IMPLEMENTASI REST - API UNTUK PORTAL AKADEMIK UKDW BERBASIS ANDROID

Yosef Koko Kurniawan
Yetli Oslan, Harianto Kristanto

Abstrak

Layanan informasi seputar akademik adalah layanan yang sangat penting bagi civitas akademik di UKDW. Dengan layanan ini stakeholder dapat mengetahui informasi perkembangan studi mahasiswa. Akan tetapi di UKDW layanan seperti ini belum ada yang dalam versi mobile.

Dalam penelitian ini penulis membuat portal akademik UKDW berbasis mobile. Aplikasi harus terkoneksi dengan basis data SITMPT UKDW agar dapat terintegrasi dengan sistem informasi yang lain. Untuk itu sistem yang dibangun membutuhkan API yang menjadi sarana penghubung antara aplikasi dengan basis data. Dalam penelitian ini API dibangun dengan konsep REST dan menerapkan otorisasi milik Amazon Web Service untuk sistem keamanannya.

Hasil dari perancangan adalah sebuah aplikasi Portal UKDW berbasis Android yang berjalan pada Sistem Operasi Android mulai dari versi 2.2 (Froyo) sampai 4.2 (Jelly Bean). Selain itu perancangan juga menghasilkan sebuah REST API lengkap dengan sistem keamanannya.

Kata Kunci : *Android, Portal, REST – API, Amazon Web Service*

1. Pendahuluan

Portal dan e-Class merupakan sarana bertukar informasi seputar kampus, perkembangan studi mahasiswa, dan lain sebagainya. Portal sangat penting digunakan terutama bagi mahasiswa dan orang tua untuk melihat perkembangan studi mahasiswa. Sedangkan e-Class digunakan untuk kegiatan yang berhubungan dengan kelas yang diikuti mahasiswa seperti nilai ujian, presensi, dan tugas.

Saat ini Portal dan e-Class UKDW masih berbasis web sehingga hanya dapat diakses melalui browser. Mahasiswa, orang tua, dan dosen tentu akan semakin dimudahkan memperoleh informasi dari Portal dan e-Class jika kedua sarana tersebut juga memiliki versi mobile. Akan tetapi permasalahan yang muncul adalah bagaimana memindahkan Portal dan e-Class ke platform berbeda yakni Android dengan mengakses basis data yang sama.

2. Landasan Teori

a. Portal

Menurut Hakim (2013) Portal ialah sebuah web yang menjadi starting point bagi pengunjung untuk memulai aktifitasnya di internet. Ada 2 jenis web portal, yakni web portal bersifat horizontal dan web portal bersifat vertikal.

Web portal yang bersifat horizontal menyediakan berbagai layanan informasi yang bersifat umum. Sedangkan web portal yang bersifat vertikal menyediakan layanan informasi yang bersifat lebih spesifik pada bidang tertentu saja sehingga dapat bersifat profesional bagi pengunjungnya.

b. Web Service

Menurut *World Wide Web Consortium* (W3C) selaku badan penemu dan pengembang web service, *web service* adalah salah satu bentuk sistem perangkat lunak yang didesain untuk mendukung interaksi mesin ke mesin melalui jaringan. Sistem *web service* memungkinkan sebuah

fungsi di dalam *web service* dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detail pemrograman yang terdapat di dalamnya.

Secara umum, arsitektur *web service* terdiri dari 3 komponen, yaitu:

1) **Service provide**r, merupakan pemilik *Web service* yang berfungsi menyediakan kumpulan operasi dari *Web service*.
2) **Service requesto**r, merupakan aplikasi yang bertindak sebagai klien dari *Web service* yang mencari dan memulai interaksi terhadap layanan yang disediakan.
3) **Service registry**, merupakan tempat dimana Service provider mempublikasikan layanannya. Pada arsitektur *Web service*, Service registry bersifat optional. Teknologi *web service* memungkinkan kita dapat menghubungkan berbagai jenis software yang memiliki platform dan sistem operasi yang berbeda.

Ada 2 jenis *web service* yaitu REST dan SOAP. SOAP (*Simple Object Access Protocol*) merupakan protokol untuk saling bertukar pesan antar aplikasi. Spesifikasi format pesan tersebut didefinisikan seperti amplop berbasis XML yang dikirim beserta aturan-aturan atau cara untuk menerjemahkan representasi data dari XML.

Cara kerja SOAP ialah, aplikasi klien mengirim *request* berbentuk XML kepada provider *web service*. *Web service* menerima *request* tersebut, menjalankan *service*, kemudian mengirimkan response ke aplikasi klien juga dalam bentuk XML. Baik *request* maupun *response* keduanya menggunakan protokol SOAP.

