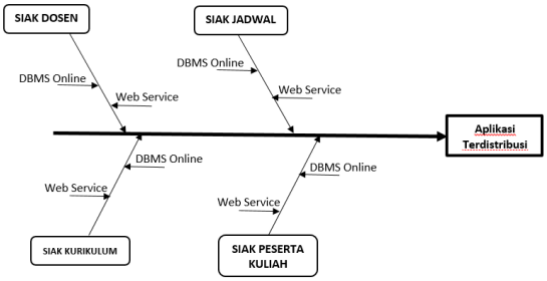
Contoh

```
<?php
// get contents of a file into a string
$filename = "data.txt";
$handle = fopen($filename, "r");
$contents = fread($handle, filesize($filename));
echo $contents;
fclose($handle);
?>
```

Dari script diatas akan ditampilkan isi dari file data.txt yaitu “123”/

3. METODOLOGI PENELITIAN

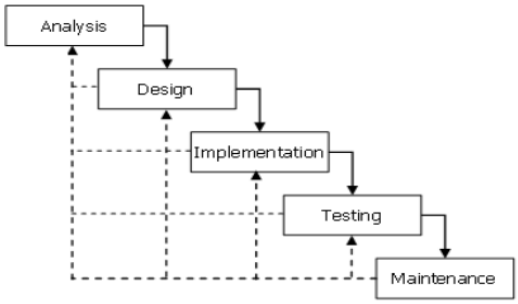
Metode penelitian dilakukan dalam skala laboratorium di Laboratorium Jaringan dan Internet Institut Sains dan Teknologi AKPRIND Yogyakarta, data atau model yang digunakan dalam penelitian ini sistem informasi akademik *online* yang disimulasikan di dalam skala laboratorium dan juga diimplementasikan secara *online*. Sedangkan untuk aplikasi terdistribusi dalam penelitian ini adalah dibatasi pada aplikasi tingkat Jurusan dan Laboratorium. Gambar 2 menunjukkan *fishbone* diagram metode penelitian.



Gambar 2. *Fishbone* Diagram Metode Penelitian

Sistem Informasi Akademik (SIAK) dalam hal ini adalah sistem imaginer kegiatan akademik di Institut Sains dan Teknologi AKPRIND Yogyakarta, SIAK yang digunakan dalam penelitian ini adalah bank data Kurikulum, Dosen, Jadwal Kuliah, Peserta Kuliah yang dikelola secara terpusat. data-data tersebut memiliki jadwal perubahan yang sudah tertata, misalnya kurikulum usia perubahan 4 tahun, dosen selama belum ada penambahan dan atau pengurangan dosen tidak akan ada perubahan, jadwal kuliah dan peserta kuliah dengan perubahan tiap semester (6 bulan sekali) dan disimulasikan dalam skala laboratorium. ***Database*** adalah *database* yang digunakan untuk menampung transaksi dari kegiatan sistem informasi. **Web Service** adalah media layanan yang digunakan untuk mengambil data-data dari SIAK secara periodik oleh *client*. **Client Service** adalah aplikasi tersebar yang memperoleh data dari *web service* dan kemudian disimpan dalam file text, *client service* dalam hal ini dibatasi pada aplikasi jurusan dan laboratorium. **File Text** adalah kumpulan file text yang mengikuti pola tertentu dan disimpan dalam aplikasi *client Service*.

Aplikasi terdistribusi tingkat Jurusan dan Laboratorium akan melakukan sinkronisasi data sesuai jadwal tertentu, dalam selama proses kegiatan administrasinya aplikasi terdistribusi ini hampir tidak pernah mengakses SIAK secara langsung, tetapi hanya menggunakan data lokal yang telah terdistribusi dan tentunya akan mengurangi beban server SIAK dan beban jalur komunikasi/internet yang ada.



