[W

;/--n

**TUGAS AKHIR – KI141502**

**PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGGUNAKAN FITUR MATRIKS KOVARIAN**

**RINA WIJAYA KUSUMA WARDHANI**

**NRP 05111440000021**

**Dosen Pembimbing I**

**Dr. Eng. Nanik Suciati, S.Kom., M.Kom.**

**Dosen Pembimbing II**

**Dini Adni Navastara, S.Kom., M.Sc.**

**DEPARTEMEN INFORMATIKA**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2018**

***[Halaman ini sengaja dikosongkan]***

**TUGAS AKHIR – KI141502**

**PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGGUNAKAN FITUR MATRIKS KOVARIAN**

**RINA WIJAYA KUSUMA WARDHANI**

**NRP 05111440000021**

**Dosen Pembimbing I**

**Dr. Eng. Nanik Suciati, S.Kom., M.Kom.**

**Dosen Pembimbing II**

**Dini Adni Navastara, S.Kom., M.Sc.**

**DEPARTEMEN INFORMATIKA**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2018**

****

**UNDERGRADUATE THESES – KI141502**

**HUMAN ACTIVITY RECOGNITION IN VIDEO USING FEATURE COVARIANCE MATRICES**

**RINA WIJAYA KUSUMA WARDHANI**

**NRP 05111440000021**

**Supervisor I**

**Dr. Eng. Nanik Suciati, S.Kom., M.Kom.**

**Supervisor II**

**Dini Adni Navastara, S.Kom., M.Sc.**

**DEPARTMENT OF INFORMATICS**

**Faculty of Information and Communication Technology**

**Sepuluh Nopember Institute of Technology**

**Surabaya 2018**

***[Halaman ini sengaja dikosongkan]***

# LEMBAR PENGESAHAN

**PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGGUNAKAN FITUR MATRIKS KOVARIAN**

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer

pada

Bidang Studi Komputasi Cerdas dan Visi

Program Studi S-1 Jurusan Teknik Informatika

Fakultas Teknologi Informasi dan Komunikasi

Institut Teknologi Sepuluh Nopember

Oleh

**RINA WIJAYA KUSUMA WARDHANI**

**NRP : 05111440000021**

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Dr. Eng. Nanik Suciati, S.Kom., M.Kom. .....................

NIP: 197906262005012002 (Pembimbing 1)

1. Dini Adni Navastara, S.Kom., M.Sc. .....................

NIP: 198701032014041001 (Pembimbing 2)

**SURABAYA**

**JANUARI, 2018**

***[Halaman ini sengaja dikosongkan]***

**PENGENALAN AKTIVITAS MANUSIA PADA VIDEO MENGGUNAKAN FITUR MATRIKS KOVARIAN**

|  |  |  |
| --- | --- | --- |
| **Nama Mahasiswa** | **:** | **RINA WIJAYA KUSUMA WARDHANI** |
| **NRP** | **:** | **05111440000021** |
| **Jurusan** | **:** | **Informatika FTIK-ITS** |
| **Dosen Pembimbing 1** | **:** | **Dr. Eng. Nanik Suciati, S.Kom., M.Kom.** |
| **Dosen Pembimbing 2** | **:** | **Dini Adni Navastara, S.Kom., M.Sc.** |

# Abstrak

*Di zaman ini, teknologi informasi sudah dan terus berkembang secara pesat. Dengan berkembangnya teknologi informasi, berkembang pula perangkat lunak yang memudahkan masyarakat dalam berbagai bidang salah satunya bidang akademik.*

*Pada suatu penelitian terkait tentang sistem informasi akademik generik telah dihasilkan purwarupa perangkat lunak dengan fungsionalitas yang umum agar dapat digunakan pada beberapa perguruan tinggi di Indonesia. Namun, sistem generik tersebut tidak dapat menangani adanya perubahan atau perbedaan proses bisnis pada suatu perguruan tinggi. Untuk itu perlu dibangun suatu perangkat lunak yang dapat menangani hal tersebut.*

*Pada tugas akhir ini, permasalahan tersebut akan ditangani dengan membuat perangkat lunak pengintegrasi yang modular dan membuat perangkat lunak generik pada penelitian sebelumnya menjadi modul-modul yang dapat diintegrasikan ke dalam perangkat lunak pengintegrasi tersebut. Perangkat lunak pengintegrasi modul berupa web beserta kerangka kerja yang berupa abstraksi/aturan yang perlu diimplementasi modul agar dapat diintegrasi. Perangkat lunak pengintegrasi akan dapat menambah, mengganti, atau menghapus modul-modul dari perangkat lunak generik tersebut tanpa melakukan perubahan pada modul lain dan perangkat lunak pengintegrasi itu sendiri sehingga jika ada perubahan proses bisnis modul, kompleksitas untuk melakukan perubahan tersebut dapat diminimalisasi.*

***Kata kunci: Perangkat Lunak Modular, Sistem Informasi Akademik***

**HUMAN ACTIVITY RECOGNITION IN VIDEO USING FEATURE COVARIANCE MATRICES**

|  |  |  |
| --- | --- | --- |
| **Student’s Name** | **:** | **RINA WIJAYA KUSUMA WARDHANI** |
| **Student’s ID** | **:** | **05111440000021** |
| **Department** | **:** | **Informatika FTIK-ITS** |
| **First Advisor** | **:** | **Dr. Eng. Nanik Suciati, S.Kom., M.Kom.** |
| **Second Advisor** | **:** | **Dini Adni Navastara, S.Kom., M.Sc.** |

# *Abstract*

*Until today, information technology has been and will still be growing fast and because of that, many softwares that help people in many areas such as education have been developed as well.*

*In some related researches about generic academic information system, some systems that share common business processes between colleges have been developed so that those same systems can be use in many colleges in Indonesia. However, those systems cannot handle any changes and business process variation in a college that shares no common business processes with those available systems. Furthermore, those systems are still a standalone system each so that they need to be integrated into a one whole academic information system. Therefore, a software that can handle such problems is needed to be developed.*

*In this undergraduate thesis, those problems will be handled by making a modular integrator software and making those available systems into modules that can be integrated into that aforementioned modular integrator software. The aforementioned software will be a web-based application and a framework for modules to implement so that those modules can be integrated. The integrator software must be able to add, change, and delete modules without doing any code changes in another modules or the integrator itself. Therefore, if any changes or business process variations occur, the complexity to handle such problems could be minimized.*

***Keywords: Modular Software, Academic Information System***

# KATA PENGANTAR



Alhamdulillahirabbil’alamin, segala puji bagi Allah SWT, yang atas izin dan karunianya penulis dapat menyelesaikan Tugas Akhir yang berjudul ***“*RANCANG BANGUN KERANGKA KERJA SISTEM INFORMASI AKADEMIK MODULAR BERBASIS WEB DENGAN POLA ARSITEKTUR *HIERARCHICAL MODEL-VIEW-CONTROLLER”*.**

Dalam pengerjaan tugas akhir ini, penulis mendapatkan banyak sekali ilmu baru dan memperdalam ilmu-ilmu yang sebelumnya telah diajarkan selama masa perkuliahan di Teknik Informatika ITS.

Terselesaikannya Tugas Akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada:

1. Allah SWT dan Nabi Muhammad SAW.
2. Ibu, Ayah, Kak Yan, Dek Farras, dan Sabrina tercinta yang senantiasa menyemangati dan menceriakan penulis selama pengerjaan tugas akhir.
3. Ibu Umi Laili Yuhana, S.Kom., M.Sc. selaku pembimbing I yang telah membantu, membimbing, dan memberikan ilmunya kepada penulis dalam menyelesaikan tugas akhir ini.
4. Bapak Rizky Januar Akbar, S.Kom., M.Eng. selaku pembimbing II yang juga telah membantu, membimbing , dan memberikan ilmunya kepada penulis dalam menyelesaikan tugas akhir ini.
5. Bapak Darlis Herumurti, S.Kom., M.Kom. selaku Kepala Jurusan Teknik Informatika ITS, Bapak Radityo Anggoro, S.Kom., M.Sc. selaku koordinator TA, dan segenap dosen Teknik Informatika yang telah memberikan ilmunya.
6. Dosen-dosen LBE, Ibu Dr. Ir. Siti Rochimah, MT. dan Ibu Ratih Nur Esti Anggraini, S.Kom, M.Sc..
7. Teman-teman seperjuangan tugas akhir di laboratorium RPL: Rahman, Tommy, Ruslan, Galih, Bustan, Teteh, dan Ujik serta teman-teman terbaik: Ibet dan kawan-kawan yang telah menemani dan menyemangati penulis selama pengerjaan tugas akhir.
8. Adik-adik admin dan penghuni laboratorium RPL yang telah menemani penulis selama pengerjaan tugas akhir.
9. Hendy yang secara sukarela telah mebuatkan beberapa logo untuk digunakan pada perangkat lunak.
10. Pihak lain yang telah membantu penulis menyelesaikan tugas akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Sehingga dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depannya.

Surabaya, Januari 2016

# DAFTAR ISI

[LEMBAR PENGESAHAN v](#_Toc440863892)

[Abstrak vii](#_Toc440863893)

[*Abstract* ix](#_Toc440863894)

[KATA PENGANTAR xi](#_Toc440863895)

[DAFTAR ISI xiii](#_Toc440863896)

[DAFTAR GAMBAR xix](#_Toc440863897)

[DAFTAR TABEL xxiii](#_Toc440863898)

[DAFTAR KODE SUMBER xxvii](#_Toc440863899)

[BAB I PENDAHULUAN 1](#_Toc440863900)

[1.1 Latar Belakang 1](#_Toc440863901)

[1.2 Rumusan Masalah 2](#_Toc440863902)

[1.3 Batasan Masalah 2](#_Toc440863903)

[1.4 Tujuan 3](#_Toc440863904)

[1.5 Manfaat 3](#_Toc440863905)

[1.6 Metodologi 4](#_Toc440863906)

[1.6.1 Penyusunan proposal tugas akhir 4](#_Toc440863907)

[1.6.2 Studi literatur 4](#_Toc440863908)

[1.6.3 Analisis dan desain perangkat lunak 4](#_Toc440863909)

[1.6.4 Implementasi perangkat lunak 5](#_Toc440863910)

[1.6.5 Pengujian dan evaluasi 5](#_Toc440863911)

[1.6.6 Penyusunan buku tugas akhir 5](#_Toc440863912)

[1.7 Sistematika Penulisan Laporan Tugas Akhir 6](#_Toc440863913)

[BAB II TINJAUAN PUSTAKA 9](#_Toc440863914)

[2.1 Sistem Informasi Akademik Generik 9](#_Toc440863916)

[2.2 Perangkat Lunak Modular 10](#_Toc440863917)

[2.3 *Hierarchical Model-View-Controller* (HMVC) 10](#_Toc440863918)

[2.4 Kerangka Kerja Spring 11](#_Toc440863919)

[2.4.1 Spring-MVC 11](#_Toc440863920)

[2.5 OSGi 13](#_Toc440863921)

[2.5.1 *Bundle* 13](#_Toc440863922)

[2.5.1.1 Status OSGi *Bundle* 15](#_Toc440863923)

[2.5.1.2 *Bundle* *fragment* 15](#_Toc440863924)

[2.5.2 OSGi *service registry* 16](#_Toc440863925)

[2.6 Apache Tomcat 17](#_Toc440863926)

[2.7 PostgreSQL 17](#_Toc440863927)

[2.8 Hibernate ORM 17](#_Toc440863928)

[2.9 Apache Maven 18](#_Toc440863929)

[2.9.1 Dependensi maven 18](#_Toc440863930)

[2.9.1.1 Dependensi transitif 18](#_Toc440863931)

[2.9.1.2 Cakupan dependensi 19](#_Toc440863932)

[2.9.2 Maven *plugin* 19](#_Toc440863933)

[2.10 Kerangka Kerja 20](#_Toc440863934)

[BAB III ANALISIS DAN PERANCANGAN PERANGKAT LUNAK 23](#_Toc440863935)

[3.1 Analisis 23](#_Toc440863938)

[3.1.1 Analisis permasalahan 23](#_Toc440863939)

[3.1.2 Deskripsi umum sistem 24](#_Toc440863940)

[3.1.3 Kasus penggunaan 26](#_Toc440863941)

[3.1.3.1 Aktor 26](#_Toc440863942)

[3.1.3.2 Menambah modul 27](#_Toc440863943)

[3.1.3.3 Menghapus modul 28](#_Toc440863944)

[3.1.3.4 Menambah akses menu 29](#_Toc440863945)

[3.1.3.5 Menghapus akses menu 29](#_Toc440863946)

[3.1.3.6 Menambah pengguna 34](#_Toc440863947)

[3.1.3.7 Menonaktifkan pengguna 35](#_Toc440863948)

[3.1.3.8 Menambah hak akses pengguna 36](#_Toc440863949)

[3.1.3.9 Menghapus hak akses pengguna 39](#_Toc440863950)

[3.1.3.10 Menambah peran 40](#_Toc440863951)

[3.1.3.11 Menghapus peran 40](#_Toc440863952)

[3.1 Perancangan Perangkat Lunak 43](#_Toc440863953)

[3.1.1 Arsitektur secara utuh 43](#_Toc440863954)

[3.1.2 Arsitektur komponen 44](#_Toc440863955)

[3.1.2.1 *Web layer* 45](#_Toc440863956)