REST adalah *web service* yang menerapkan konsep perpindahan antar *state* dimana dalam bernavigasi REST melalui *link* HTTP untuk melakukan aktivitas tertentu. Dalam pengaplikasiannya REST banyak digunakan untuk *web service* yang berorientasi pada *resource*. Maksud orientasi pada *resource* adalah orientasi yang menyediakan *resource* sebagai layanannya dan bukan kumpulan dari aktifitas yang mengolah *resource* itu. *Response* dari *web service* REST dapat berupa XML atau JSON.

c. Amazon Web Service

Request data pada REST API sangat dimudahkan karena hanya menggunakan alamat URI. Akan tetapi hal tersebut menjadi sebuah hal yang beresiko karena siapa saja dapat megakses API hanya dengan mengetikkan URI. Untuk mengatasi kemungkinan hal-hal yang tidak diinginkan maka API memerlukan proses otorisasi untuk mengecek keabsahan setiap *request*. Salah satu proses otorisasi yang sering digunakan ialah mengadopsi proses otorisasi milik AWS.

Berikut ini adalah poin-poin penting yang digunakan dalam proses otorisasi seperti dilansir dari dokumentasi *Web Service* milik Amazon.

1) API Key

API Key bersifat unik dan digunakan sebagai penanda dari *user*. Fungsi API Key sama dengan *username*. API key selalu dikirimkan setiap kali aplikasi klien melakukan *request* pada API.

2) Secret Key

Secret Key digunakan untuk menghasilkan hash.

3) StringToSign

StringToSign merupakan penggabungan kata dari *method*, *date*, dan *resource*.

4) Date

Date dibutuhkan untuk mengetahui waktu dari *request* yang dikirimkan. Dengan menggunakan nilai Date ini API dapat melakukan pembatasan waktu, sehingga *request* yang sama tidak dapat digunakan kembali dalam rentang waktu yang sudah ditentukan. Hal ini dilakukan untuk alasan keamanan.

5) Signature

Signature merupakan sekumpulan karakter acak yang dibutuhkan untuk menentukan apakah *request* tersebut valid atau tidak. Signature dihasilkan dari StringToSign yang dienkripsi

menggunakan algoritma HMAC dengan SHA1 sebagai fungsi *hash* dan Secret KEY sebagai kuncinya. Rumus umum algoritma HMAC dapat dijelaskan dengan persamaan dibawah ini:

$$HMAC_K(m) = h\left((K \oplus opad) \quad h\left((K \oplus ipad) \quad m\right)\right) \qquad [1]$$

dengan K adalah kunci privat yang diketahui oleh pengirim dan penerima, h adalah fungsi *hash* yang digunakan, m adalah pesan yang akan diautentikasi, opad adalah 0x5c5c5c...5c dan ipad adalah 0x363636...36 dengan panjang yang sama. Hasil dari enkripsi menggunakan HMAC kemudian diubah ke dalam bentuk Base64.

d. Android

Android adalah sistem operasi yang dikembangkan untuk perangkat *mobile* berbasis Linux. Setiap aplikasi pada Android memiliki tingkatan yang sama baik aplikasi inti maupun aplikasi pihak ketiga. API yang disediakan menawarkan akses ke *hardware*, data-data ponsel sekalipun, maupun data sistem sendiri.

Menurut Safaat H. (2012:3), Android disebut sebagai platform masa depan karena Android dipuji sebagai *platform nobile* pertama yang lengkap, terbuka, dan bebas. Lengkap artinya Android menyediakan banyak *tools*, terbuka berarti Android merupakan *open source platform*, sedangkap bebas artinya pengembang dapat membuat aplikasi secara bebas tanpa dipungut biaya.

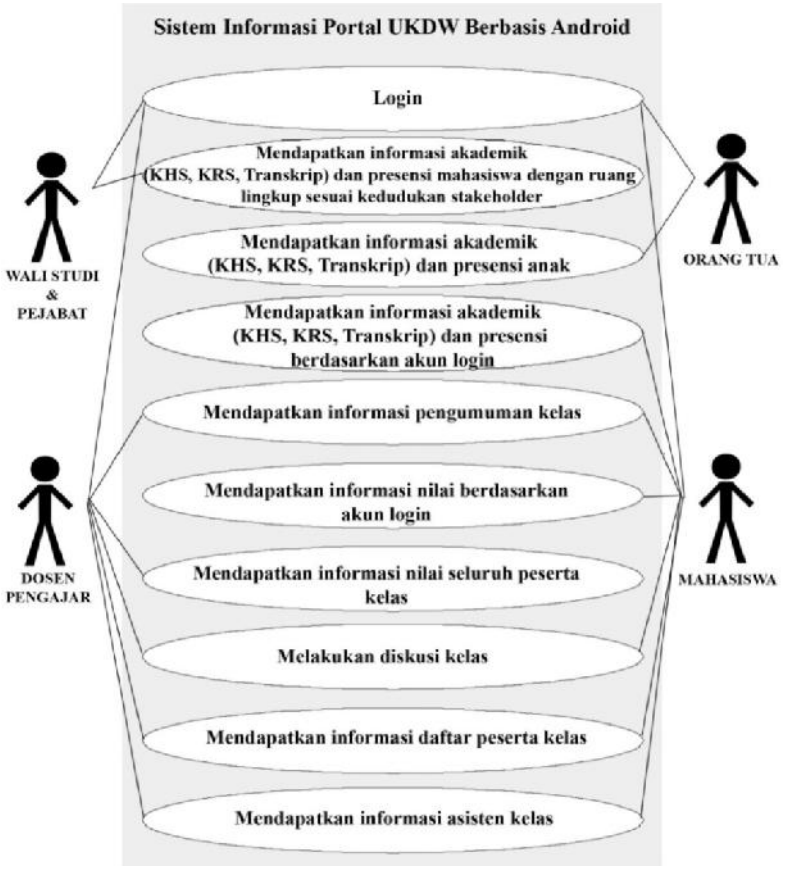
Hermawan (2011:5) memaparkan fitur yang tersedia pada Android sebegai berikut:

