ISSN 2477-0043
e-ISSN 2460-7908

Jurnal Teknologi Terpadu
Vol. 2, No. 2, Desember, 2016

ANALISIS DAN PERANCANGAN *REPRESENTATIONAL STATE TRANSFER* (REST) *WEB SERVICE* SISTEM INFORMASI AKADEMIK STT TERPADU NURUL FIKRI MENGGUNAKAN YII FRAMEWORK

Mukhammad Agus Arianto[(1)] (Teknik Informatika, STT Terpadu Nurul Fikri)

Email :massagz@gmail.com

Sirojul Munir[(2)] (Teknik Informatika, STT Terpadu Nurul Fikri)

Email : rojulman@nurulfikri.ac.id

Khusnul Khotimah[(3)] (Teknik Informatika, STT Terpadu Nurul Fikri)

Email : kkhotimah33@gmail.com

ABSTRAK : Kesulitan dalam mengintegrasikan data pada beberapa sistem yang berbeda menjadi salah satu masalah yang sering dialami oleh *developer*, mulai dari bahasa pemrograman, *Platform*, dan perangkat yang digunakan. Oleh sebab itu, perlu adanya pembuatan *web service* untuk sistem informasi akademik STT Terpadu Nurul Fikri dengan teknologi REST. Perancangan ini dibuat dua model API untuk dua modul yakni modul mahasiswa dan modul dosen. Pada modul ini akan menghasilkan data dengan format JSON. Teknik pengujian API menggunakna teknik blackbox testing dengan *tools* aplikasi penguji Postman. Pengujian API dilakukan pada *prototype* aplikasi *web service.* Metode penelitian yang digunakan adalah *Unified Process* yang menggunakan kerangka kerja *Yii Framework 2.0*

Kata Kunci: *REST,web service,JSON, API*

1. PENDAHULUAN

Keberadaan teknologi internet kini sudah menjadi kebutuhan yang menunjang kemudahan manusia dalam mendapatkan informasi. Hal ini ditandai dengan berkembang pesatnya teknologi informasi. Selain itu, internet juga telah mengubah cara pertukaran data dan informasi menjadi lebih cepat dan akurat . Oleh karena itu sarana komunikasi teknologi internet dimanfaatkan untuk meningkatkan produktivitas perusahaan atau organisasi.

Pengolahan data dan informasi menjadi bagian terpenting dalam suatu organisasi.semakin besar dan kompleknya suatu sistem informasi maka kebutuhan akan pengolahan dan integrasi data menjadi perhatian bagi organisasi. Proses bisnis yang berubah seiring kebutuhan organisasi sehingga diperlukan pengembangan sistem dan aplikasi yang ada.F Kapojos, Rumagit, & Wowor, 2012). Namun, bagaimana membangun sistem dan aplikasi yang dapat digunakan kembali di masa mendatang yang dapat berjalan baik lintas *platform*, bahasa pemgrograman, maupun berbagai sistem operasi.

Hal tersebut yang mendasari konsep dan pemikiran lama akan sistem informasi khusunya mengenai arsitektur perangkat lunak yang terus berkembang. *Service Oriented Architecutre (SOA)* merupakan salah satu konsep arsitektur perangkat lunak yang menyediakan layanan bagi suatu sistem untuk bisa digunakan pada sistem lain sesuai kebutuhan. SOA bertujuan untuk memberikan layanan yang dapat diakses sistem lain, sehingga mendukung integrasi antar sistem (Thomas, 2005).

Dalam mengimplementasikan SOA, *web service* dapat digunakan untuk membuat pertukaran data yang diakses melalui standar *internet protocol.* Dalam perkembangan *web service* telah dikembangkan REST (*Representational State Transfer*) *web service*. Dengan mengimplementasikan REST *web service* dalam SOA tentu akan memudahkan pengembangan aplikasi perangkat lunak di luar sistem maupun dengan bahasa pemrograman atau *platform* berbeda.

Sekolah Tinggi Teknologi Terpadu Nurul Fikri merupakan perguruan tinggi teknologi yang memadukan antara keilmuan praktis di bidang teknologi informasi dengan pengembangan kepribadian islami (Menteri Pendidikan dan Kebudayaan, 2012). STT Terpadu Nurul Fikri berdiri pada tahun 2012 dan saat ini sudah memiliki lima angkatan (STT Terpadu Nurul Fikri, 2016) sehingga kebutuhan akan informasi yang mudah dan cepat sangat diperlukan guna menunjang proses kegiatan perkuliahan mahasiswa.