Gambar 3. Fase pada model *waterfall*

Pengembangan aplikasi menggunakan metode pengembangan *waterfall*. Metode *waterfall* ini dibangun oleh *Winston W. Royce* pada tahun 1970 untuk menggambarkan praktik produk perangkat lunak. Model *waterfall* terdiri dari lima fase yaitu *analysis*, *design*, *implementation*, *testing* dan *maintenance*. Setiap fase pada model *waterfall* dapat dilihat pada Gambar 3.

a. *Analysis*
   Pada tahap analisis dilakukan proses pengumpulan data untuk mendapatkan informasi mengenai perangkat lunak dan database yang akan dikembangkan sesuai dengan kebutuhan pengguna. Teknik pengumpulan data untuk tahap analisis ini bisa dengan observasi, wawancara maupun kuesioner.
b. *Design*
   Tahap *design* digunakan untuk menerjemahkan kebutuhan perangkat lunak dari tahap analisis ke desain. Desain yang dibuat berupa sketsa desain Model Transaksi.

c. *Implementation*
   Setelah desain dibuat, tahap selanjutnya yaitu membuat atau mengembangkan Model Transaksi.
d. *Testing*
   Pada tahap *testing*, verifikasi dan validasi model kontrol transaksi dan aplikasi perangkat lunak dilakukan. Tahap *testing* digunakan untuk menemukan kesalahan atau *debug* pada saat pembuatan aplikasi. Tahapan pengujian secara keseluruhan mencakup pengujian unit, pengujian integrasi, pengujian sistem, dan pengujian penerimaan.
e. *Maintenance*
   Tahap terakhir dari model *waterfall* ini adalah tahapan *maintenance*. Proses instalasi

produk dan pemeliharaan dilakukan dalam tahap *maintenance*.

4. PEMBAHASAN

Perancangan arsitektur tersebar pada penilitian ini mengikuti pola fishbone diagram pada gambar 2.

a. Membangun API Server Service

Dalam penelitian ini ada 4 sumber sistem informasi yang akan diambil datanya seperti terlihat pada tabel 1.

Tabel 1. API Server Service

Domain	Nama API	Perubahan	Parameter
SIAK Kurikulum	kurikulumIST	4 tahun sekali	Kode Jur dan Tahun Kurikulum
SIAK Dosen	dosenIST	Tiap awal semester	Kode Jur dan Tahun Akademik
SIAK Jadwal	jadwalIST	Tiap awal semester	Kode Jur dan Tahun Akademik
SIAP Peserta Kuliah	kuliahIST	Tiap awal semester	Kode Jur dan Tahun Akademik

Dari tabel 1 terlihat bahwa masing-masing domain memiliki API sendiri-sendiri dengan waktu perubahan yang berbeda-beda. Terlihat bahwa kegiatan kuliah terdiri dari Dosen, Jadwal dan Peserta kuliah mengalami perubahan data tiap awal semester yaitu setelah mahasiswa melakukan proses pengisian KRS. Sedangkan Kurikulum menurut peraturan yang berlaku, maka perubahan hanya terjadi tiap 4 tahun sekali.

Dari tabel tersebut maka data tersebut bisa didistribusikan ke aplikasi terdistribusi sehingga tidak membebani server utama.

1. API Server kurikulumIST

API server ini diletakkan pada domain SIAK Kurikulum, untuk mendapatkan data kurikulum yang dipakai saat ini.

Script API Server kurikulumST yang bisa diakses pada url https://lapak.akprind.info/service/kurikulum/kurikulumIST.php?jur=05&thkurik=2016

Paramater yang digunakan adalah kode jurusan dan tahun kurikulum, karena client ini berbasis jurusan dan tahun kurikulum yang akan diambil.

```
<?php
include("buka.php");
$sql = "SELECT * FROM kurikulum where jur='".$_GET['jur']."' and thkurik='".$_GET[thkurik]."' order by smt";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
$arr=array();
   while($row = $result->fetch_assoc()) {
         array_push($arr,$row);
         }
         $x=json_encode($arr);
         echo $x;
}
$conn->close();
?>
```

Hasil dari script tersebut berupa data kurikulum format JSON sebagai berikut

```
[{"id":"4318","jur":"05","thkurik":"2016","kodemtk":"INT 1101","matakuliah":"Agama Islam","kredit":"2","smt":"1"},
{"id":"4342","jur":"05","thkurik":"2016","kodemtk":"INT 1201","matakuliah":"Bahasa Indonesia","kredit":"2","smt":"1"},
{"id":"4339","jur":"05","thkurik":"2016","kodemtk":"TFP 1301","matakuliah":"Pemrograman Dasar","kredit":"4","smt":"1"},
dst
```

Dengan komponen ID, Jur, ThKurik, KodeMtk, Matakuliah, Kredit dan SMT

2. API Server dosenIST

Api server ini diletakkan pada domain SIAK Dosen, untuk mendapatkan informasi dosen mengajar pada semester yang aktif dengan parameter Kode Jur dan Tahun Akademik.

