Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 5 Nomor 2, September 2021, pp. 852-865
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti

Web Service Untuk Transaksi Data Pada Aplikasi Fasilitas Keuangan Dengan Metode REST

Diki Darmawan[1], Fitrah Satrya Fajar Kusumah[2], Safaruddin Hidayat Al Ikhsan[3]
Universitas Ibn Khaldun Bogor
Jl. K.H. Sholeh Iskandar Raya Km. 2, Kedung Badak, Bogor
[1] dikidarmawan6698@gmail.com, [2] fitrah@uika-bogor.ac.id, [3] safaruddin@uika-bogor.ac.id

Abstract

According to Law No. 14/1967 concerning banking principles, it is explained that financial institutions are all entities that carry out their activities in the financial sector. withdraw money from the people and channel it into the people. Financial institutions have large data on financial facilities. In order to improve the quality of information disclosure to the public, financial institutions require the application of financial facilities as a means to register the financial facilities they have. This financial facility application consists of two platforms, namely web-based and mobile-based, so that a mechanism is needed to process data transactions between different platforms and can support interoperability between systems. Web service is a standard used in conducting data transactions between systems, because applications that exchange data can be written in a variety of programming languages and run on a variety of different platforms. The financial facility web service is built using node js with asynchronous javascript programming language so that it has better performance in handling requests. In this study, the web service was built using a REST architecture that uses the HTTP protocol to exchange data. Tests have been carried out to see the results of the web service using insomnia software. The results of this study indicate that web services can be a bridge for communication and data transactions between financial facility search applications and can run well and are able to support interoperability between systems.

Keywords*: Web Service, Node js, REST, JSON.*

Abstrak

Menurut Undang-undang No. 14 tahun 1967 tentang pokok-pokok perbankan dijelaskan bahwa lembaga keuangan adalah semua badan yang melalui kegiatan-kegiatannya di bidang keuangan, menarik uang dari dan menyalurkannya ke dalam masyarakat. Lembaga keuangan memiliki data fasilitas keuangan yang besar. Dalam rangka meningkatkan kualitas keterbukaan informasi kepada masyarakat, lembaga keuangan memerlukan aplikasi fasilitas keuangan sebagai sarana untuk mendaftarkan fasilitas keuangan yang mereka mililki. Aplikasi fasilitas keuangan terdiri dari dua platform yaitu web dan juga mobile, sehingga dibutuhkan mekanisme dalam proses transaksi data antar platform yang berbeda serta dapat menunjang interoperabilitas antar sistem. Web service merupakan suatu standar yang digunakan dalam melakukan transaksi data antar sistem, karena aplikasi yang melakukan pertukaran data dapat ditulis dalam berbagai bahasa pemrograman dan berjalan pada berbagai platform yang berbeda. Web service fasilitas keuangan dibangun menggunakan node js dengan bahasa pemrograman javascript yang bersifat asinkronis sehingga memiliki performa yang lebih baik dalam menangani request. Pada penelitian ini web service dibangun menggunakan arsitektur REST yang menggunakan protokol HTTP dalam melakukan pertukaran data. Pengujian telah dilakukan untuk melihat hasil dari web service menggunakan perangkat lunak insomnia. Hasil dari penelitian ini menunjukkan bahwa web service dapat menjadi jembatan komunikasi dan transaksi data antar aplikasi pencarian fasilitas keuangan dan dapat berjalan dengan baik serta mampu menunjang interoperabilitas antar sistem.

Kata kunci*: Web Service, Node js, REST, JSON.*

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 5 Nomor 2, September 2021, pp. 852-865
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti

1. PENDAHULUAN

Lembaga keuangan adalah semua badan yang melalui kegiatan-kegiatannya di bidang keuangan, menarik uang dari dan menyalurkannya ke dalam masyarakat [1]. Lembaga keuangan berfungsi untuk memberikan jaminan hukum dan moral mengenai keamanan dana masyarakat, menghimpun dana dari masyarakat dalam bentuk simpanan dan menyalurkan ke masyarakat dalam bentuk pinjaman serta berkewajiban dalam memberikan pengetahuan dan informasi yang berguna dan menguntungkan bagi nasabahnya [2]. Lembaga keuangan memiliki data fasilitas keuangan yang besar. Dalam rangka meningkatkan kualitas keterbukaan informasi kepada masyarakat, lembaga keuangan memerlukan aplikasi fasilitas keuangan sebagai sarana untuk mendaftarkan fasilitas keuangan yang mereka mililki. Diharapkan aplikasi tersebut dapat dimanfaatkan oleh masyarakat untuk melakukan pencarian fasilitas keuangan yang *valid* dan terdaftar secara resmi dan juga dapat dikendalikan secara penuh oleh masing-masing lembaga keuangan. Aplikasi fasilitas keuangan yang akan dibangun berbasis *web* dan juga *mobile*, sehingga dibutuhkan mekanisme dalam proses transaksi data antar *platform* yang berbeda. Terlebih lagi sistem yang akan dibangun ini akan berhubungan dengan sistem lainnya sehingga dibutuhkan mekanisme transaksi data yang menunjang interoperabilitas antar sistem.