1) ***Framework* aplikasi**: memungkinkan penggunaan dan pemindahan komponen yang tersedia.
2) ***Dalvik virtual machine*:** *virtual machine* yang dioptimalkan untuk perangkat *mobile*.
3) **Grafik:** grafik 2D dan 3D yang didasarkan pada *library* OpenGL.
4) ***SQLite*:** untuk penyimpanan data.
5) **Mendukung media:** audio, video, dan berbagai format gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
6) **GSM, Bluetooth, EDGE, 3G, dan WiFi** (tergantung hardware)
7) **Kamera, GPS, Kompas, dan *accelerometer*** (tergantung hardware)
8) ***Lingkungan pengembangan yang kaya***, termasuk emulator, peralatan debugging, dan plugin untuk Eclipse IDE

3. Perancangan Sistem dan Basis Data

a. Use Case Diagram

Sistem yang dibangun memiliki 4 *stakeholder* yang terlibat, yakni mahasiswa, dosen pengajar, wali studi, pejabat (prodi dan dekan), dan orang tua. Untuk lebih jelas mengenai peran keempat *stakeholder* dapat dilihat pada diagram *use case* berikut.

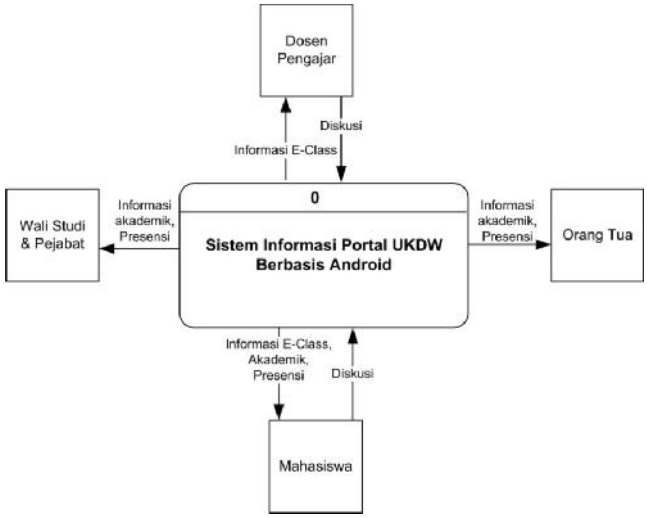


Gambar 1. Use Case Diagram

b. Perancangan Proses

Berikut ini adalah *Data Flow Diagram* (DFD) yang menggambarkan aliran data dari setiap proses.

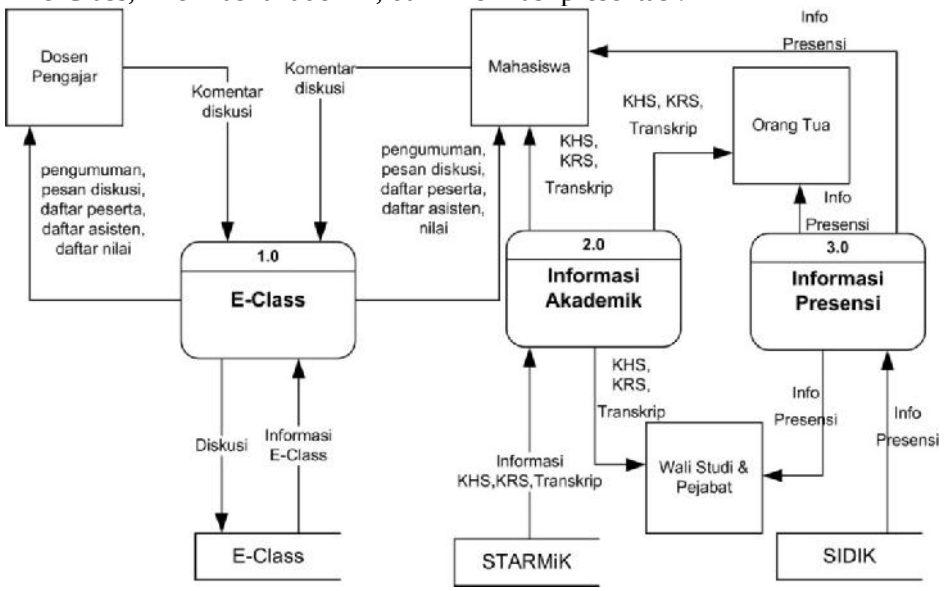
1) Diagram Konteks



Gambar 2. Diagram Konteks

2) Data Flow Diagram Level 1

Secara garis besar Aplikasi Portal Akademik UKDW Berbasis Android memiliki 3 fitur utama, yakni e-Class, informasi akademik, dan informasi presentasi.

