TELEMATIKA, Vol. 13, No. 02, JULI, 2016, Pp. 107 – 122
ISSN 1829-667X

UTILIZATION OF MOODLE WEB SERVICE BASED SYSTEM TO SYSTEM WITH SIAKAD AND SSO UNS

Ristu Saptono(1), Meiyanto Eko Sulistyo(2), Joko Susilo(3)
1,3)Program Studi Informatika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Sebelas Maret, Surakarta
2)Program Studi Teknik Elektro
Fakultas Teknik
Universitas Sebelas Maret, Surakarta
ristu.saptono@staff.uns.ac.id(1), mekosulistyo@staff.uns.ac.id(2), jokosusilo@student.uns.ac.id(3)

Abstract

The development of information technology in education allows for integration between systems so every system can be optimized. Elearning, SIAKAD, and SSO UNS are education system in UNS (Universitas Sebelas Maret) but they are not integrated yet. Course data for elearning is still manual and SSO which can not be used to log into SIAKAD. In this study the integration of elearning, SIAKAD, and SSO utilizing REST web service and exchange data using JSON. As a result, the integration of additional system must use a bridge application as a customizer data between elearning and SIAKAD. While the results of the testing to include one course, 40 lecturers, and 40 students, including automatically enroll is 60.22 seconds, while the time required for unenroll lecturers and students is 2:13 seconds. To enroll course, lecturers and students when there are previously data was 28.5 seconds.

Keywords: Elearning, JSON, SIAKAD, SSO, Web Service

Abstrak

Perkembangan teknologi informasi dalam pendidikan memungkinkan untuk integrasi antar sistem sehingga setiap sistem dapat dioptimalkan. *Elearning*, SIAKAD, dan SSO UNS yang sistem pendidikan di UNS (Universitas Sebelas Maret) tetapi mereka belum terintegrasi. Data mata kuliah untuk *elearning* masih manual dan SSO yang tidak dapat digunakan untuk *login* ke SIAKAD. Dalam penelitian ini integrasi *elearning*, SIAKAD, dan SSO memanfaatkan REST *web service* dan pertukaran data menggunakan JSON. Akibatnya, integrasi sistem tambahan harus menggunakan aplikasi *bridge* sebagai data penyesuai antara *elearning* dan SIAKAD. Sedangkan hasil pengujian untuk memasukkan satu mata kuliah, 40 dosen dan 40 mahasiswa, termasuk secara otomatis mendaftarkan diri adalah 60,22 detik, sedangkan waktu yang dibutuhkan untuk dosen dan mahasiswa membatalkan pendaftaran 2,13 detik. Untuk mendaftarkan mata kuliah, dosen, dan mahasiswa ketika ada data sebelumnya adalah 28,5 detik.

Kata Kunci: Elearning, JSON, SIAKAD, SSO, Web Service

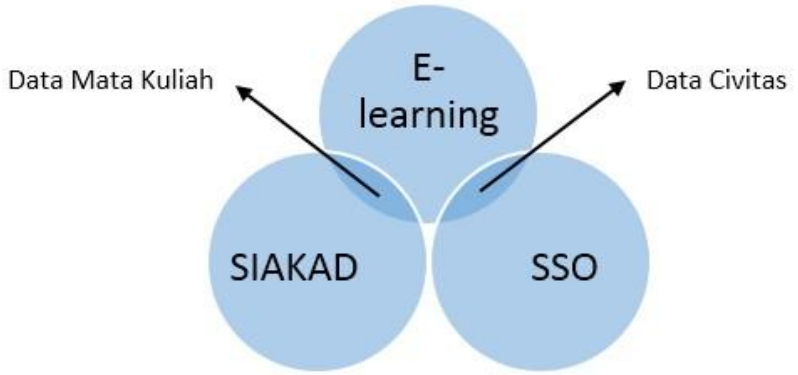
1. PENDAHULUAN

Universitas Sebelas Maret (UNS) mempunyai visi menjadi pusat pengembangan ilmu, teknologi, dan seni yang unggul di tingkat internasional dengan berlandaskan pada nilai-nilai luhur budaya nasional [1]. Salah satu penerapan teknologi adalah dengan menggunakan *elearning* sebagai komplementer proses belajar mengajar. *Elearning* merupakan suatu jenis belajar mengajar yang memungkinkan tersampaikannya bahan ajar ke siswa dengan menggunakan media *internet*, *intranet*, atau media jaringan komputer lain [2].

Elearning yang dipakai oleh Universitas Sebelas Maret adalah adalah MOODLE (*Modular Object-Oriented Dynamic Learning Environment*) [3]. Menurut penelitian salah satunya [4] serta [5] dilihat dari beberapa kategori dan fitur yang dimiliki seperti *Usability*, *Adaptation*, *Course*

Management, dan lain-lain, MOODLE adalah yang terbaik. Selain *elearning* yang memakai MOODLE, aplikasi lain yang mendukung sistem belajar mengajar adalah Sistem Informasi Akademik (SIAKAD) UNS yang digunakan untuk manajemen data civitas akademik. Sedangkan, *Single Sign On* (SSO) digunakan untuk pemusatan data diri civitas akademik, *email*, dan akun untuk *login* ke *hotspot*.

Masalah yang dihadapi dari hadirnya beberapa aplikasi tersebut, adalah setiap kali masuk ke semester baru, dosen akan membuat *course* (kelas), melakukan *enroll* (membuka kelas), dan *unenroll* (menutup kelas) manual terhadap mahasiswa yang ada. Sedangkan mahasiswa baru akan mendaftar *elearning* secara manual. Hal ini terasa kurang maksimal penggunaanya karena data mata kuliah (*course*) di *elearning* bisa diambil dari data mata kuliah yang ada di SIAKAD, sedangkan data mahasiswa dan dosen di *elearning* bisa diambil dari SSO sebagaimana terlihat pada gambar 1.