[3.1.2.2 *Service layer* 46](#_Toc440863957)

[3.1.2.3 *Data layer* 49](#_Toc440863958)

[3.1.2.4 *Plugin layer* 49](#_Toc440863959)

[3.1.2.5 *Domain layer* 50](#_Toc440863960)

[3.1.3 Perancangan modularitas 53](#_Toc440863961)

[3.1.3.1 Lingkungan pengembangan modul 54](#_Toc440863962)

[3.1.3.2 Arsitektur modul 56](#_Toc440863963)

[3.1.3.3 Dependensi antar modul 56](#_Toc440863964)

[3.1.3.4 Pendaftaran *servlet* MVC modul 57](#_Toc440863965)

[3.1.4 Perancangan keamanan 59](#_Toc440863966)

[3.1.5 Perancangan akses data 60](#_Toc440863967)

[3.1.6 Perancangan *template* antarmuka pengguna 61](#_Toc440863968)

[3.1.7 Perancangan fitur administratif 62](#_Toc440863969)

[3.1.7.1 Hubungan antar kelas 62](#_Toc440863970)

[3.1.7.2 Perancangan basis data 75](#_Toc440863971)

[3.1.7.3 Perancangan antarmuka pengguna 79](#_Toc440863972)

[BAB IV IMPLEMENTASI 93](#_Toc440863973)

[4.1 Lingkungan Pengembangan 93](#_Toc440863975)

[4.1.1 Persiapan lingkungan dan implementasi kerangka kerja OSGi 94](#_Toc440863976)

[4.1.1.1 Apache Tomcat dalam lingkungan OSGi 94](#_Toc440863977)

[4.1.1.2 Pemasangan dependensi 94](#_Toc440863978)

[4.2 Implementasi Fungsionalitas Administratif 99](#_Toc440863979)

[4.2.1 *Web* *layer* 100](#_Toc440863980)

[4.2.1.1 *Package* *Controllers* 100](#_Toc440863981)

[4.2.1.2 Package JSON *Model* 107](#_Toc440863982)

[4.2.1.3 *Package* OSGi dan OSGi *Service Reference* 109](#_Toc440863983)

[4.2.1.4 Antarmuka pengguna 111](#_Toc440863984)

[4.2.2 *Service* *layer* 117](#_Toc440863985)

[4.2.2.1 *Package* *Service* 118](#_Toc440863986)

[4.2.2.2 Package Service Implementation 119](#_Toc440863987)

[4.2.2.3 *Package* *Module* 121](#_Toc440863988)

[4.2.2.4 OSGi *Service Reference* 123](#_Toc440863989)

[4.2.3 DAO *layer* 124](#_Toc440863990)

[4.2.3.1 *Package* DAO 124](#_Toc440863991)

[4.2.3.2 *Package* DAO *Implementation* 125](#_Toc440863992)

[4.2.3.3 OSGi *Service Reference* 126](#_Toc440863993)

[4.2.4 *Domain* *layer* 126](#_Toc440863994)

[4.3 Pembuatan Perangkat Lunak Web Standard dengan Spring MVC 128](#_Toc440863995)

[4.3.1 Pembuatan proyek baru 128](#_Toc440863996)

[4.3.2 Pom.xml 129](#_Toc440863997)

[4.3.3 *Servlet* dan *Web Application Context* 132](#_Toc440863998)

[4.3.3.1 Konfigurasi *Servlet* 132](#_Toc440863999)

[4.3.3.2 Web.xml 134](#_Toc440864000)

[4.3.4 Controllers 135](#_Toc440864001)

[4.3.5 View 136](#_Toc440864002)

[4.3.6 Pertukaran Data 136](#_Toc440864003)

[4.3.6.1 @RequestParam 137](#_Toc440864004)

[4.3.6.2 @RequestBody dan JSON 138](#_Toc440864005)

[4.3.6.3 @PathVariable 138](#_Toc440864006)

[4.3.6.4 @ResponseBody 139](#_Toc440864007)

[4.3.6.5 *ModelAndView* 139](#_Toc440864008)

[4.3.6.6 JSP *Expression* *Language* 139](#_Toc440864009)

[4.3.6.7 JSTL 140](#_Toc440864010)

[4.4 *Refactoring* Perangkat Lunak Web Menjadi OSGi *Bundle* 141](#_Toc440864011)

[4.4.1 Perubahan dependensi pada pom.xml 142](#_Toc440864012)

[4.4.2 Penambahan maven-bundle-plugin pada pom.xml 142](#_Toc440864013)

[4.4.3 *ContextClass* pada *DispactherServlet* 146](#_Toc440864014)

[4.5 Implementasi Modularitas 147](#_Toc440864015)

[4.5.1 Pembuatan modul 147](#_Toc440864016)

[4.5.1.1 Pembuatan Proyek Maven Baru 149](#_Toc440864017)

[4.5.1.2 Pengaturan POM pada maven project 149](#_Toc440864018)

[4.5.1.3 Pengisian OSGi *bundle* properties 154](#_Toc440864019)

[4.5.1.4 Implementasi Modul 155](#_Toc440864020)

[4.5.2 Pembacaan modul 167](#_Toc440864021)

[4.5.3 Pendaftaran *servlet* modul 169](#_Toc440864022)

[4.6 Implementasi Keamanan 172](#_Toc440864023)

[4.6.1 *Package* *Security* 173](#_Toc440864024)

[4.6.1.1 Kelas *RedirectAccessDeniedHandler* 173](#_Toc440864025)

[4.6.1.2 Kelas *SessionSavedRequestAwareAuthenticationSuccessHandler* 173](#_Toc440864026)

[4.6.1.3 Kelas *SIAUser* 174](#_Toc440864027)

[4.6.1.4 Kelas *SIAUserProvider* 174](#_Toc440864028)

[4.6.1.5 Sia-web-security.xml 175](#_Toc440864029)

[4.7 Implementasi Akses Data 176](#_Toc440864030)

[4.8 Implementasi *Template* Antarmuka Pengguna 178](#_Toc440864031)

[BAB V UJI COBA DAN EVALUASI 181](#_Toc440864032)

[5.1 Lingkungan Uji Coba 181](#_Toc440864034)

[5.2 Uji Coba 182](#_Toc440864035)

[5.2.1 Kasus uji fungsionalitas administratif 182](#_Toc440864036)

[5.2.1.1 Kasus uji menambah modul 182](#_Toc440864037)

[5.2.1.2 Kasus uji menghapus modul 187](#_Toc440864038)

[5.2.1.3 Kasus uji menambah akses menu 188](#_Toc440864039)

[5.2.1.4 Kasus uji menghapus akses menu 188](#_Toc440864040)

[5.2.1.5 Kasus uji menambah pengguna 189](#_Toc440864041)

[5.2.1.6 Kasus uji menonaktifkan pengguna 190](#_Toc440864042)

[5.2.1.7 Kasus uji menambah hak akses pengguna 191](#_Toc440864043)

[5.2.1.8 Kasus uji menghapus hak akses pengguna 192](#_Toc440864044)

[5.2.1.9 Kasus uji menambah peran 193](#_Toc440864045)

[5.2.1.10 Kasus uji menghapus peran 193](#_Toc440864046)

[5.2.2 Uji coba integrasi modul 194](#_Toc440864047)

[5.2.3 Uji coba hak akses 197](#_Toc440864048)

[5.3 Evaluasi 201](#_Toc440864049)

[5.3.1 Evaluasi fungsionalitas administratif 202](#_Toc440864050)

[5.3.2 Evaluasi Integrasi modul 202](#_Toc440864051)

[5.3.3 Evaluasi hak akses modul 208](#_Toc440864052)

[BAB VI KESIMPULAN DAN SARAN 215](#_Toc440864053)

[6.1 Kesimpulan 215](#_Toc440864055)

[6.2 Saran 216](#_Toc440864056)

[DAFTAR PUSTAKA 217](#_Toc440864057)

[BIODATA PENULIS 221](#_Toc440864058)

***[Halaman ini sengaja dikosongkan]***

# DAFTAR GAMBAR

[Gambar 2.1 Arsitektur Modul Penilaian [6] 9](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082048)

[Gambar 2.2 Hierarchical Model-View Controller [8] 11](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082049)

[Gambar 2.3 Kerangka kerja Spring [9] 12](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082050)

[Gambar 2.4 Spring MVC [21] 12](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082051)

[Gambar 2.5 OSGi [10] 14](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082052)

[Gambar 2.6 Siklus hidup sebuah OSGi bundle [6] 15](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082053)

[Gambar 2.7 *Bundle* *Fragment* 16](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082054)

[Gambar 2.8 Hibernate [9] 17](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082055)

[Gambar 3.1 Arsitektur sistem 25](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082056)

[Gambar 3.2 Diagram kasus penggunaan 26](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082057)

[Gambar 3.3 Diagram aktivitas menambah modul 28](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082058)

[Gambar 3.4 Diagram aktivitas menghapus modul 30](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082059)

[Gambar 3.5 Diagram aktivitas menambah akses menu setelah Menambah modul 31](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082060)

[Gambar 3.6 Diagram aktivitas menambah akses menu 32](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082061)

[Gambar 3.7 Diagram aktivitas menghapus akses menu 33](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082062)

[Gambar 3.8 Diagram aktivitas menambah pengguna 35](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082063)

[Gambar 3.9 Diagram aktivitas menonaktifkan pengguna 37](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082064)

[Gambar 3.10 Diagram aktivitas menambah hak akses pengguna 38](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082065)

[Gambar 3.11 Diagram aktivitas menghapus hak akses pengguna 40](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082066)

[Gambar 3.12 Diagram aktivitas menambah peran 41](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082067)

[Gambar 3.13 Diagram aktivitas menghapus peran 42](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082068)

[Gambar 3.14 Diagram arsitektur 45](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082069)

[Gambar 3.15 Diagram kelas *web layer* 47](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082070)

[Gambar 3.16 Diagram kelas *service layer* 48](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082071)

[Gambar 3.17 Dependensi *web layer* 50](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082072)

[Gambar 3.18 Pengambilan *OSGi service* yang didaftarkan *service layer* oleh *web layer* 51](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082073)

[Gambar 3.19 Pengambilan *OSGi service* yang didaftarkan *data layer* oleh *web layer* dan *service layer* 51](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082074)

[Gambar 3.20 Diagram kelas *data layer* 52](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082075)

[Gambar 3.21 Diagram kelas *plugin layer* 52](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082076)

[Gambar 3.22 Diagram kelas *domain layer* 53](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082077)

[Gambar 3.23 *Workflow* pengembangan modul 54](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082078)

[Gambar 3.24 Diagram komponen modul dan dependensinya dengan komponen lain 55](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082079)

[Gambar 3.25 Diagram komponen modul 56](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082080)

[Gambar 3.26 Dependensi antar modul 57](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082081)

[Gambar 3.27 Diagram *sequence* pendaftaran modul 58](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082082)

[Gambar 3.28 Diagram kelas keamanan perangkat lunak 59](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082083)

[Gambar 3.29 Diagram alur otorisasi perangkat lunak 60](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082084)

[Gambar 3.30 Diagram kelas akses data 60](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082085)

[Gambar 3.31 Halaman *template* fungsi administratif 61](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082086)

[Gambar 3.32 Halaman *template* modul 61](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082087)

[Gambar 3.33 *Flow Chart* penambahan modul ke basis data 66](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082088)

[Gambar 3.34 sequence diagram Menambah modul 67](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082089)

[Gambar 3.35 sequence diagram Menghapus modul 68](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082090)

[Gambar 3.36 sequence diagram Menambah akses menu 69](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082091)

[Gambar 3.37 sequence diagram Menghapus akses menu 69](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082092)

[Gambar 3.38 sequence diagram Menambah pengguna 70](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082093)

[Gambar 3.39 sequence diagram Menambah pengguna (dengan *CSV*) 71](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082094)

[Gambar 3.40 sequence diagram Menonaktifkan pengguna 72](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082095)

[Gambar 3.41 sequence diagram Menonaktifkan pengguna (*multiple*) 72](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082096)

[Gambar 3.42 sequence diagram Menambah hak akses pengguna 73](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082097)

[Gambar 3.43 sequence diagram Menghapus hak akses pengguna 74](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082098)

[Gambar 3.44 sequence diagram Menambah peran 74](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082099)

[Gambar 3.45 sequence diagram Menghapus peran 75](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082100)

[Gambar 3.46 Model Data Konseptual 77](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082101)

[Gambar 3.47 Model Data Fisik 78](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082102)

[Gambar 3.48 Halaman penambahan modul 84](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082103)

[Gambar 3.49 Halaman penambahan akses menu 84](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082104)

[Gambar 3.50 Halaman daftar modul 85](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082105)

[Gambar 3.51 Halaman daftar pengguna 86](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082106)

[Gambar 3.52 Halaman penambahan pengguna manual 87](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082107)

[Gambar 3.53 Halaman penambahan pengguna dengan unggah CSV 88](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082108)

[Gambar 3.54 Halaman pengubahan informasi pengguna 89](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082109)

[Gambar 3.55 Halaman pengelolaan hak akses pengguna 90](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082110)

[Gambar 3.56 Halaman daftar peran 91](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082111)

[Gambar 3.57 Halaman penambahan peran 92](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082112)

[Gambar 4.1 Halaman penambahan modul 113](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082113)

[Gambar 4.2 Halaman penambahan akses menu 113](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082114)