3. API Server jadwalIST

API Server ini diletakkan pada domain SIAK Jadwal, untuk mendapatkan jadwal kuliah semester saat ini dengan parameter Kode Jur dan Tahun Akademik.

4. API Server kuliahIST

API Server ini diletakkan pada domain SIAK Peserta Kuliah, untuk mendapatkan data peserta kuliah semester saat ini dengan parameter Kode Jur dan Tahun Akademik.

b. Membangun Client Service

Client Service digunakan untuk mendapatkan data dari Server service dan mendistribusikan ke masing-masing aplikasi terdistribusi dengan format JSON dan tersimpan pada file text seperti terlihat pada tabel 2.

Tabel 2. Client Service

Domain	Nama Client	File Text
SIAK Dosen	c_dosenIST	dosenIST.txt
SIAK Kurikulum	c_kurikulumIST	kurikulumIST.txt
SIAK Jadwal	c_jadwalIST	jadwalIST.txt
SIAP Peserta Kuliah	c_kuliahIST	kuliahIST.txt

Script berikut digunakan untuk client SIAK Kurikulum jurusan 05 dengan tahun kurikulum (thkurik) 2016

```
<?php
$x=file_get_contents("https://lapak.akprind.info/service/kurikulum/kurikulumIST.php?jur=05&thkurik=2016");
$myfile = fopen("data/kurikulumIST.txt", "w");
fwrite($myfile, $x);
fclose($myfile);
?>
```

Hasil dari script tersebut akan membuat file text pada folder data seperti terlihat pada gambar 4.

Gambar 4. Hasil Script Client c_kurikulumIST.php

Proses ini akan dilakukan mengikuti pola pada tabel 1, sehingga tidak tiap saat menjalankan script ini, sehingga akses ke database utama akan sangat minimal. Juga client ini berbasis jurusan sehingga data-data akan terdistribusi untuk masing-masing jurusan.

c. Aplikasi terdistribusi

Menampilkan data dari file text ke aplikasi terdistribusi, sehingga aplikasi terdistribusi tidak akan melakukan koneksi database ke server atau masing-masing domain, tetapi cukup mengambil data file text yang telah tersedia di aplikasi terdistribusi.

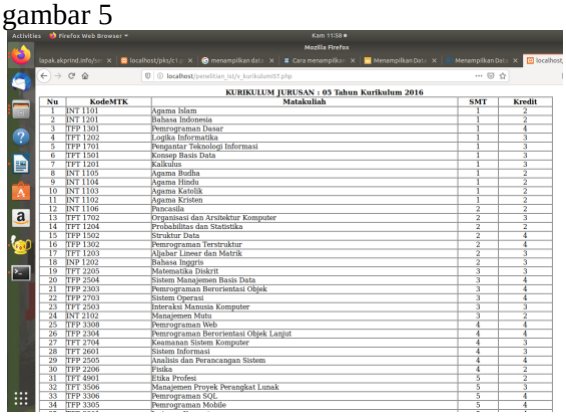
Aplikasi terdistribusi ini berbasis jurusan, dan hanya akan memproses atau menampilkan data jurusannya sendiri.

Untuk mengambil data dari file text dan ditampilkan ke aplikasi terdistribusi bisa dilihat pada script berikut ini:

```
<?php
$myfile = fopen("data/kurikulumIST.txt", "r");
$tek=fread($myfile,filesize("data/kurikulumIST.txt"));
$y=json_decode($tek,true);
$i=0; foreach($y as $key => $value):
if($i==0)
{
echo "<div align=center><strong>KURIKULUM JURUSAN : ".$value['jur']." Tahun Kurikulum ".$value['thkurik']."</strong></div>
<table border=1 width=90% cellpadding=0 cellspacing=0>
<th>Nu</th><th>KodeMTK</th><th>Matakuliah</th><th>SMT</th><th>Kredit</th>";
$i++;
}
?>
<tr>
<td align=center><?php echo $key+1; ?></td>

<td><?php echo $value['kodemtk']; ?></td>
<td><?php echo $value['matakuliah']; ?></td>
<td align=center><?php echo $value['smt']; ?></td>
<td align=center><?php echo $value['kredit']; ?></td>
</tr>
<?php endforeach; ?>
</table>
```

Hasil dari script diatas seperti terlihat pada gambar 5



Gambar 5. Hasil script v_kurikulumIST.php

Implementasi dari aplikasi ini bisa dimasukkan kedalam plugin web masing-masing jurusan, sehingga web jurusan akan tidak harus selalu tergantung terhadap web utama sistem informasi.

