

SEKOLAH TINGGI ILMU KOMPUTER BINANIAGA
PENGESAHAN PROPOSAL PENELITIAN

Judul : Penerapan Resource Oriented Architecture Untuk Pendistribusian Data Menggunakan RESTful APIs Berbasis Multiplatform
Oleh : Yanuar Nurcahyo
Jenjang : Strata 1 (S1), NPM : 1512018
Program Studi : Teknik Informatika

Tanggal,..... Pembimbing I

Dahlia Widhyaestoeiti, S.Kom, M.Kom
NIP. 11.120.1102

Tanggal,..... Pembimbing II

Anggra Triawan, S.Kom
NIP. 11.120.1003

Tanggal,..... a.n. Ketua
Pembantu Ketua Bidang Akademik

Irmayansyah,S.Kom,M.Kom
NIP. 11.120.0404

RIWAYAT PENULIS



Yanuar Nurcahyo lahir di Jakarta pada tanggal 26 Januari 1995. Ia tinggal di Jakarta hanya 5 bulan dan pindah ke daerah kabupaten Bojonggede kota Bogor. Pada tahun 2006 lulus Sekolah Dasar di SDN Bojonggede 3, tahun 2009 menyelesaikan Sekolah Menengah Pertama di SMP Al-Basyariah, kemudian menamatkan Sekolah Menengah Kejuruan di SMK Tri Dharma 2 Bogor, jurusan RPL (Rekayasa Perangkat Lunak) pada tahun 2012. Tahun 2012 melanjutkan pendidikan di S1 Jurusan Sistem Informasi di STIKOM BINANIAGA Bogor.

Sampai saat ini, ia masih melanjutkan kuliahnya dengan membuat karya tulis ilmiah. Karya ilmiah tersebut adalah “Resource Oriented Architecture Untuk Pendistribusian Data Menggunakan RESTful APIs Berbasis Multiplatform”. Disela-sela aktivitasnya, ia juga bekerja sebagai freelancer dibidang web programming sejak tahun 2014 sampai saat ini.

KATA PENGANTAR

Puji dan syukur dipanjangkan kehadiran Allah SWT karena atas berkat rahmat dan hidayah-Nya penyusunan proposal skripsi berjudul *“Resource Oriented Architecture Untuk Pendistribusian Data Menggunakan RESTful APIs Berbasis Multiplatform”* dapat diselesaikan tepat pada waktunya. Meskipun banyak hambatan yang dialami dalam proses penggerjaanya, namun Alhamdulillah berhasil diselesaikan.

Penulisah skripsi ini diajukan untuk memenuhi salah satu syarat memperoleh gelar Sarjana pada jurusan Teknik Informatika di STIKOM Binaniaga Bogor. Permasalah yang diangkat adalah kurang lengkapnya fasilitas diperpustakaan saat ini mengenai data-data journal diberbagai universitas lokal maupun internasional mengakibatkan mahasiswa sulit mencari referensi journal.

Dalam penyusunan dan penulisan skripsi ini tidak terlepas dari bantuan, bimbingan serta dukungan dari berbagai pihak. Penulis menyadari bahwa dalam menyusun laporan ini masih jauh dari kesempurnaan, untuk itu sangat diharapkan kritik dan saran yang bersifat membangun kemajuan pendidikan di masa yang akan datang.

Akhir kata, penulis sampaikan terima kasih kepada semua pihak yang telah berperan serta dalam penyusunan proposal skripsi ini dari awal sampai akhir. Semoga Allah SWT senantiasa memberkahi segala usaha kita. Aamiin.

Bogor, Maret 2016

Penulis

DAFTAR ISI

PENGESAHAN PROPOSAL PENELITIAN	i
RIWAYAT PENULIS.....	ii
KATA PENGANTAR.....	iii
DAFTAR ISI.....	iv
DAFTAR TABEL.....	v
DAFTAR GAMBAR	vi
BAB I	1
PENDAHULUAN.....	1
A. LATAR BELAKANG.....	1
B. RUMUSAH MASALAH	2
1. Identifikasi Masalah	2
2. Problem Statement.....	2
3. Research Question.....	2
C. MAKSUD DAN TUJUAN	2
D. KEGUNAAN DAN MANFAAT PENGEMBANGAN	2
E. PENTINGNYA PENGEMBANGAN	3
F. RUANG LINGKUP DAN KETERBATASAN	3
G. DEFINISI ISTILAH.....	3
BAB II	5
KERANGKA TEORITIS	5
A. TINJAUAN PUSTAKA	5
B. LANDASAN TEORI	7
C. KERANGKA PEMIKIRAN.....	14
BAB III	19
METODE PENGEMBANGAN.....	19
A. MODEL DAN PROSEDUR PENGEMBANGAN.....	19
B. UJI COBA PRODUK.....	21
1. Desain Uji Coba.....	21
2. Subjek Uji Coba.....	21
C. INSTRUMEN PENGUMPULAN DATA	21
D. UJI ANALISIS DATA	21
E. PENJADWALAN.....	21
F. SISTEMATIKA PENULISAN	24
BAB IV	25

HASIL dan PEMBAHASAN	25
A. DESKRIPSI OBJEK PENELITIAN	25
1. Tentang Aplikasi	25
B. HASIL PENGEMBANGAN	30
C. PEMBAHASAN.....	30
DAFTAR PUSTAKA	31

DAFTAR TABEL

Table 1: HTTP methods, (Roberto Lucchi, 2008)	7
Table 2: Menampilkan kode HTTP status, (Roberto Lucchi, 2008)	8
Table 3: Rincian jadwal kegiatan penelitian	22

DAFTAR GAMBAR

Gambar 1: Konsep dan relasi pada Resource Oriented Model, (W3C, 2003)I	8
Gambar 2: Struktur (object) pada format data JSON, (json.org, 2000)	10
Gambar 3: Struktur (array) pada format data JSON, (json.org, 2000)	10
Gambar 4: Struktur (value) pada format data JSON, (json.org, 2000)	10
Gambar 5: Deployment Diagram pada sistem Node.js, (Dahl, 2013)	11
Gambar 6: Gambar diatas menggambarkan proses NodeJS merequest data akses kedatabase menurut (Benjamin San Souci, 2014)	12
Gambar 7: Ilustrasi arsitektur aplikasi pada cross platform (Xamarin, 2014)	13
Gambar 8: Environtment Distribute Database System, (Aspects of the design of distributed databases, 2011).....	14
Gambar 9: Kerangka pemikiran	15
Gambar 10: Model prototype menurut Roger S. Pressman.....	19
Gambar 11: Prosedur pengembangan.....	20
Gambar 12: Penjadwalan.....	22

BAB I

PENDAHULUAN

A. LATAR BELAKANG

Dalam peradaban baru dunia global, kemajuan teknologi dan informasi menjadi infrastruktur penopang bergeraknya globalisasi dan ekonomi neoliberal. Melalui teknologi informasi, pemegang modal raksasa di sektor keuangan dan industri dengan mudah memindahkan modalnya dari satu negara kenegara yang lain. Saat ini Indonesia telah satu tahun memasuki pasar bebas dan bersiap memasuki *AEC (Asean Economic Community)* di tahun 2015, perkembangan industri di Indonesia telah melonjak sangat tinggi, termasuk perkembangan di sektor pendidikan pun meningkat pesat.

Perpustakaan pada sektor universitas pada saat ini sangat berperan penting untuk kemajuan bangsa disuatu negara terutama untuk sebuah penelitian. Penting bagi perpustakaan universitas menyediakan refrensi journal-jurnal nasional maupun internasional secara lengkap. Artinya, untuk mencapai keberhasilan peneliti merupakan aspek yang paling menentukan bagi pengetahuan umum. Salah satu kemampuan dasar yang harus dimiliki oleh seorang peneliti adalah kemampuan profesional di bidang tertentu. Kemajuan teknologi dibidang perpustakaan juga sangat berkembang seiringnya zaman digital saat ini bahkan berbagai platform digunakan untuk membangun sistem tersebut.

Masih banyak pada zaman yang sudah serba digital perpustakaan universitas tidak memanfaatkan dengan maksimal. Pendistribusian jurnal keberbagai perpustakaan juga masih menggunakan teknik konvensional, sehingga menyebabkan kurangnya efesien dalam penggunaan waktu pendistribusian.

Resource Oriented Architecture (ROA) suatu teknologi arsitektur pengembangan perangkat lunak dengan pendekatan layanan, memungkinkan hubungan dan pertukaran data atau informasi antar bagian menjadi mudah. Resources ini dapat memaparkan fungsi-fungsi dari mereka melalui tampilan interface. Melalui standar bahasa *Web Service Definition Language (WSDL)* atau *Web Application Definition Language (WADL)* sehingga penamaan deskripsi pada suatu Resource mudah dimengerti oleh manusia.

Arsitektur REST, yang umumnya dijalankan via *HTTP (Hypertext Transfer Protocol)*, melibatkan proses pembacaan halaman web yang memuat sebuah file JSON. File inilah yang akan menguraikan dan memuat konten yang hendak disajikan. (Roberto Lucchi, 2008) pada penelitiannya dengan judul “*Resource Oriented Architecture and Rest*”, di mereview konseptual REST style, dengan kesimpulan bahwa RESTful Web Service telah menjadi alternatif yang sangat baik untuk kompleksitas pada WS-*.

Berdasarkan kesimpulan diatas maka Platform teknologi *Resource Oriented Architecture (ROA)* dan REST sebagai solusi untuk spasialis infrastruktur data terdistribusi. Sehingga journal-jurnal diberbagai universitas dapat diberikan kepada beberapa perpustakaan dan mendapatkan

informasi yang lengkap dan terseruktur di berbagai platform dan mengetahui *performance* pada setiap platform.

B. RUMUSAH MASALAH

1. Identifikasi Masalah

Identifikasi masalah dalam pendistribusian data journal menggunakan RESTful APIs berbasis multiplatform.

- a. Perpustakaan yang memiliki journal tidak menggunakan konsep ROA.
- b. Sistem journal masih konvensional.
- c. Belum ada sistem pendistribusian data yang bersifat multiplatform..

2. Problem Statement

Belum adanya sistem pendistribusian datayang berbasis multiplatform sehingga journal diberbagai universitas lambat jika dilakukan secara konvensional .