Berdasarkan permasalahan di atas maka dibutuhkan teknologi *web service* sebagai mekanisme interaksi antar sistem yang menunjang interoperabilitas untuk kepentingan integrasi data yang dapat diakses oleh berbagai pihak melalui internet dengan menggunakan berbagai macam perangkat milik masing-masing pengguna [3]. Interoperabilitas perangkat lunak secara sederhana merupakan kerjasama antara dua atau lebih program atau aplikasi yang berbeda *platform* sistem operasi, bahasa pemrograman, serta basis data yang memproses dan mengolah data tertentu [4]. Dengan demikian, *web service* memungkinkan sarana operasi antar sistem menjadi standar pada berbagai *platform* yang berbeda [5].

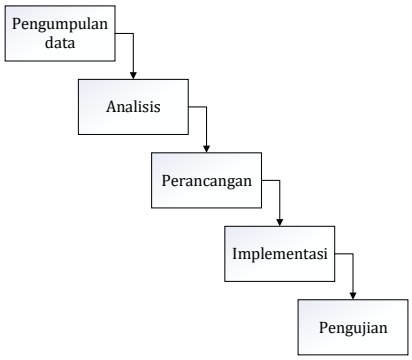
Salah satu arsitektur yang lazim digunakan dalam pengembangan *web service* yaitu REST [6]. *REpresentational State Transfer* (REST) adalah suatu arsitektur metode komunikasi yang sering diterapkan dalam pengembangan layanan berbasis web. Arsitektur REST yang umumnya dijalankan *via Hypertext Transfer Protocol* (HTTP), melibatkan proses pembacaan laman *web* tertentu yang memuat sebuah *file* XML atau JSON. *File* inilah yang menguraikan dan memuat konten yang hendak disajikan. Setelah melalui sebuah proses definisi tertentu, konsumen akan bisa mengakses antarmuka aplikasi yang dimaksudkan [7]. Kekhasan REST terletak pada interaksi antara *client* dan *server* yang difasilitasi oleh sejumlah tipe operasional (verba) dan *Universal Resource Identifiers* (URIs) yang unik bagi tiap-tiap sumber daya. Masing-masing verba *GET*, *POST*, *PUT* dan *DELETE* memiliki makna operasional khusus untuk menghindari ambiguitas. REST kerap dipergunakan dalam *mobile application*, situs *web* jejaring sosial, *mashup tools* dan *automated business processes* [7].

Penelitian terkait *web service* juga telah dilaksanakan oleh Ali firdaus dkk yang membangun *web service* untuk sistem informasi perpustakaan yang menghasilkan sebuah server yang lebih *flexible* digunakan dan dapat melayani dalam permintaan *resource* yang berada di perpustakaan [7]. Penelitian lainnya oleh Edhy dan Khabib yang membangun *web service* untuk sinkronisasi data antar sistem informasi dalam e-Gov di Pemkab Bantul Yogyakarta dengan hasil *web service* yang mendukung konsep interoperabilitas yang dapat menjadi sebuah alternatif solusi untuk proses pertukaran data antar sistem informasi serta dengan kemampuan proses pertukaran data antar sistem informasi, maka dimungkinkan untuk melakukan proses sinkronisasi data antar sistem informasi [3].

Berdasarkan uraian latar belakang di atas, maka pada penelitian ini akan dibangun suatu "*Web Service* untuk Transaksi Data pada Aplikasi Fasilitas Keuangan dengan Metode REST" yang diharapkan dapat mengoptimalkan proses transaksi dan integritas data pada aplikasi fasilitas keuangan menjadi lebih cepat dan aman.

2. METODOLOGI PENELITIAN

Metode penelitian yang digunakan mengadopsi metode *Waterfall*. Meliputi pengumpulan data, analisis, perancangan, implementasi dan pengujian.



Gambar 1. Metode *waterfall*

2.1. Pengumpulan data

Pengumpulan data dilakukan dari berbagai sumber *literature* seperti jurnal, buku dan lainnya.

2.2. Analisis (*Requirement Definition*)

Pada tahap ini dilakukan proses analisis kebutuhan sistem diantaranya analisis kebutuhan fungsional sistem, analisis kebutuhan non-fungsional sistem dan analisis arsitektur system.

2.3. Perancangan (*System and Software Design*)

Pada tahap ini dilakukan proses perancangan desain dengan menggunakan UML dan merancang pemasangan *web service* dengan metode REST.