Gambar 1. Hubungan data antar sistem

Dengan mengintegrasikan ketiga sistem tersebut, maka pembuatan *course* (kelas), pembuatan akun *elearning*, *enroll course* (membuka kelas), dan *unenroll course* (menutup kelas) akan bisa dilakukan secara otomatis. Hal ini tentu akan menjadikan dosen dan mahasiswa lebih fokus pada proses belajar mengajar di kelas dan tidak perlu melakukan pembuatan *course* (kelas) atau pendaftaran di *elearning*. Integrasi ini bisa dilakukan karena MOODLE menyediakan fitur atau modul *web service*.

Web service adalah sebuah sistem *software* yang didesain untuk mendukung interoperabilitas interaksi mesin ke mesin melalui sebuah jaringan [6]. MOODLE versi 2.9 mendukung empat *engine web service*, yaitu AMF, SOAP, REST, dan XML-RPC. Pemanfaatan *web service* MOODLE sebelumnya oleh Bayu Wicaksono adalah Model Pemanfaatan *Web Service* Moodle Berbasis REST-JSON Secara *Mobile* Dengan Perancangan *Native Client* Berbasis Android [7]. Memanfaatkan *web service* moodle untuk membuat sebuah aplikasi android, yang bisa mengakses beberapa fungsi yang ada di moodle. Dari penelitian tersebut belum membahas mengenai pemanfaatan *web service* MOODLE yang dihubungkan dengan *web service* lain. Dalam hal ini adalah sistem SIAKAD dan SSO. Karena pada dasarnya *web service* moodle dibangun untuk bisa digunakan untuk *system to system* yang artinya bisa menghubungkan antara dua aplikasi.

Dibutuhkan sebuah aplikasi untuk menjembatani *web service* satu dengan *web service* lainnya. Pada penelitian ini, digunakan REST sebagai *engine* karena REST lebih *powerfull* dibanding dengan SOAP, dan untuk format pertukaran datanya menggunakan JSON karena lebih ringan jika digunakan untuk data yang isinya banyak [8]. Sedangkan metode yang dipakai adalah *waterfall* karena sangat cocok dengan *project* yang *requirement*-nya sudah pasti.
Dengan adanya integrasi antara *elearning* (MOODLE), SIAKAD dan SSO UNS maka akan di dapat *elearning* yang berjalan secara *seamlessly*, artinya pembuatan *course* (kelas), pembuatan akun *elearning*, *enroll course* (membuka kelas), dan *unenroll course* (menutup kelas) akan bisa dilakukan secara otomatis, sehingga proses belajar mengajar akan lebih optimal.

2. DASAR TEORI

2.1. *Elearning*

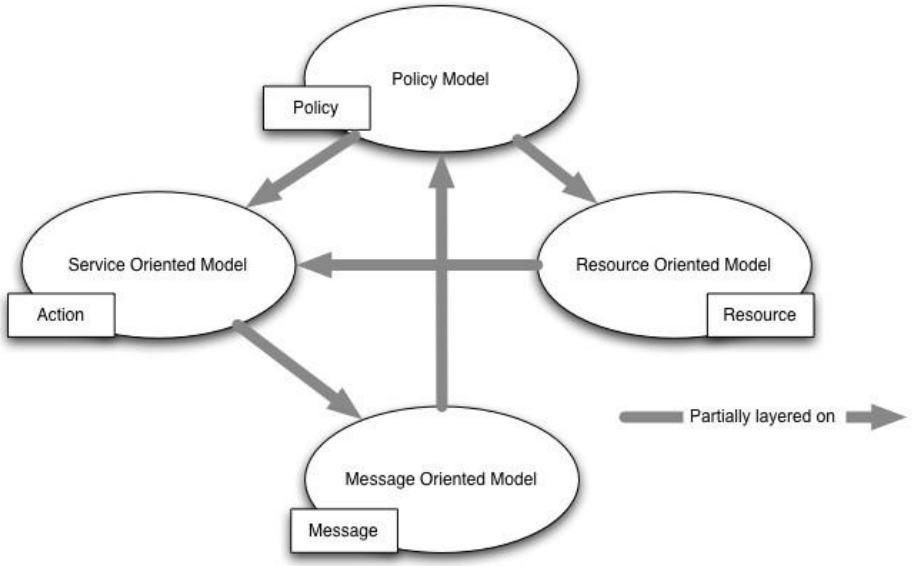
Elearning merupakan suatu jenis belajar mengajar yang memungkinkan tersampaikannya bahan ajar ke siswa dengan menggunakan media *internet*, *intranet*, atau media jaringan komputer yang lain [2]. Sesuai dengan namanya, arti *elearning* adalah sistem pembelajaran menggunakan media elektronik. Sistem pembelajaran ini mulai berkembang seiring perkembangan teknologi itu sendiri. Salah satu implementasinya dengan menggunakan LMS.

LMS atau singkatan dari *Learning Management System* adalah suatu perangkat lunak atau *software* untuk keperluan administrasi, dokumentasi, laporan sebuah kegiatan, kegiatan belajar mengajar dan kegiatan secara *online* (terhubung ke *internet*), *elearning*, dan materi-materi pelatihan. Dan semua itu dilakukan dengan *online*. [9].

Salah satu LMS yang paling popular adalah MOODLE. MOODLE atau singkatan dari *Modular Object-Oriented Dynamic Learning Environment* adalah sebuah paket aplikasi untuk *elearning* yang bersifat *open source*. Karena sifat *open source* inilah, maka pengguna MOODLE cepat berkembang dan memiliki andil dalam mengembangkan moodle itu sendiri. Caranya dengan pembuatan *plugin* atau *module* untuk MOODLE.

2.2. *Web Service*

Web service adalah sebuah sistem yang didesain untuk mendukung interaksi mesin dengan mesin yang melalui sebuah jaringan [10]. Dalam bahasa yang lebih mudah, *web service* adalah sebuah sistem yang dibuat untuk pertukaran data dengan sistem atau aplikasi lain. Arsitektur *web service* memiliki 4 model [11] yaitu sebagai berikut :



Gambar 2. Arsitektur *Web Service* [11]

1. *Message Oriented Model*
   Fokus pada aspek arsitektur yang berhubungan dengan pesan dan prosesnya seperti struktur pesan, transportasi pesan dan yang lainnya.
2. *Service Oriented Model*
   Fokus pada arsitektur yang berhubungan dengan servis dan aksi.
3. *Resource Oriented Model*
   Fokus pada aspek yang berhubungan dengan *resource*. *Resource* adalah konsep paling fundamental dalam *web service*.
4. *Policy Oriented Model*
   Fokus pada aspek arsitektur yang berhubungan dengan aturan, contohnya keamanan dan kualitas dari servis yang dihasilkan.