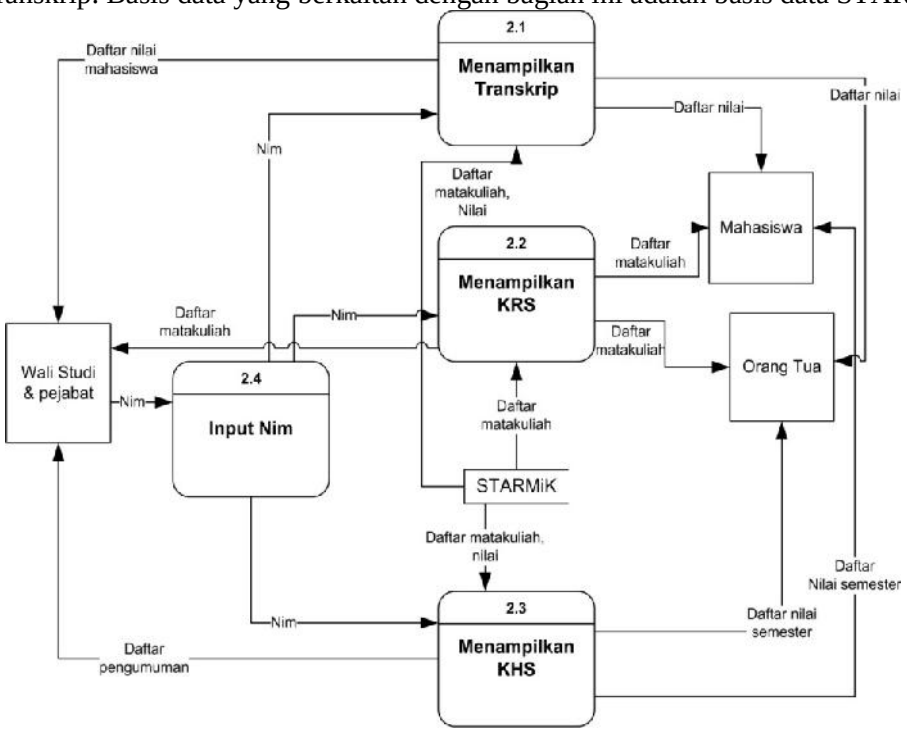


Gambar 3. Data Flow Diagram Level 1

3) Data Flow Diagram Level 2

a) DFD Level 2 Informasi Akademik

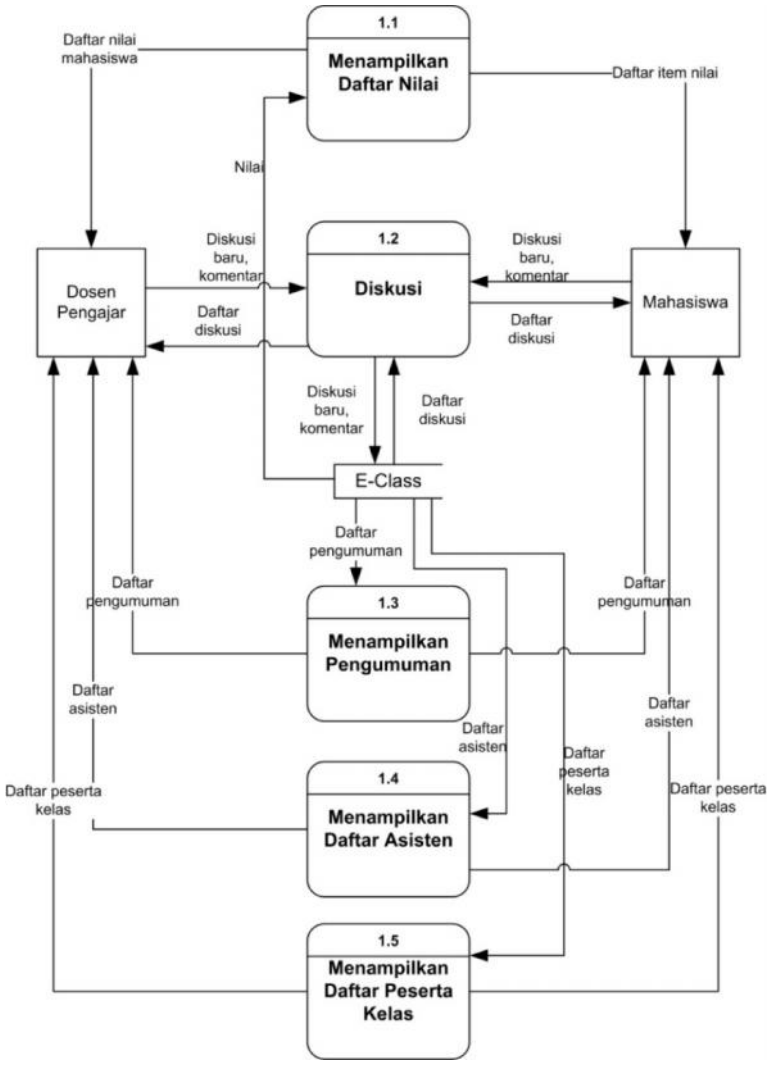
Bagian ini menyediakan informasi seputar akademik mahasiswa seperti KHS, KRS, dan Transkrip. Basis data yang berkaitan dengan bagian ini adalah basis data STARMiK.