3. Research Question

Bagaimana penerapan *Resource Oriented Architecture* untuk pendistribusian data journal keberbagai perpustakaan yang berbasis secara multiplatform.

C. MAKSUD DAN TUJUAN

1. Maksud

Menerapkan metode *Resource Oriented Architecture* menggunakan RESTful API pada web service berbasis multiplatform.

2. Tujuan Penelitian

memberikan suatu solusi untuk pendistribusian data yang bersifat multiplatform diterapkan di sistem perpustakaan. Serta meninjau platform yang mana yang cocok digunakan untuk data terdistribusi dan mencari kelemahan atau kelebihan antar platform.

D. KEGUNAAN DAN MANFAAT PENGEMBANGAN

1. Kegunaan

Kegunaan penelitian ini adalah untuk dapat mendistribusikan data journal ke perpustakaan universitas dengan berbagai multiplatform.

2. Manfaat

- a. Manfaat teoritis ini adalah melanjutkan penelitian sebelumnya dan memberikan sumbangan terhadap IPTEK khususnya untuk membuktikan *Resource Oriented Architecture* dapat digunakan untuk pendistribusian data ke berbagai platform menggunakan *RESTful API* sebagai *development*-nya.
- b. Manfaat praktis pada penelitian ini adalah untuk memudahkan pendistribusian data journal dari berbagai universitas secara efektif dan efisien.

E. PENTINGNYA PENGEMBANGAN

Penelitian ini dibangun karena adanya masalah dalam pendistribusian data-data journal keberbagai perpustakaan yang tidak terorganisir dengan baik.

Penelitian ini memberikan solusi untuk permasalahan tersebut, yang diuraikan sebagai berikut:

1. Dapat memberikan referensi journal bagi peneliti yang sedang melakukan penelitian.
2. Melengkapi journal yang ada diperpustakaan tersebut.
3. Memudahkan pengintegrasian data yang terdistribusi karna bersifat multiplatform

F. RUANG LINGKUP DAN KETERBATASAN

Adapun keterbatasan dari penerapan ini adalah:

1. Data journal yang disediakan harus lengkap dan valid
2. Server yang digunakan harus besar untuk menampung journal diberbagai universitas
3. Security system harus sangat diperhatikan untuk menjaga sebuah database, karena data journal pada setiap sistem perpustakaan tergantung pada system yang didistribusikan oleh provider

G. DEFINISI ISTILAH

1. WSDL (*Web Services Description Language*) adalah sebuah XML-based language untuk mendeskripsikan XML.
2. Web service adalah aplikasi perangkat lunak yang dapat diakses secara remote oleh berbagai piranti dengan sebuah perantara tertentu.
3. Representasi adalah proses dimana sebuah objek ditangkap oleh indra seseorang, lalu masuk ke akal untuk diproses yang hasilnya adalah sebuah konsep/ide yang dengan bahasa akan disampaikan/diungkapkan kembali.
4. Resources adalah segala informasi yang dapat diberikan nama, seperti, dokumen, gambar atau daftar masalah terbuka dalam versi software.
5. *Hypertext Transfer Protocol* (HTTP) adalah sebuah protokol lapisan jaringan aplikasi yang digunakan untuk sistem informasi terdistribusi, kolaboratif, dan menggunakan hipermedia.
6. URI (*Uniform Resource Identifier*) adalah pengidentifikasi dokumen tunggal dan dituliskan dalam satu baris teks.
7. URL (*Uniform Resource Locator*) adalah pengidentifikasi sumberdaya di Internet yang dituliskan dalam satu baris teks.

[Halaman ini sengaja dikosongkan]

BAB II

KERANGKA TEORITIS

A. TINJAUAN PUSTAKA

1. "Developing Distributed System with Service Resource Oriented Architecture" oleh Hermawan dari Fakultas Teknik Informatika, Universitas Trunojoyo Madura, Indonesia dan Rianarto Sarnom, Fakultas Teknik Informatika, Institut Teknologi Sepuluh Nopember, Indonesia, 2012. Penelitian ini menjelaskan mengenai fungsionalitas aplikasi sebagai sebuah layanan melalui teknologi layanan web yang disebut dengan *Simple Object Access Protocol (SOAP)*. Adapun SOAP bersifat statis dan berorientasikan pada metode layanan sehingga memiliki keterbatasan dalam pembuatan dan pengaksesannya pada jumlah layanan yang besar. Karena itu dalam penelitian tersebut dilakukan kombinasi *Service Oriented Architecture (SOA)* dan *Resource Oriented Architecture (ROA)* yang berorientasi pada sumberdaya layanan menggunakan *REpresentational State Transfer (REST)* untuk memperluas cakupan layanan. Kombinasi arsitektur ini disebut sebagai *Service Resource Oriented Architecture (SROA)*. SROA mampu mengoptimalkan distribusi aplikasi dan integrasi layanan yang diimplementasikan untuk membangun perangkat lunak manajemen. Untuk merealisasikan model ini, perangkat lunak dibangun sesuai dengan kerangka kerja *Agile Model Driven Development (AMDD)* untuk mengurangi kompleksitas pada semua tahapan proses pembangunan perangkat lunak.
2. "Design & Development of a REST based Web Service Platform for Applications Integration on Cloud" oleh Ritesh Sinha, Manisha Khatkar, Subhash Chand Gupta dari Amity University, India, September 2014. Penelitian ini mengenai pendekatan REST based method untuk komunikasi dan transfer data dari satu aplikasi ke aplikasi lain menggunakan Cloud Platform seperti Force.com platform pada Salesforce. Penelitian ini menggunakan standar *Simple Object Access Protocol (SOAP)*, *Extensible Markup Language (XML)* dan *Java Script Object Notation (JSON)* untuk berinteraksi dengan Cloud. Tujuan pada penelitian ini adalah untuk Design & Development pada REST berbasis Web Service platform untuk integrasi dengan Cloud. Penelitian ini juga diselesaikan dengan Apex language, The Proprietary Language pada Salesforce.
3. "Preventing Buffer Overflow DOS Attack in Restful Services" oleh Hussam A. Elkurd , Tawfeeq S.Barhoom, Fakultas Teknologi Informasi, Islamic University Gaza, Gaza, Palestine 2015 yang diterbitkan oleh International Journal of Information and Communication Technology Research. Pada penelitian ini bermaksud untuk membuat sebuah mekanisme pada pencegahan serangan DOS buffer overflow dan membatasi waktu berlaku *token authorized* yang telah ditentukan. Selain itu penelitian ini menerapkan sebuah web RESTful API dan menunjukkan hasil yang mampu mencegah serangan DOS Attack. Salah satu cara mereka untuk mencegah serangan tersebut dengan Token Lifetime biasa digunakan pada akses Authorized pada resources, token tersebut digenerated dari provider dengan limit interval, jika batas waktu telah berakhir maka token menjadi tidak valid dan token baru harus digenerated kembali.

Untuk mekanisme pencegahan ini mereka membuat sebuah REST API menggunakan PHP script.

4. "Rancang Bangun Perangkat Lunak Aplikasi Kesehatan Berbasis Service Oriented Architecture" oleh Sarwosri, Farah Naja, Fakultas Teknologi Informasi, Jurusan Teknik Informatika di Institute Teknologi Sepuluh Nopember, 2011. Dalam penelitian ini peneliti membangun sebuah aplikasi rekam medis untuk memudahkan pasien dalam mendapatkan info dan fasilitas kesehatan yang dibutuhkan tanpa perlu mengunjungi satu persatu web rumah sakit menggunakan *Service Oriented Architecture* (SOA). Di penelitian ini juga mereka membagi proses bisnis menjadi dua, yaitu: proses bisnis aplikasi pelayanan kesehatan dan proses bisnis aplikasi rumah sakit.
5. "Perancangan Basis Data dan Layanan Akses Berbasis Service Oriented Architecture (SOA) Untuk Dinas Kesehatan Kabupaten Sleman" oleh Inna Yoana Sari Tarigan S, Soedjatmiko, Rudy Hartanto, Program Studi Magister Teknologi Informasi, Program Pasca Sarjana Universitas Gadjah Mada, Yogyakarta, Indonesia, 2010. Pada penelitian ini peneliti membantu sebuah proses pengambilan keputusan kebijakan dan sector kesehatan dengan memanfaatkan teknologi komunikasi dan informasi. Penelitian ini juga bertujuan untuk merancang sebuah basis data dan layanan akses berbasis SOA untuk pusat data transaksional pada Dinas Kesehatan Kabupaten Sleman. Data transaksional dalam penelitian ini adalah data rekam medis pasien, logistic alat-alat kesehatan, pengirimannya, catatan penyimpanan, catatan perpindahan tempat termasuk data tenaga medis disebuah puskesmas. Jalannya penelitian yang dilakukan meliputi tahap analisa kebutuhan dan tahap pengembangan sistem. Tahap analisa kebutuhan dilakukan dengan pengumpulan data studi literatur dan survey kondisi data yang tersedia dipuskesmas dan kebutuhan informasi Dinas Kesehatan Kabupaten Sleman. Sedangkan tahap pengembangan sistem meliputi: Melakukan analysis terhadap sistem informasi puskesmas dan basis data puskesmas, membuat rancangan basis data yang dapat mengintegrasikan data transaksional puskesmas, membuat *prototype* aplikasi untuk mengirimkan data puskesmas ke basis data yang dirancang, menguji *prototype* aplikasi dengan mengirimkan data puskesmas ke basis data, membuat report dan *prototype* sistem untuk mengakses report tersebut, guna memenuhi kebutuhan informasi Dinas Kesehatan Kabupaten Sleman, menentukan layanan berbasis SOA untuk mengakses data (informasi) yang dimiliki Dinas Kesehatan, membuat sample client untuk mengakses layanan yang tersedia (untuk kepentingan pengujian), melakukan pengujian ketersediaan layanan dengan mengakses layanan menggunakan program sample yang telah dibuat.
6. "REST and Resource-Oriented Architecture" oleh Roberto Lucchi, University of Western Ontario London, Romania 2011. Pada penelitian ini peneliti mereview REST style, konseptual dan karakteristik ROA, Security pada RESTful Web Service. Pada kesimpulannya peneliti menyimpulkan bahwa, REST adalah sebuah gaya arsitektur yang dibuat untuk menjelaskan sebuah hypermedia distributed system. Kontribusi utama pada journal ini adalah untuk meninjau aspek penting dari arsitektual REST dan Resource Oriented Architecture pada

perbandingan dari WS-*. Dan juga hasil yang lain bahwa, RESTful Web Service telah menjadi alternatif yang sangat baik untuk kompleksitas pada WS-*.