Dari 4 model di atas, *Service Oriented Model* berkembang menjadi SOAP dan *Resource Oriented Model* berkembang menjadi REST. SOAP adalah sebuah protokol yang diusulkan oleh W3C untuk antarmuka *web service* dan digunakan untuk memanggil fungsi *remote* (XML-RPC), sedangkan REST adalah sebuah arsitektur metode komunikasi untuk sistem *hypermedia* terdistribusi, seperti *World Wide Web* yang diperkenalkan oleh Roy Fielding di tahun 2000 melalui disertasinya. REST memiliki karakteristik sebagai berikut [12]:

1. Menggunakan HTTP method secara eksplisit
2. *Stateless*
3. Memiliki struktur direktori URI
4. Pesan yang ditransfer dalam format XML, JSON, atau keduanya.

2.3. Format Data

Format data yang sering dipakai dalam *web service* adalah XML dan JSON.

1. XML
   XML (*Extensible Markup Language*) adalah sebuah format data yang berasal dari SGML (ISO 8879). Awalnya di desain untuk memenuhi kebutuhan penerbitan elektronik dalam skala besar. Kemudian berkembang dan memiliki peranan penting dalam pertukaran data di di web [13]. Contoh penggunaan XML untuk mendeklarasikan data adalah sebagai berikut:

```
<employees>
    <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
    </employee>
</employees>
```

Gambar 3. Contoh Penggunaan XML [14]

2. JSON
   JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemrograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh *programmer* keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python, dan lain-lain. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data [15]. JSON terbuat dari dua struktur [15]:
   1. 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*.
   2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

   Contoh penggunaan JSON untuk mendeklarasikan data adalah sebagai berikut :

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

Gambar 4. Contoh Penggunaan JSON [14]

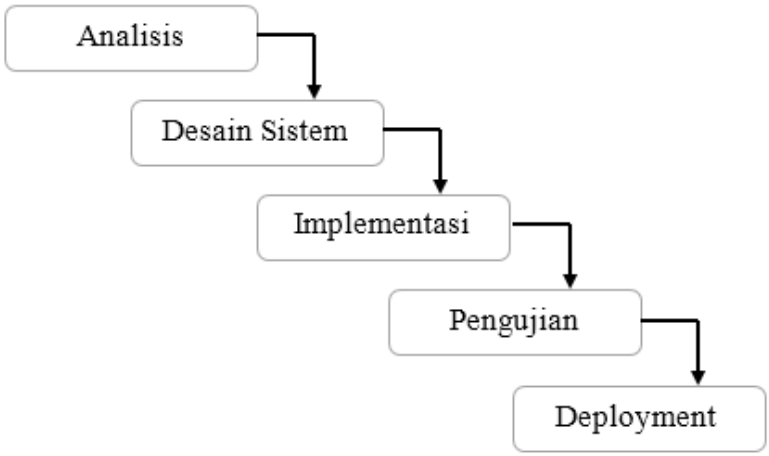
2.4. SSO

Single Sign-On (SSO) adalah sebuah sesi atau proses autentikasi yang memperbolehkan seorang user menggunakan satu *username* dan *password* untuk mengakses beberapa aplikasi [16]. Beberapa keuntungan penggunaan SSO adalah sebagai berikut

1. User tidak perlu mengingat banyak *username* dan *password*.
2. Kemudahan pemrosesan data, karena data yang ada sudah terpusat
3. Tidak perlu membuat data pengguna yang sama di setiap aplikasi. Menghemat biaya untuk pemeliharaan akun.

3. METODOLOGI

Tahapan penelitian ini adalah sebagai berikut :



Gambar 5. Tahapan Penelitian

3.1. Analisis

Dalam proses analisis ini dilakukan identifikasi *business case*, selanjutnya menentukan *requirement* (kebutuhan sistem) yang akan digunakan dalam proses pembuatan aplikasi. Selain itu, juga dilakukan studi terhadap rencana aplikasi yang akan digunakan, menggunakan *platform* apa dan *tools*-nya apa saja.

3.2. Desain Sistem

Setelah mengetahui kebutuhan dan fitur yang akan dibuat dari proses analisis, maka selanjutnya pada desain sistem dilakukan perancangan sistem. Perancangan ini meliputi design UI (*User Interface*), data model, alur program dan rancangan sistem. Rancangan sistem nantinya akan dibuat berdasarkan diagram UML.

3.3. Implementasi

Merupakan tahap menterjemahkan model atau desain yang telah ditetapkan pada proses analisis dan desain sistem ke dalam bahasa yang dimengerti komputer.

3.4. Pengujian

Setelah penyelesaian proses implementasi, selanjutnya dilakukan *testing*. Tahapan ini ialah tahapan di mana developer harus menguji kelayakan aplikasi apakah sesuai dengan yang diharapkan atau tidak. Proses *testing* dan *debugging* dibagi menjadi 2 macam. Pengujian yang pertama, *black box testing*, yaitu pengujian untuk mengetahui apakah semua fungsional sistem berjalan sesuai yang direncanakan atau tidak. Sedangkan, pengujian kedua adalah pengujian *response time*, yaitu pengujian untuk mengetahui berapa waktu yang dibutuhkan sistem untuk melakukan *enroll* dan *unenroll*.

3.5. *Deployment*

Ini merupakan tahap terakhir dalam model *waterfall*. *Software* yang sudah jadi dilakukan pemasangan atau dijalankan serta dilakukan pemeliharaan.
Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya.