[Gambar 4.3 Halaman daftar modul 114](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082115)

[Gambar 4.4 Halaman daftar pengguna 114](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082116)

[Gambar 4.5 Halaman pengunggahan berkas CSV 115](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082117)

[Gambar 4.6 Halaman penambahan pengguna 115](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082118)

[Gambar 4.7 Halaman penyuntingan informasi pengguna 116](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082119)

[Gambar 4.8 Halaman pengelolaan hak akses pengguna 116](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082120)

[Gambar 4.9 Halaman penambahan peran 116](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082121)

[Gambar 4.10 Halaman daftar peran 117](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082122)

[Gambar 4.11 Perbedaan Import-Package dan Export-Package 144](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082123)

[Gambar 4.12 Perbedaan Import-Library, Import-Bundle, dan Export-Package 145](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082124)

[Gambar 4.13 Halaman *template* fungsionalitas administratif 179](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082125)

[Gambar 4.14 Halaman *template* modul 180](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082126)

[Gambar 5.1 *Error* jika properti modul tidak lengkap 186](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082127)

[Gambar 5.2 *Error* jika URL *mapping* sudah digunakan 187](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082128)

[Gambar 6.1 Modul-modul SIAKAD yang sudah terintegrasi dan yang dapat dikembangkan lebih lanjut 216](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc441082129)

***[Halaman ini sengaja dikosongkan]***

# DAFTAR TABEL

[Tabel 3.1 Spesifikasi kasus penggunaan menambah modul 27](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864141)

[Tabel 3.2 Spesifikasi kasus penggunaan menghapus modul 29](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864142)

[Tabel 3.3 Spesifikasi kasus penggunaan menambah akses menu 30](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864143)

[Tabel 3.4 Spesifikasi kasus penggunaan menghapus akses menu 32](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864144)

[Tabel 3.5 Spesifikasi kasus penggunaan menambah pengguna 34](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864145)

[Tabel 3.6 Spesifikasi kasus penggunaan menonaktifkan pengguna 36](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864146)

[Tabel 3.7 Spesifikasi kasus penggunaan menambah hak akses pengguna 37](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864147)

[Tabel 3.8 Spesifikasi kasus penggunaan menghapus hak akses pengguna 39](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864148)

[Tabel 3.9 Spesifikasi kasus penggunaan menambah peran 41](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864149)

[Tabel 3.10 Spesifikasi kasus penggunaan menghapus peran 42](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864150)

[Tabel 3.11 Spesifikasi atribut antarmuka halaman *template* admin dan modul 83](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864151)

[Tabel 3.12 Spesifikasi atribut antarmuka penambahan modul 84](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864152)

[Tabel 3.13 Spesifikasi atribut antarmuka penambahan akses menu 85](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864153)

[Tabel 3.14 Spesifikasi atribut antarmuka daftar modul 86](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864154)

[Tabel 3.15 Spesifikasi atribut antarmuka daftar pengguna 87](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864155)

[Tabel 3.16 Spesifikasi atribut antarmuka penambahan pengguna manual 88](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864156)

[Tabel 3.17 Spesifikasi atribut antarmuka penambahan pengguna dengan unggah CSV 89](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864157)

[Tabel 3.18 Spesifikasi atribut antarmuka pengubahan informasi pengguna 89](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864158)

[Tabel 3.19 Spesifikasi atribut antarmuka pengelolaan hak akses pengguna 90](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864159)

[Tabel 3.20 Spesifikasi atribut antarmuka daftar peran 91](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864160)

[Tabel 3.21 Spesifikasi atribut antarmuka penambahan peran 92](file:///C:\Users\sagungw93\Dropbox\Tugas%20Akhir\LBE%20SIA\Framework\Buku\Buku-Salahudin-Agung-Wijaya.docx#_Toc440864161)

[Tabel 5.1 Tabel lingkungan uji coba klien 181](#_Toc440864162)

[Tabel 5.2 Tabel lingkungan uji coba server 181](#_Toc440864163)

[Tabel 5.3 Tabel kasus uji menambah modul 183](#_Toc440864164)

[Tabel 5.4 Tabel kasus uji menghapus modul 187](#_Toc440864165)

[Tabel 5.5 Tabel kasus uji menambah akses menu 188](#_Toc440864166)

[Tabel 5.6 Tabel kasus uji menghapus akses menu 189](#_Toc440864167)

[Tabel 5.7 Tabel kasus uji menambah pengguna 189](#_Toc440864168)

[Tabel 5.8 Tabel kasus uji menonaktifkan pengguna 191](#_Toc440864169)

[Tabel 5.9 Tabel kasus uji menambah hak akses pengguna. 191](#_Toc440864170)

[Tabel 5.10 Tabel kasus uji menghapus hak akses pengguna. 192](#_Toc440864171)

[Tabel 5.11 Tabel kasus uji menambah peran 193](#_Toc440864172)

[Tabel 5.12 Tabel kasus uji menghapus peran 194](#_Toc440864173)

[Tabel 5.13 Tabel daftar modul yang diuji coba 194](#_Toc440864174)

[Tabel 5.14 Tabel daftar menu pada modul kurikulum [3] 195](#_Toc440864175)

[Tabel 5.15 Tabel daftar menu pada modul pembelajaran [5] 195](#_Toc440864176)

[Tabel 5.16 Tabel daftar menu pada modul penilaian [6] 196](#_Toc440864177)

[Tabel 5.17 Tabel daftar menu pada modul ekuivalensi [4] 197](#_Toc440864178)

[Tabel 5.18 Tabel daftar peran 198](#_Toc440864179)

[Tabel 5.19 Tabel daftar akses menu dari modul kurikulum [3] 198](#_Toc440864180)

[Tabel 5.20 Tabel daftar akses menu dari modul pembelajaran [5] 199](#_Toc440864181)

[Tabel 5.21 Tabel daftar akses menu dari modul penilaian [6] 200](#_Toc440864182)

[Tabel 5.22 Tabel daftar akses menu dari modul ekuivalensi [4] 201](#_Toc440864183)

[Tabel 5.23 Tabel evaluasi kasus uji fungsionalitas administratif 202](#_Toc440864184)

[Tabel 5.24 Tabel evaluasi integrasi modul kurikulum [3] 203](#_Toc440864185)

[Tabel 5.25 Tabel evaluasi integrasi modul pembelajaran [5] 204](#_Toc440864186)

[Tabel 5.26 Tabel evaluasi integrasi modul penilaian [6] 205](#_Toc440864187)

[Tabel 5.27 Tabel evaluasi integrasi modul ekuivalensi [4] 206](#_Toc440864188)

[Tabel 5.28 Tabel evaluasi hak akses untuk modul kurikulum [3] 208](#_Toc440864189)

[Tabel 5.29 Tabel evaluasi hak akses untuk modul pembelajaran [5] 210](#_Toc440864190)

[Tabel 5.30 Tabel evaluasi hak akses untuk modul penilaian [6] 212](#_Toc440864191)

[Tabel 5.31 Tabel evaluasi hak akses untuk modul ekuivalensi [4] 213](#_Toc440864192)

***[Halaman ini sengaja dikosongkan]***

# DAFTAR KODE SUMBER

[Kode Sumber 4.1 Berkas XML hibernate.plan 98](#_Toc440864193)

[Kode Sumber 4.2 Berkas XML jackson.plan 98](#_Toc440864194)

[Kode Sumber 4.3 Berkas XML sia.plan 99](#_Toc440864195)

[Kode Sumber 4.4 Fungsi pengunggahan berkas modul 101](#_Toc440864196)

[Kode Sumber 4.5 Fungsi penambahan akses menu modul 102](#_Toc440864197)

[Kode Sumber 4.6 Fungsi penghapusan modul 102](#_Toc440864198)

[Kode Sumber 4.7 Fungsi penambahan pengguna 104](#_Toc440864199)

[Kode Sumber 4.8 Fungsi penambahan hak akses pengguna 105](#_Toc440864200)

[Kode Sumber 4.9 Fungsi penghapusan hak akses pengguna 106](#_Toc440864201)

[Kode Sumber 4.10 Fungsi penambahan peran 107](#_Toc440864202)

[Kode Sumber 4.11 Fungsi penghapusan peran 107](#_Toc440864203)

[Kode Sumber 4.12 Kelas *Role* 108](#_Toc440864204)

[Kode Sumber 4.13 Kelas *RoleMenu* 108](#_Toc440864205)

[Kode Sumber 4.14 Kelas SatMan 109](#_Toc440864206)

[Kode Sumber 4.15 Kelas *ServiceLocator<T>* 110](#_Toc440864207)

[Kode Sumber 4.16 berkas XML Blueprint 111](#_Toc440864208)

[Kode Sumber 4.17 Kelas *interface ModulService* 118](#_Toc440864209)

[Kode Sumber 4.18 Kelas *ModulServiceImpl* 121](#_Toc440864210)

[Kode Sumber 4.19 Kelas *XmlModuleReaderBean* 123](#_Toc440864211)

[Kode Sumber 4.20 Berkas XML Blueprint untuk mengambil *service* dari komponen DAO *layer* 124](#_Toc440864212)

[Kode Sumber 4.21 Kelas *interface GenericDAO* 125](#_Toc440864213)

[Kode Sumber 4.22 Berkas XML Blueprint pada DAO *layer* 126](#_Toc440864214)

[Kode Sumber 4.23 Kelas entitas Modul 128](#_Toc440864215)

[Kode Sumber 4.24 *Maven compiler plugin* 130](#_Toc440864216)

[Kode Sumber 4.25 Berkas pom.xml 131](#_Toc440864217)

[Kode Sumber 4.26 Contoh konfigurasi *servlet* 133](#_Toc440864218)

[Kode Sumber 4.27 Contoh berkas web.xml 135](#_Toc440864219)

[Kode Sumber 4.28 Contoh kelas *controller* 136](#_Toc440864220)

[Kode Sumber 4.29 Berkas JSP 136](#_Toc440864221)

[Kode Sumber 4.30 Contoh penggunaan @RequestParam 137](#_Toc440864222)

[Kode Sumber 4.31 Contoh penggunaan @RequestParam dengan *form* 137](#_Toc440864223)

[Kode Sumber 4.32 Dependensi Jackson pada pom.xml 138](#_Toc440864224)

[Kode Sumber 4.33 Contoh penggunaan @RequestBody 138](#_Toc440864225)

[Kode Sumber 4.34 Contoh penggunaan @PathVariable 138](#_Toc440864226)

[Kode Sumber 4.35 Contoh penggunaan @ResponseBody 139](#_Toc440864227)

[Kode Sumber 4.36 Contoh penggunaan fungsi *addObject*() 139](#_Toc440864228)

[Kode Sumber 4.37 Contoh penggunaan fungsi *addObject*() untuk diakses pada JSP 140](#_Toc440864229)

[Kode Sumber 4.38 Contoh pengambilan data dengan *expression* *language* 140](#_Toc440864230)

[Kode Sumber 4.39 Contoh penggunaan fungsi *addObject*() untuk ditampilkan dengan JSTL 141](#_Toc440864231)

[Kode Sumber 4.40 Contoh penggunaan JSTL 141](#_Toc440864232)

[Kode Sumber 4.41 maven-bundle-plugin pada berkas pom.xml 143](#_Toc440864233)

[Kode Sumber 4.42 maven-war-plugin pada pom.xml 146](#_Toc440864234)

[Kode Sumber 4.43 Berkas web.xml yang mendukung OSGi 147](#_Toc440864235)

[Kode Sumber 4.44 Dependensi minimal modul pada berkas pom.xml 152](#_Toc440864236)

[Kode Sumber 4.45 Maven *plugin* modul pada berkas pom.xml 154](#_Toc440864237)

[Kode Sumber 4.46 Berkas osgi.properties pada modul 154](#_Toc440864238)

[Kode Sumber 4.47 Penggunaan @Secured pada @RequestMapping 156](#_Toc440864239)

[Kode Sumber 4.48 Penggunaan @Secured pada @Controller 156](#_Toc440864240)

[Kode Sumber 4.49 Pengambilan obyek *session* pengguna yang terautentikasi 157](#_Toc440864241)

[Kode Sumber 4.50 Penambahan dependensi sia-domain pada berkas pom.xml 157](#_Toc440864242)

[Kode Sumber 4.51 Contoh kelas DAO 158](#_Toc440864243)

[Kode Sumber 4.52 Konigurasi *servlet* modul 160](#_Toc440864244)

[Kode Sumber 4.53 Daftar *stylesheet* yang telah disediakan template 162](#_Toc440864245)

[Kode Sumber 4.54 Daftar *script* yang telah disediakan template 163](#_Toc440864246)

[Kode Sumber 4.55 Contoh struktur berkas JSP 164](#_Toc440864247)

[Kode Sumber 4.56 Penambahan menu yang sedang diakses 164](#_Toc440864248)

[Kode Sumber 4.57 Contoh struktur berkas properti modul 166](#_Toc440864249)

[Kode Sumber 4.58 Kode sumber fungsi *readModule*() 169](#_Toc440864250)

[Kode Sumber 4.59 Kode sumber kelas WebAppInitializer 170](#_Toc440864251)

[Kode Sumber 4.60 Kode sumber kelas SIAMainWebApplication sebagai kelas pendaftar servlet 172](#_Toc440864252)

[Kode Sumber 4.61 Kelas *SIAUserProvider* 174](#_Toc440864253)

[Kode Sumber 4.62 Berkas sia-web-security.xml 176](#_Toc440864254)

[Kode Sumber 4.63 Berkas konfigurasi akses data fungsionalitas admin 178](#_Toc440864255)

[Kode Sumber 4.64 Berkas konfigurasi *template* antarmuka pengguna 179](#_Toc440864256)

***[Halaman ini sengaja dikosongkan]***

# BAB I PENDAHULUAN

## Latar Belakang

Saat ini, pengenalan aktivitas manusia pada data video merupakan persoalan yang cukup menantang di bidang visi komputer. Hal ini disebabkan oleh kompleksitas pada data video berupa halangan, kekacauan, interaksi objek ganda, perubahan cahaya, dan lain-lain. Selain itu, adanya persoalan pada proses pengambilan video seperti penyimpangan, pergerakan, dan sudut pandang kamera, serta kompleksitas pada aktivitas manusia itu sendiri sebagai objek non-rigid yang memiliki beragam kelas aktivitas [1]. Pengenalan aktivitas manusia pada data video telah diterapkan di beberapa bidang serta digunakan sebagai sistem keamanan atau sarana hiburan. Untuk itulah penulis ingin menerapkan pengenalan aktivitas manusia pada data video yang berasal dari Closed-Circuit Television (CCTV) yang berada di Departemen Informatika ITS Surabaya.