Berdasarkan penelitian yang berhubungan diatas, maka judul yang diajukan pada penelitian ini adalah “Penerapan *Resource Oriented Architecture* Untuk Pendistribusian Data Menggunakan RESTful APIs Berbasis Multiplatform”. Judul tersebut didasari oleh permasalahan di perpustakaan universitas, yaitu belum adanya pendistribusian data journal dari berbagai perpustakaan ke perpustakaan lain yang mengakibatkan lambatnya pencarian data journal oleh peneliti sebagai referensinya. Sedangkan variable yang digunakan adalah *http method* diantaranya yaitu GET dan POST. Berdasarkan penelitian-penelitian sebelumnya terdapat saran untuk menggunakan metode lain selain *Resource Oriented Architecture* yaitu dengan metode *Service Oriented Architecture* sebagai pendistribusian data.

B. LANDASAN TEORI

1. Resource Oriented Architecture (ROA)

ROA didasarkan pada the concept of resource, setiap resource adalah komponen sebuah pendistribusian yang dapat diakses secara langsung yang ditangani oleh beberapa standar. Resource interface semantics didasari oleh operasi-operasi HTTP (Roberto Lucchi, 2008). Pada ROA, prinsip REST menjadi sangat terkait dengan Web protocols, seperti *Uniform Resource Identifier* (URIs), *Hypertext Transfer Protocol/HTTP* and *eXtensible Markup Language* (XML) (R. Fähnrich, 2009). Berikut table rangkuman metode-metode resource dan bagaimana mereka dapat diterapkan pada protokol *Hypertext Transfer Protocol* (HTTP) menurut, (Roberto Lucchi, 2008).

HTTP operations	CRUD	Description
POST	Create	Menciptakan sebuah resource
GET	Read	Membaca sebuah resource atau collection
PUT	Update	Biasa digunakan untuk update sebuah data
DELETE	Delete	Menghapus sebuah resource

Table 1: *HTTP methods*, (Roberto Lucchi, 2008)

Definisi status HTTP codes pada HTTP protocol, berikut table2 yang dijelaskan oleh (Ritesh Sinha, 2014).

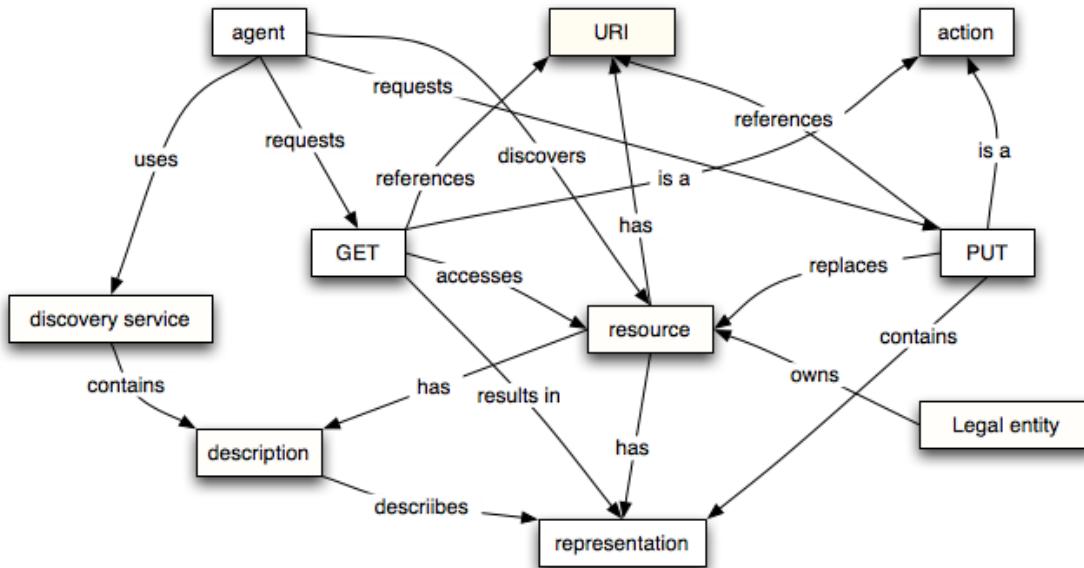
Status	Code	Description
Success	200	Berhasil mengambil data, mengubah dan menghapus

Unauthorized	401	kredensial tidak tersedia atau salah
Not Found	404	Service tidak disediakan oleh provider
Internal server error	500	Kesalahan tak terduga
Unprocessable Entity	422	Tidak berhasil menyimpan, mengubah atau menghapus data

Table 2: Menampilkan kode HTTP status, (Roberto Lucchi, 2008)

Jelas, bahwa konsep utama ROA adalah *resource/informasi*. Dalam struktur ini, sebuah *resource* memiliki nama alamat yang dapat direpresentasikan oleh URI (Universal Resource Identifier).

Istilah 'RESTful Web Services' pada umumnya digunakan untuk menunjukkan Web Service berdasarkan style ini. Menurut (Leonard Richardson, 2007), mereka mengatakan, definisi sebuah arsitektur baru, dikenal sebagai *Resource Oriented Architecture (ROA)*, untuk mendekvelop sebuah RESTful Web Services.



Gambar 1: Konsep dan relasi pada Resource Oriented Model, (W3C, 2003)I

Resource Oriented Model dapat menampilkan bagaimana *resources* merupakan konsep yang independen, namun manipulasi pada *resources* adalah sebuah turunan dari Service Model: dengan jenis tertentu pada *services* dan diidentifikasi pada *resources* (W3C, 2003).

2. RESTful APIs

REST adalah suatu metode komunikasi yang sering diterapkan dalam pengembangan layanan berbasis web. REST, yang umumnya dijalankan via HTTP (Hypertext Transfer Protocol), 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 (Saputra, 2015)[6].

Kekhasan REST terletak pada interaksi antara klien dan server yang difasilitasi oleh sejumlah tipe operasional (verba) dan Universal Resource Identifiers (URIs) yang unik bagi tiap-tiap sumberdaya. Masing-masing verba – GET, POST, PUT dan DELETE – memiliki makna operasional khusus untuk menghindari ambiguitas (Saputra, 2015).

RESTful platform itu sendiri didasari oleh *RESTdevelopment technology* yang dihasilkan oleh ROA (Roberto Lucchi, 2008). Layanan berbasis web yang biasa menggunakan arsitektur REST, semacam itu dinamakan RESTful APIs (*Application Programming Interface*) atau REST APIs.

API (*Application Programming Interface*) telah menjadi bagian penting dari industri komputer sejak awal. mereka mendasar untuk cara komputer, perangkat lunak, dan jaringan arsitektur telah berkembang.

Sebuah representasi dapat dimasukkan kedalam pesan dan diproses oleh penerima sesuai dengan data kontrol pesan dan sifat dari jenis mediayang dikirimkan (Fielding, 2000). Beberapa jenis yang dapat dimasukan diantaranya berupa pesan yang dapat dilihat rapih oleh pengguna, sehingga pesan yang disampaikan mudah dimengerti.

3. JSON

JSON (*JavaScript Object Notation*) adalah adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer (json.org, 2000). JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data.

JSON memiliki dua struktur:

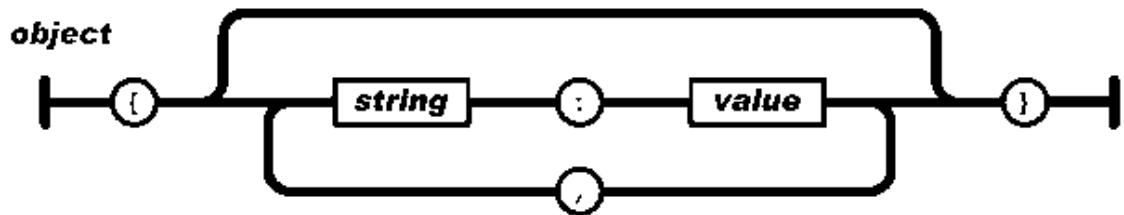
8. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (object), rekaman (record), struktur (struct), kamus (dictionary), tabel hash (hash table), daftar berkunci (keyed list), atau associative array.
9. Daftar nilai terurutkan (an ordered list of values). Pada kebanyakan bahasa, hal ini dinyatakan sebagai (array), vektor (vector), daftar (list), atau urutan (sequence).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemrograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini (json.org, 2000).

Dibawah ini beberapa bentuk dari format data JSON

1. Object

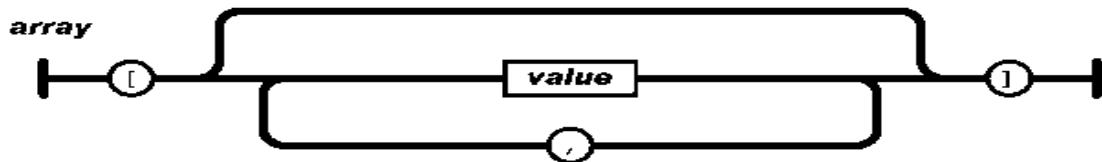
Object adalah sepasang nama/nilai yang tidak terurutkan. Biasa dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma), (json.org, 2000).



Gambar 2: Struktur (object) pada format data JSON, (json.org, 2000)

2. Array

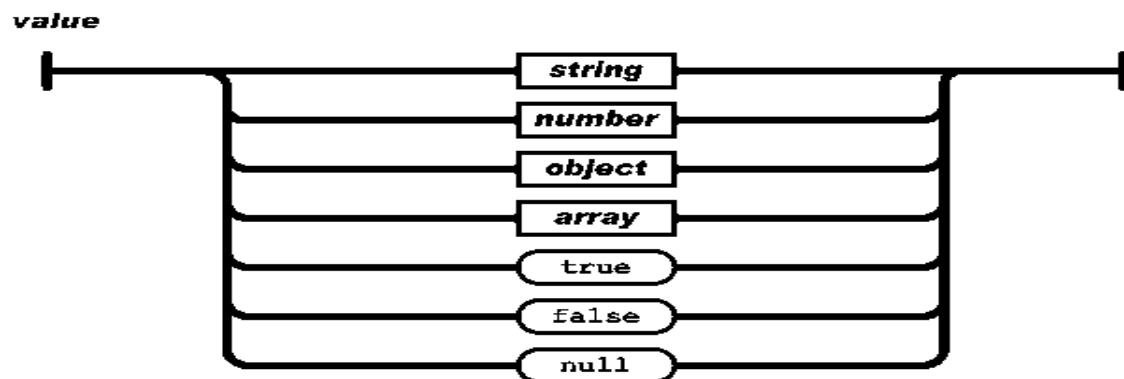
Array adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma), (json.org, 2000).