Gambar 4. DFD Level 2 Informasi Akademik

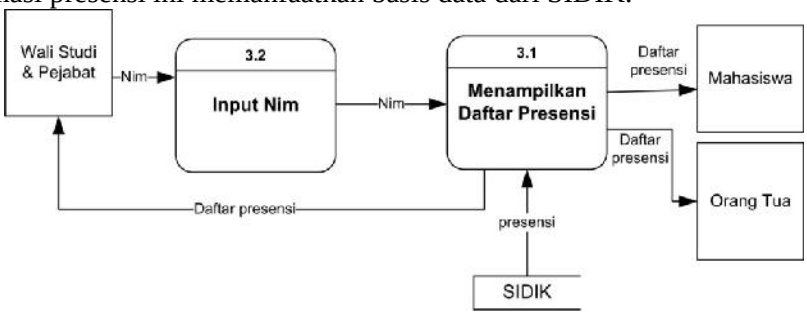
b) DFD Level 2 e-Class

Pada bagian e-Class, sistem menyediakan informasi seputar kelas seperti pengumuman, daftar nilai, daftar diskusi, daftar peserta, dan daftar asisen. Bagian ini juga terdapat proses input yakni saat pengguna mengirimkan pesan diskusi. Basis data yang berkaitan dengan bagian ini adalah basis data e-Class.



Gambar 5. DFD Level 2 e-Class

c) DFD Level 2 Informasi Presensi
Informasi presensi ini memanfaatkan basis data dari SIDIK.



Gambar 6. DFD Level 2 Informasi Presensi

c. Perancangan API

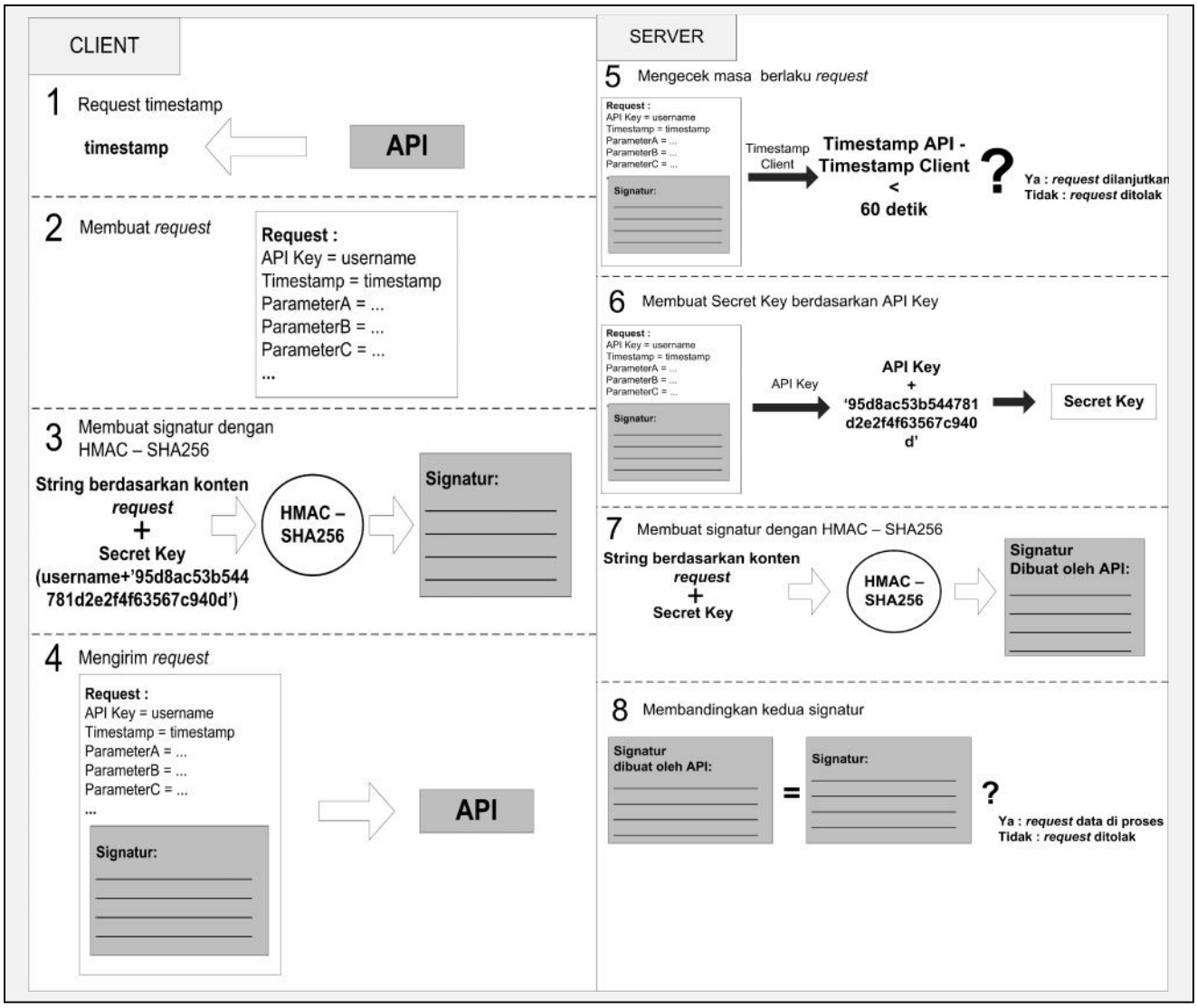
API dirancang dengan konsep *Representational State Transfer* (REST). REST memungkinkan klien dapat melakukan *request* melalui protokol HTTP dengan mudah menggunakan URI. Berikut ini adalah format URI untuk melakukan *request* pada API :

```
http://{nama_domain}/{sub_domain}/{nama_fungsi}?{parameter_1}...
&{parameter_n}
```