Ada dua komponen dasar pada setiap algoritma pengenalan aktivitas yang memengaruhi hasil akurasi dan efisiensi yaitu representasi model aktivitas dan metode klasifikasi yang digunakan. Model yang digunakan untuk merepresentasikan aktivitas dapat dibagi menjadi lima kategori yaitu shape models, motion models, geometric human body models, interest-point models, dan dynamic models. Metode klasifikasi standar yang paling sering digunakan antara lain nearest-neighbor (NN) classifier, support vector machine (SVM), dan boosting [1].

Pengenalan aktivitas dengan shape-based models menggunakan estimasi akurasi dari siluet objek yang bergerak di setiap frame video sehingga membentuk silhouette tunnel. Model ini bersifat invarian terhadap pencahayaan, warna, dan tekstur dari objek yang bergerak, namun tidak semua data video memiliki model ini [1-10]. Pengenalan aktivitas dengan geometric human body models menggunakan bagian tubuh manusia sebagai modelnya sehingga akan sulit didapat dari data video dengan objek yang kecil dan background yang luas [11-14]. Pengenalan aktivitas dengan interestpoint models dilakukan dengan mengekstrak intereset-point dari pertimbangan informasi struktural dan mendeteksi cuboids di daerah yang memiliki kemungkinan besar mengalami pergerakan [15-20]. Dynamic models merupakan model yang paling baru digunakan dalam pengenalan aktivitas [21], [22]. Model ini mendeskripsikan postur statis dari suatu aksi sebagai sebuah state dan mendeskripsikan dynamics (variasi temporal) dari suatu aksi menggunakan transisi model state-space. Model yang terakhir yaitu motion models. Model ini mengekstrak berbagai karakteristik pergerakan dan deformasi objek. Model ini menggunakan fitur kinematic yang berasal dari optical flow untuk merepresentasikan aktivitas [23-30]. Motion-model dikatakan sebagai atribut yang paling diskriminatif dalam merepresentasikan aksi [1].

Komponen dasar yang akan digunakan dalam pembuatan tugas akhir ini adalah penggunaan fitur matriks kovarian sebagai representasi model aktivitas. Fitur matriks kovarian merupakan representasi ringkas dari koleksi fitur lokal sehingga memiliki dimensi yang lebih sedikit. Koleksi fitur lokal yang digunakan berasal dari optical flow data video yang termasuk ke dalam motion-model. Komponen dasar lainnya adalah metode klasifikasi aktivitas yang digunakan yaitu nearest-neighbor (NN) classifier yang diketahui tahan terhadap keanekaragaman bentuk aktivitas, perubahan sudut pandang kamera, dan objek dengan resolusi rendah. Secara konseptual, framework ini cukup sederhana, membutuhkan ruang penyimpanan yang kecil, serta memiliki perhitungan yang memungkinkan untuk implementasi secara real time [1].

## Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini dapat dipaparkan sebagai berikut:

Bagaimana cara mendapatkan koleksi fitur lokal berbasis *optical flow* dari data video?

Bagaimana cara mendapatkan representasi model aktivitas manusia menggunakan fitur matriks kovarian?

Bagaimana cara klasifikasi aktivitas manusia menggunakan *nearest-neighbor* (NN) *classifier*?

Bagaimana mengetahui kinerja dari aplikasi pengenalan aktivitas manusia pada data video?

Bagaimana cara mengimplementasi program agar dapat berjalan secara *real-time*?

## Batasan Masalah

Permasalan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, diantaranya sebagai berikut:

Data yang digunakan pada tugas akhir ini berupa data video yang diperoleh dari CCTV di Departemen Informatika ITS Surabaya.

Karena kompleksitas dan keberagaman aktivitas yang ada, pengenalan aktivitas hanya berupa aktivitas yang berasal dari objek tunggal.

Meskipun dapat digunakan untuk mengenali aktivitas berbagai macam jenis objek, tugas akhir ini hanya berfokus pada aktivitas manusia.

Aktivitas yang dapat dikenali berjumlah tiga yaitu aktivitas berlari, berjalan, dan duduk.

## Tujuan

Tujuan dari tugas akhir ini adalah menciptakan aplikasi pengenalan aktivitas manusia pada video yang terekam oleh CCTV.

## Manfaat

Manfaat tugas akhir ini adalah sebagai langkah awal dalam membangun sistem pengawasan dan keamanan di Departemen Informatika ITS Surabaya melalui data rekaman video CCTV.

## Metodologi

### Penyusunan proposal tugas akhir

Usulan tugas akhir ini berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah, tujuan, dan manfaat dari pembuatan tugas akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Kemudian pada sub bab metodologi berisi penjelasan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku tugas akhir. Terdapat pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan tugas akhir.

### Studi literatur

Tahap ini merupakan tahap pengumpulan informasi dan pembelajaran yang akan digunakan pada tugas akhir ini. Studi literatur meliputi diskusi dan pemahaman mengenai topik tugas akhir ini, diantaranya mengenai:

1. Modularitas perangkat lunak
2. Pola Arsitektur *Hierarchical Model-View-Controller*
3. PostgreSQL sebagai basis data

### Analisis dan desain perangkat lunak

Adapun pembagian tahap analisa kebutuhan dan perancangan dari kerangka kerja sistem informasi akademik sebagai berikut:

* + 1. Mempelajari kebutuhan kerangka kerja secara umum

Pada tahap ini yaitu mempelajari kebutuhan aplikasi kerangka kerja web secara umum melalui aplikasi serupa yang telah dikembangkan.

* + 1. Mempelajari kebutuhan umum sistem informasi akademik secara garis besar

Tahap ini dilakukan agar dapat membuat rancang bangun kerangka kerja yang sesuai kebutuhan pada tahap selanjutnya.

* + 1. Merancang kerangka kerja sistem informasi akademik

Pada tahap ini dilakukan perancangan struktur kelas dalam bentuk *class diagram* dan basis data dalam bentuk *physical data model* dan *conceptual data model*.

### Implementasi perangkat lunak

Kerangka kerja dibuat dengan menggunakan bahasa pemrograman Java dengan kakas bantu spring MVC dan kemungkinan penggunaan JavaScript, JSON, dan jQuery serta dengan sistem manajemen basis data PostgreSQL.

### Pengujian dan evaluasi

Pengujian sistem ini dilakukan dengan metode *blackbox*. Pengujian *blackbox* merupakan pengujian untuk mengetahui apakah semua fungsi perangkat lunak telah berjalan semestinya sesuai dengan kebutuhan fungsional yang telah didefinisikan [2].

### Penyusunan buku tugas akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Pendahuluan
   1. Latar Belakang
   2. Rumusan Masalah
   3. Batasan Tugas Akhir
   4. Tujuan
   5. Metodologi
   6. Sistematika Penulisan
2. Tinjauan Pustaka
3. Desain dan Implementasi
4. Pengujian dan Evaluasi
5. Kesimpulan dan Saran
6. Daftar Pustaka

## Sistematika Penulisan Laporan Tugas Akhir

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini:

**Bab I Pendahuluan**

Bab yang berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan tugas akhir. Selain itu permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

**Bab II Tinjauan Pustaka**

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.

**Bab III Perancangan Perangkat Lunak**

Bab ini berisi tentang desain sistem yang disajikan dalam bentuk *pseudocode*.

**Bab IV Implementasi**

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa *code* yang digunakan untuk proses implementasi.

**Bab V Uji Coba dan Evaluasi**

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.

**Bab VI Kesimpulan dan Saran**

Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

***[Halaman ini sengaja dikosongkan]***

# BAB II TINJAUAN PUSTAKA

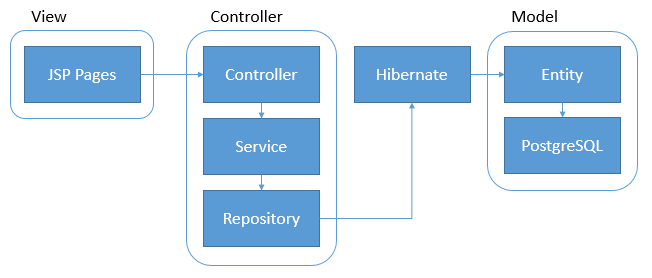


## Sistem Informasi Akademik Generik

Sistem informasi akademik generik merupakan sebuah sistem informasi dengan proses bisnis yang bersifat umum dan dapat digunakan untuk variasi proses bisnis akademik di perguruan tinggi lainnya [3]. Sistem informasi akademik ini dibagi menjadi beberapa modul yaitu:

* + - 1. Modul Ekivalensi [4]
      2. Modul Kurikulum [3]
      3. Modul Pembelajaran [5]
      4. Modul Penilain [6]

Masing-masing modul tersebut berupa satu perangkat lunak yang masih berdiri sendiri. Modul-modul ini dikembangkan dengan pola arsitektur Model-View-Controller dengan kerangka kerja Spring MVC. Sebagai contoh arsitektur modul, Gambar 2.1 menunjukkan arsitektur yang digunakan pada Modul Penilaian.



Gambar 2.1 Arsitektur Modul Penilaian [6]

## Perangkat Lunak Modular

Perangkat lunak modular merupakan perangkat lunak yang terbagi menjadi beberapa modul dimana modul-modul tersebut terfokus pada satu fungsionalitas (*high cohession*) dan memiliki sedikit ketergantungan dengan modul lain (*loosely-coupled*). Yang dimaksud dengan sedikitnya ketergantungan suatu modul dengan modul lain artinya suatu modul dapat berhubungan dengan modul lain melalui abstraksi kelas yang stabil dan tanpa mengetahui implementasi kelas sehingga perubahan pada modul lain tidak banyak mempengaruhi modul yang berhubungan [7].

Dengan perancangan yang modular, suatu perangkat lunak memiliki kelebihan [7] seperti:

Modul dapat lebih mudah diganti selama modul memiliki abstraksi yang sama

Perangkat lunak dapat dipelajari per modul sehingga lebih mudah dipahami

Modul dapat dikembangkan, diuji, dan di-*deploy* secara paralel

Modul memiliki *reusability* yang tinggi karena adanya abstraksi modul tersebut.

## *Hierarchical Model-View-Controller* (HMVC)

Pengembangan dari arsitektur perangkat lunak *Model-View-Controller* (selanjutnya disebut MVC), *Hierarchical Model-View-Controller* (HMVC), adalah sebuah arsitektur perangkat lunak MVC yang diimplementasikan secara hirarkis.

Seperti yang ditunjukkan pada Gambar 2.2, arsitektur HMVC memiliki beberapa *triad* MVC yang dapat berdiri sendiri. Triad MVC dapat berupa request internal ataupun external. *Request* ke dalam suatu *triad* MVC dilakukan dengan memanggil *controller* pada *triad* MVC tersebut melalui *controller* dari *triad* MVC yang membutuhkan [8].



Gambar 2.2 Hierarchical Model-View Controller [8]

## Kerangka Kerja Spring

Spring adalah salah satu kerangka kerja aplikasi untuk aplikasi berbasis Java atau sering disebut dengan *Java Enterprise Edition* (selanjutnya disebut JEE). Spring menangani infrastruktur sistem, sehingga kita dapat berfokus pada lapisan aplikasi. Kerangka kerja Spring terdiri dari fitur-fitur yang diatur hingga 20 modul. Modul-modul ini dikelompokkan menjadi *Core Container, Data Access/Integration, Web, AOP* (*Aspect Oriented Programming*), *Instrumentation* dan *Test* [9].

Dalam pengerjaan tugas akhir ini, penggunaan modul kerangka kerja Spring yang ditunjukkan pada Gambar 2.3 adalah JDBC dan *Transaction* pada akses data/integrase, Web-Servlet, Beans dan Context pada *Core Container*.