2.4. Implementasi (*Implementation and Unit Testing*)

Pada tahap ini dilakukan proses implementasi ke dalam kode program menggunakan bahasa pemrograman berbasis Javascript menggunakan Node.js. Proses ini merupakan penerjemah desain ke dalam bahasa yang dikenali oleh komputer. Setelah dilakukan pengkodean, akan dilakukan *testing* terhadap sistem yang dibuat.

2.5. Pengujian (*Integration and System Testing*)

Pada tahap ini dilakukan proses pengujian sistem dengan menggunakan metode *blackbox* untuk pengujian fungsi dari setiap *web service* yang dibuat.

3. HASIL DAN PEMBAHASAN

3.1. Analisis

3.1.1. Analisis kebutuhan fungsional

Kebutuhan fungsional adalah kebutuhan pada sistem yang merupakan layanan dalam *web service* yang harus disediakan untuk aplikasi pencarian fasilitas keuangan berbasis *mobile* android dan *web*. Berdasarkan kebutuhan aplikasi pencarian fasilitas keuangan maka modul-modul yang akan disediakan oleh *web service* diantaranya adalah sebagai berikut:

a) Modul Pendaftaran Pengguna
   Modul ini digunakan oleh pengguna untuk mendaftar pada aplikasi pencarian fasilitas keuangan.
b) Modul Pencarian Fasilitas Keuangan
   Modul ini digunakan oleh pengguna untuk mencari sebaran fasilitas keuangan berdasarkan kriteria tertentu.
c) Modul Simpan Lokasi Fasilitas Keuangan
   Modul ini digunakan untuk menyimpan dan menandai data fasilitas keuangan yang telah dikunjungi oleh pengguna.
d) Modul *Rating*
   Modul ini digunakan pengguna untuk memberikan *rating* atau penilaian terhadap fasilitas keuangan yang telah mereka kunjungi.
e) Modul *Update* Profil
   Modul ini digunakan pengguna untuk mengupdate data profil pengguna.
f) Modul Notifikasi
   Modul ini digunakan oleh pengguna untuk mendapatkan pemberitahuan dari admin seputar aplikasi pencarian fasilitas keuangan.
g) Modul Manajemen Fasilitas Keuangan
   Modul ini digunakan oleh admin lembaga keuangan untuk melakukan manajemen data seperti menambah data, mengedit data, melihat data, mencari data dan menghapus data fasilitas keuangan.
h) Modul Grafik Fasilitas Keuangan
   Modul ini digunakan untuk menampilkan statistik *hitcounter* dari data fasilitas keuangan.

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 5 Nomor 2, September 2021, pp. 852-865
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti

3.1.2. Analisis kebutuhan non-fungsional

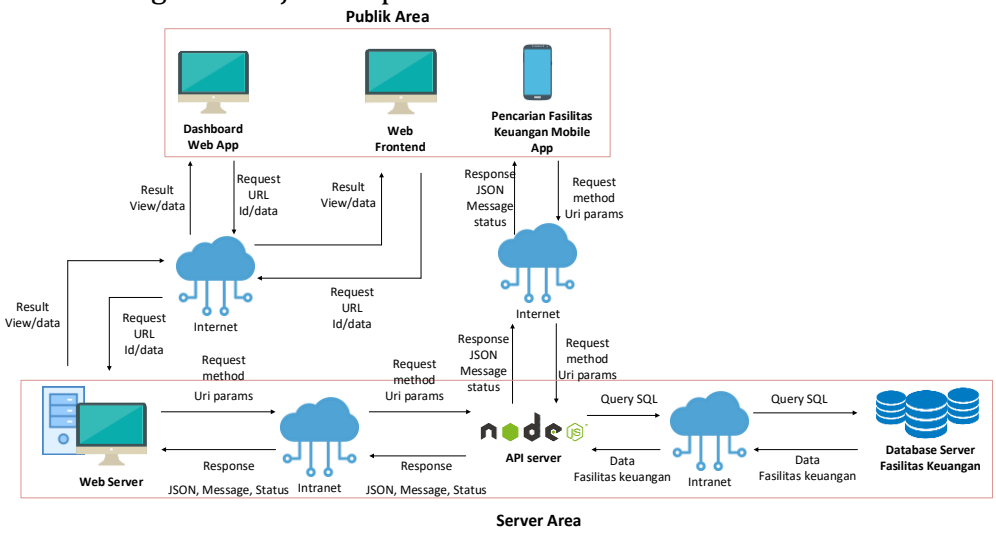
Analisis kebutuhan non-fungsional merupakan analisis yang digunakan untuk mengetahui bagian-bagian yang berinteraksi dengan sistem yang sedang berjalan. Berikut ini adalah kebutuhan non-fungsional sistem:

a) *Reliability web service* dapat berjalan selama 24 jam /hari.
b) *Portability web service* dapat diakses melalui jaringan internet.
c) *Interoperability web service* dapat digunakan oleh *platform* dan *operating system* yang berbeda.