4. KESIMPULAN

Dari hasil penelitian dan pengujian dapat disimpulkan sebagai berikut:

a. Perancangan arsitektur aplikasi tersebar berbasis text bisa berjalan dengan sempurna.

b. Penerapan API Server Service akan mengurangi akses ke database utama secara langsung, sehingga akan lebih mengamankan database tersebut.
c. Penerapan client Service yang hanya dilakukan secara periodik tertentu akan sangat mengurangi beban server database.
d. Aplikasi terdistribusi tidak pernah secara langsung berhubungan dengan server utama, sehingga akan mengurangi beban server utama.
e. Aplikasi terdistribusi tidak pernah tahu bentuk database pada server utama, sehingga keamanan dan rahasia server utama akan lebih terjamin.
f. Akses data aplikasi terdistribusi hanya terhadap data lokalnya sendiri dalam bentuk file text.
g. Metode aplikasi terdistribusi ini bisa diterapkan atau diimplementasikan ke sistem localhost yang tidak harus selalu terkoneksi ke internet sehingga terjadi penghematan akses internet.

5. UCAPAN TERIMA KASIH

Atas selesainya penelitian ini, peneliti sangat-sangat mengucapkan banyak terimakasih terkusus kepada
1. IST AKPRIND Yogyakarta sebagai pemberi dana simultan.
2. LPM IST AKPRIND Yogyakarta yang telah meloloskan topik penelitian ini.
3. Rekan peneliti yang telah bersusah payah untuk ikut memecahkan dan menghasilkan penelitian ini.
4. Keluarga tercinta yang telah mengikhlaskan tersita waktunya demi menyelesaikan penelitian ini.
5. dan kepada para pihak yang tidak bisa peneliti sebutkan satu per satu, sehingga penelitian ini bisa berjalan dengan lancar.

DAFTAR PUSTAKA

Groff, J. et al. (2010). *SQL The Complete Refference - Third Edition.* United States: The McGraw-Hill Companies.

Haryani, P. (2016). Evaluasi Kualitas Layanan E-Government Pemerintah Kota Yogyakarta Dengan Metode E-GovQual Modifikasi. *Simposium Nasional Ke-15 RAPI 2016* (pp. 379-386). Surakarta: Fakultas Teknik Universitas Muhammadiyah Surakarta.

Kamus Besar Bahasa Indonesia. (n.d.). Retrieved 04 28, 2020, from http://www.kamuskbbi.id/inggris/indonesia.php?mod=view&reminder&id=27160-kamus-inggris-indonesia.html

Marrs, T. (2017). *JSON at Work.* . Sebastopol: O'Reilly Media inc.

PHP:fread-manual. (2021, 05 26). Retrieved from PHP-Documentation: https://www.php.net/manual/en/function.fread

PHP:fwrite-manual. (2021, 05 26). Retrieved from PHP Documentation: https://www.php.net/manual/en/function.fwrite

Smith, B. (2015). *Beginning JSON.* New York: Apress.

Triyono, J. (2015). Sistem Informasi Agroteknologi berbasis Web dan Jejaring Sosial Twitter. *Seminar Nasional IENACO* (pp. 205-212). Surakarta: UMS Surakarta.

Triyono, J. dan Haryani, P. (2017). Membangun Aplikasi Front End Web Dosen Berbasis Web APPML (Application Modeling Language). *Simposium Nasional RAPI XVI – 2017 FT UMS* (pp. 289-298). Surakarta: FT UMS.

Triyono, dkk. (2018). Membangun Aplikasi Web Front End dengan Teknik JSON (Java Script Object Notation). Seminar Nasional Aplikai Sains dan Teknologi (SNAST 2018) (pp. B237-B246). Yogyakarta: IST AKPRIND Yogyakarta.