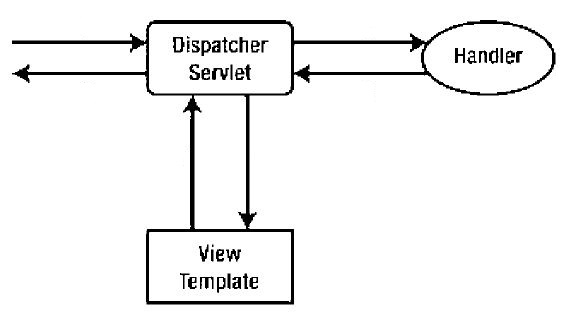
### Spring-MVC

*Model View Controller* atau MVC merupakan pola perancangan yang membagi lapisan presentasi pada suatu perangkat lunak menjadi beberapa komponen, yaitu *model*, *view*, dan *controller*, yang memiliki fokus fungsionalitasnya masing-masing.

* *View* bertugas untuk menampilkan *Model* kepada pengguna. *View* biasanya berupa halaman web seperti JSP dan JSF pada Java.
* *Model* merupakan informasi yang dibutuhkan *View* untuk dapat ditampilkan dan ditukarkan antara *View* dan *Controller*.
* *Controller*, secara garis besar, bertugas mengatur alur pertukaran informasi dan perintah yang diberikan pengguna.



Gambar 2.3 Kerangka kerja Spring [9]



Gambar 2.4 Spring MVC [21]

Pada Spring MVC, terdapat beberapa komponen tambahan selain komponen dasar. Komponen tersebut antara lain:

* *Front Controller* berupa kelas org.springframework.web.servlet.DispatcherServlet.Java yang bertugas meneruskan *HTTP Request* ke *Controller* yang tepat pada *Handler*.
* *Handler* merupakan kumpulan-kumpulan komponen yang berfungsi menentukan bagaimana http request akan ditangani. Komponen-komponen yang terdapat pada handler yaitu *Controller* dan *View Resolver*.
* *View Template* merupakan komponen yang berisi macam-macam kelas yang bertugas membentuk *View* yang sesuai dengan tipe *View* masing-masing kelas tersebut yang telah ditentukan oleh *View Resolver*.

## OSGi

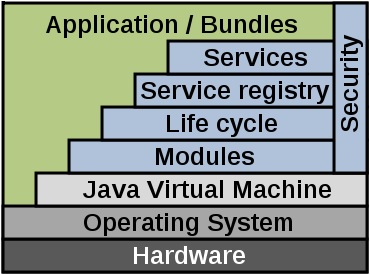
OSGi adalah sistem modular dan *service platform* pada Java. Dengan OSGi, pengembang bisa membagi perangkat lunak nya ke dalam beberapa modul yang saling *loosely coupled*. Modul-modul tersebut dapat dikembangkan, diubah, dicoba, dan di-*deploy* secara terpisah dengan sedikit atau tanpa mempengaruhi modul lain [7].

Seperti yang ditunjukka pada Gambar 2.5, OSGi berjalan di atas *Java Virtual Machine* (JVM) dengan beberapa komponen seperti *bundle*, *bundle life cycle*, dan *service registry*. Berikut Penjelasan komponen tersebut.

### *Bundle*

*Bundle* adalah berkas hasil *build* dari suatu aplikasi Java. *Bundle* memiliki ekstensi berkas “.bundle” ataupun “.jar”. Beberapa container OSGi juga mendukung ekstensi “.war” sebagai *web-bundle*. Perbedaan antara *bundle* dan berkas jar ataupun war biasa adalah pada berkas manifest nya dimana bundle memiliki beberapa properti khusus [10] yaitu:

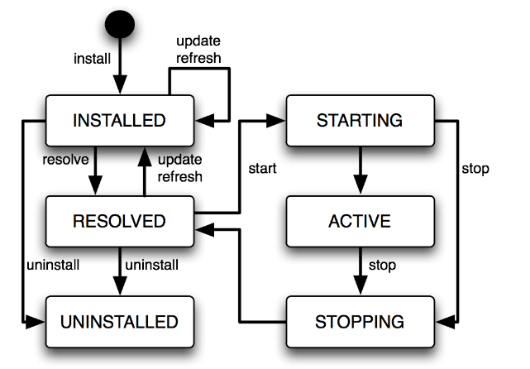
* *Bundle-Name*: Berisi nama *bundle*. Nama *bundle* biasanya dituliskan dalam bahasa manusia (bisa dibaca).
* *Bundle-SymbolicName*: Berisi nama *bundle* yang dituliskan secara simbolik dan unik. Properti ini digunakan sebagai identifier suatu *bundle* dalam OSGi *container*.
* *Bundle-Description*: Berisi deskripsi ringkasan fungsionalitas suatu *bundle*.
* *Bundle-ManifestVersion*: Berisi versi dari spesifikasi OSGi dimana bundle ini dapat di-*deploy*
* *Bundle-Version*: Berisi versi *bundle*
* *Bundle-Activator*: Berisi nama kelas yang digunakan sebagai activator *bundle*. OSGi menyediakan API yang dapat digunakan pengembang untuk membuat kelas baru yang dapat digunakan sebagai aktivator *bundle*.
* *Export-Package*: Berisi nama-nama *package* dari suatu *bundle* yang dapat diakses sebagai *classpath* oleh *bundle* lain.
* *Import-Package*: Berisi nama-nama *package* dari *bundle* lain yang diperlukan di dalam *classpath* dari suatu *bundle*.



Gambar 2.5 OSGi [10]

#### Status OSGi *Bundle*

Tiap OSGi *bundle* memiliki siklus hidup yang terdiri beberapa status seperti *ACTIVE*, *STOPPING*, *STARTING*, *RESOLVED*, *INSTALLED*, dan *UNINSTALLED* [10]. Status yang paling berpengaruh dalam siklus hidup OSGi *bundle* adalah *ACTIVE*, *RESOLVED*, dan *INSTALLED*. Siklus hidup OSGi ditunjukkan pada Gambar 2.6.



Gambar 2.6 Siklus hidup sebuah OSGi bundle [6]

Status *INSTALLED* merupakan status yang menandakan bahwa *bundle* sudah dipasang ke dalam sistem OSGi namun dengan dependensi yang belum terpenuhi. Dependensi merupakan pustaka yang diperlukan sebuah *bundle* agar dapat berfungsi. Status *RESOLVED* menandakan bahwa bundle sudah terpasang dan semua dependensi yang diperlukan sudah terpenuhi namun belum dijalankan. Status *ACTIVE* merupakan status dimana bundle sudah *RESOLVED* dan berjalan.

#### *Bundle* *fragment*

*Bundle* *fragment* adalah *bundle* yang tidak bisa berdiri sendiri melainkan harus digunakan sebagai *bundle* tambahan untuk *bundle* lain yang menjadi *bundle* induk dari *bundle* ini. Karena *bundle* *fragment* tidak bisa berdiri sendiri, *bundle* ini tidak memiliki siklus hidup standard sebuah *bundle* melainkan mengikuti siklus hidup *bundle* induknya. *Bundle* ini bekerja dengan menambahkan semua isi nya (kelas, berkas, dan lain-lain) ke dalam *bundle* induk [7]. Penjelasan tersebut digambarkan pada Gambar 2.7.

Pada tugas akhir ini, modul-modul akan di-*build* sebagai *bundle* *fragment* dari *bundle* perangkat lunak utama. Metode ini dilakukan karena beberapa masalah dengan metode modul sebagai bundle standard. Masalah tersebut antara lain pada *ViewResolver* dan *ServletContext* dari Spring MVC. Pembahasan lebih detil akan dibahas pada Bab IV.

C:\Users\sagungw93\Downloads\bundle-fragment.png

Gambar 2.7 *Bundle* *Fragment*

### OSGi *service registry*

*Service registry* merupakan komponen utama dari modularitas OSGi. Komponen ini adalah tempat untuk menyimpan service yang disediakan oleh *bundle-bundle* yang ter-*install* pada *container* OSGi. *Bundle* mendaftarkan *service* yang berupa obyek baik melalui kelas yang telah disediakan API OSGi yaitu *ServiceReference* dan *ServiceRegistration* ataupun menggunakan fitur *dependency injection* yang disediakan bundle pihak ketiga yang ter-*install* pada *container* OSGi [7].

## Apache Tomcat

Apache Tomcat adalah perangkat lunak *web server* yang digunakan untuk mempublikasikan perangkat lunak Sistem Informasi Akademik ini. *Web server* ini mendukung beberapa implementasi *Java EE* seperti *servlet*, *JSP*, dan *Java Expression Language* [11]. Perangkat lunak utama harus di-*build* menjadi berkas *Web Application Archive* (WAR) untuk dapat dipublikasikan ke dalam *web server* ini.

## PostgreSQL

PostgreSQL merupakan salah satu RDBMS. PostgreSQL bersifat *open-source* sehingga dapat digunakan, dimodifikasi dan disebarkan. ORDBMS ini dapat mendukung bagian besar dari standar SQL dan menawarkan berbagai macam fitur seperti *query* yang bersifat kompleks, *foreign key* dan lainnya [12]. Pada tugas akhir ini, PostgreSQL digunakan karena adanya fitur UUID sebagai tipe data untuk *primary key.*

## Hibernate ORM

Hibernate adalah kerangka kerja ORM (*Object-Relational Mapping*) untuk bahasa pemrograman Java. Hibernate digunakan untuk memetakan sebuah tabel pada basis data relasional (SQL) ke sebuah kelas *domain model* pada Java (obyek persistensi) [13]. Hibernate mencocokan tipe data pada sebuah kelas *domain model* dengan tipe data yang ada pada RDBMS.



Gambar 2.8 Hibernate [9]

Hibernate mendukung hampir semua RDBMS. Berikut RDBMS yang didukung:

* HSQL Database Engine
* DB2/NT
* MySQL
* PostgreSQL
* Oracle
* Microsoft SQL Server Database

## Apache Maven

Maven adalah alat bantu pengelolaan dependensi proyek pada Java.

### Dependensi maven

Dependensi merupakan pustaka Java yang berbentuk berkas .jar yang dapat biasanya diperlukan pengembang dalam mengembangkan perangkat lunak [14].

#### Dependensi transitif

Fitur dependensi transitif sudah ada sejak maven versi 2.x. Fitur ini memudahkan pengembang pada saat menentukan dependensi maven yang diperlukan dimana pohon dependensi yang dibutuhkan oleh dependensi proyek akan disertakan secara otomatis [15].

Pohon dependensi dapat memiliki banyak level selama tidak ada dependensi sirkular. Pohon dependensi mungkin saja memiliki sangat banyak level dependensi. Oleh karena itu, maven menyediakan beberapa cara untuk masalah tersebut antara lain:

* *Dependency Scope*
* Dependensi *excluded*. Pengembang mendefinisikan dependensi apa saja yang tidak diperlukan oleh dependensi di dalam proyek.
* Dependensi optional. Merupakan dependensi yang secara *default* tidak disertakan.

#### Cakupan dependensi

Fitur ini berguna untuk mengatur transitifitas dari sebuah dependensi. Terdapat 6 *scope* [16] yaitu:

* *Compile*. *Scope* *default* pada maven jika elemen *scope* tidak didefinisikan dalam sebuah dependensi. Dependensi dengan *scope* ini tersedia di dalam *classpath* proyek. *Scope* ini secara *default* menyertakan dependensi transitif.
* *Provided*. *Scope* ini mirip dengan *scope* *compile* namun tidak menyertakan dependensi ke dalam *classpath* proyek.
* *Runtime*. *Scope* ini menandakan bahwa dependensi diperlukan saat proses eksekusi aplikasi bukan saat proses kompilasi
* *Test*. *Scope* ini menandakan bahwa dependensi diperlukan saat proses *testing* aplikasi bukan saat proses kompilasi atau eksekusi.
* *System*. *Scope* ini mirip dengan *scope* *provided* namun pengembang diharuskan menyediakan pustaka .jar secara eksplisit.
* *Import*. Merupakan *scope* yang digunakan untuk menyertakan dependensi yang bertipe pom. Dependensi ini hanya bisa digunakan didalam elemen *<dependencyManagement>*

### Maven *plugin*

Maven menyediakan maven *build lifecycle* pada saat melakukan *build* pada proyek maven. Maven *build lifecycle* adalah runtutan fase yang diperlukan aplikasi java sehingga aplikasi siap di-*deploy*. Fase-fase tersebut adalah *validate*, *initialize*, *generate-sources*, *process-sources*, *generate-resources*, *process-resources*, *compile*, *process-classes*, *generate-test-sources*, *process-test-sources*, *generate-test-resources*, *process-test-resources*, *test-compile*, *process-test-classes*, *test*, *prepare-package*, *package*, *pre-integration-test*, *integration-test*, *post-integration-test*, *verify*, *install*, dan *deploy* [17]*.*

Masing-masing fase di atas memiliki *default goal*. Selain *default goal*, pengembang dapat menggunakan maven *plugin* untuk menambahkan *goal* pada fase *default*. Pada tugas akhir ini penulis menambahkan *plugin* *maven-bundle-plugin* dari Apache Felix untuk memanipulasi hasil berkas MANIFEST.MF pada hasil *build* agar dapat digunakan di lingkungan OSGi.

## Kerangka Kerja

Kerangka kerja adalah abstraksi dari suatu perangkat lunak yang memiliki fungsionalitas yang generic untuk dikembangkan lebih lanjut agar menghasilkan fungsionalitas yang lebih spesifik. Kerangka kerja perangkat lunak bersifat universal, *reuseable*, dan dapat memfasilitasi pengembangan perangkat lunak lebih lanjut [18]. Salah satu contoh bentuk kerangka kerja adalah kerangka kerja aplikasi web.