3.1.3. Analisis kebutuhan arsitektur sistem

Web service yang akan dibangun akan berjalan pada dua *platform* yang berbeda yaitu *platform mobile* dan *web* juga *web service* ini kemungkinan akan diakses oleh sistem lainnya yang membutuhkan data dari fasilitas keuangan. Adapun aplikasi fasilitas keuangan yang akan terhubung dengan *web service* adalah aplikasi pencarian fasilitas keuangan berbasis *mobile*, aplikasi pencarian fasilitas keuangan berbasis *web* dan aplikasi *dashboard* berbasis *web*. Ketiga aplikasi ini akan melakukan transaksi data menggunakan *web service* dengan akses *response* dan *request*, sehingga komunikasi data yang ada pada *web service* adalah komunikasi dua arah atau lebih dikenal dengan hubungan *client-server*. *Client* akan melakukan *request* dengan parameter tertentu terhadap *server*, kemudian *request* itu akan diolah oleh *server* dan disajikan dalam bentuk *response* yang dikembalikan kepada *client*.

Hubungan *client* dan *server* dijembatani oleh *file web service* yang pada umumnya adalah format XML dan JSON. Pada penelitian ini, format yang digunakan adalah JSON. Sehingga aplikasi fasilitas keuangan yang mengirimkan *request* data terhadap *web service* akan menerima *response* data berupa *file* JSON. Sehingga *file* JSON inilah yang akan diolah oleh aplikasi fasilitas keuangan dan digunakan sesuai kebutuhan masing-masing aplikasi. Adapun penjelasan arsitektur sistem dari *web service* aplikasi pencarian fasilitas keuangan ditunjukkan pada Gambar 4.1.



Gambar 2. Analisis arsitektur sistem

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 5 Nomor 2, September 2021, pp. 852-865
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti

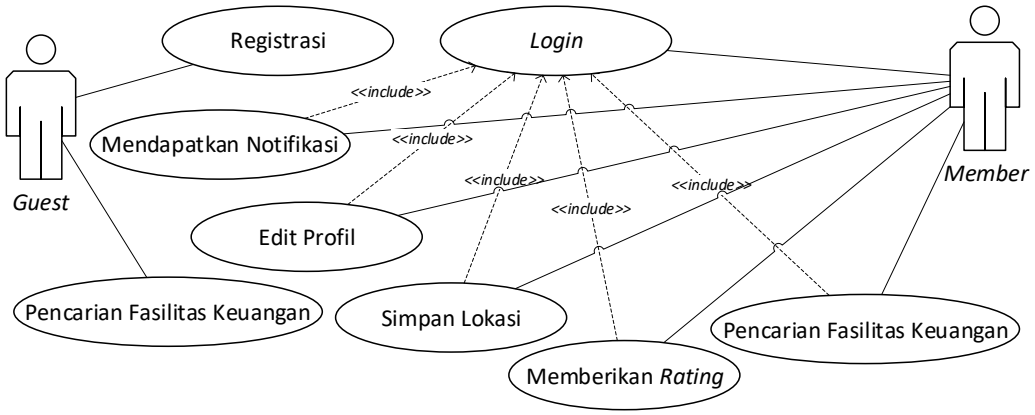
3.2. Perancangan

Pada tahap ini dilakukan proses perancangan desain dengan menggunakan salah satu diagram UML yaitu *usecase diagram*.

3.2.1. *Usecase diagram*

Use case diagram adalah sebuah diagram yang menunjukkan hubungan antara aktor dan *use cases.* Aktor adalah seorang pengguna yang berinteraksi dengan sistem sedangkan *use case* merupakan sebuah tindakan/aksi yang dilakukan oleh aktor.