Di STT Terpadu Nurul Fikri kini telah terdapat sistem informasi akademik yang saat ini menggunakan pola *client-server (two-tier system).* Tetapi kesulitan timbul pada saat akan dikembangkan, yakni belum tersedianya layanan (*web service*) yang mampu

mengintegrasikan sistem tersebut dengan sistem lain sehingga mempersulit pengembangan terutama dalam hal pertukaran, integrasi dan pengelolaan data. Oleh karenanya perlu dilakukan perancangan dan pembuatan layanan (*web service*) tersebut guna menyelesaikan kesulitan tersebut.

Berawal dari hal tersebut, penelitian ini merancang dan membuat REST *web service* sistem informasi akademik STT Terpadu Nurul Fikri yang menyediakan layanan bagi sistem lain yang membutuhkan juga menawarkan kemudahan dalam menjembatani pertukuran data tanpa mempermasalahkan perbedaan *platform* ,ataupun bahasa pemrograman.

2. TEORI DASAR

2.1 Sistem Informasi

Sistem informasi adalah seperangkat komponen yang saling terkait yang mengumpulkan, memproses, menyimpan atau mendistribusikan informasi untuk mendukung koordinasi, pengendalian dan tahap pengambilan keputusan pada suatu organisasi. Informasi adalah data yang berhubungan dengan keputusan. (Harahap, 2013).

Menurut Abdilah,2016 Sistem informasi terdiri dari tiga konsep dasar, yaitu: masukan (*input),* proses (*processing*) dan keluaran (*output).* Ketiga elemen dasar ini menghasilkan informasi yang dibutuhkan untuk pengambilan keputusan, pengendalian operasi, analisis permasalahan dan menciptakan produk atau jasa baru.

Gambar 2.1: Konsep dasar sistem informasi (Abdillah, 2006)

2.2 *Service* Oriented Architecture

Service oriented architecture atau SOA didefinisikan sebagai kebijakan, praktek, kerangka kerja yang memungkinkan fungsionalitas aplikasi disediakan dan dikonsumsi sebagai seperangkat *service* pada sebuah unit yang sesuai dengan kebutuhan *service* customer. *Service* dapat digunakan, dipulikasikan, ditemukan, dan diabstraksikan menggunakan standar antarmuka (Sprott & Wilkes, 2004).

Gambar 2.2 memberikan struktur hirarki dari SOA.

Gambar 2.2: Antarmuka *web service* dengan sistem lainnya (Newcomer, 2002)

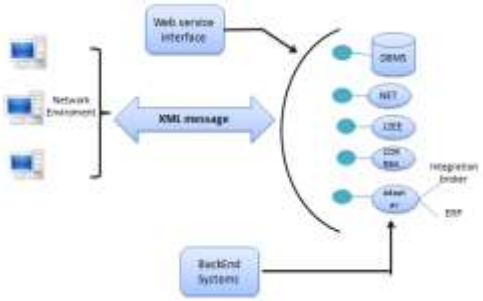
2.3 Web service

Web service adalah sebuah *software* yang dirancang untuk mendukung interoperabilitas interaksi mesin-ke-mesin melalui sebuah jaringan. *Web service* secara teknis memiliki mekanisme interaksi antar sistem sebagai penunjang interoperabilitas, baik berupa agregasi (pengumpulan) maupun sindikasi (penyatuan). *Web service* memiliki layanan terbuka untuk kepentingan integrasi data dan kolaborasi informasi yang bisa diakses melalui internet oleh berbagai pihak menggunakan teknologi yang dimiliki oleh masing-masing pengguna (Sutanta, 2012).

Web service bertujuan untuk meningkatkan kolaborasi antar pemrograman dan perusahaan, yang memungkinkan sebuah fungsi di dalam *web service* dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detil pemrograman yang terdapat di dalamnya. Ilustrasi mengenal posisi *web service* terhadap aplikasi lainnya dijelaskan pada gambar dibawah ini.