Kerangka kerja aplikasi web adalah kerangka kerja perangkat lunak yang dirancang untuk mendukung pengembangan website dinamis, aplikasi web, *web services*, dan *web resources*. Kerangka kerja ini bertujuan untuk meringankan pengerjaan aktifitas-aktifitas yang sama dalam pengembangan suatu web. Contoh aplikasi kerangka kerja web yaitu *Content Management System* (CMS).

Fitur-fitur umum yang terdapat pada aplikasi kerangka kerja web antara lain:

*Web template system*

Merupkan kumpulan templat web yang diolah prosesor templat agar menjadi satu web utuh.

*Caching*

Merupakan metode untuk mengurangi *bandwidth* dan beban server pada penyimpanan dokumen dan konten web.

Keamanan

Kemanan pada aplikasi kerangka kerja web meliputi otentikasi dan otorisasi yang berguna untuk mengidentifikasi pengguna aplikasi dan hak akses fungsi-fungsi tertentu.

Konfigurasi dan akses basis data

Untuk menyimpan data yang diperlukan aplikasi. Kerangka kerja yang berorientasi obyek biasanya menyediakan kakas bantu *object-relational mapping* contohnya *Hibernate* untuk Java dan *Entity Framework* untuk .NET.

*Scaffolding*

Merupakan metode pada *model-view controller* yang menyediakan template kode kepada programmer untuk menentukan bagaimana basis data pada aplikasi digunakan. Templat kode yang disediakan biasanya meliputi *create, read, update,* dan *delete.*

*URL mapping*

Merupakan metode untuk mengubah URL halaman web URL tertentu yang telah ditentukan. Fitur ini memudahkan pengguna membaca, menulis, mengingat alamat web, dan memberikan informasi yang lebih jelas tentang struktur web tersebut.

***[Halaman ini sengaja dikosongkan]***

# BAB III ANALISIS DAN PERANCANGAN PERANGKAT LUNAK

Pada bab ini akan dibahas analisis dan perancangan perangkat lunak dari kerangka kerja Sistem Informasi Akademik ini. Hasil dari proses ini berupa diagram yang akan digunakan sebagai acuan untuk proses implementasi perangkat lunak. Selain digunakan sebagai acuan untuk proses selanjutnya, beberapa diagram hasil dari proses perancangan digunakan sebagai dokumentasi dari implementasi perangkat lunak. Diagram yang dihasilkan pada proses ini disajikan dalam bentuk *Unified Modelling Language* (UML).



## Analisis

Analisis meliputi analisis domain permasalahan, pendeskripsian perangkat lunak secara umum, penggambaran dan penjelasan kasus penggunaan dalam bentuk diagram kasus penggunaan, dan penggambaran dan penjelasan alur aktivitas tiap kasus penggunaan dalam bentuk diagram aktivitas.

### Analisis permasalahan

Pada penelitian terkait, yaitu sistem informasi akademik generik yang terbagi ke dalam modul-modul, telah menghasilkan perangkat lunak sistem informasi akademik dengan fungsionalitas umum pada proses ekivalensi, kurikulum, pembelajaran, dan penilaian. Perangkat lunak tersebut masing-masing berupa perangkat lunak berbasis web dengan pola arsitektur *Model-View-Controller.* Namun, perangkat lunak tersebut perlu digabungkan menjadi satu perangkat lunak sistem informasi akademik. Proses penggabungan yang diinginkan adalah dengan membentuk perangkat lunak pada penelitian terkait sebagai modul kemudian modul tersebut ditambahkan ke dalam perangkat lunak utama tanpa mengubah kode sumber dari perangkat lunak utama. Selain penambahan modul perangkat lunak pada penelitian sebelumnya, perangkat lunak utama perlu dapat menambahkan modul lain untuk menangani kasus dimana suatu perguruan tinggi memiliki modul lain selain empat modul pada penelitian sebelumnya. Selain itu, perangkat lunak utama juga harus bisa mengganti modul-modul tersebut untuk menangani kasus adanya perubahan atau variasi proses bisnis pada modul yang telah diintegrasi. Selain itu, karena modul perangkat lunak sudah dijadikan satu pada perangkat lunak utama, keamanan juga harus dikelola pada perangkat lunak utama untuk mengamankan data dan proses bisnis modul-modul yang terintegrasi.

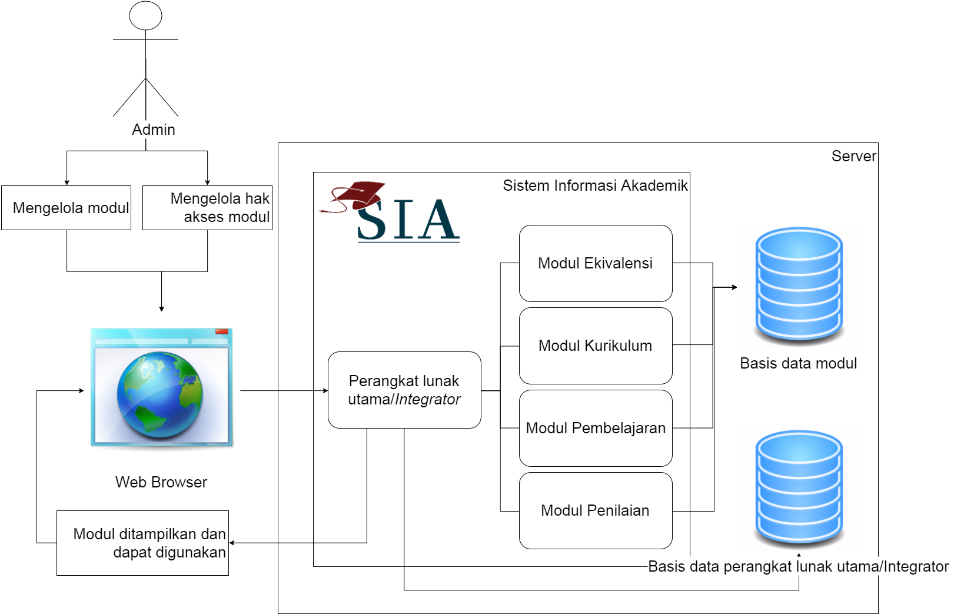
Pada perangkat lunak ini akan digunakan sistem modularitas OSGi dengan pola arsitektur perangkat lunak yang menyerupai *Hierarchical Model-View-Controller* saat modul-modul sudah digabungkan. Modularitas perangkat lunak dirancang dengan dibuatnya abstraksi/aturan modul yang harus diimplementasi oleh modul yang ingin ditambahkan agar modul dapat dibaca. Dengan sistem modular, perangkat lunak utama dapat menambah, mengganti, dan menghapus modul tanpa mengubah kode sumber dari perangkat lunak utama. Selain itu, untuk mengatur keamanannya perangkat lunak utama dapat mengelola pengguna, hak akses pengguna, peran, dan pemetaan peran ke dalam menu-menu pada modul.

### Deskripsi umum sistem

Pada tugas akhir ini akan dikembangkan sebuah perangkat lunak berbasis web yang dapat mengintegrasikan perangkat lunak sistem informasi akademik generik yang telah dikembangkan pada tugas-tugas akhir sebelumnya [3] [4] [5] [6] yang dibentuk menjadi modul untuk setiap perangkat lunak tersebut. Untuk dapat mengintegrasikan modul-modul tersebut, perangkat lunak dirancang modular. Artinya perangkat lunak dapat menambahkan, mengganti, dan menghapus modul tanpa mengubah kode sumber dari perangkat lunak utama selama modul mengimplementasi abstraksi atau aturan yang ditetapkan perangkat lunak utama. Modul yang diintegrasikan akan berupa sebuah berkas arsip Java (.jar).

Proses pengintegrasian modul dilakukan oleh administrator dengan mengunggah berkas arsip Java dari modul ke dalam perangkat lunak utama melalui sebauh *web browser*. Setelah itu modul akan terdaftar dan fungsionalitas modul tersebut dapat dijalankan.

Perangkat lunak berbasis web ini akan dioperasikan oleh seorang administrator yang dapat mengelola modul dan mengelola keamanan perangkat lunak. Pengelolaan keamanan yang dimaksud adalah otentikasi yang berupa data pengguna yang dapat mengakses perangkat lunak utama sistem informasi akademik dan otorisasi yang meliputi pengelolaan hak akses pengguna dan hak akses pengguna terhadap menu pada modul. Arsitektur sistem ini ditunjukkan pada Gambar 3.1.



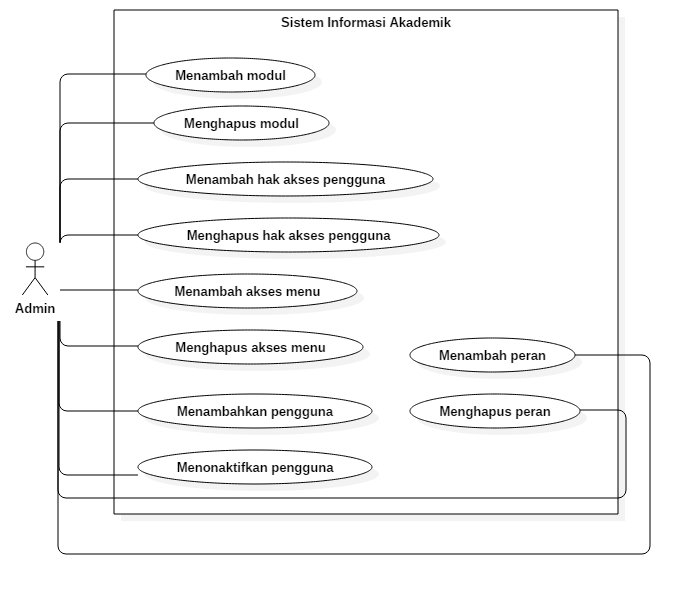
Gambar 3.1 Arsitektur sistem

### Kasus penggunaan

Bagian ini menjelaskan analisa dan perancangan kasus-kasus penggunaan yang terdapat pada fitur administratif pada perangkat lunak. Kasus penggunaan tersebut ditunjukkan pada Gambar 3.2 dan tiap kasus penggunaan akan disertakan spesifikasi dan diagram aktivitasnya.

#### Aktor

Aktor adalah entitas luar yang terlibat langsung dalam penggunaan perangkat lunak. Pada kerangka kerja ini aktor adalah administrator atau admin. Admin memiliki hak akses penuh ke dalam kerangka kerja perangkat lunak. Admin berkewajiban menjalankan segala kebutuhan administratif sistem yaitu mengelola modul beserta keamanan/hak akses tiap pengguna ke menu pada modul yang tersedia.



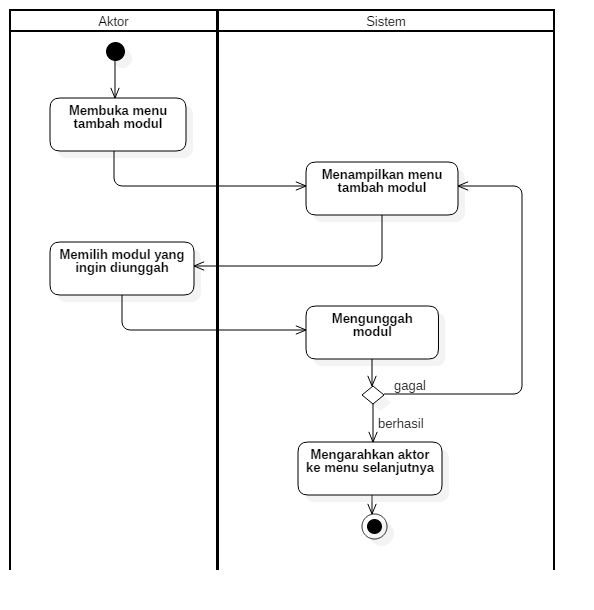
Gambar 3.2 Diagram kasus penggunaan

#### Menambah modul

Modul yang dibuat wajib dimuat ke dalam berkas Java *Archive* kemudian aktor mengunggah berkas modul tersebut ke dalam perangkat lunak. Modul yang dibangun harus mengikuti konvensi dan memiliki properti yang telah ditentukan pada kerangka kerja sehingga bisa dibaca. Properti yang telah didefinisikan di dalam modul akan otomatis dibaca sehingga aktor tidak perlu mengisi kembali properti modul tersebut melalui formulir web.

Tabel 3.1 Spesifikasi kasus penggunaan menambah modul

|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menambah modul |
| Nomor | UC-001 |
| Deskripsi | Kasus penggunaan ini adalah kasus dimana aktor menambah modul ke dalam perangkat lunak web Sistem Informasi Akademik. |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Modul belum terdaftar |
| Kondisi Akhir | Modul terdaftar |
| Alur Normal | 1. Aktor memilih menu menambah modul 2. Sistem menampilkan menu tambah modul 3. Aktor memilih modul yang ingin diunggah 4. Sistem mengunggah modul 5. Modul terunggah dan proses dilanjutkan ke penambahan hak akses menu |
| Alur Alternatif | * 1. Modul gagal terunggah      1. Perangkat lunak mengeluarkan pesan gagal      2. Aktor diarahkan kembali ke menu awal penambahan modul |



Gambar 3.3 Diagram aktivitas menambah modul

#### Menghapus modul

Kasus penggunaan ini merupakan kasus penggunaan dimana aktor menghapus modul dari perangkat lunak Sistem Informasi Akademik. Tabel 3.2 menunjukan spesifikasi kasus penggunaan dan Gambar 3.4 menunjukan diagram aktivitas kasus penggunaan.