4. HASIL DAN PEMBAHASAN

4.1. Analisis

Aplikasi yang akan dikembangkan adalah aplikasi penengah (*Bridge App*) antara *elearning* dan SIAKAD. Aplikasi ini akan dikelola oleh admin dengan cara mengisi jadwal kapan dilakukan *enroll* dan *unenroll* terhadap dosen dan mahasiswa. Kondisi awal dari semua sistem adalah *elearning* dan aplikasi berada dalam kondisi kosong hanya berisi data *default*. Admin akan menambahkan jadwal kapan dilakukan *enroll* dan *unenroll* tiap semester pada sistem. *Enroll* akan dilakukan setelah jadwal pengambilan mata kuliah di SIAKAD selesai. Karena data pengambilan mata kuliah digunakan untuk membuat *course* (kelas), *user* dosen, dan *user* mahasiswa di *elearning*. Setelah itu jadwal dimasukkan, sistem akan melakukan pencocokan jadwal setiap hari. Jika jadwal *enroll* sama dengan hari sekarang maka akan dilakukan proses *enroll*, begitu juga saat jadwal *unenroll*. Proses *enroll* adalah proses dimana aplikasi akan mengambil data dari SIAKAD berupa data mata kuliah, *user* dosen beserta mata kuliah yang diampu, dan mahasiswa beserta mata kuliah yang diambil untuk kemudian dimasukkan ke dalam *elearning* (MOODLE). Setelah proses pemasukan data tadi, kemudian dilakukan proses *enroll* atau proses memasukkan dosen dan mahasiswa ke dalam *course* (kelas) masing-masing. Sedangkan, proses *unenroll* adalah proses dimana *course* (kelas) akan ditutup sementara waktu sampai semester berikutnya, sehingga mahasiswa dan dosen tidak akan bisa lagi masuk ke dalam kelas.

4.1.1. Kebutuhan Fungsional

1. Sistem dapat menerima *login* dari admin.
2. Setelah *login*, admin dapat melihat daftar mata kuliah, mahasiswa dan dosen yang ada di MOODLE melalui sistem.
3. Setelah *login*, admin dapat melakukan proses pengelolaan jadwal *running* aplikasi, seperti menambahkan jadwal dan melihat jadwal.
4. Sistem dapat secara otomatis membuatkan akun untuk dosen dan mahasiswa berdasarkan NIP atau NIM. NIP atau NIM inilah yang nantinya dicocokkan dengan data SSO sehingga dosen dan mahasiswa bisa login elearning menggunakan SSO.
5. Sistem dapat melakukan proses *enroll* dosen dan mahasiswa terhadap kelas di *elearning* secara otomatis.
6. Dosen dan mahasiswa dapat melakukan *login* di *elearning* menggunakan SSO.

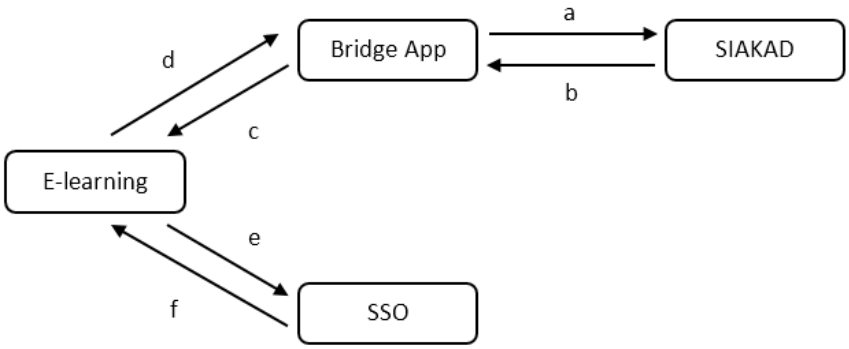
7. Setelah *login*, mahasiswa dapat melihat data *course* (kelas) di *elearning*, sedangkan dosen dapat pula melakukan perubahan dari kelas yang dibimbing.

4.1.2. Kebutuhan Non-Fungsional

1. Sistem menyimpan data mata kuliah, mahasiswa dan dosen.
2. Sistem berjalan 24 jam perhari
3. Sistem melakukan pengecekan jadwal *enroll* dan *unenroll* setiap hari sekali

4.2. Desain Sistem

Arsitektur pengembangan sistem secara keseluruhan dapat dilihat pada gambar berikut:



Gambar 6. Desain Aplikasi

Keterangan Gambar 6 :
a. *Bridge App* me-*request* data ke SIAKAD
b. *Bridge App* mendapat data dari SIAKAD
c. Data dari SIAKAD dikirim ke *elearning*
d. *Elearning* mengirim data kembalian berupa hasil eksekusi fungsi MOODLE
e. *Elearning* melakukan *request* login ke SSO
f. SSO mengirim data kembalian ke *elearning*

Setelah dilakukan studi terhadap data dan *requirement* yang dibutuhkan dalam aplikasi. Ada 3 sistem yang akan digunakan pada penelitian ini, yaitu Sistem Informasi Akadmik (SIAKAD), elearning (MOODLE), *Single Sign On* (SSO).

4.2.1. SIAKAD

SIAKAD yang digunakan bukanlah SIAKAD sungguhan. SIAKAD yang digunakan berupa *web service* dengan 3 fungsi yang akan mengembalikan data JSON tergantung dari fungsi yang dipanggil. Tabel 1 berikut berisi daftar fungsi API SIAKAD.

Tabel 1. Daftar fungsi API SIAKAD

Nama fungsi	Deskripsi
getMata kuliah	*Return* JSON berisi daftar mata kuliah berupa nama panjang dan singkatan
getDosen	*Return* JSON berisi daftar dosen beserta mata kuliah yang diampu
getMahasiswa	*Return* JSON berisi daftar mahasiswa beserta mata kuliah yang diambil

4.2.2. *Elearning* (MOODLE)

Untuk menggunakan *web service* MOODLE, hal yang harus dilakukan adalah dengan *login* sebagai *administrator* kemudian menghidupkan fungsi *web service*. Setelah itu, hidupkan tipe *web service* REST. Dan lakukan *generate key*. *Key* ini yang akan digunakan di aplikasi. Dan kemudian *web service* MOODLE siap digunakan. Salah satu data yang dibutuhkan adalah tipe *user* MOODLE. Tipe *user* di MOODLE dibedakan menjadi beberapa kategori. Tipe user inilah

yang nantinya digunakan untuk membedakan antara dosen dan mahasiswa. Tabel 2 berikut berisi daftar tipe *user* yang ada di MOODLE.