2.4 REST *Web service*

REST merupakan singkatan dari *Representational State Transfer*. Istilah REST atau RESTful pertama kali diperkenalkan oleh Roy Fielding pada disertasinya di tahun 2000 (Alonso, Casati.F, H., & V, 2003). REST bukanlah sebuah standar protokol *web service*, melainkan hanya sebuah gaya arsitektur. Ide dasar dari arsitektur REST adalah bagaimana menghubungkan jalur komunikasi antar mesin atau aplikasi melalui HTTP sederhana. Sebelum adanya REST, komunikasi antar mesin atau aplikasi dilakukan dengan



menggunakan beberapa mekanisme atau protokol *middleware* yang cukup kompleks seperti DCE, CORBA, RPC, ataupun SOA.

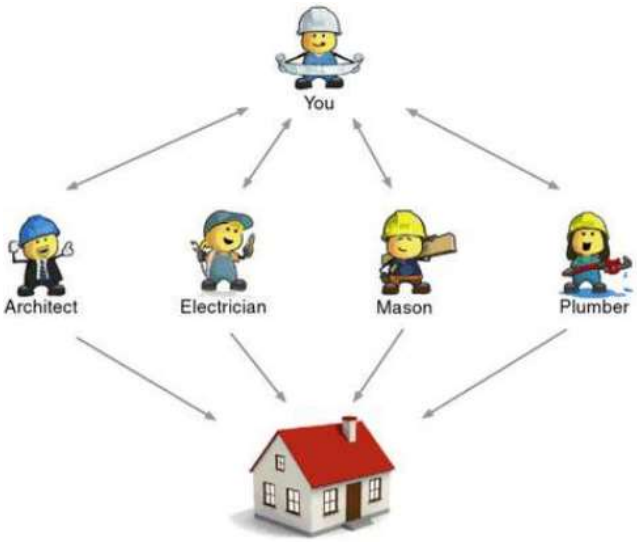
Arsitektur REST mampu mengeksploitasi berbagai kelebihan dari HTTP yang digunakan untuk kebutuhan *web service*. Walaupun SOAP juga dapat menggunakan protokol HTTP, namun hanya terbatas untuk kebutuhan transport saja. HTTP sendiri merupakan sebuah protokol standar di dunia *World Wide Web* yang berbasis *synchronous request/response*. Protokol tersebut sangat sederhana: *client* mengirimkan sebuah *request message* yang mencakup HTTP *method* yang akan di invokasi, lokasi *resource* dalam format URI, serta pilihan format pesan (pada dasarnya dapat berupa format apa saja seperti HTML, *plain text*, XML, JSON, ataupun data binary), Kemudian *server* akan mengirimkan *response* sesuai dengan spesifikasi yang diminta oleh *client*. Selama ini, yang berfungsi sebagai aplikasi *client* adalah sebuah *web browser* yang memfasilitasi komunikasi antara mesin dengan manusia. Dengan adanya REST, aplikasi *client* dapat berupa aplikasi apa saja hanya dengan memanfaatkan HTTP.

Berikut ini beberapa prinsip arsitektur dari REST yang dikutip dari sebuah buku berjudul "RESTful Java with JAX-RS":

1. *Addressablitiy*
2. *Constrained & Uniform Interface*
3. *Stateless Communication*
4. *Format Pesan Pertukaran*

2.6 API (*Application Programming Interface*).

API merupakan *software interface* yang terdiri atas kumpulan instruksi yang disimpan dalam bentuk *library* dan menjelaskan bagaimana agar suatu *software* dapat berinteraksi dengan *software* lain. Penjelasan ini dapat dicontohkan dengan analogi apabila akan dibangun suatu rumah. Dengan menyewa kontraktor yang dapat menangani bagian yang berbeda, pemilik rumah dapat memberikan tugas yang perlu dilakukan oleh kontraktor tanpa harus mengetahui bagaimana cara kontraktor menyelesaikan pekerjaan tersebut. Dari analogi tersebut, rumah merupakan *software* yang akan dibuat, dan kontraktor merupakan API yang mengerjakan bagian tertentu dari *software* tersebut tanpa harus diketahui bagaimana prosedur dalam melakukan pekerjaan tersebut. (Reddy, 2011)



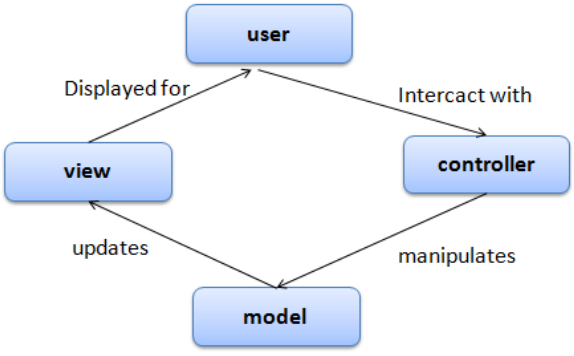
Gambar 2.3: Analogi API dalam pembangunan Rumah (Reddy, 2011)

2.7 *Model-View-Controller* (MVC).