Gambar 3: Struktur (array) pada format data JSON, (json.org, 2000)

3. Value

Nilai (value) dapat berupa sebuah **string** dalam tanda kutip ganda, atau **angka**, atau



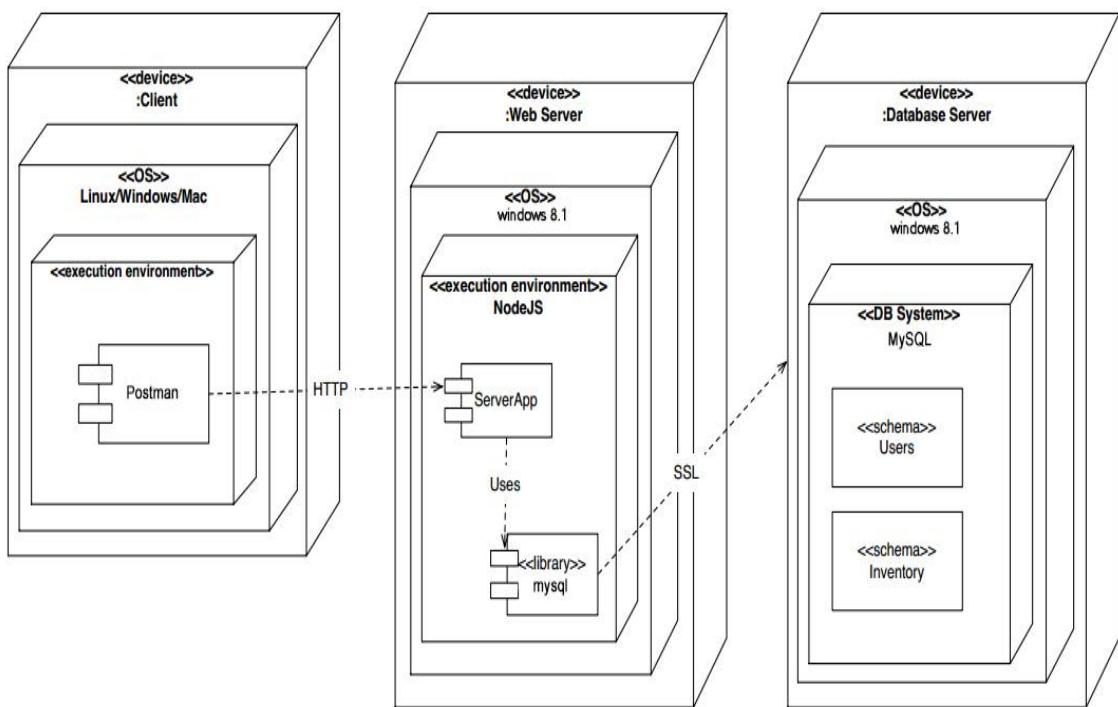
Gambar 4: Struktur (value) pada format data JSON, (json.org, 2000)

true atau false atau null, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat, (json.org, 2000).

4. Node.js

Node.js merupakan bahasa pemrograman Javascript yang dijalankan di serverside. Tapi walaupun begitu Node.js tidak termasuk dalam Javascript Framework (Orsini, 2013).Javascript merupakan bahasa pemrograman yang lengkap hanya saja selama ini di pakai sebagai bahasa untuk pengembangan aplikasi web yang berjalan pada sisi client atau browser saja. Tetapi sejak ditemukannya Node.js oleh Ryan Dahl pada tahun 2009, Javascript bisa digunakan sebagai bahasa pemrograman di sisi server kelas dengan PHP, ASP, C#, Ruby dll dengan katak lain Node.js menyediakan platform untuk membuat aplikasi Javascript dapat dijalankan di sisi server.

Javascript biasa digunakan dibagian interface dengan Node bisa dilakukan dengan 2 prespektif yaitu client dan server, berikut gambar deployement diagaram (Dahl, 2013).

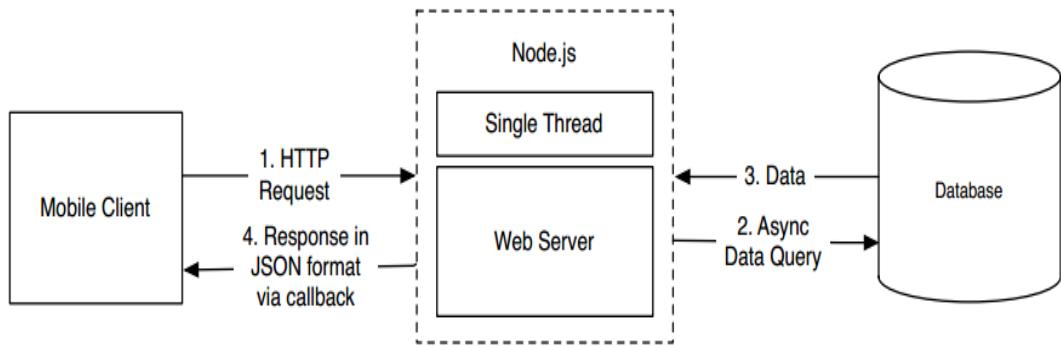


Gambar 5: Deployment Diagram pada sistem Node.js, (Dahl, 2013)

Node.js dibagi menjadi dua komponen: core dan modules. Core dibuild menggunakan C dan C++. Itu kombinasi dari Google V8 Javascript dengan Node's Libuv library dan protokol yang digunakan seperti sockets dan HTTP. Proses instalasi Node sangatlah mudah dengan mengakses platform pada website sehingga user bisa menginstall dengan versi yang sesuai (Benjamin San Souci, 2014).

a. Architectural Description

Node JS menggabungkan jaringan arsitektur untuk efisiensi penginputan data dan menghasilkan sesuai dengan output. Ini mengikuti struktur yang sangat modular, tapi tetap erat digabungkan ke *library*, khususnya mesin V8 (Letaifa, 2011). Sebagai platform harus berkoordinasi input dan output data pada berbagai *sistem terdistribusi*.



Gambar 6: Gambar diatas menggambarkan proses NodeJS merequest data akses kedatabase menurut (Benjamin San Souci, 2014)

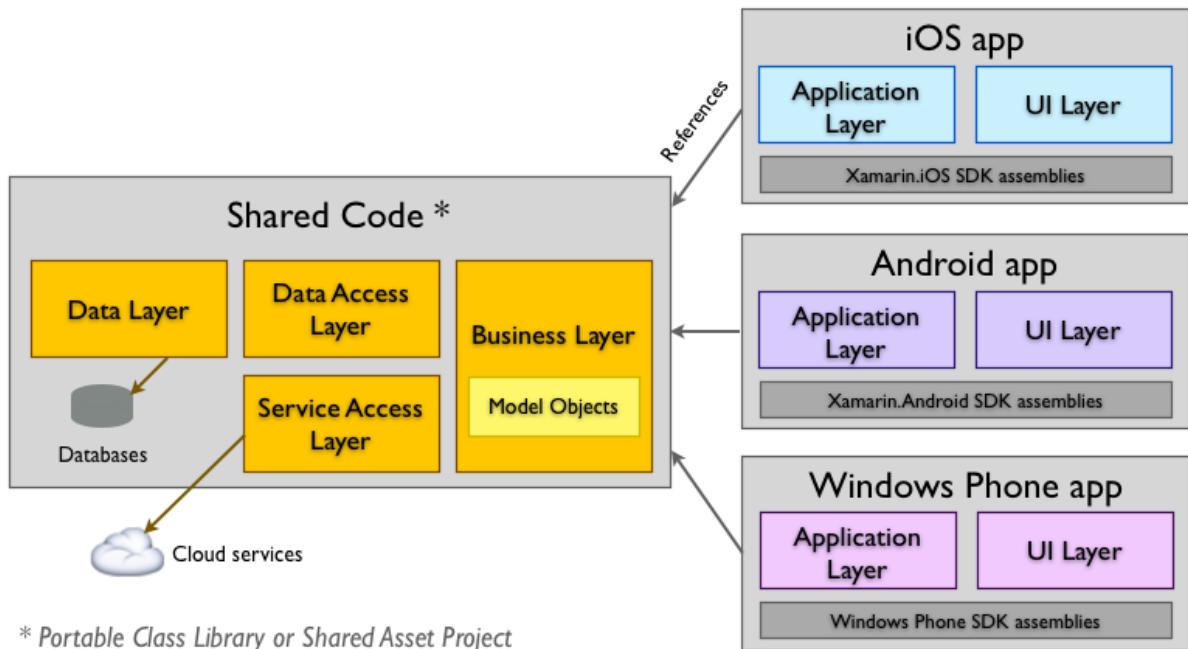
b. Distributed Style

Distributed Style architecture membahas keputusan desain yang penting sehingga terkait dengan collection computational dan perangkat penyimpanan yang berkomunikasi melalui jaringan. Pada umumnya, ini terdiri dari komputer remote secara independen mengeksekusi aplikasi, dan berkomunikasi dengan server independen melalui jaringan.

5. Cross Platform Development

Cross platform development yang muncul untuk menghadapi developers yang akan menerapkan aplikasi mereka dalam satu tahapan untuk berbagai platform, menghindari pengulangan development dan meningkatkan produktivitas (Henning Heitkötter, 2012).

Tahap integrasi adalah dimana aplikasi mobile bisa dilakukan pada fasilitasi IT perusahaan, *cloud* infrastruktur dan kemampuan perangkat asli. Integrasi akan membantu memperluas fungsi dengan fitur tambahan yang bergantung pada sistem lain (Redda, 2012).