Tabel 2. Daftar tipe *user* dan *level* di *elearning* MOODLE

No	Tipe User	Deskripsi
1	*Site administrator*	Dapat 'melakukan apa saja' di *website*
2	*Manager*	Role yang sedikit lebih sempit dibanding admin
3	*Course Creator*	Dapat membuat kelas (*course*)
4	*Teacher*	Dapat mengelola dan menambah konten ke kelas (*course*)
5	*Non-editing teacher*	Dapat mengelola kelas (*course*), tapi tidak bisa mengedit
6	*Student*	Dapat mengakses dan berpartisipasi di kelas

Data selanjutnya adalah daftar fungsi *web service* yang ada di MOODLE. Fungsi-fungsi inilah yang bisa diakses untuk melakukan perubahan di dalam aplikasi moodle melalui *web service*.

Setelah melakukan studi literatur dan analisa terhadap masalah yang ada, dipilih beberapa fungsi yang sesuai kebutuhan aplikasi. Dari daftar fungsi tersebut, fungsi yang dipakai dalam aplikasi ditunjukkan oleh tabel 3 sebagai berikut.

Tabel 3. Daftar fungsi *web service* moodle yang dipakai

Nama Fungsi
core_course_get_categories
core_course_create_categories
core_course_get_courses
core_user_create_users
core_course_create_courses
enrol_manual_enrol_users

4.2.3. SSO

Untuk dapat menggunakan *Single Sign On* (SSO) Universitas Sebelas Maret, perlu ditambahkan *plugin* auth_saml ke dalam MOODLE. Selain itu di mesin yang sama harus ter-*install* SimpleSAMLphp. Setelah sistem yang lain siap, selanjutnya adalah daftar fungsi aplikasi. Tabel 4 merupakan daftar fungsi di *Bridge App*.

Tabel 4. Daftar fungsi *bridge app*

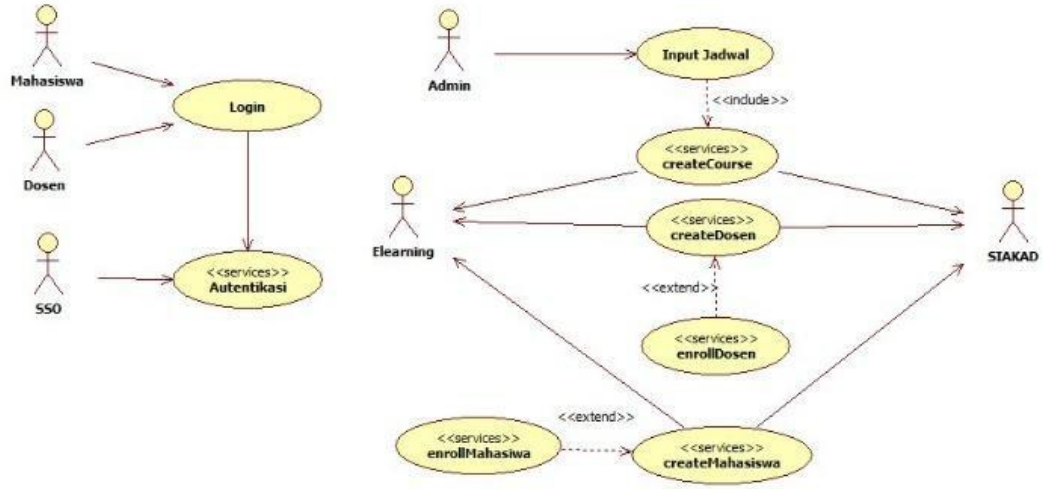
Golongan	Nama fungsi	Deskripsi
Course	checkCourse	Melakukan cek ke *database* apakah suatu *course* sudah di masukkan ke *elearning* atau belum
	Send	Melakukan pemanggilan *web service* ke *elearning*
	saveCourse	Melakukan *save* data *course* ke *database*
Dosen	checkCourse	Melakukan cek ke *database* apakah dosen dengan nama yang dicek sudah di masukkan ke *elearning* atau belum
	Send	Melakukan pemanggilan *web service* ke *elearning*
	saveCourse	Melakukan *save* data dosen ke database
Mahasiswa	checkCourse	Melakukan cek ke *database* apakah mahasiswa dengan nama yang dicek sudah di masukkan ke *elearning* atau belum
	Send	Melakukan pemanggilan *web service* ke *elearning*
	saveCourse	Melakukan *save* data mahasiswa ke *database*

Use Case Glossary sebagai bagian dari perancangan aplikasi dapat dilihat di bawah ini.

Tabel 5. *Use Case Glossary*

Kode	Use Case	Deskripsi
UC-001	Login	Dosen dan Mahasiswa login ke *elearning*
UC-002	Autentikasi	SSO member akses autentikasi ke *elearning*
UC-003	Input Jadwal	Admin mengelola jadwal
UC-004	Create Course	Sistem menambahkan *course* ke *elearning*
UC-005	Create Dosen	Sistem menambahkan dosen ke *elearning*
UC-006	Enroll Dosen	Sistem melakukan *enroll* dosen ke *elearning*
UC-007	Create Mahasiswa	Sistem menambahkan mahasiswa ke *elearning*
UC-008	Enroll Mahasiswa	Sistem melakukan *enroll* mahasiswa ke *elearning*

Use Case diagram dapat dilihat pada gambar berikut.



Gambar 7. *Use Case Diagram*

4.3. Implementasi

Hasil implementasi dari prototipe sistem yang menghubungkan SIAKAD dengan *elearning* adalah sebagai berikut :

1. Data JSON untuk fungsi getMataKuliah

Gambar 8. Data JSON untuk fungsi getMataKuliah

Gambar 8 diatas adalah JSON hasil request dari aplikasi ke SIAKAD untuk mata kuliah.