dengan keterangan dapat dijelaskan sebagai berikut :

1) `{nama_domain}` adalah nama domain letak API berada.
2) `{sub_domain}` adalah folder sub domain letak API berada.
3) `{nama_fungsi}` adalah nama fungsi yang akan diakses.
4) `{parameter_1}...&{parameter_n}` adalah parameter-parameter yang dikirimkan.

Berikut ini adalah proses implementasi yang mengadopsi proses otorisasi AWS.



Gambar 7. Otorisasi Sisi Client dan Sisi Server

4. Uji Coba Sistem

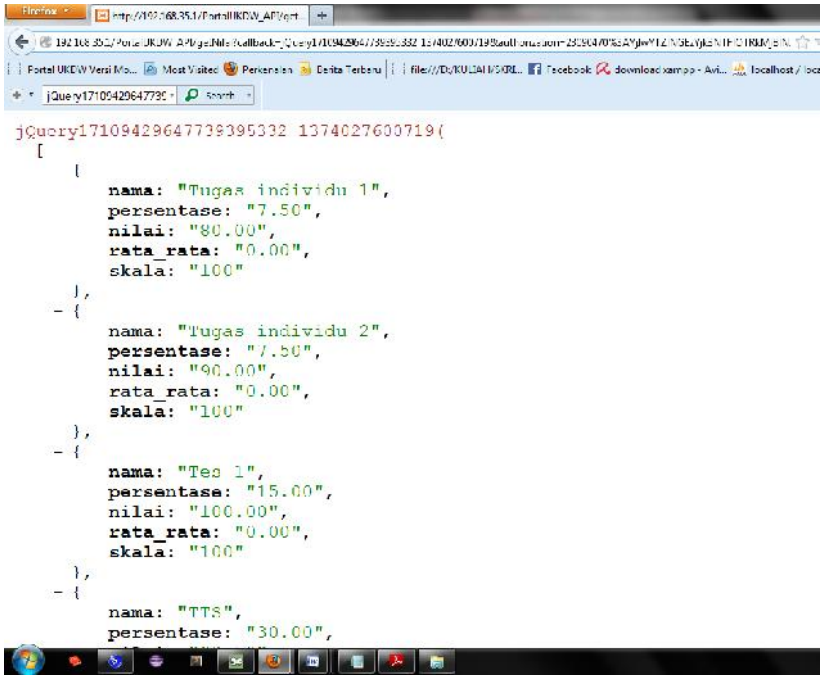
a. Uji Coba API

Uji coba API dilakukan untuk mengetahui apakah proses otorisasi sudah dapat berjalan dengan baik. Untuk menguji coba API, berikut ini adalah sebuah URI untuk *request* data nilai mahasiswa dengan nim 23090470 pada kelas dengan kode matakuliah SI1016, semester gasal dan tahun ajaran 2012/2013:

```
http://192.168.35.1/PortalUKDW_API/getNilai?callback=jQuery1710
9429647739395332_1374027600719&authorization=23090470%3AYjIwYTZlNGEzYj
k3NTFlOTRkMjBlNzcyMmJiN2U5NDIzNjQwYmNmMDg3NjQ0MjUxOWFhZWZkMGFlYjJhNjkw
MQ%3D%3D&timestamp=1374027675&username=23090470&kode=SI1016&semester=G
ASAL&thn_ajaran=2012%2F2013&grup=A&_=1374027675672
```