Model-View-Controller adalah pola desain yang mengijinkan pengembang untuk membagi kode mereka menjadi tiga kategori (Myer, 2008):

a) *Model* untuk mengelola data
b) *View* untuk menampilkan data dan elemen *user interface.*
c) *Controller* menangani *user events* yang mempengaruhi *model dan view.*

Konsep hubungan *Model-View-Controller* selengkapnya dijelaskan pada gambar 2.7.



Gambar 2.4: Konsep Hubungan *Model-View-Controller* (Myer, 2008)

MVC dipisah menjadi 3 bagian, sehingga pengembang dapat membuat *multiple views* dan *controllers* untuk banyak model tanpa mengubah desain model.

Pemisahan ini memungkinkan untuk kemudahan pengelolaan, *portable*, dan mengorganisasi aplikasi. *Model-View-Controller* pertama kali digagas oleh peneliti Xerox PARC yang bekerja pada bahasa pemrograman *smalltalk* pada akhir 1970-an dan awal 1980-an. *Smalltalk* berorientasi objek, *dynamically typed*, dan bahasa pemrograman yang reflektif (Myer, 2008).

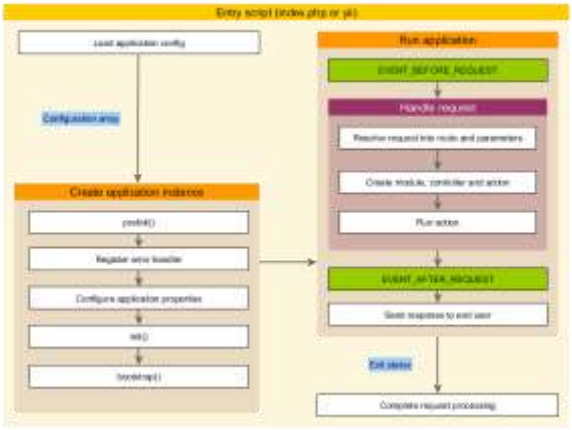
Tahap selanjutnya dari MVC yaitu dengan kedatangan sistem operasi NeXT dan perangkat lunaknya. NeXT merupakan perusahaan yang didirikan oleh Steve Jobs pada akhir 1980-an dan dibeli oleh *Apple* pada awal 1990-an. Pengembang MVC di NeXT menemukan cara untuk membuat *view* dan *controller* yang lebih baik (Myer, 2008).

Implementasi MVC pada dunia *web* yaitu dengan munculnya produk Django, Struts, dan yang sangat baik yaitu Ruby on Rails. Perangkat seperti Ruby on Rails memungkinkan tim pengembang untuk membuat aplikasi berbasis *web* pada waktu yang singkat, dengan sedikit proses pengujian. Situs web berbasis Ruby on Rails memiliki *interface* yang rapi, URL yang mudah dipahami, dan secara umum lebih lengkap dan *secure* daripada situs yang dibuat secara biasa dengan PHP. Keuntungan bagi pengembang web, beberapa *framework* PHP berbasis MVC bermunculan seperti CakePHP, Symfony, dan CodeIgniter (Myer, 2008).

2.8 Yii 2.0 *Framework*

Yii adalah sebuah *framework* yang digunakan untuk membuat sebuah aplikasi berbasis *web* yang disusun dengan menggunakan bahasa PHP. Yii merupakan sebuah *web framework programming* umum yang dapat digunakan untuk membangun berbagai macam aplikasi *web* menggunakan PHP. Sebab Yii sendiri merupakan *component-based architecture* dan dengan dukungan *caching* yang canggih dan sangat cocok untuk membangun aplikasi berskala besar seperti portal, forum, *Content Management System* (CMS), *e-commerce*, RESTful *web service* dan banyak lagi.