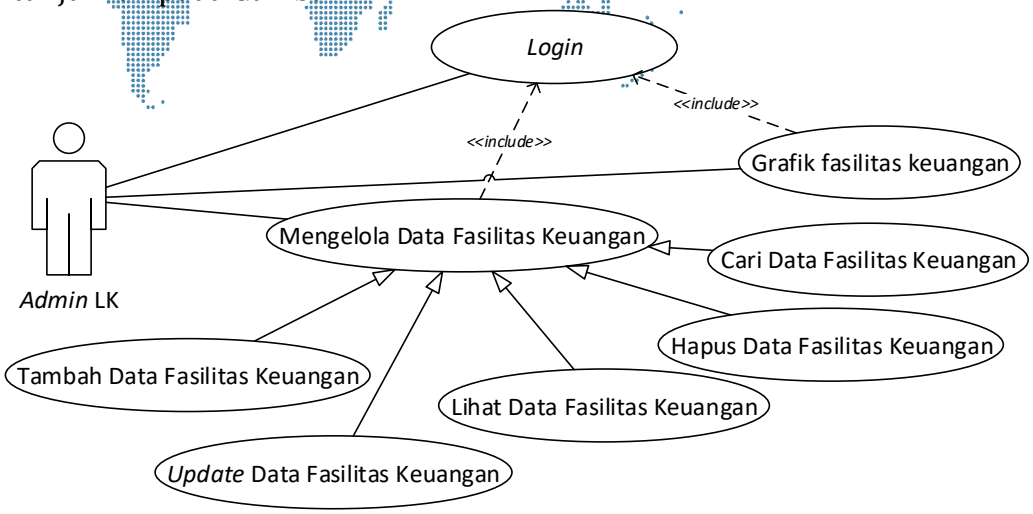
Pada perancangan *web service* terdapat 3 aktor yang akan berinteraksi dengan *service* yaitu *member*, *guest* dan admin lembaga keuangan. *Member* merupakan pengguna yang *login* dan telah mendaftar pada aplikasi pencarian fasilitas keuangan, *guest* merupakan tamu yang melakukan pencarian fasilitas keuangan menggunakan aplikasi pencarian fasilitas keuangan berbasis *web* dan admin lembaga keuangan yang *login* ke dalam aplikasi *dashboard web*. *Member* dapat melakukan beberapa tindakan/aksi namun diharuskan untuk *login* terlebih dahulu. Aksi yang dapat dilakukan *member* seperti *login*, melakukan pencarian fasilitas keuangan, mendapatkan notifikasi, *update* profil, simpan lokasi fasilitas keuangan dan memberikan *rating* terhadap fasilitas keuangan. *Guest* dapat melakukan registrasi dan dapat melakukan pencarian fasilitas keuangan menggunakan aplikasi pencarian fasilitas keuangan berbasis *web* tanpa diharuskan untuk *login* terlebih dahulu namun terdapat batasan aksi/tindakan yang dapat dilakukan oleh *guest*. *Use case diagram* untuk aktor *member* dan *guest* ditunjukkan pada Gambar 3.



Gambar 3. *Usecase Diagram Member* dan *Guest*

Adapun admin lembaga keuangan dapat mengelola data fasilitas keuangan namun diharuskan untuk *login* terlebih dahulu untuk masuk ke dalam aplikasi *dashboard* fasilitas keuangan. Aksi yang dapat dilakukan oleh admin adalah mengelola data fasililtas keuangan serta dapat melihat grafik *statistic counter* dari fasilitas keuangan yang dimilikinya. Aksi mengelola data dapat di generalisasi menjadi beberapa aksi yaitu seperti menambah data fasilitas keuangan, *update* data fasilitas keuangan, melihat data fasilitas keuangan, mencari data fasilitas keuangan dan menghapus data fasilitas

keuangan. *Use case diagram* untuk aktor admin lembaga keuangan ditunjukkan pada Gambar 4.



Gambar 4. *Usecase Diagram* Admin LK

3.3. Implementasi

Berdasarkan analisis dan percangan yang telah dilakukan, maka dibangun *web service* untuk transaksi data antar aplikasi pencarian fasilitas keuangan serta untuk mendukung interoperabilitas antar sistem. *Web service* yang telah dihasilkan akan diuji dengan menggunakan *software* Insomnia. Terdapat 10 *web service* yang dibangun dalam aplikasi pencarian fasilitas keuangan yaitu sebagai berikut:

a) Registrasi pengguna

Registrasi pengguna merupakan *web service* yang berfungsi untuk menangani aksi dari penggguna yang mendaftar pada aplikasi *mobile* pencarian fasilitas keuangan. *Method* yang digunakan adalah *POST*. Berikut data dokumentasi *service* registrasi pengguna ditunjukkan pada Tabel 1.

Tabel 1. *Web service* registrasi pengguna

URL	http://*localhost*:3000/v1/*user*/*insert*
Method	*POST*
URI Parameter	nik, nama_user, email_user, sandi_user, latitude_ user, longitude_ user, notlpn_ user, foto_ user, jeniskelamin_ user, status_verifikasi

Success Response	{
	“status”:”200”
	"success": true,
	"message": "Create data success"
	}

b) Edit profil pengguna

Edit profil pengguna merupakan *web service* yang berfungsi untuk menangani aksi dari pengguna yang melakukan pembaharuan data diri. *Method* yang digunakan adalah *PUT*. Berikut adalah data dokumentasi *service* edit profil pengguna ditunjukkan pada Tabel 2.