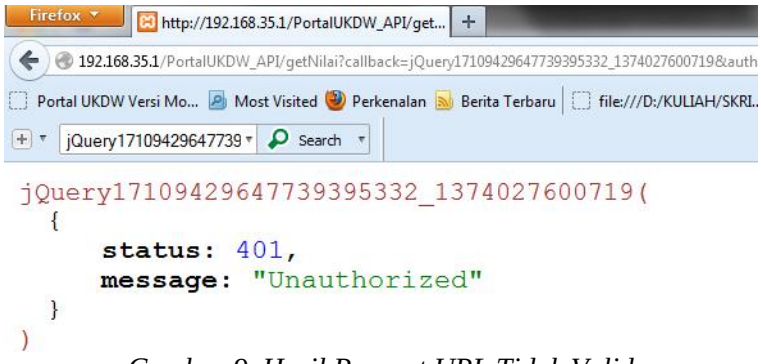
Untuk melakukan uji coba, penulis menyiapkan dua fungsi pada API yang melakukan sama-sama mengembalikan data nilai. Fungsi pertama diberi nama `getNilai()` dan fungsi ke dua diberi nama `getNilai2()`. `getNilai()` menerapkan proses otorisasi sedangkan `getNilai2()` tidak menerapkan proses otorisasi.

Ketika URI dikirim ke fungsi `getNilai()` pada API maka pengirim akan mendapat respon dari API berupa data nilai seperti pada gambar berikut.



Gambar 8. Hasil Request URL Valid

Request tersebut dinilai valid oleh API karena berhasil melewati proses otorisasi pada fungsi `getNilai()` dan mengembalikan data nilai. Namun URI ini ketika digunakan kembali setelah 60 detik tidak berhasil melewati proses otorisasi sebab API hanya mengijinkan umur sebuah URI hanyalah 60 detik. Jika lebih dari itu URI dianggap tidak valid. Berikut ini adalah contoh bila URI tersebut digunakan setelah 60 detik.



Gambar 9. Hasil Request URL Tidak Valid

Selanjutnya URL yang sama diuji coba pada fungsi `getNilai2()` yang tidak menggunakan proses otorisasi. Untuk melakukan URL dimodifikasi pada bagian nama fungsi dengan mengganti `getNilai` menjadi `getNilai2.` Berikut ini hasilnya.

Gambar 10. Hasil Request pada Fungsi getNilai2()

Setelah dilakukan request hasilnya adalah kembalian data nilai. *Request* ini selalu mengembalikan data nilai ketika dilakukan kapan pun, sebab tidak ada proses otorisasi yang terjadi.

Dari hasil uji coba di atas dapat diketahui bahwa proses otorisasi pada API dapat berjalan dengan baik sehingga keamanan data dapat terjaga.

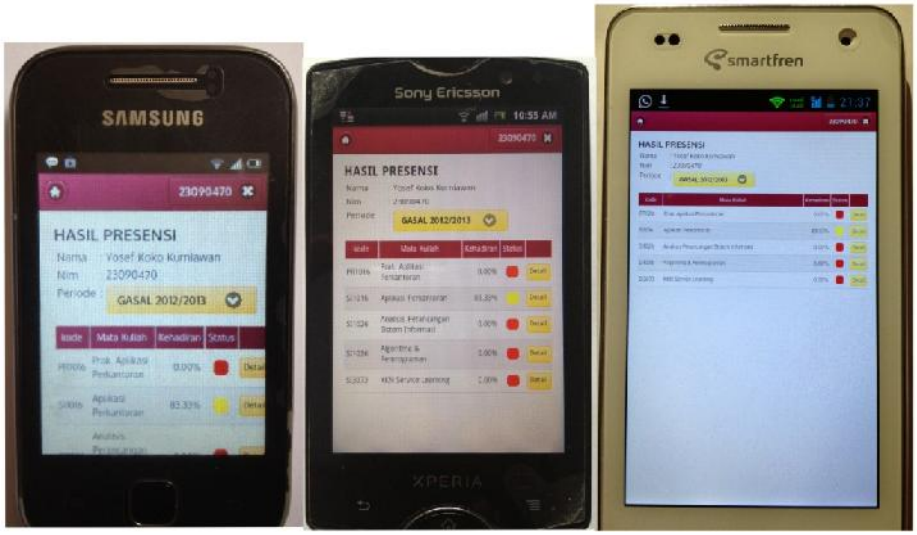
b. Uji Coba Applikasi

Uji coba aplikasi dilakukan pada beberapa perangkat *mobile* berbasis Android. Uji coba ini bertujuan untuk mengetahui performa aplikasi pada berbagai ukuran layar, berbagai OS Android, serta beberapa vendor yang mengusung Android sebagau OS-nya. Uji coba mengabaikan faktor banyaknya aplikasi yang sudah terinstal pada perangkat, processor, serta memori RAM yang digunakan.