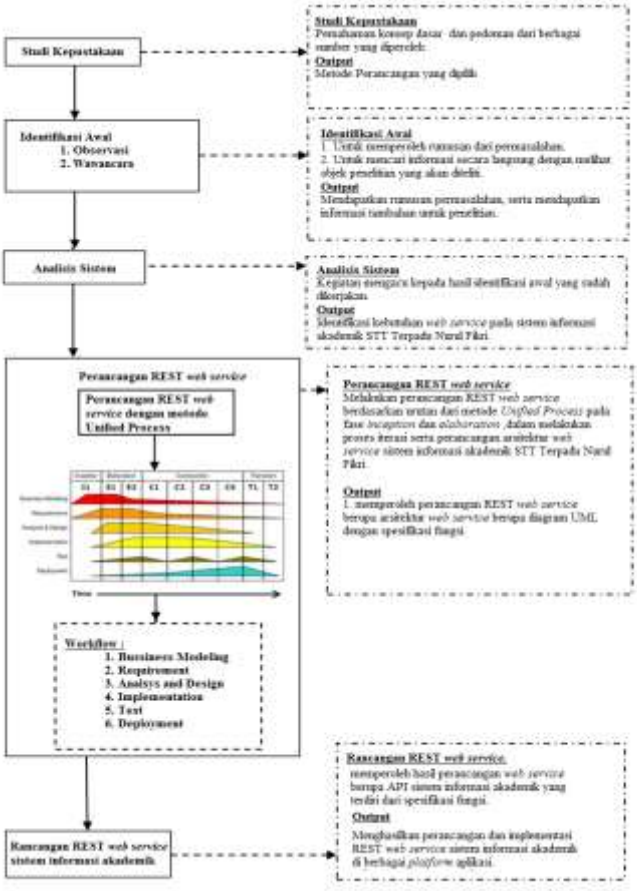
Yii memiliki dua versi utama yang tersedia yaitu: Yii 1.1 dan Yii 2.0. versi 1.1 adalah generasi lama dan sekarang ini dalam kondisi *maintenance*. Versi 2.0 adalah Yii yang telah disempurnakan yang mengadopsi teknologi terakhir dan protokol, termasuk *composer*, PSR, *namespaces*, *traits* dan lain-lain. Versi 2.0 merepresentasikan generasi *framework* masa kini dan akan mendapatkan fokus pengembangan lebih lanjut untuk beberapa tahun kedepan. Berikut adalah gambar yang menunjukan *lifecycle* dari aplikasi *Framework* Yii 2.0. (Yii Framework).



3. KERANGKA PIKIR PENELITIAN

tahapan-tahapan yang akan dilakukan dalam proses penyelesain penelitian dalam rangka untuk memudahkan memecahkan masalah dari awal perencanaan strategis hingga tercapainya tujuan.

Adapun tahapan-tahapan tersebut dapat dilihat pada Gambar 3.1

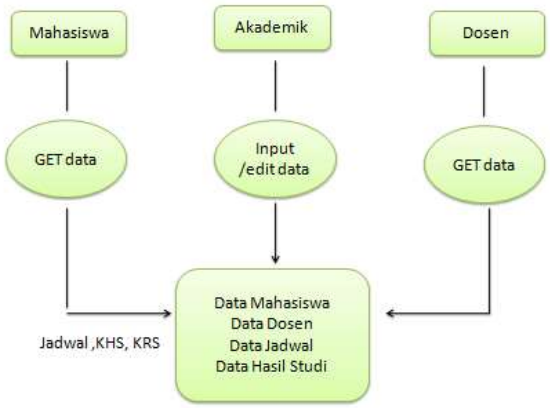


Gambar 3.5: Tahapan penelitian

4. PEMBAHASAN DAN ANALISIS

Analisia Kebutuhan

Proses analisis kebutuhan pengembangan *web service* pada sistem informasi akademik STT Terpadu Nurul Fikri menggunnakan USDP yang dimulai dengan mendefenisikan atau menentukan persyaratan (*requirements*). Persyaratan tersebut terdiri atas persyaratan fungsional dan persyaratan non fungsional