Gambar 7: Ilustrasi arsitektur aplikasi pada cross platform (Xamarin, 2014).

6. Database

Sebuah *database* dapat digambarkan sebagai repository untuk data. Dengan jelas ini bahwa membangun database benar-benar merupakan kelanjutan kegiatan yang dimiliki manusia. Database dapat diterapkan pada hasil setiap pembukuan atau kegiatan perekaman yang terjadi jauh sebelum munculnya era komputer (Robbins, 1995). DBMS (Database Management System) adalah sistem perangkat lunak untuk membuat dan mengelola database (Rouse, 2015). DBMS juga melakukan fungsi pengaturan, pengawasan, pengendalian, pengolahan, dan koordinasi terhadap semua proses yang terjadi pada sistem basis data. Dibawah ini adalah komponen-komponen utama dalam DBMS:

a. Query Language

Digunakan oleh bagian lain dengan sedikit perintah sederhana. Contoh : SQL (Structure Query Language), QBE (Query By Example)

b. DML (Data Manipulation Language)

Terdiri dari perintah-perintah yang disediakan dalam program aplikasi untuk melakukan manipulasi data seperti append, list, atau update

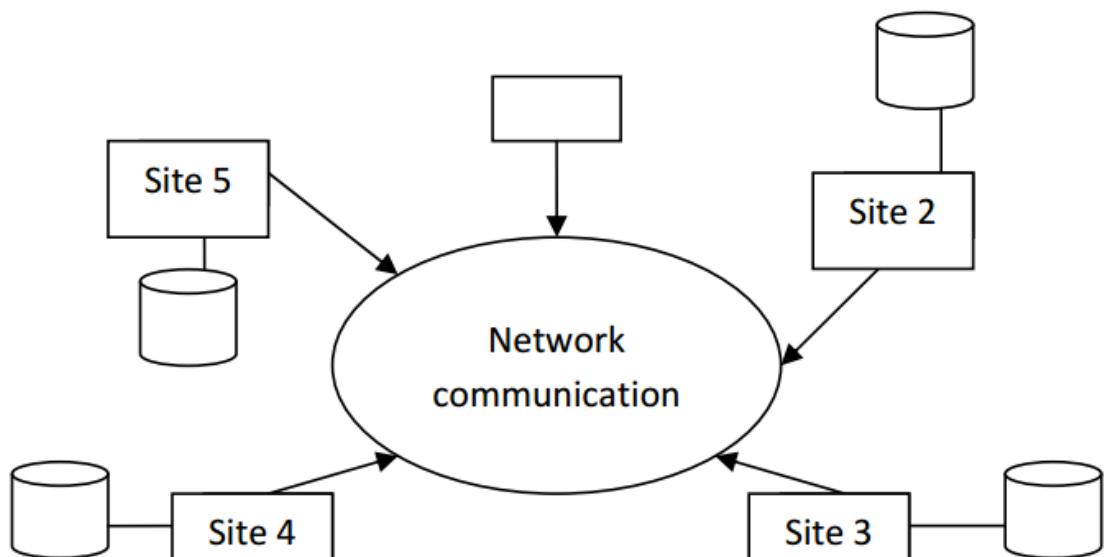
c. DDL (Data Definition Language)

Dengan bahasa ini kita dapat membuat tabel baru, membuat indeks, mengubah tabel, menentukan struktur tabel, dll. Hasil dari kompilasi perintah DDL menjadi Kamus Data, yaitu data yang menjelaskan data sesungguhnya. Contoh : Create, Modify report, Modify structure

7. Distributed Database System

Database terdistribusi sebagai kumpulan beberapa database secara logis saling terkait didistribusikan melalui jaringan komputer. Sebuah sistem manajemen database terdistribusi (DBMS terdistribusi) lalu didefinisikan sebagai sistem perangkat lunak yang memungkinkan manajemen database terdistribusi dan membuat distribusi transparan kepada pengguna (Aspects of the design of distributed databases, 2011). Biasanya (DDBS) digunakan bersama-sama untuk merujuk ke database terdistribusi dan DBMS terdistribusi. Dua istilah penting dalam definisi ini "secara logis saling terkait" dan "didistribusikan melalui jaringan komputer."

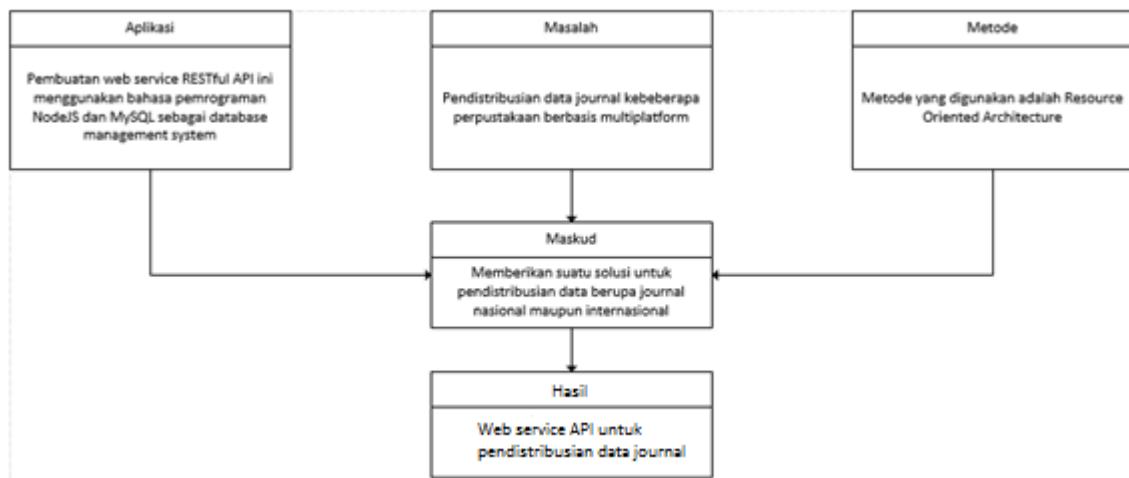
Distribute Database System tidak hanya "sebuah koleksi pada file" tapi itu bisa menjadi media penyimpanan individual dalam setiap node jaringan pada komputer (Aspects of the design of distributed databases, 2011).



Gambar 8: Environtment Distribute Database System, (Aspects of the design of distributed databases, 2011)

C. KERANGKA PEMIKIRAN

Berikut merupakan kerangka pemikiran pemecahan masalah dalam penelitian ini yang digambarkan pada gambar dibawah ini.



Gambar 9: Kerangka pemikiran

Dari gambar 8 maka dapat dijelaskan kerangka pemikirannya, yaitu:

1. Terdapat permasalahan distribusi data jurnal keberbagai perpustakaan
2. Dengan adanya metode *Resource Oriented Architecture* (ROA) distribusi data dapat dilakukan dengan efisien.
3. RESTful API digunakan sebagai media penerapan ROA
4. Pemrograman NodeJS digunakan untuk menarik resource data jurnal dari database
5. Database yang digunakan adalah MySQL, dengan adanya MySQL yang berorientasi RDBMS dapat memungkinkan relasi data keberbagai table menjadi lebih mudah
6. Database ditangani oleh aplikasi sehingga data dapat terdistribusi keberbagai multiplatform menggunakan format data JSON sebagai pertukaran data.

[Halaman ini sengaja dikosongkan]

BAB III

METODE PENGEMBANGAN

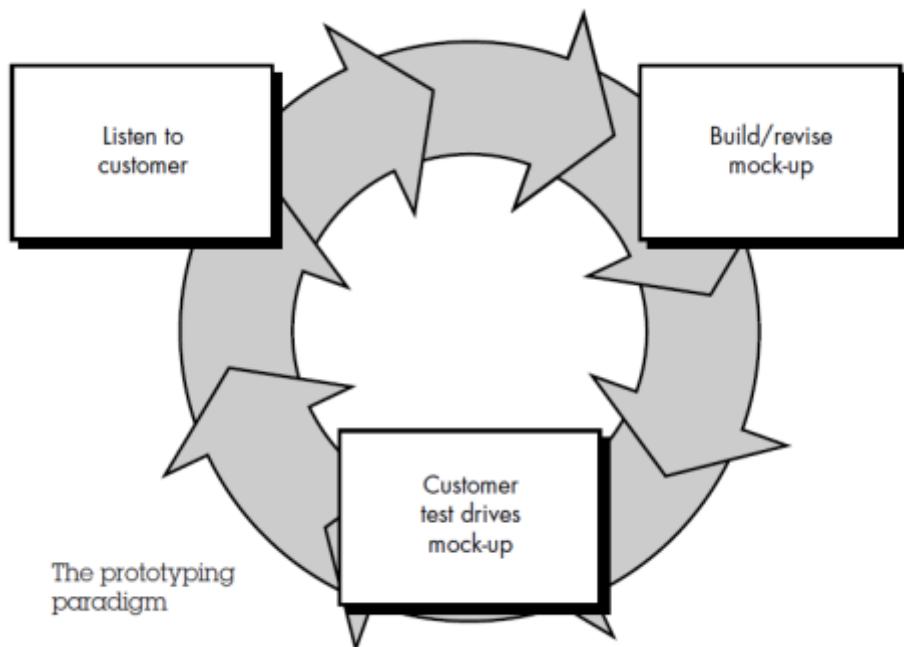
A. MODEL DAN PROSEDUR PENGEMBANGAN

1. Model Pengembangan

Proses pengembangan sistem seringkali menggunakan pendekatan prototipe (prototyping). Metode ini sangat baik digunakan untuk menyelesaikan masalah kesalahpahaman antara user dan analis yang timbul akibat user tidak mampu mendefinisikan secara jelas kebutuhannya (Mulyanto, 2009).

Prototyping adalah pengembangan yang cepat dan pengujian terhadap model kerja (prototipe) dari aplikasi baru melalui proses interaksi dan berulang-ulang yang biasa digunakan ahli sistem informasi dan ahli bisnis. Prototyping disebut juga desain aplikasi cepat (rapid application design/RAD) karena menyederhanakan dan mempercepat desain sistem (O'Brien, 2005).