2. Data JSON untuk fungsi getDosen

Gambar 9. Data JSON untuk fungsi getDosen

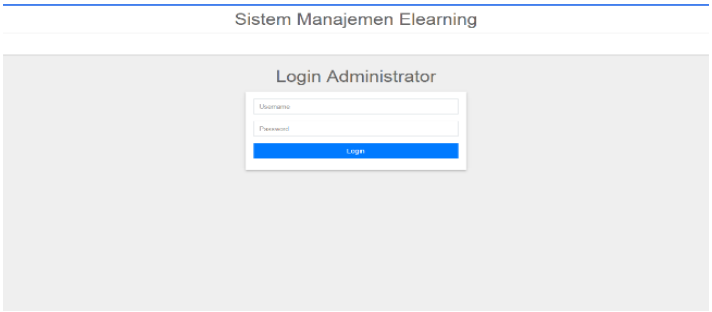
Gambar 9 diatas adalah JSON hasil request dari aplikasi ke SIAKAD untuk mata kuliah.

3. Data JSON untuk fungsi getMahasiswa

Gambar 10. Data JSON untuk fungsi getMahasiswa

Gambar 10 di atas adalah JSON hasil request dari aplikasi ke SIAKAD untuk mahasiswa.

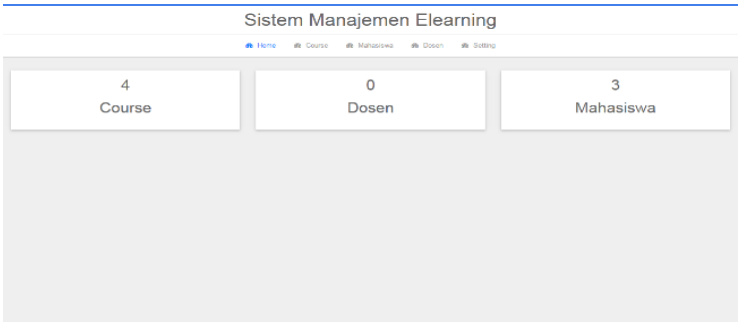
4. Halaman Login Administrator



Gambar 11. Tampilan Halaman Login Administrator

Halaman ini digunakan *administrator* untuk *login* ke dalam sistem.

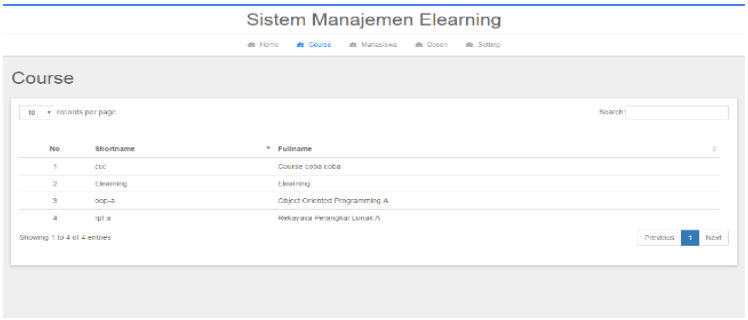
5. Halaman *Home*



Gambar 12. Tampilan Halaman *Home*

Halaman *home* menyajikan jumlah *course*, dosen, dan mahasiswa yang ada di *elearning*, terlihat di gambar 12.

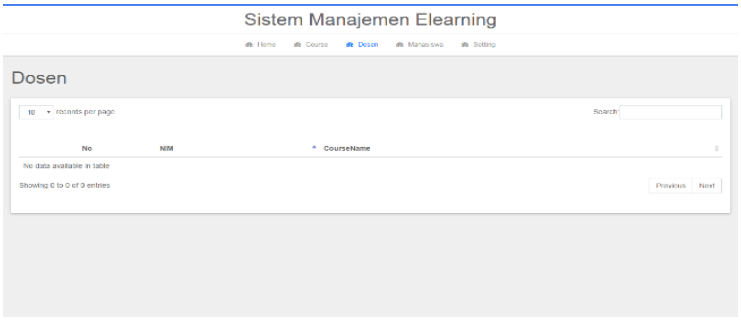
6. Halaman *Course*



Gambar 13. Tampilan Halaman *Course*

Beralih ke halaman *course* yang berisi semua *course* yang ada di *elearning*. Data yang di simpan dari *course* adalah *shortname* dan *fullname*-nya. Tampilan data *course* dapat di lihat di gambar 13.

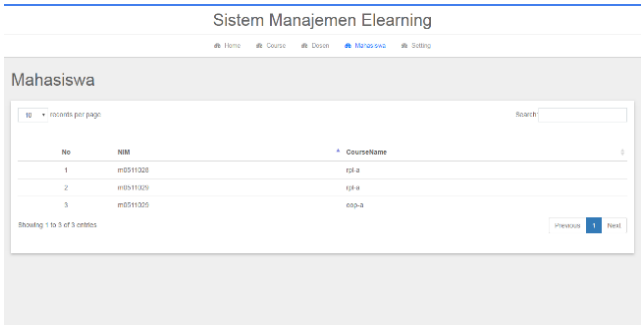
7. Halaman Dosen



Gambar 14. Tampilan Halaman Dosen

Menu selanjutnya adalah dosen, yang berisi data semua dosen yang ada di *elearning*. Seperti *course*, datanya berbentuk tabel. Untuk *field* yang disimpan adalah NIM dan *course* yang diambil.

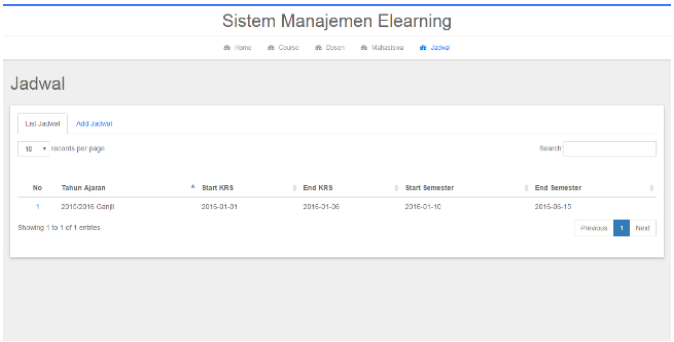
8. Halaman Mahasiswa



Gambar 15. Tampilan Halaman Mahasiswa

Kemudian, menu yang disebelahnya adalah mahasiswa. Berisi data semua mahasiswa yang ada di *elearning*. Seperti *course*, datanya berbentuk tabel. Untuk *field* yang disimpan adalah NIM dan *course* yang diambil.