Gambar 4.6 Rancangan kebutuhan *web service*

Analisis kebutuhan perancangan REST *web service.*

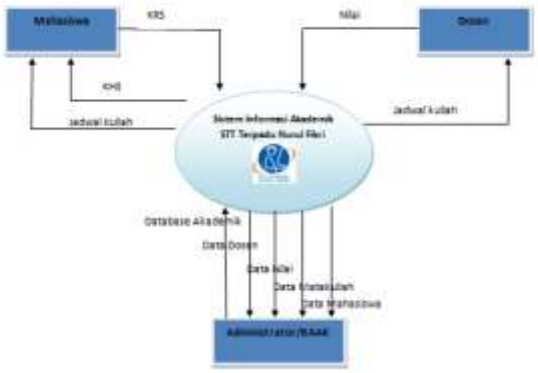
berdasarkan hasil analisis kebutuhan yang dilakukan melalui observasi dan wawancara. Hasil analisis dapat dilihat pada tabel 4.1 berikut :

Tabel 4.1: Hasil Analisis kebutuhan perancangan *web service*

No	Kebutuhan Perancangan	Kondisi Awal
1	Basis data SIAK *PostgreSQL*	Tersedia
2	Framework untuk membangun *web service*	Tidak Tersedia
3	Rancangan arsitektur *web service*	Harus dirancang
4	Format data transfer	Belum ditentukan
5	Kandidat- kandidat modul *web service*	Harus ditentukan
6	Dokumentasi API	Belum Tersedia

Analisis Sistem yang Berjalan

Setelah dilakukan pengamatan pada sistem informasi akademik STT Terpadu Nurul Fikri yang dilakukan pada tahun 2015, maka sistem yang berjalan digambarkan melalui *data flow diagram* (DFD) berikut :



Gambar 4.7: Analisis sistem berjalan

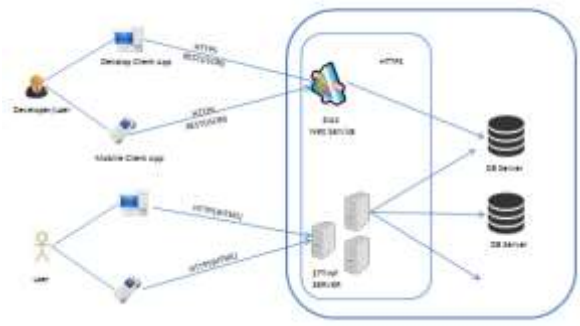
Analisis Arsitektur yang Berjalan

sistem informasi akademik yang ada di STT Terpadu Nurul Fikri terdiri dari perangkat lunak berbasis web dimana secara fisik diletakan di beberapa *web server* dan terhubung dengan *database server* sebagai tempat penyimpanan data. Perangkat lunak tersebut hanya dapat diakses oleh *web browser* (komunikasi *machine-to-human*) melalui jaringan internet. Aplikasi sistem informasi akademik tersebut hanya dapat diakses dari jaringan lokal.

Jika disederhanakan dalam sebuah model *diagram*, model keadaan sistem informasi yang di STT Terpadu Nurul Fikri dapat digambarkan sebagai berikut :

Gambar 4.8: Model Sistem kini

Berdasarkan analisis sistem dengan kondisi sistem yang telah dipaparkan sebelumnya , berikut ini diusulkan sebuah model untuk antarmuka REST *web service* pada sistem informasi akademik yang ada di STT Terpadu Nurul Fikri:



Selanjutnya sebagaimana yang tertera pada batasan masalah, manfaat dan tujuan penelitian, maka akan dikembangkan perancangan hasil implementasi dari *Web service* yang telah dirancang Perancangan berikut mensimulasikan mekanisme aliran data yang terjadi pada berbagai *platform* aplikasi.

Gambar 4.9: model *interoperability* modul *web service*

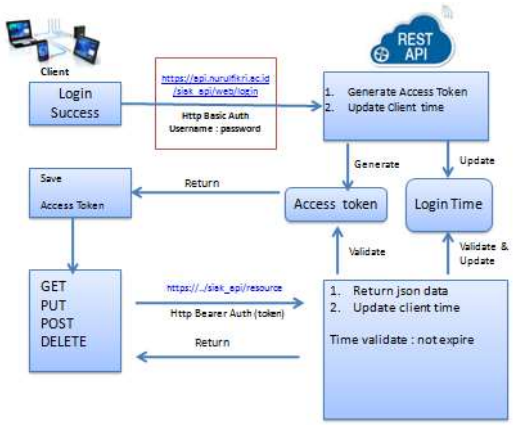
Oleh karena *web service* yang akan dirancang menggunakan konsep *Representational State Transfer* (REST) dimana memungkinkan klien dapat melakukan *request* melalui protokol HTTP dengan mudah menggunakan URI. Struktur dari URI nya adalah sebagai berikut :

https://{nama_domain}/{module}/{class}/{fungsi}?{parameter}..&{parameter_n}

Dengan keterangan dapat dijelaskan sebagai berikut:

1. {nama_domain} adalah nama domain letak API berada.
2. {module} adalah folder subdomain letak API berada.
3. {class} adalah class dimana letak API berada.
4. {nama_fungsi} adalah nama fungsi yang akan diakses.
5. {paremeter_1}...{parameter_n} adalah parameter yang dikirimkan .

Selanjutnya dalam perancangan skema messaging dan otentifikasi. Berikut ini menggambarkan bagaimana skema *messaging* dan otentikasi yang diterapkan dalam perancangan API untuk sistem informasi akademik STT Terpadu Nurul Fikri.



Gambar 4.10: *Skema Messaging* & otentikasi *token*

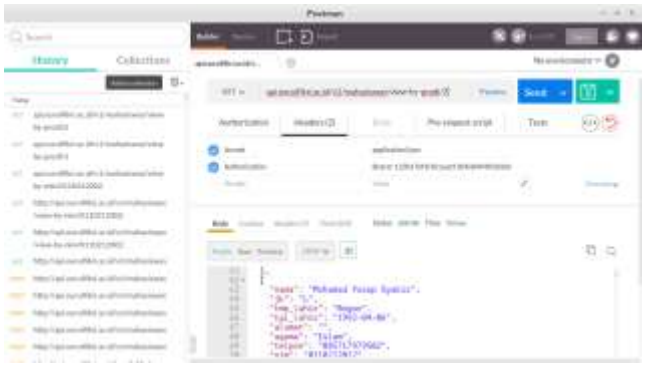
Adapun skema *messaging* nya ialah melalui beberapa tahapan dalam pengaksesan URI yakni, pertama *user* harus *login* terlebih dahulu untuk mendapatkan *token* , kemudian *server* mengautentikasi *request* melalui *Http Basic Auth*. *Server* men*generate token* yang digunakan untuk *request user*. Kemudian setelah *user* mendapatkan *token*, *token* tersebut digunakan untuk *request* ke *resource* data. *Server* mengotentikasi *request* melalui *Http Bearer Auth* dimana *token* diletakan didalam *header* saat melakukan *request* dan memvalidasi *login time*. Terdapat empat macam *response* diantaranya sukses dengan menampilakan data JSON ,kedua *token* salah, *token expire*. Terakhir *server* memperbaharui *login time user* setelah menampilkan *response* data.

5. HASIL IMPLEMENTASI

Hasil dari Modul Mahasiswa *service*.

Pengujian fungsional yang dilakukan menggunakan aplikasi Postman. Data yang diuji adalah data mahasiswa yang diakses menggunakan *token* yang

telah di dapatkan sebelumnya. Hasil pengujian dapat dilihat pada gambar dibawah ini :



Gambar 5.11: Hasil pengujian mahasiswa *service* pada aplikasi Postman

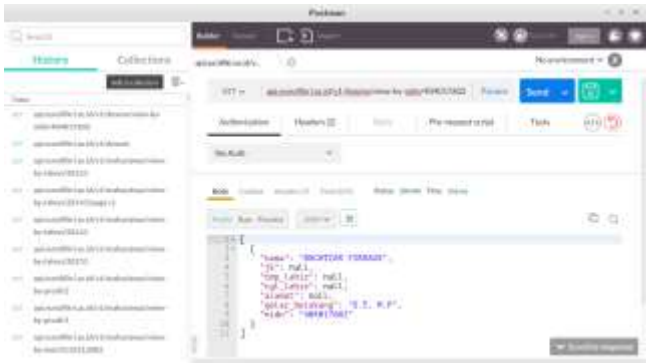
Pengujian dilakukan pada aplikasi Postman pada alamat https:api.nurulfikri.ac.id/v1/mahasiswas/*view*-by-prodi dengan menyertakan *token* pada *header*. Untuk hasil keluaran berupa daftar mahasiswa berdasarkan prodi tertentu sesuai yang di isikan dalam parameter yang di format dalam bentuk JSON format. Hasil teks tersebut dapat dilihat secara terstruktur seperti dibawah ini :


```
{
  "items": [
    {
      "nama": "Mukhamad Agus Arianto",
      "jk": "L",
      "tmp_lahir": "Tangerang",
      "tgl_lahir": "1993-02-23",
      "alamat": "Jl. Sultan Ageng Tirtayasa Kunciran Indah Pinang",
      "noktp": null,
      "email": null,
```