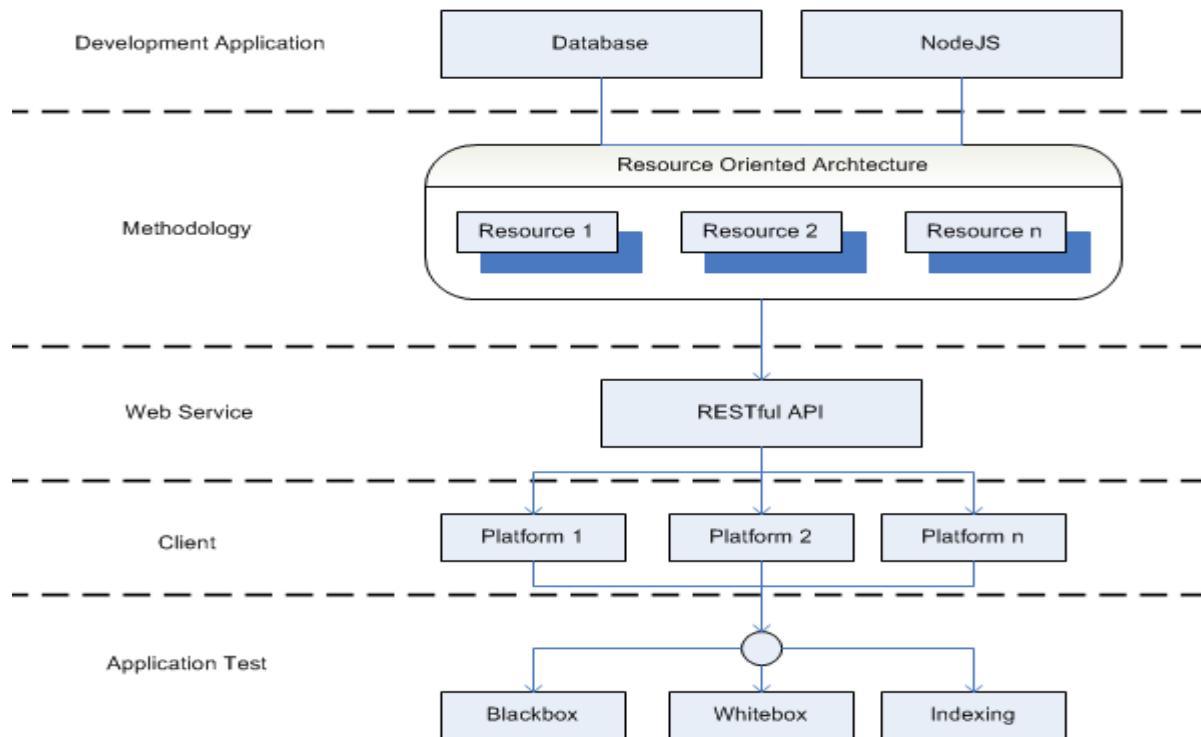
Model tersebut dapat diamati pada gambar 9 dibawah ini



Gambar 10: Model prototype menurut Roger S. Pressman

2. Prosedur Pengembangan

Merujuk pada model pengembangan yang menggunakan *Prototyping*, maka prosedur dalam pengembangan ini berisisi langkah-langkah pembuatan *webservice* untuk pendistribusian data journal



Gambar 11: Prosedur pengembangan

a. Development Application

Pada tahap ini adalah tahap untuk membangun sebuah aplikasi berbasis *web service* yang menggunakan database MySQL dan menggunakan NodeJS sebagai bahasa pemrogramannya.

b. Methodology

Metode yang digunakan adalah *Resource Oriented Architecture* sebagai komponen pendistribusian data berbasis *resource* atau sumber data.

c. Web Service

RESTful diperuntukan sebagai *middleware* aplikasi yang mengintegrasikan *resource* kepada platform lain dan HTTP sebagai protokolnya.

d. Client

Pada penelitian pengembangan ini platform-platform adalah sebagai client yang nantinya akan menjadi experiment pendistribusian data

e. Application Test

Pada tahap ini dilakukan pengujian terhadap platform-platform yang menggunakan teknik RESTful API dan diuji dampak yang terjadi saat pendistribusian data dilakukan.

B. UJI COBA PRODUK

1. Desain Uji Coba

Pengujian hanya dilakukan dengan menggunakan metode blackbox, terfokus pada apakah unit program memenuhi kebutuhan yang disebutkan dalam spesifikasi. Pengujian hanya dilakukan dengan menjalankan atau mengeksekusi unit atau modul, kemudian diamati apakah hasil dari unit itu sesuai dengan proses bisnis yang diinginkan

2. Subjek Uji Coba

Yang menjadi subjek uji coba adalah platform dengan bahasa pemrograman Java, Python dan PHP sebagai media pengujian. Pengujian dilakukan untuk mengukur kecepatan dan juga ketepatan data distribusi bagi pengguna. Selain itu pengujian dilakukan untuk mendeteksi kesalahan yang mungkin terjadi dalam penulisan kode program sehingga diharapkan aplikasi dapat berjalan dengan baik.

C. INSTRUMEN PENGUMPULAN DATA

Instrumen pengumpulan data dalam proses penelitian pengembangan ini adalah MySQL untuk database pengumpulan data dan Sublime Text 3 sebagai *text editor* bahasa pemrograman NodeJS, dan uji indexing sebagai mengukur selisih waktu antara setiap platform yang menggunakan RESTful.

D. UJI ANALISIS DATA

Uji analisis data dilakukan untuk mengetahui tujuan penelitian pengembangan yaitu menentukan kecepatan rata-rata waktu antara platform dalam pendistribusian data menggunakan RESTful. Variabel yang diuji adalah menggunakan waktu antara platform-platform yang menggunakan *HTTP request* (GET, POST) dan banyak data yang diberikan.

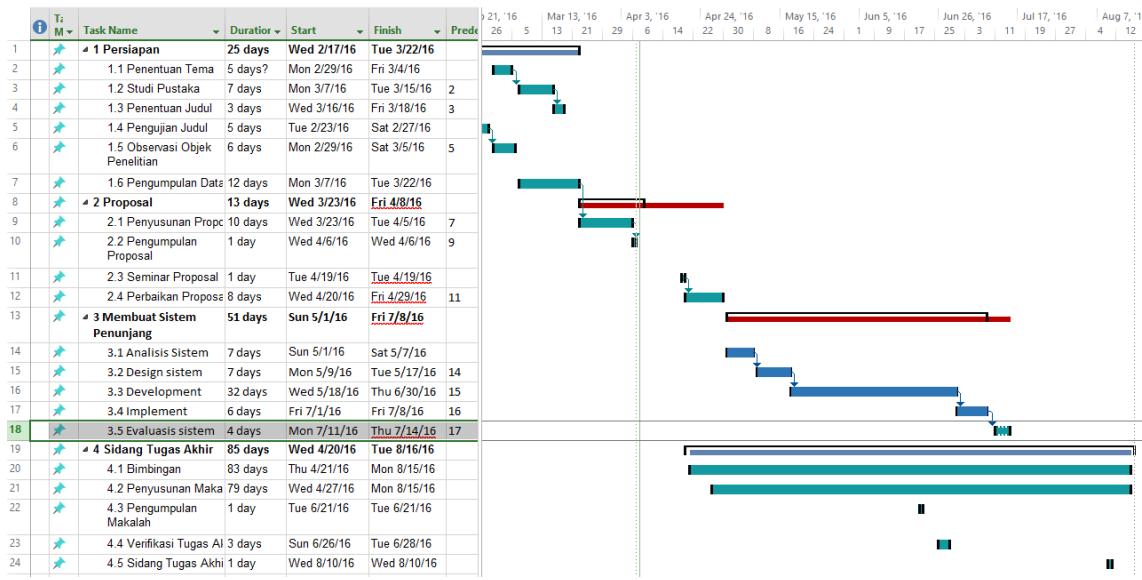
PLATFORM	METHOD	REQUEST	DATA(s)	WAKTU (t)
1				
↓				
n				
Total				

Table 3: Table analisis data

Untuk menentukan kecepatan rata-rata yaitu dengan rumus $v = \frac{s_{total}}{t_{total}}$

E. PENJADWALAN

Penjadwalan dibuat untuk mengetahui proses dari pengembangan dan waktu yang dibutuhkan dalam melakukan pengembangan. Penjadwalan dari penelitian pengembangan ini dapat digambarkan pada gambar 10 berikut:



Gambar 12: Penjadwalan

Dari gambar 12 penjadwalan kegiatan pada penelitian ini dapat dirincikan pada tabel 4.

No.	Kegiatan	Waktu Kegiatan/hari
Persiapan		
1	Penentuan Tema	5
	Studi Pustaka	7
	Penentuan Judul	3
	Pengajuan Judul	5
	Observasi Objek Penelitian	6
	Pengumpulan Data	12
Proposal		
2	Penyusunan Proposal	10
	Pengumpulan Proposal	1
	Seminar Proposal	1
	Perbaikan Proposal	8
	Membuat Sistem Penunjang	
3	Analisis Sistem	7
	Design Sistem	7
	Development	32
	Implement	6
	Evaluasi	4
Sidang Tugas Akhir		
4	Bimbingan	83
	Penyusunan Makalah	79
	Pengumpulan Makalah	1
	Verifikasi Tugas Akhir	3
	Sidang Tugas Akhir	1

Table 4: Rincian jadwal kegiatan penelitian

F. SISTEMATIKA PENULISAN

BAB I PENDAHULUAN

Dalam bab ini berisi latar belakang masalah, rumusah masalah, maksud dan tujuan penelitian, pentingnya penelitian, serta keterbatasan.

BAB II KERANGKA TEORITIS

Dalam bab ini dibahas tentang tinjauan pustaka yang berkaitan dengan penelitian ini, landasan teori mengenai metode *Resource Oriented Architecture* dan RESTful API serta kerangka pemikiran sebagai pemecahan masalah.

BAB II METODE PENGEMBANGAN

Dalam bab ini diuraikan metode yang dipakai dalam pembuatan web service ini.

BAB IV HASIL DAN PEMBAHASAN

Bab ini menampilkan pembahasan dari hasil penelitian.

BAB V KESIMPULAN

Bab ini berisi kesimpulan dari hasil penelitian.