Tabel 2. *Web service* edit profil pengguna

URL	http://*localhost*:3000/v1/*user/update*/id
Method	*PUT*
URI Parameter	nik, nama_user, email_user, sandi_user, latitude_ user, longitude_ user, notlpn_ user, foto_ user, jeniskelamin_ user, status_verifikasi

Success Response	{
	“status”:”200”
	"success": true,
	"message": "Update data success"
	}

c) Pendaftaran fasilitas keuangan

Pendaftaran fasilitas keuangan merupakan *web service* yang berfungsi untuk menangani aksi dari lembaga-lembaga keuangan yang mendaftarkan fasilitas keuangan yang mereka miliki pada aplikasi pencarian fasilitas keuangan. *Method* yang digunakan adalah *POST*. Berikut adalah data dokumentasi *service* pendaftaran fasilitas keuangan ditunjukkan pada Tabel 3.

Tabel 3. *Web service* pendaftaran fasilitas keuangan

URL	http://*localhost*:3000/v1/ fasilitas_keuangan/*insert*
Method	*POST*
URI Parameter	kode_fasilitas, nama_ fasilitas, alamat_ fasilitas, image_ fasilitas, lembaga_id, induk_id, kategori_id, longitude_ fasilitas, latitude_ fasilitas, status_ fasilitas, aktif_weekend, kodepos_ fasilitas, call_point.

Success Response	{
	“status”:”200”
	"success": true,
	"message": "Create data success"
	}

d) *Update* Fasilitas Keuangan

Update fasilitas keuangan merupakan *web service* yang berfungsi untuk menangani aksi dari lembaga-lembaga keungan yang mengedit data fasilitas keuangan yang mereka miliki pada aplikasi pencarian fasilitas keuangan. *Method* yang digunakan adalah *PUT*. Berikut adalah data dokumentasi *service* update fasilitas keuangan ditunjukkan pada Tabel 4.

Tabel 4. *Web service update* fasilitas keuangan

URL	http://*localhost*:3000/v1/ fasilitas_keuangan/*update*/id
Method	*PUT*
URI Parameter	kode_fasilitas, nama_ fasilitas, alamat_ fasilitas, image_ fasilitas, lembaga_id, induk_id, kategori_id, longitude_ fasilitas, latitude_ fasilitas, status_ fasilitas,

	aktif_weekend, kodepos_ fasilitas, call_point.

Success Response	{
	“status”:”200”
	"success": true,
	"message": "Update data success"
	}

e) Hapus Fasilitas Keuangan

Hapus fasilitas keuangan merupakan *web service* yang berfungsi untuk menangani aksi dari lembaga-lembaga keungan yang menghapus data fasilitas keuangan yang mereka miliki pada aplikasi *dashboard* fasilitas keuangan. *Method* yang digunakan adalah *DELETE*. Berikut adalah data dokumentasi *service* hapus fasilitas keuangan ditunjukkan pada Tabel 5.

Tabel 5. *Web service* hapus fasilitas keuangan

URL	http://*localhost*:3000/v1/ fasilitas_keuangan/*delete*/id
Method	*DELETE*
URI Parameter	Id_fasilitas

Success Response	{
	“status”:”200”
	"success": true,
	"message": "Delete data success"
	}

f) Pencarian fasilitas keuangan

Pencarian fasilitas keuangan merupakan *web service* yang berfungsi untuk melakukan pencarian terhadap fasilitas keuangan berdasarkan kriteria tertentu baik melalui *platform mobile* ataupun melalui *platform web frontend*. *Method* yang digunakan adalah *GET*. Adapun data dokumentasi *service* pencarian fasilitas keuangan ditunjukkan pada Tabel 6.

Tabel 6. *Web service* pencarian fasilitas keuangan

URL	http://*localhost*:3000/v1/*search*
Method	*GET*
URI Parameter	idLembaga, idInduk, max_distance, id_user, lat, lon, search_terms, idKategori, jenisTarik

Success Response	{
	"status": 200,
	"success": true,
	"message": "Success",
	"total_data": 1,
	"data": [
	{
	" fasilitas ": {
	"*distance*": 2.9198438860057,

```
"id_fasilitas": 5221,
"nama_ fasilitas ": "ATM Pabuaran",
"kode_ fasilitas ": "008-3910-1132212",
"alamat": " Jl. Pabuaran No. 101 Bogor ",
"lembaga_id": 1,
"kodepos_ fasilitas ": "35773",
"image": null,
"call_point": null,
"latitude_ fasilitas ": -6.56586,
"longitude_ fasilitas ": 106.76399,
"aktif_weekend": "1",
"status_fasilitas ": "1",
"nama_fasilitas ": "Bank Bogor",
"profil_fasilitas ": "Bank Bogor didirikan pada 2 Oktober 1998, sebagai bagian dari program restrukturisasi perbankan",
"call_center": "14000",
"status_lembaga": "1",
"tahun_lembaga": "1998",
"nama_induk": "Bank",
"kategori_id": 23,
"nama_kategori": "ATM",
"induk_in_kategori": 1,
"icon_induk": "Bank.png",
"logo_kategori": "ATM.png",
"nama_provinsi": "Provinsi Jawa Barat ",
"images": [
],
"layanan": [
 {
  "id_pro": 9,
  "nama_pro": "produk 1",
  "deskripsi_produk": "deskripsi produk 1",
  "sp_id": 1,
  "created_at": "2020-03-13T17:54:50.180Z",
  "updated_at": "2020-03-13T17:54:50.180Z",
  "deleted_at": null,
  "induk_id": 1
 }],

"rating": [
 {
  "rating": null
 }
]
},
```