#### Menambah akses menu

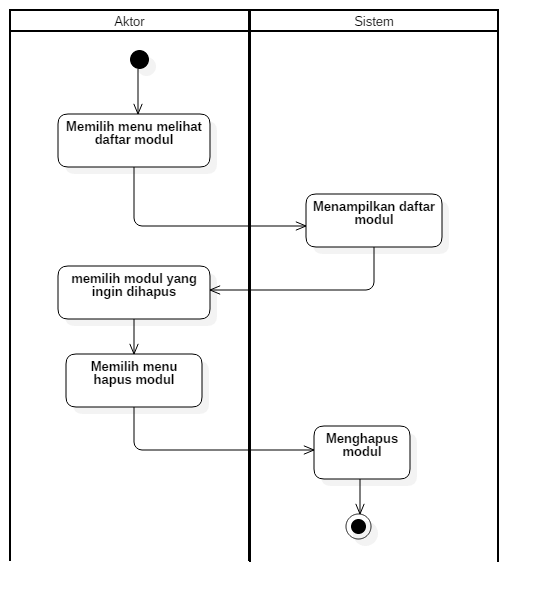
Menambah akses menu yang dimaksud pada kasus penggunaan ini adalah memetakan peran apa saja yang bisa mengakses suatu menu. Menu menambah akses menu dapat diakses dengan dua cara yaitu setelah menambah modul ataupun melalui menu yang disediakan sistem. Tabel 3.3 menunjukan spesifikasi kasus penggunaan dan Gambar 3.5 dan Gambar 3.6 menunjukan diagram aktivitas kasus penggunaan.

#### Menghapus akses menu

Kasus penggunaan merupakan kasus penggunaan dimana aktor menghapus akses menu dari modul. Tabel 3.4 menunjukan spesifikasi kasus penggunaan dan Gambar 3.7 menunjukan diagram aktivitas kasus penggunaan.

Tabel 3.2 Spesifikasi kasus penggunaan menghapus modul

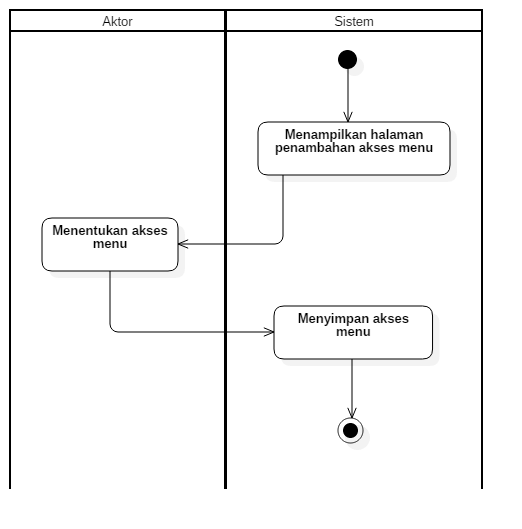
|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menghapus modul |
| Nomor | UC-002 |
| Deskripsi | Kasus penggunaan ini merupakan kasus penggunaan dimana aktor menghapus modul dari perangkat lunak Sistem Informasi Akademik. |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Modul terdaftar |
| Kondisi Akhir | Modul terhapus |
| Alur Normal | 1. Aktor memilih menu melihat daftar modul 2. Sistem menampilkan daftar modul 3. Aktor memilih modul yang ingin dihapus 4. Aktor menghapus modul yang dipilih 5. Sistem menghapus modul 6. Modul terhapus |
| Alur Alternatif |  |



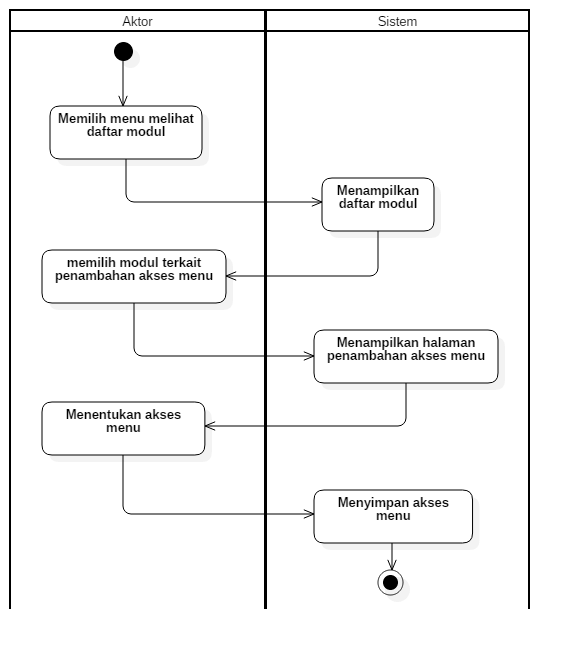
Gambar 3.4 Diagram aktivitas menghapus modul

Tabel 3.3 Spesifikasi kasus penggunaan menambah akses menu

|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menambah akses menu |
| Nomor | UC-003 |
| Deskripsi | Kasus penggunaan ini merupakan kasus penggunaan dimana aktor menambah akses menu dari modul yang sudah terdaftar |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Akses menu tidak terbatas |
| Kondisi Akhir | Akses menu lebih terbatas |
| Alur Normal | 1. Aktor memilih menu menampilkan daftar modul 2. Sistem menampilkan daftar modul yang terpasang 3. Aktor memilih modul yang terkait dengan penambahan akses menu 4. Sistem menampilkan halaman yang berisi menu-menu yang terdapat pada modul yang dipilih dan peran-peran yang ada pada sistem 5. Aktor menentukan akses menu 6. Sistem menyimpan akses menu |
| Alur Alternatif | * 1. Aktor selesai mengunggah modul      1. Lanjut ke alur nomor 4 dengan modul yang baru diunggah sebagai modul yang dipilih |



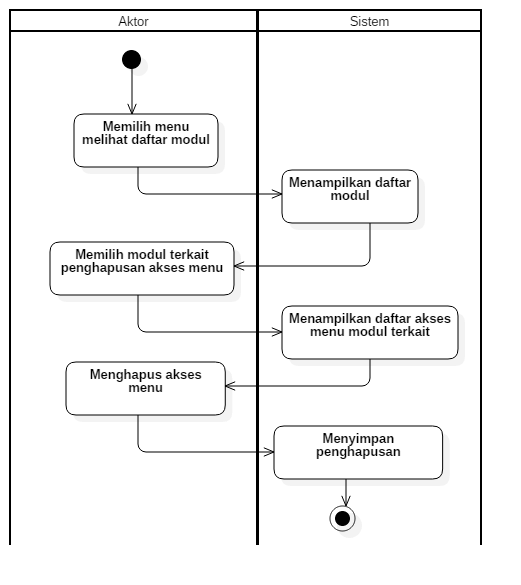
Gambar 3.5 Diagram aktivitas menambah akses menu setelah Menambah modul



Gambar 3.6 Diagram aktivitas menambah akses menu

Tabel 3.4 Spesifikasi kasus penggunaan menghapus akses menu

|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menghapus akses menu |
| Nomor | UC-004 |
| Deskripsi | Kasus penggunaan merupakan kasus penggunaan dimana aktor menghapus akses menu dari modul |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Akses menu terbatas |
| Kondisi Akhir | Akses menu kurang terbatas |
| Alur Normal | 1. Aktor memilih menu menampilkan daftar modul 2. Sistem menampilkan daftar modul yang terpasang 3. Aktor memilih modul yang terkait dengan penghapusan akses menu 4. Sistem menampilkan halaman yang berisi daftar akses menu 5. Aktor menghapus akses menu 6. Sistem menyimpan akses menu |
| Alur Alternatif |  |



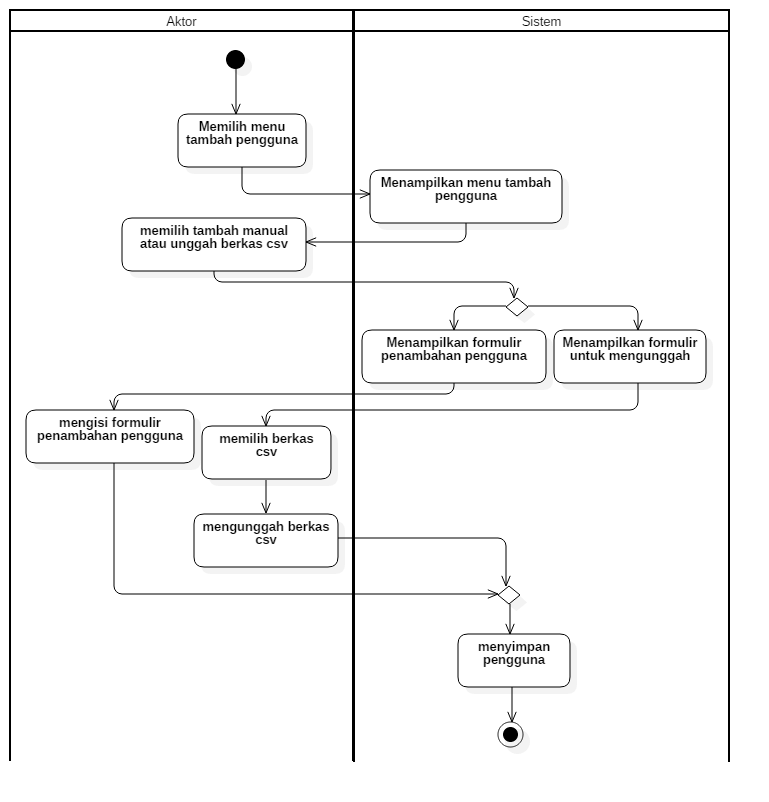
Gambar 3.7 Diagram aktivitas menghapus akses menu

#### Menambah pengguna

Kasus penggunaan ini adalah kasus dimana aktor memberikan akses perangkat lunak ini ke peserta didik atau pendidikan dan tenaga kependidikan yang terdaftar pada Sistem Informasi Akademik sehingga mereka bisa terlibat dalam sistem. Tabel 3.5 menunjukan spesifikasi kasus penggunaan dan Gambar 3.8 menunjukan diagram aktivitas kasus penggunaan.

Tabel 3.5 Spesifikasi kasus penggunaan menambah pengguna

|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menambah pengguna |
| Nomor | UC-005 |
| Deskripsi | Kasus penggunaan ini adalah kasus dimana aktor memberikan akses perangkat lunak ini ke peserta didik atau pendidikan dan tenaga kependidikan yang terdaftar pada Sistem Informasi Akademik sehingga mereka bisa terlibat dalam sistem. |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Pengguna belum terdaftar |
| Kondisi Akhir | Pengguna terdaftar |
| Alur Normal | 1. Aktor memilih menu tambah pengguna 2. Sistem menampilkan menu tambah pengguna 3. Aktor memilih menambahkan pengguna dengan cara manual 4. Sistem menampilkan formulir penambahan pengguna 5. Aktor mengisi formulir 6. Sistem menyimpan pengguna |
| Alur Alternatif | * 1. Aktor memilih menambahkan pengguna dengan cara mengunggah berkas *CSV*      1. Sistem menampilkan formulir unggah      2. Aktor memilih berkas untuk diunggah      3. Aktor mengunggah berkas      4. Lanjut ke alur normal nomor 6 |



Gambar 3.8 Diagram aktivitas menambah pengguna

#### Menonaktifkan pengguna

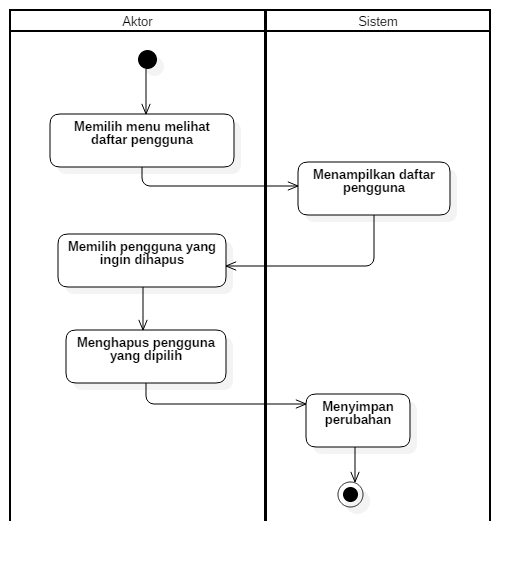
Merupakan kasus penggunaan dimana aktor menonaktifkan pengguna yang sudah tidak memiliki hak akses ke dalam sistem. Tabel 3.6 menunjukan spesifikasi kasus penggunaan dan Gambar 3.9 menunjukan diagram aktivitas kasus penggunaan.

#### Menambah hak akses pengguna

Penambahan hak akses pengguna yang dimaksud di sini adalah pemetaan peran-peran terhadap pengguna pada sistem. Penambahan hak akses pengguna dilakukan melalui daftar pengguna. Dari menu tersebut aktor memilih pengguna yang ingin diubah hak aksesnya. Tabel 3.7 menunjukan spesifikasi kasus penggunaan dan Gambar 3.10 menunjukan diagram aktivitas kasus penggunaan.