BAB IV

HASIL dan PEMBAHASAN

A. DESKRIPSI OBJEK PENELITIAN

Objek dari penelitian ini ditekankan kepada data journal dari berbagai universitas ataupun instansi yang dikumpulkan dalam satu database dan di *compile* menjadi beberapa *resource* dengan tipe data *json*. Hal ini dimaksudkan untuk menguji seberapa besar pengaruh antara *resource – resource* tersebut terhadap *multiplatform programming* sebagai objek penelitian diantaranya yaitu Android, PHP dan Python. Penelitian yang akan digunakan adalah menerapkan teknik RESTful APIs dan NodeJS sebagai development-nya serta *Resource Oriented Architecture* sebagai metode penerapan teknik-teknik tersebut.

1. Tentang Aplikasi

Aplikasi ini dibuat menggunakan NodeJS v4.4.7 dan menggunakan database MySQL v5.6. Beberapa plugin untuk mendukung aplikasi RESTful ini antara lain yaitu: express, mysql, body-parser, response-time, crypto.

Aplikasi ini lah yang akan digunakan sebagai web server yang membuat resource-resource itu dapat didistribusikan kepada platform lain. Beberapa resource yang dapat digunakan pada aplikasi ini diantaranya adalah memberikan seluruh data journal, mencari data journal, melihat isi journal, menyimpan data journal dan mengedit data journal.

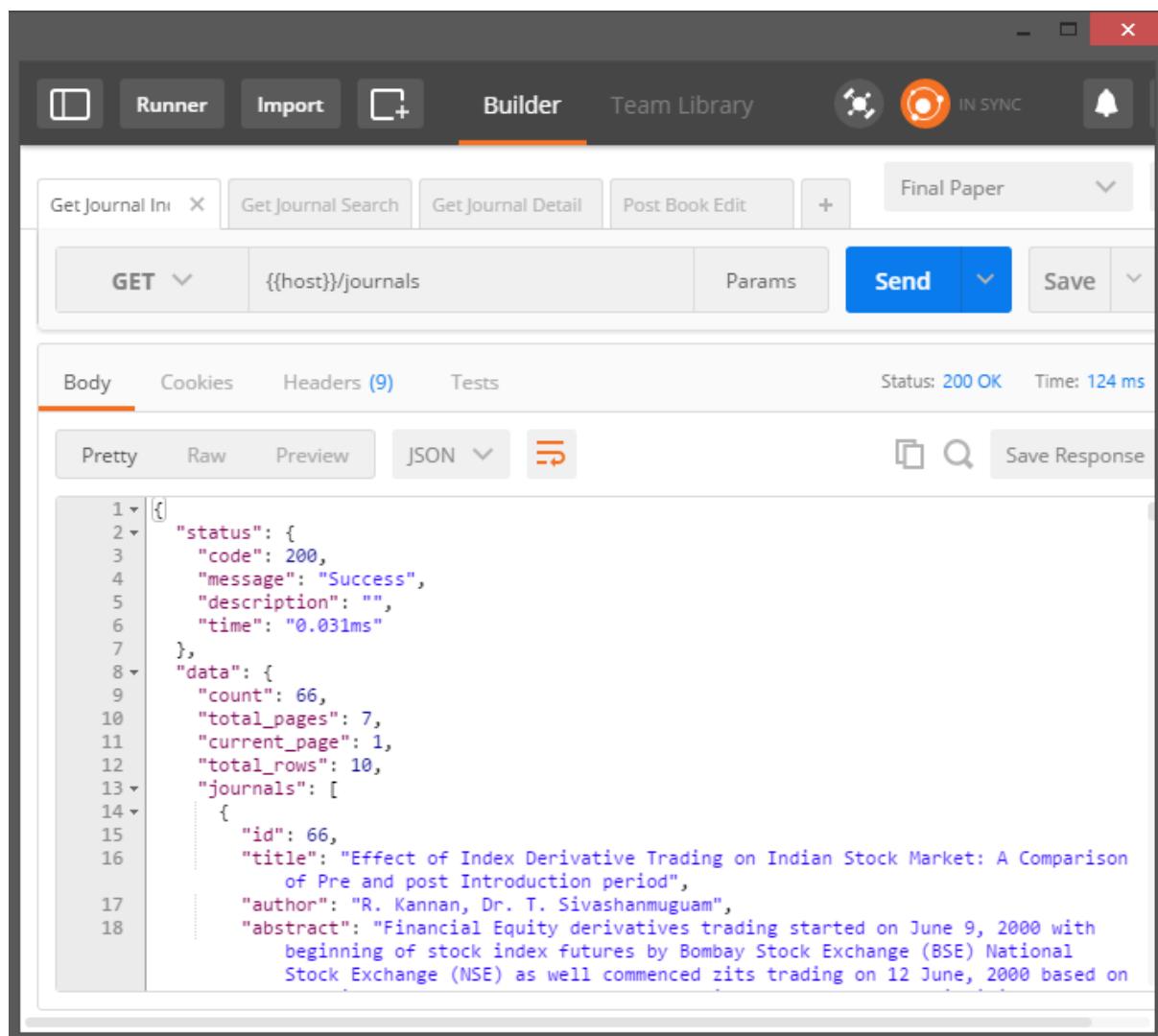
Resource – resource tersebut terbentuk dalam sebuah URL yang berisi data journal bertipe JSON. Request URL yang dapat digunakan dan sesuai standar HTTP diantara lain yaitu:

Method	URL	Deskripsi
GET	<code>{{host}}/books</code>	Untuk menampilkan seluruh journal
GET	<code>{{host}}/books/search/{{keyword}}</code>	Untuk pencarian data journal
GET	<code>{{host}}/books/detail/{{id_buku}}</code>	Untuk menampilkan detail pada journal
POST	<code>{{host}}/books/add</code> Parameter: title, author, abstract, keywords, image, file, issn, publisher, volume, page, number, year, month, date,	Untuk meyimpan data journal
PUT	<code>{{host}}/books/edit/6</code>	Untuk mengedit data journal

Table 5: URL yang dapat digunakan pada aplikasi. Dimana host yang digunakan pada aplikasi RESTful tersebut

2. Pengujian Aplikasi

Untuk pengujian aplikasi ini saya menggunakan aplikasi POSTMAN sebagai tester pada resource-resource yang dapat digunakan.



The screenshot shows the POSTMAN application interface. The top navigation bar includes 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and status indicators for 'IN SYNC' and a bell icon. Below the navigation is a toolbar with buttons for 'Get Journal In', 'Get Journal Search', 'Get Journal Detail', 'Post Book Edit', and 'Final Paper'. The main request configuration area shows a 'GET' method, the URL '{{host}}/journals', and a 'Params' section. To the right are 'Send' and 'Save' buttons. Below the configuration, tabs for 'Body', 'Cookies', 'Headers (9)', and 'Tests' are visible, along with status information: 'Status: 200 OK' and 'Time: 124 ms'. The 'Body' tab is active, displaying a JSON response with line numbers. The response object contains a 'status' object with fields like 'code' (200) and 'message' ('Success'), and a 'data' object with fields like 'count' (66) and 'total_pages' (7). The 'journals' array contains one item, which is expanded to show its details: 'id' (66), 'title' ('Effect of Index Derivative Trading on Indian Stock Market: A Comparison of Pre and post Introduction period'), 'author' ('R. Kannan, Dr. T. Sivashanmugam'), and 'abstract' ('Financial Equity derivatives trading started on June 9, 2000 with beginning of stock index futures by Bombay Stock Exchange (BSE) National Stock Exchange (NSE) as well commenced zits trading on 12 June, 2000 based on').

```
1  {
2    "status": {
3      "code": 200,
4      "message": "Success",
5      "description": "",
6      "time": "0.031ms"
7    },
8    "data": {
9      "count": 66,
10     "total_pages": 7,
11     "current_page": 1,
12     "total_rows": 10,
13     "journals": [
14       {
15         "id": 66,
16         "title": "Effect of Index Derivative Trading on Indian Stock Market: A Comparison of Pre and post Introduction period",
17         "author": "R. Kannan, Dr. T. Sivashanmugam",
18         "abstract": "Financial Equity derivatives trading started on June 9, 2000 with beginning of stock index futures by Bombay Stock Exchange (BSE) National Stock Exchange (NSE) as well commenced zits trading on 12 June, 2000 based on"
19       }
20     ]
21   }
22 }
```

Gambar 13: Hasil dari menampilkan data journal

The screenshot shows the Postman application interface. At the top, there are tabs for 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and status indicators ('IN SYNC' and a bell icon). Below the tabs, a navigation bar includes buttons for 'Get Journal Index', 'Get Journal Se', 'GetJournal Detail', 'Post Book Edit', and a '+' button, along with a dropdown for 'Final Paper'. The main area shows a 'GET' request to {{host}}/journals/search/Effect of index. The response status is 200 OK with a time of 177 ms. The response body is displayed in 'Pretty' format, showing JSON data with a count of 1, total pages of 1, current page of 1, and a single journal entry. The journal entry details include id 66, title "Effect of Index Derivative Trading on Indian Stock Market: A Comparison of Pre and post Introduction period", author "R. Kannan, Dr. T. Sivashanmugam", and abstract text about financial derivatives trading starting on June 9, 2000.

```
8  "data": {  
9   "count": 1,  
10  "total_pages": 1,  
11  "current_page": 1,  
12  "journals": [  
13    {  
14      "id": 66,  
15      "title": "Effect of Index Derivative Trading on Indian Stock Market: A Comparison  
16      of Pre and post Introduction period",  
17      "author": "R. Kannan, Dr. T. Sivashanmugam",  
18      "abstract": "Financial Equity derivatives trading started on June 9, 2000 with  
19      beginning of stock index futures by Bombay Stock Exchange (BSE) National Stock  
20      Exchange (NSE) as well commenced zits trading on 12 June, 2000 based on S&P  
21      Nifty. Consequently, new products like stock futures on individual securities,  
22      index options and options on being securities were introduced. This study  
23      tries to examine the impact of derivatives trading on the stock market  
24      volatility of BSE- 30 Index it also examines the behavior of volatility index  
25      to see whether the market wide volatility has declined over the sample period.  
26      Advance, surrogate indices like BSE-30 Index pre and post second-hand to  
27      assess whether the introduction of derivatives for every se has been  
28      instrumental or the volatility has plummeted in line up with general fall in  
29  }  
30}  
31}
```

Gambar 14: Hasil dari menampilkan data pencarian journal

The screenshot shows the Postman application interface. At the top, there are tabs for 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and status indicators ('IN SYNC' and a bell icon). Below the tabs, a navigation bar contains several items: 'Get Journal Index', 'Get Journal Se' (with a red error dot), 'Get Journal De' (with a close button), 'Post Book Edit', a '+' button, 'Final Paper' (with a dropdown arrow), and a 'Save' button.