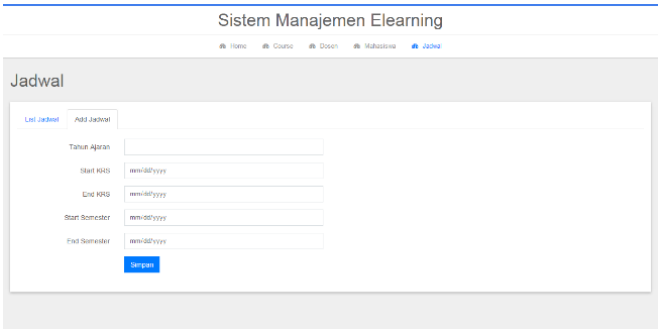
9. Halaman Jadwal



Gambar 16. Tampilan Halaman Jadwal

Menu terakhir yang utama adalah jadwal. Jadwal yang diinputkan disini nantinya yang akan melakukan *trigger* terhadap *elearning* dan SIAKAD. Jadwal yang dipakai untuk melakukan *trigger* adalah jadwal terakhir.

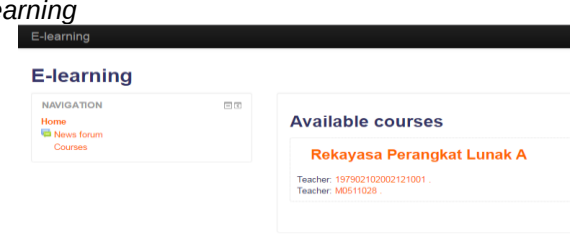
10. Halaman Input Jadwal



Gambar 17. Tampilan Halaman Jadwal

Untuk menambahkan jadwal baru, klik *add* jadwal. Maka akan muncul tampilan *form* seperti gambar 17. Semua *field* wajib diisi dan setelah mengklik tombol Simpan, maka akan kembali ke tab list jadwal.

11. Halaman *Home Elearning*



Gambar 18. Tampilan Halaman *Home Elearning*

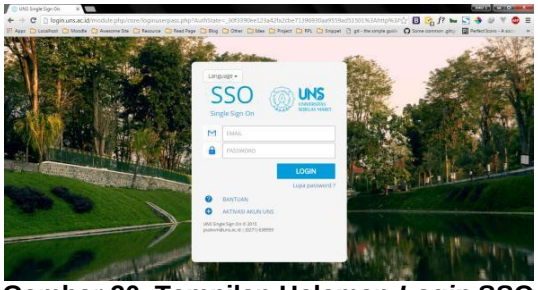
Halaman tampilan awal di elearning yang berisi daftar *course* (kelas) yang sudah dimasukkan dari SIAKAD.

12. Halaman *Login Elearning*

Gambar 19. Tampilan Halaman *Login Elearning*

Gambar 19 adalah halaman *login elearning*. Akan tersedia *link* untuk *login* melalui SSO UNS.

13. Halaman *Login* SSO



Gambar 20. Tampilan Halaman *Login* SSO

Setelah *link login* dengan SSO UNS diklik, maka kita akan di-*redirect* ke halaman seperti tampak pada gambar 20. Selanjutnya tinggal masukkan *email* UNS dan *password*-nya.

14. Halaman Detail *Course* Mahasiswa

Gambar 21. Tampilan Detail *Course* Mahasiswa

Ketika sudah *login* melalui SSO, maka akan di-*redirect* lagi ke halaman awal *elearning*. Setelah itu, kita bisa masuk ke dalam *course* (kelas) yang diambil. Gambar 21 di atas adalah tampilan ketika mahasiswa membuka *course* yang diambil.

15. Halaman Detail *Course* Dosen

Gambar 22. Tampilan Halaman Detail *Course* Dosen

Gambar 22 di atas adalah tampilan dimana yang melakukan *login* dan membuka *course* (kelas) adalah seorang dosen. Bedanya adalah dosen bisa melakukan perubahan konten yang ada dikelas.

4.4. Pengujian

4.1.3. *Black Box Testing*

Black Box Testing digunakan untuk menguji kebutuhan fungsional apakah sudah sesuai dengan yang diharapkan atau belum. Pengujian dilakukan dengan dua macam scenario, yaitu skenario dengan inputan benar dan scenario dengan inputan yang salah. Hasil dari pengujian black box menunjukkan sistem dapat menjalankan semua fungsional dengan baik

4.1.4. *Response Time*

Setelah sistem selesai dibuat, dilakukan uji *web service* untuk *response time*. Pengujian ini terdiri dari 3 jenis, yang pertama untuk data baru, artinya di *elearning* (MOODLE) belum ada data, kemudian sistem akan memasukkan data dari SIAKAD dan melakukan *enroll* terhadap dosen dan mahasiswa. Kedua, *unenroll* dosen dan mahasiswa terhadap mata kuliah di MOODLE. Dan yang ketiga adalah *enroll existing* data, artinya data mata kuliah dan *user* sudah ada di MOODLE setelah dilakukan *unenroll*, kemudian dilakukan proses *enroll* lagi. Ketiga pengujian dilakukan masing-masing sebanyak 10 kali. Berikut adalah hasil yang diperoleh dari proses pengujian.