	"*distance*": "2.92", "unit": "Km", "distance_spell": "2.92Km" }] }

g) Simpan lokasi fasilitas keuangan

Simpan lokasi fasilitas keuangan merupakan *web service* yang berfungsi untuk menyimpan atau menandai fasilitas keuangan yang telah dilihat oleh pengguna. *Method* yang digunakan adalah *POST*. Adapun data dokumentasi *service* simpan lokasi fasilitas keuangan ditunjukkan pada Tabel 7.

Tabel 7. *Web service* simpan lokasi fasilitas keuangan

URL	http://*localhost*:3000/v1/simpan_lokasi/*insert*
Method	*GET*
URI Parameter	fasilitas _id, user_id

Success Response	{
	"status": 200,
	"success": true,
	"message": "create data success!"
	}

h) *Rating* fasilitas keuangan

Rating fasilitas keuangan merupakan *web service* yang berfungsi untuk menangani proses pemberian nilai oleh *member* terhadap fasilitas keuangan yang telah mereka kunjungi. *Method* yang digunakan adalah *POST*. Berikut adalah data dokumentasi *service* rating fasilitas keuangan ditunjukkan pada Tabel 8.

Tabel 8. *Web service* rating fasilitas keuangan

URL	http://*localhost*:3000/v1/*rating/insert*
Method	*POST*
URI Parameter	fasilitas_id, pujk_id, user_id, *rating*, komentar

Success Response	{
	"status": 200,
	"success": true,
	"message": "create data success!"
	}

i) Notifikasi Pengguna

Notifikasi pengguna merupakan *web service* yang berfungsi untuk menampilkan pemberitahuan kepada pengguna aplikasi *mobile*. *Method* yang digunakan adalah *GET*. Adapun data dokumentasi notifikasi pengguna ditunjukkan pada Tabel 9.

Tabel 9. *Web service* notifikasi pengguna

URL	http://*localhost*:3000/v1/notifikasi
Method	*GET*
URI Parameter	-

Success Response	{
	"status": 200,
	"success": true,
	"message": "Success",
	"total_data": 11,
	"data": [
	{
	"id_pemberitahuan": 2006,
	"judul_pemberitahuan": " pemberitahuan 1",
	"isi_pemberitahuan": " pemberitahuan 1 ",
	"admin_id": 4,
	"created_at": "2020-05-03T10:55:35.900Z",
	"updated_at": "2020-05-03T10:55:35.900Z",
	"deleted_at": *null*,
	"admin": {
	"nama_admin": "admin"
	}
	}]

j) Grafik fasilitas keuangan

Grafik fasilitas keuangan merupakan *web service* yang berfungsi untuk menampilkan statistik jumlah fasililtas keuangan dan jumlah *hit* dari fasilitas keuangan. *Method* yang digunakan adalah *GET*. Adapun data dokumentasi grafik fasilitas keuangan ditunjukkan pada Tabel 10.

Tabel 10. *Web service* grafik fasilitas keuangan

URL	http://*localhost*:3000/v1/grafik_fk/grafik_fk
Method	*GET*
URI Parameter	Id_lk, id_kab_kot, id_provinsi, id_kategori, *limit, offset*, tanggal_awal, tanggal_akhir, bulan, tahun
Success Response	{ "status": 200, *"success": true,* *"message": "Success",* "total_data": 3, "data": [{ "jumlah_counter": 133, "id_fasilitas": 1225, "nama_fasilitas": "ATM pabuaran", "kode_fasilitas ": "16829", "provinsi_id": 33, "kab_kot_id": 74,

```
    "nama_provinsi": "Provinsi Jawa Barat ",
    "nama_kab_kot": null,
    "alamat_fasilitas ": "Jl. Pabuaran No. 101 Bogor",
    "latitude_fasilitas ": -6.987284,
    "longitude_ fasilitas ": 110.416664,
    "nama_lembaga": "Bank Bogor",
    "nama_kategori": "ATM",
    "lembaga_id": 6
  } ]
}
```


3.4. Pengujian

Pengujian dilakukan dengan metode Blackbox *testing.* Blackbox *testing* merupakan pengujian yang dilakukan dengan mengamati hasil eksekusi melalui data uji dan mengamati fungsional dari perangakat lunak. Blackbox *testing* pada *web service* aplikasi pencarian fasilitas keuangan ditunjukkan pada Tabel 11.