Tabel 1.
Hasil Uji Coba Applikasi di Berbagai Perangkat

Nama Perangkat	Spesifikasi	Parameter		
		Waktu Loading	Animasi transisi dan tombol	Tampilan
Cyrus TV Pad	OS : Ginger Bread Display : 800x480	11-12 detik	transisi tidak jalan, delay saat tekan tombol	OK
Samsung Galaxy Young	OS : Ginger Bread Display : 320x240	3-4 detik	transisi tidak jalan, tombol lancar	Beberapa tabel terpotong
Sony Ericsson Xperia Mini Pro	OS : Ginger Bread Display : 320x480	9-11 detik	transisi tidak jalan, tombol lancar	OK
LG P350	OS : Froyo Display : 320x240	16-18 detik	Animasi tidak jalan, tombol delay	OK
Orly Tablet A700	OS : Jelly Bean Display : 800x480	10 detik	Animasi dan tombol lancar	OK
Sony Ericson Xperia Active	OS : Ice Cream Sandwich Display : 320x480	6-7 detik	Animasi dan tombol lancar	OK
LG P500	OS : Jelly Bean Display : 320x480	8-9 detik	Animasi dan tombol lancar	OK
Smartfren Andromax	OS : Jelly Bean Display : 320x480	4-5 detik	Animasi dan tombol lancar	OK

Pada saat uji coba, tampilan aplikasi menjadi kurang maksimal di beberapa perangkat yang mempunyai resolusi layar 320px X 240px. Berikut ini adalah perbandingan tampilan aplikasi pada beberapa ukuran resolusi layar perangkat.



Gambar 11. Perbandingan Tampilan Applikasi di Berbagai Ukuran Resolusi Layar

Gambar di atas merupaka aplikasi yang berjalan pada resolusi layar 320px X 240px (a), 320px X 480px (b), dan 800px X 480px. Pada gambar dapat diketahui tampilan aplikasi pada layar resolusi 320px X 240px nampak kurang maksimal dengan terpotongnya sisi kanan Tabel Hasil Presensi.

Berdasarkan data serta analis di atas maka dapat diketahui beberapa hal berikut ini.

1) Animasi transisi tidak berjalan pada OS Android versi Ginger Bread ke bawah.
2) Untuk mendapatkan tampilan yang maksimal maka layar setidaknya harus berukuran 320px X 480px. Jika di bawah itu maka ada kemungkinan ada bagian-bagian tabel yang tidak namapak di layar.
3) Aplikasi dapat berjalan dengan cukup baik pada Sistem Operasi Android versi Froyo sampai Jelly Bean.

5. Kesimpulan

Berikut ini adalah kesimpulan yang diperoleh dalam penelitian ini :

a. Dalam membangun Aplikasi Portal Akademik UKDW Berbasis Android yang terintegrasi dengan sistem yang sudah ada maka diperlukan sebuah API sebagai pihak yang mengelola permintaan data dari aplikasi dan menghubungkannya dengan basis data.
b. REST API mudah untuk diakses dengan URI melalui protokol HTTP. Namun kemudahan ini dapat menjadi sebuah kelemahan. URI dapat dicuri, begitu pula data. Untuk itu diperlukan proses otorisasi untuk mencegah *request* yang tidak valid masuk.
c. Aplikasi Informasi Portal UKDW berbasis Android dapat berjalan dengan cukup baik di berbagai perangkat *mobile* berbasis Android. Akan tetapi tampilan tidak dapat maksimal untuk perangkat dengan resolusi layar 320px X 240px. Hal ini disebabkan aplikasi menampilkan informasi di beberapa fitur menggunakan tabel. Semakin kecil ukuran resolusi layar maka kemungkinan bagian tabel yang tidak terlihat di layar akan semakin besar.
d. Fitur-fitur aplikasi bersifat membaca basis data. Semua fitur tersebut dapat diimplementasikan di berbagai perangkat *mobile* berbasis Android. Tampilan aplikasi mampu menyesuaikan berbagai ukuran layar. Dengan demikian aplikasi diharapkan mampu membantu pengguna dalam memperoleh informasi seputar akademik, presensi, dan e-Class menggunakan perangkat *mobile* berbasis Android.

Daftar Pustaka

Amazon 2006. *Authenticating Requests Using the REST API*. Diakses dari World Wide Web: http://docs.aws.amazon.com/AmazonS3/latest/dev/S3_Authentication2.html pada 5 Juli 2013

Hakim, Zainal. *Pengertian Web Portal*. Dikutip dari http://www.zainalhakim.web.id/pengertian-web-portal.html. Di akses pada 23 April 2013

Hermawan S, Stephanus. 2011. *Mudah Membuat Aplikasi Android*. Yogyakarta: Andi Offset.

Safaat H, Nazruddin. 2012. *Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android.* Bandung: Informatika Bandung.

World Wide Web Consortium. 2004. *Web Service Architecture*. Diakses dari World Wide Web: http://www.w3.org/TR/ws-arch/ pada 7 Juli 2013