Tabel 6. Pengujian *New Data*

#	Timestamp	Waktu Eksekusi (detik)
1	2016-04-04 20:10:50	63,16
2	2016-04-04 20:14:50	58,86
3	2016-04-04 20:22:49	58,57
4	2016-04-04 20:27:17	58,04
5	2016-04-04 22:37:13	60,77
6	2016-04-04 22:40:02	65,54
7	2016-04-04 22:42:33	58,96
8	2016-04-04 22:45:46	57,47
9	2016-04-04 22:48:27	60.22
10	2016-04-04 22:52:03	60,61
Rata-Rata		60,22

Tabel 7. Pengujian *Unenroll* Data

#	Timestamp	Waktu Eksekusi (detik)
1	2016-04-04 23:06:26	1,88
2	2016-04-04 23:26:03	2,29
3	2016-04-04 23:30:47	1,83
4	2016-04-04 23:31:24	2,10
5	2016-04-04 23:32:01	2,16
6	2016-04-04 23:32:37	2,15
7	2016-04-04 23:33:15	2,62
8	2016-04-04 23:33:50	2,55
9	2016-04-04 23:34:27	1,96
10	2016-04-04 23:35:24	1,81
Rata-Rata		2,13

Tabel 8. Pengujian *enroll existing* data

#	Timestamp	Waktu Eksekusi (detik)
1	2016-04-04 23:25:05	27,84
2	2016-04-04 23:27:32	35,18
3	2016-04-04 23:31:18	29,15
4	2016-04-04 23:31:56	30,43
5	2016-04-04 23:32:32	28,50
6	2016-04-04 23:33:09	27,72
7	2016-04-04 23:33:44	26,65
8	2016-04-04 23:34:22	26,65
9	2016-04-04 23:34:58	26,70
10	2016-04-04 23:35:56	26,16
Rata-Rata		28,50

Dari data pengujian, didapatkan *response time* untuk pengujian *new data* memiliki rata-rata waktu 60,22 detik. Sedangkan, untuk pengujian *unenroll* data memiliki rata-rata waktu 2,13 detik. Dan untuk pengujian *enroll existing* data memiliki rata-rata waktu 28,5 detik.

5. PENUTUP

5.1. Kesimpulan

1. Integrasi dari SIAKAD, *E-learning* (MOODLE) dan SSO UNS memanfaatkan *web service*, dapat dilakukan dengan membuat sebuah aplikasi penengah (*Bridge App*). Aplikasi ini berfungsi sebagai penyesuai data antara SIAKAD dan *Elearning*.
2. Fungsi *web service* dari MOODLE, yang digunakan adalah core_course_get_categories, core_course_create_categories, core_course_get_courses, core_course_create_courses, core_user_create_users, enrol_manual_enrol_users, Enam fungsi ini adalah fungsi inti untuk membuat *course*, melihat daftar *course*, membuat *user*, melihat daftar *user*, melakukan *enroll*, dan *unenroll* terhadap *user*.
3. Rata-rata waktu yang dibutuhkan untuk memasukkan 1 *course*, 1 dosen, dan 40 mahasiswa adalah 60,22 detik. Sedangkan, waktu yang dibutuhkan untuk *unenroll* dosen dan mahasiswa adalah 2,13 detik dan untuk *enroll* jika sudah ada data sebelumnya dibutuhkan waktu 28,5 detik.

5.2. Saran

Untuk pengembangan penelitian ini selanjutnya dapat dilakukan penelitian lebih lanjut tentang metode pengarsipan *course* yang ada pada MOODLE. Pengarsipan *course* digunakan untuk menyimpan *course* yang telah digunakan. Hal ini bertujuan agar *course* yang telah disimpan tersebut dapat dilihat dikemudian hari. Selain itu, perlu dilakukan penerapan system dengan menggunakan SIAKAD yang sebenarnya, sehingga *response time* yang didapat dapat akan lebih sesuai dengan kondisi yang sebenarnya.

DAFTAR PUSTAKA

Universitas Sebelas Maret, “Visi, Misi, Tujuan dan Budaya Kerja - Universitas Sebelas Maret,” 2015. [Online]. Available: http://uns.ac.id/id/tentang-uns/visi-misi-dan-tujuan/ [Diakses 1 September 2015].

E. H. Darin, “Selling E-Learning, American Society for Training and Development,” 2011. Universitas Sebelas Maret, “Elearning UNS,” 2015. [Online]. Available: http://elearning.uns.ac.id. [Diakses 20 Desember 2015].

N. Cavus dan T. Zabadi, “A Comparison of Open Source Learning Management Systems,” Procedia - Social and Behavioral Sciences, pp. 521-526, 2014.

R. S. Wahono, “Memilih Sistem e-Learning Berbasis Open Source, RomiSatriaWahono.Net,” 2008. [Online]. Available: http://romisatriawahono.net/2008/01/24/memilih-sistem-e-learning-berbasis-open-source/. [Diakses 10 Februari 2016].

The World Wide Web Consortium, “Web Services Architecture,” 2015. [Online]. Available: http://www.w3.org/TR/ws-arch/. [Diakses 4 Agustus 2015].

B. Wicaksono, “Pemanfaatan Web Service Moodle Berbasis Rest-Json Untuk Membangun Moodle Online Learning Extension,” 2014.

A. Wijayanto, “Implementasi Rest Web Service Dengan Menggunakan Json Pada Aplikasi Mobile Enterprise Resource Planning,” 2011.

R. K. Ellis, A Field Guide to Learning Management Systems, Learning Circuits, 2009.

The World Wide Web Consortium, "Web Services Glossary," 2015. [Online]. Available: http://www.w3.org/TR/ws-gloss/. [Diakses 2 Agustus 2015].

The World Wide Web Consortium, "Web Services Architecture," 2015. [Online]. Available: http://www.w3.org/TR/ws-arch/. [Diakses 4 Agustus 2015].

A. Rodriguez, "RESTful Web services: The basics," 2015. [Online]. Available: http://www.ibm.com/developerworks/webservices/library/ws-restful/. [Diakses 20 Desember 2015].

The World Wide Web Consortium, "Extensible Markup Language (XML)," 2015. [Online]. Available: http://www.w3.org/XML/. [Diakses 6 Agustus 2015].

Json Indonesia, "JSON Tutorial," 2015. [Online]. Available: http://www.w3schools.com/json/. [Diakses 26 Februari 2015].

Json Indonesia, "Pengenalan JSON," 2015. [Online]. Available: http://www.json.org/json-id.html. [Diakses 4 Agustus 2015].

Tech Target, "What is single sign-on (SSO)? - Definition from WhatIs.com," 2015. [Online]. Available: http://searchsecurity.techtarget.com/definition/single-sign-on. [Diakses 3 Agustus 2015].