Tabel 11. Pengujian *Blackbox*

No	URI API Pengujian	Skenario Pengujian	Hasil yang Diharapkan	Hasil Uji
1.	http://*localhost* :3000/v1/ *user/insert*	Kirim *request* dengan *method POST* beserta data pengguna	*Success response* dengan *status code* 200	Berhasil
2.	http://*localhost* :3000/v1/ user/ *update/id*	Kirim *request* dengan *method PUT* beserta data baru pengguna	*Success response* dengan *status code* 200	Berhasil
3.	http://*localhost* :3 000/v1/ fasilitas_keuangan /*insert*	Kirim *request* dengan *method POST* beserta data fasilitas keuangan	*Success response* dengan *status code* 200	Berhasil
4.	http://*localhost* :3000/v1/ fasilitas_keuangan /*update*/id	Kirim *request* dengan *method PUT* beserta data baru fasilitas keuangan	*Success response* dengan *status code* 200	Berhasil
5.	http://*localhost* :3000/v1/ fasilitas_keuangan /*delete*/id	Kirim *request* dengan *method DELETE* beserta id fasilitas keuangan	*Success response* dengan *status code* 200	Berhasil
6.	http://*localhost* :3000/v1 /*search*	Kirim *request* dengan *method GET* beserta parameter yang dibutuhkan	*Success response* beserta data dan *status code* 200	Berhasil
7.	http://*localhost* :3 000/v1/ simpan_lokasi/ *insert*	Kirim *request* dengan *method POST* beserta data *bookmark*	*Success response* dengan *status code* 200	Berhasil
8.	http://*localhost* :3000/v1/	Kirim *request* dengan *method POST* beserta	*Success response* dengan *status*	Berhasil

	rating/insert	data *rating*	*code* 200	
9.	http://*localhost* :3000/v1/ notifikasi	Kirim *request* dengan *method GET*	*Success response* beserta data dan *status code* 200	Berhasil
10.	http://*localhost* :3000/v1/ grafik_fk/	Kirim *request* dengan *method GET* beserta parameter yang dibutuhkan	*Success response* beserta data dan *status code* 200	Berhasil

4. SIMPULAN

Hasil penelitian ini dapat disimpulkan bahwa *web service* dengan metode REST telah berhasil diimplementasikan pada aplikasi pencarian fasilitas keuangan menggunakan Node.js sebagai teknologi dalam mengembangkannya. *Web service* menjadikan komunikasi dan transasksi data antar aplikasi pencarian fasilitas keuangan berbasis *mobile* dan aplikasi pencarian fasilitas keuangan berbasis *web* dapat berjalan dengan baik serta mampu menunjang interoperabilitas antar sistem, sehingga aplikasi pencarian fasilitas keuangan dapat dikembangkan untuk berinteraksi dengan sistem lain yang membutuhkan *resource* baik itu sistem internal dari lembaga keuangan ataupun sistem lainnya.

DAFTAR PUSTAKA

[1] R. Indonesia, Undang-Undang Republik Indonesia Nomor 14 Tahun 1967 Tentang Pokok-Pokok Perbankan, 1967.

[2] P. Sawitri and E. Hartanto, "Bank dan Lembaga Keuangan Lain", Jakarta: Gunadarma, 2007.

[3] E. Sutanta and K. Mustofa, "Kebutuhan Web Service Untuk Sinkronisasi Data Antar Sistem Informasi Dalam E-Gov Di Pemkab Bantul Yogyakarta", *JURTIK - STMIK BANDUNG (edisi Mei 2012),* 2012.

[4] A. M. M, "Interoperabilitas perangkat lunak menggunakan RESTful web service", *Jurnal Ilmiah Teknologi Sistem Informasi,* vol. 4, no. 1, pp. 14-22, 2018.

[5] M. I. Perkasa and E. B. Setiawan, "Pembangunan Web Service Data Masyarakat Menggunakan REST API dengan Access Token", *ULTIMA Computing,* vol. X, no. 1, pp. 19-26, 2018.

[6] V. Kumari, "Web Services Protocol: SOAP vs REST", *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET),* vol. 4, no. 5, pp. 2467-2469, 2015.

[7] A. Firdaus1, S. Widodo, A. Sutrisman, S. . G. F. Nasution and R. Mardiana, "Rancang Bangun Sistem Informasi Perpustakaan Menggunakan Web Service Pada Jurusan Teknik Komputer Polsri", *Jurnal Informanika,* vol. 5, no. 2, pp. 81-87, 2019.