The main area displays a 'GET' request to the URL `{{host}}/journals/detail/10`. Below the URL are buttons for 'Params', 'Send' (which is blue), and 'Save'. The 'Body' tab is selected, showing the JSON response. The response is a structured object:

```
8  "data": {  
9    "book": {  
10      "id": 10,  
11      "title": "The Impact of Religion in University on Young University Students an  
Ethnographic Inquiry",  
12      "author": "Indraka Hemantha Ubeysekara",  
13      "abstract": "The existing literature reveals that there are two main reasons  
explaining why youth practice religion in universities: to fulfill psychological  
well-being and to gain life satisfaction. However, these studies failed to  
examine how religion impacts on youth in university contexts. Besides, the  
research studies were largely based on empirical data gathered from limited  
samples, thus, the context-specific differences and diversity of human actions  
were not taken into consideration. Given this scenario, this study explores the  
motivations of young university students to participate in the activities of the  
religion based organizations in universities. The study further examined two sub  
-questions in order to understand why young university students choose religious  
associations in the university; and to gain insights into the specific youth  
-related benefits that university students expect to enjoy as members of the  
those religious groups.",  
14      "keywords": "Religion, Youth, University, Social Capital, Empowerment",  

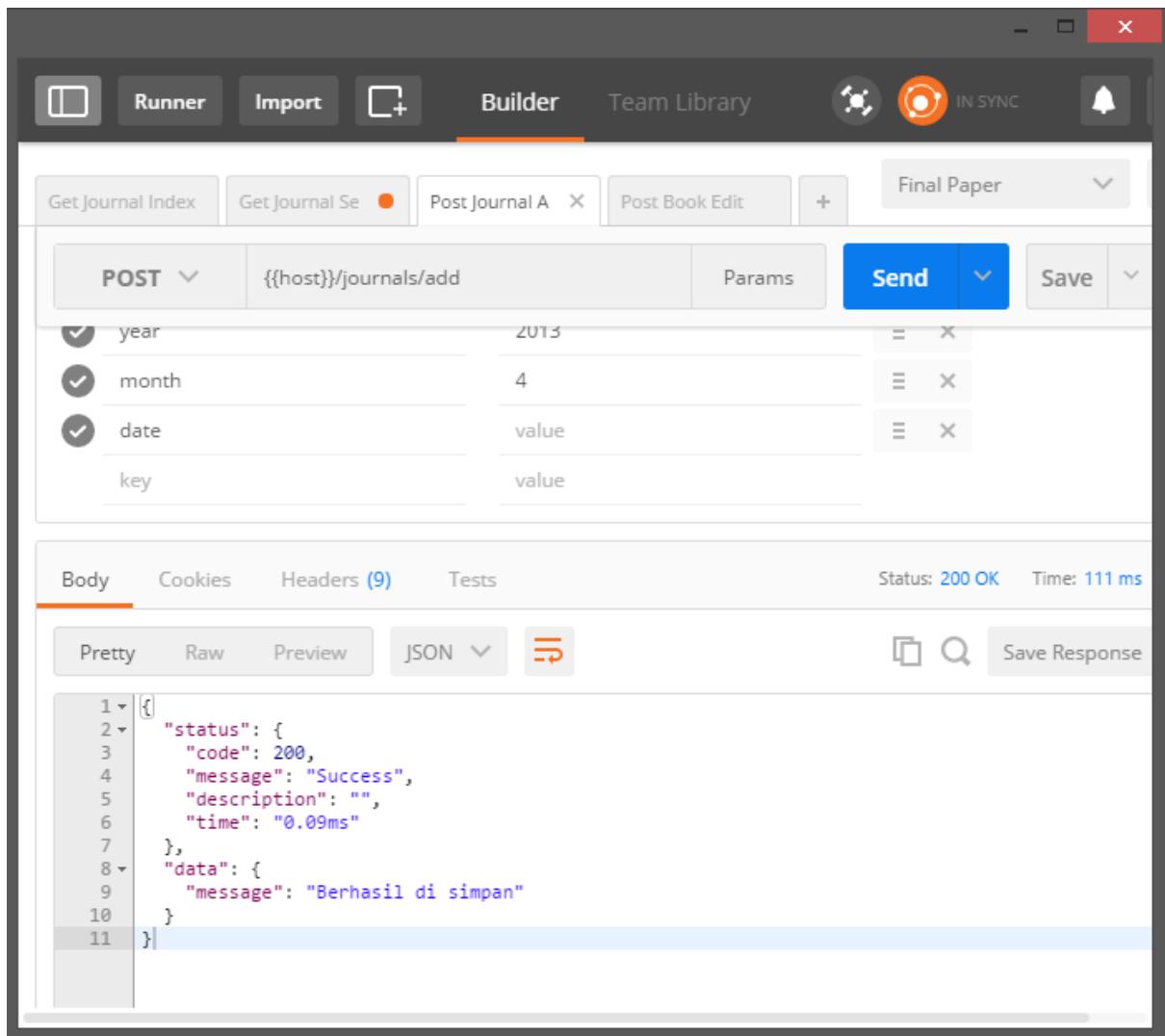
```

Gambar 15: Hasil menampilkan detail journal

The screenshot shows the Postman application interface. At the top, there are tabs for 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and status indicators ('IN SYNC' and a bell icon). Below the tabs, a navigation bar includes 'Get Journal Index', 'Get Journal Se...', 'Post Journal Add', 'Post Journal E...', and 'Final Paper'. A search bar contains the URL {{host}}/journals/edit/66. To the right of the URL are buttons for 'Send' (highlighted in blue) and 'Save'. Underneath the URL, there is a table for parameters with columns 'key' and 'value'. The 'Body' tab is selected, showing a JSON response with status 200 OK and time 210 ms. The response body is a JSON object:

```
1 [{}]
2   "status": {
3     "code": 200,
4     "message": "Success",
5     "description": "",
6     "time": "0.191ms"
7   },
8   "data": {
9     "post": {
10       "title": "Effect of Index Derivative Trading on Indian Stock Market: A Comparison of Pre and post Introduction period",
11       "id": "66"
12     },
13     "message": "Berhasil di update"
14   }
15 }
```

Gambar 16: Hasil dari mengedit data journal



Gambar 17: Hasil dari menambahkan data journal

B. HASIL PENGEMBANGAN

C. PEMBAHASAN

DAFTAR PUSTAKA

- Aspects of the design of distributed databases. (2011). *Journal of Knowledge Management, Economics and Information Technology*.
- Benjamin San Souci, M. L. (2014). An Inside Look at the Architecture of NodeJS. *McGill University*.
- Dahl, R. (2013, December 13). About node. (D. Synodinos, Interviewer)
- Danks, S. (2011, 11). The ADDIE Model. *ASQ The Global Voice of Quality, IV*.
- Fielding, R. T. (2000). Architectures, Architectural Styles and the Design of Network-based Software. *University of California*.
- Henning Heitkötter, S. H. (2012, April). Evaluating Cross-Platform Development Approaches for Mobile Applications. *Springer Berlin Heidelberg*, 140, 120-138.
- json.org. (2000). *JSON is a lightweight data-interchange format. It is easy for humans to read and write*. Retrieved from JSON (JavaScript Object Notation): <http://www.json.org/json-id.html>
- Leonard Richardson, S. R. (2007). *RESTful Web Services*. (M. Loukides, Ed.) United States of America: O'Reilly Media.
- Letaifa, N. (2011, April 10). *Zenika is a firm specializing in Computer Architecture and Agile methods*. Retrieved from Zenika: <http://blog.zenika.com/2011/04/10/nodejs/>
- Orsini, L. (2013, November 7). *ReadWrite is now one of the most widely read and respected tech news sites in the world*. Retrieved from ReadWrite: <http://readwrite.com/2013/11/07/what-you-need-to-know-about-nodejs/>
- R. Fähnrich, K. F. (2009). Proceedings First International Symposium on Services Science ISSS'09. *REST and Resource-Oriented Architecture*, 161.
- Redda, Y. A. (2012, Juni). Cross platform Mobile Applications Development. *Norwegian University of Science and Technology*.
- Ritesh Sinha, M. K. (2014, 8 7). Design & Development of a REST based Web Service Platform for Applications Integration on Cloud. *IJISET - International Journal of Innovative Science, Engineering & Technology*, 1, 387.
- Robbins, R. J. (1995). Database Fundamentals. In R. J. Robbins, *Database Fundamentals*. Johns Hopkins University.
- Roberto Lucchi, M. M. (2008). Resource Oriented Architecture and Rest – Assesment of impact and advantage on INSPIRE. *JRC Scientific and Technical Reports*.
- Rouse, M. (2015, January 23). *This definition is part of our Essential Guide: Relational database management system guide: RDBMS still on top*. Retrieved from database management system (DBMS): <http://searchsqlserver.techtarget.com/definition/database-management-system>
- Saputra, E. Y. (2015, July 30). *ekajogja is an Indonesian web designer-developer, WordPress consultant, who loves working on software localization & documentation*. Retrieved from ekajogja: <http://ekajogja.com/definisi/rest-representational-state-transfer/>
- W3C. (2003, March). *The World Wide Web Consortium (W3C) is an international community where Member organizations, a full-time staff, and the public work together to develop Web standards*. (D. Booth, Editor) Retrieved from Web Services Architecture: <https://www.w3.org/TR/2003/WD-ws-arch-20030808/>
- Xamarin. (2014, Juni). *Setting Up A Xamarin Cross Platform Solution*. Retrieved from We officially open sourced the Xamarin SDK for Android, iOS, and Mac under the same MIT license used for the Mono project. Now, .NET is open source and native on every single device, from mobile to desktop to cloud: <https://developer.xamarin.com/guides/cross->

[platform/application_fundamentals/building_cross_platform_applications/part_3_-
_setting_up_a_xamarin_cross_platform_solution/](#)