Gambar 5.12: Hasil keluaran mahasiswa *service* dalam format JSON

Hasil dari Modul Dosen *Service*

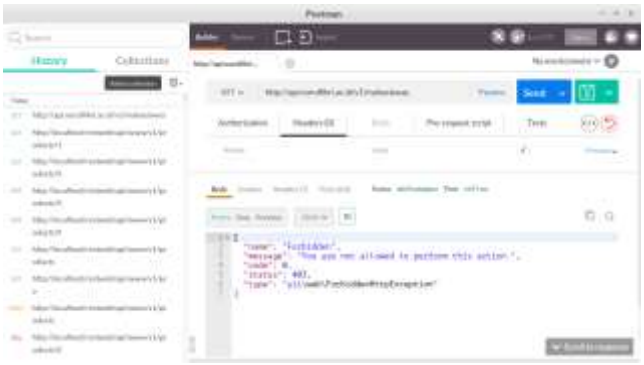
Pengujian fungsional yang dilakukan menggunakan aplikasi Postman. Data yang diuji adalah data jadwal yang diakses menggunakan *token* yang diselipkan di dalam *header*. Hasil Pengujian dapat dilihat pada Gambar dibawah ini :



Gambar 5.13: Hasil pengujian jadwal *service* pada Postman

Hasil pengujian dari Modul *service* tanpa *token*

Setelah dilakukan pengujian terhadap API yang berhasil diakses dengan *token*, kini saatnya menguji bagaimana API yang diakses tanpa menggunakan *token* sebagai parameterrya. Berikut adalah gambar yang ditampilkan pada aplikasi Postman saat *request* API tanpa menyelipkan *token* pada *header*.



Gambar 5.14: hasil keluran modul *service* tanpa parameter *token* pada *header.*

Pengujian API *web service* mahasiswa pada *web client* aplikasi Yii 2.0

Gambar 5.15 pengaksesan API pada *controller web client*

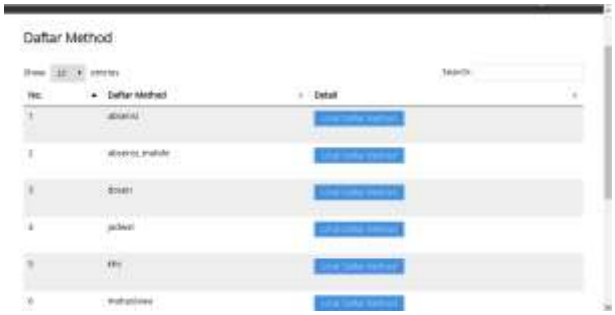
Gambar 5.16 pengaturan *view* pada *web-client*

Gambar 5.17 hasil yang ditampilkan pada *browser*

***prototype web* untuk dokumentasi API**

Prototype web dokumentasi berikut bertujuan untuk memberikan penjelasan mengenai API yang telah di rancang sehingga dapat dengan mudah digunakan oleh *developer* yang ingin menggunakan API tersebut. Berikut ini adalha tampilan awal prototype web dokumentasi API sistem informasi akademik STT Terpadu Nurul Fikri.

Gambar 5.18 Tampilan Muka Web API SIAK STT-NF



Gambar 5.19 Tampilan Daftar *method*

Gambar 5.20 Tampilan *Method-Method* kelas Mahasiswa

Gambar 5.211 Tampilan Penjelasan detail *Method*

Tabel 5.2: Pengujian dengan menggunakan metode *black box*

6. KESIMPULAN

Berdasarkan hasil penelitian tugas akhir diatas dapat disimpulkan:

1. Dihasilkannya rancangan model *web service* sistem informasi akademik STT Terpadu Nurul Fikri berbasis teknologi REST menggunakan Yii Framework 2.0.
2. REST *web service* yang di rancang dan dibuat dapat di uji dengan baik dengan menggunakan *tools* pengujian.