Tabel 3.6 Spesifikasi kasus penggunaan menonaktifkan pengguna

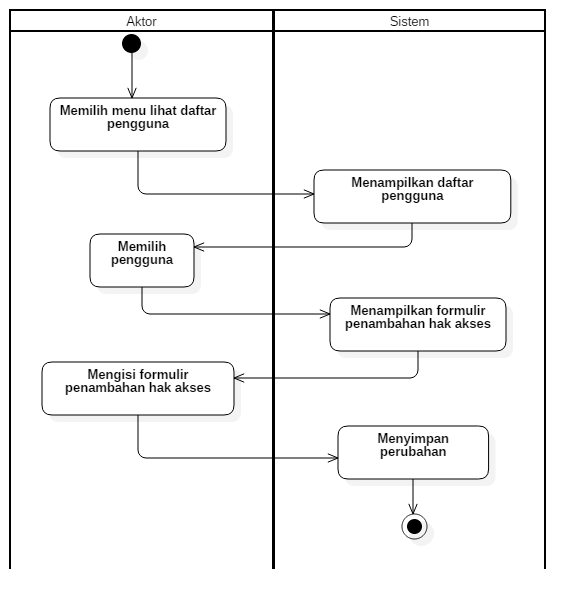
|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menonaktifkan pengguna |
| Nomor | UC-006 |
| Deskripsi | Kasus penggunaan ini merupakan kasus penggunaan dimana aktor menonaktifkan pengguna yang sudah tidak memiliki hak akses ke dalam sistem. |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Pengguna terdaftar |
| Kondisi Akhir | Pengguna tidak terdaftar |
| Alur Normal | 1. Aktor memilih menu melihat daftar pengguna 2. Sistem menampilkan daftar pengguna 3. Aktor memilih pengguna yang ingin dinonaktifkan 4. Aktor menonaktifkan pengguna yang dipilih 5. Sistem menyimpan perubahan |
| Alur Alternatif |  |



Gambar 3.9 Diagram aktivitas menonaktifkan pengguna

Tabel 3.7 Spesifikasi kasus penggunaan menambah hak akses pengguna

|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menambah hak akses pengguna |
| Nomor | UC-007 |
| Deskripsi | Kasus penggunaan ini merupakan kasus penggunaan dimana aktor membatasi hak akses pengguna terhadap menu pada sistem |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Pengguna dapat mengakses lebih sedikit menu |
| Kondisi Akhir | Pengguna dapat mengakses lebih banyak menu |
| Alur Normal | 1. Aktor memilih menu melihat daftar pengguna 2. Sistem menampilkan daftar pengguna 3. Aktor memilih pengguna terkait hak akses yang ingin ditambahkan 4. Sistem menampilkan formulir hak akses pengguna yang dipilih 5. Aktor mengisi formulir 6. Sistem menyimpan perubahan |
| Alur Alternatif |  |



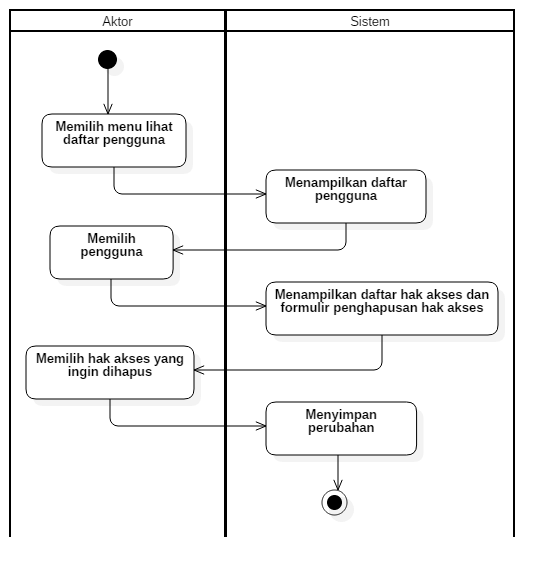
Gambar 3.10 Diagram aktivitas menambah hak akses pengguna

#### Menghapus hak akses pengguna

Hampir sama dengan kasus penggunaan menambah hak akses pengguna, kasus penggunaan ini diakses melalui daftar pengguna dan ditampilkan setelah aktor memilih pengguna yang terkait dengan kasus penggunaan ini. Tabel 3.8 menunjukan spesifikasi kasus penggunaan dan Gambar 3.11 menunjukan diagram aktivitas kasus penggunaan.

Tabel 3.8 Spesifikasi kasus penggunaan menghapus hak akses pengguna

|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menghapus hak akses pengguna |
| Nomor | UC-008 |
| Deskripsi | Kasus penggunaan dimana aktor mengurangi batas akses pengguna ke menu pada sistem |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Pengguna dapat mengakses lebih banyak menu |
| Kondisi Akhir | Pengguna dapat mengakses lebih sedikit menu |
| Alur Normal | 1. Aktor memilih menu melihat daftar pengguna 2. Sistem menampilkan daftar pengguna 3. Aktor memilih pengguna terkait hak akses yang ingin dihapus 4. Sistem menampilkan formulir penghapusan hak akses 5. Aktor memilih hak akses yang ingin dihapus dari pengguna 6. Sistem menyimpan perubahan |
| Alur Alternatif |  |



Gambar 3.11 Diagram aktivitas menghapus hak akses pengguna

#### Menambah peran

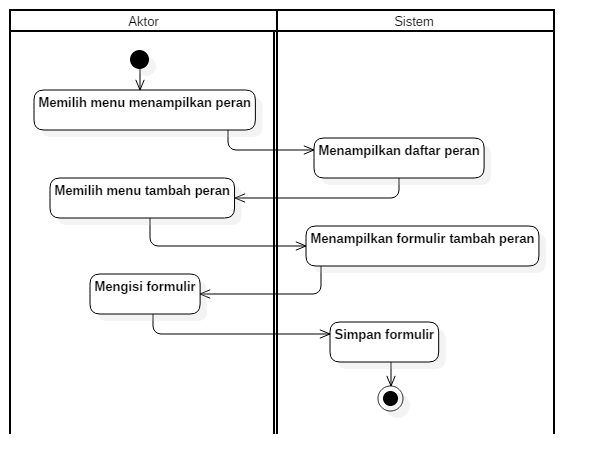
Peran pada sistem ini digunakan sebagai otorisasi pengguna untuk masuk ke dalam sebuah menu. Kasus penggunaan ini digunakan untuk mendaftarkan peran yang bisa digunakan oleh pengguna. Tabel 3.9 menunjukkan spesifikasi kasus penggunaan dan Gambar 3.12 menunjukan diagram aktivitas kasus penggunaan.

#### Menghapus peran

Peran pada sistem digunakan untuk menghapus peran pada sistem. Aktor menghapus peran dengan memilih menu hapus pada yang ada pada tiap baris daftar peran. Tabel 3.10 menunjukkan spesifikasi kasus penggunaan dan Gambar 3.13 menunjukan diagram aktivitas kasus penggunaan.

Tabel 3.9 Spesifikasi kasus penggunaan menambah peran

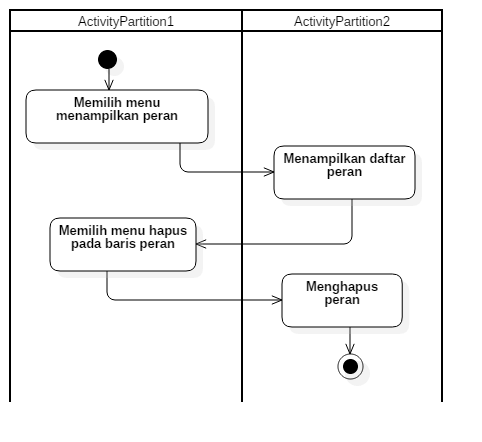
|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menambah Peran |
| Nomor | UC-009 |
| Deskripsi | Kasus penggunaan dimana aktor menambah peran yang dapat digunakan sebagai hak akses pengguna. |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Peran belum terdaftar |
| Kondisi Akhir | Peran terdaftar |
| Alur Normal | 1. Aktor memilih menu melihat daftar peran 2. Sistem menampilkan daftar peran 3. Aktor memilih menu menambah peran 4. Sistem menampilkan formulir penambahan peran 5. Aktor mengisi formulir 6. Sistem menyimpan peran baru |
| Alur Alternatif |  |



Gambar 3.12 Diagram aktivitas menambah peran

Tabel 3.10 Spesifikasi kasus penggunaan menghapus peran

|  |  |
| --- | --- |
| Komponen | Deskripsi |
| Nama | Menghapus Peran |
| Nomor | UC-010 |
| Deskripsi | Kasus penggunaan dimana aktor menghapus peran yang digunakan sebagai hak akses pengguna. |
| Tipe | Fungsional |
| Aktor | Administrator |
| Kondisi Awal | Peran terdaftar |
| Kondisi Akhir | Peran tidak terdaftar |
| Alur Normal | 1. Aktor memilih menu melihat daftar peran 2. Sistem menampilkan daftar peran 3. Pengguna memilih menu hapus pada peran yang ingin dihapus 4. Sistem menghapus peran |
| Alur Alternatif |  |



Gambar 3.13 Diagram aktivitas menghapus peran

## Perancangan Perangkat Lunak

Arsitektur perangkat lunak yang dibahas meliputi arsitektur secara utuh yang disajikan dalam diagram arsitektur, arsitektur tiap komponen yang meliputi *web layer*, *service layer*, *data layer*, *domain layer*, dan *plugin layer* yang masing-masing berisi diagram kelas dan diagram *sequence*, model basis data yang disajikan dalam bentuk *Conceptual Data Model* (CDM) dan *Physical Data Model* (PDM), dan perancangan antarmuka pengguna.

### Arsitektur secara utuh

Pada dasarnya, arsitektur yang digunakan oleh perangkat lunak ini adalah arsitektur *multi-layer*, dan pola modularitas yang kemudian di-*deploy* ke dalam *OSGi Environment* yang telah terpasang dependensi yang diperlukan. Perangkat lunak dibagi menjadi lima *layer* yaitu *web layer*, *service layer*, *data layer*, *domain layer*, dan *plugin layer*  yang masing-masing dapat di-*deploy* secara terpisah kedalam berkas .war untuk *web layer* dan .jar untuk *layer* lainnya. Arsitektur secara utuh ditunjukkan pada Gambar 3.14.

*Web layer*, yang diberi nama sia-web, merupakan *layer* yang mempresentasikan perangkat lunak sebagai perangkat lunak berbasis web dan menangani pendaftaran modul ke dalam perangkat lunak.

*Web layer* memiliki ketergantungan pada *service* *layer* untuk menjalankan logika bisnis perangkat lunak, *data layer* untuk mengakses data pada basis data, *domain layer* untuk mengakses kelas entitas, dan *plugin layer* untuk mengakses kelas yang berhubungan dengan pendaftaran modul. Untuk menjalankan logika bisnisnya, *layer* ini tidak menggunakan kelas melainkan *interface* dan menerima *service* dari *service layer* sehingga implementasi logika bisnis pada *service layer* tersembunyi dari *web layer* dan hubungan antara *web layer* dan *service layer* menjadi *loosely-coupled.* Begitu pula dengan hubungan *layer* ini dengan *data layer* dimana *Data Access Object* (DAO) yang diperlukan berbentuk *interface* dengan implementasi yang tersembunyi pada *data layer*.

*Service layer*, yang diberi nama sia-service, merupakan *layer* yang menyediakan logika bisnis untuk fungsionalitas admin. *Layer* ini menyediakan *interface* *service* untuk digunakan oleh *layer* lain sekaligus implementasinya yang bersifat tertutup untuk meminimalisasi *coupling*.

*Service layer* memiliki ketergantungan terhadap *data layer* untuk mengakses data pada basis data, *domain layer* untuk mengakses kelas entitas, dan *plugin layer* untuk mengakses kelas yang berhubungan dengan pendaftaran modul. Sama seperti pada *web layer*, *layer* ini menggunakan *interface* DAO untuk menggunakan kelas implementasi DAO pada *data layer*.

*Data layer*, diberi nama sia-data, berisi *interface* dan kelas implementasi DAO untuk mengakses data. Seperti pada *service layer*, *interface* pada *layer* ini digunakan oleh *layer* lain untuk mengakses kelas implementasi pada *layer* ini.

*Data layer* memiliki ketergantungan terhadap *domain layer* untuk mengakses kelas entitas.

*Plugin layer*, diberi nama sia-plugin, merupakan *layer* yang berisi *interface* untuk yang digunakan untuk mendaftarkan modul ke dalam perangkat lunak sistem informasi akademik.

*Domain layer*, diberi nama sia-domain, merupakan *layer* yang berisi kelas entitas yang digunakan pada penyimpanan data dan pertukaran data antar *layer*.

### Arsitektur komponen

Kerangka kerja perangkat lunak Sistem Informasi Akademik ini tersusun atas lima layer sebagai komponennya yaitu *Web Layer, Service Layer, Data Layer, Plugin Layer, dan Domain Layer*. *Layer* tersebut dibangun dan di-*deploy* secara terpisah ke dalam *OSGi* *container*.



Gambar 3.14 Diagram arsitektur

#### *Web layer*

*Web layer* dari perangkat lunak ini menggunakan pola perancangan *Model-View-Controller* yang didukung oleh kerangka kerja *Spring MVC*. Seperti yang digambarkan pada Gambar 3.15, pada *layer* ini terdapat kelas-kelas *controller* utama yang merupakan fitur admin, kelas-kelas yang bertugas untuk mendaftarkan modul, kelas-kelas yang mengatur keamaan dari kerangka kerja ini, dan beberapa komponen kerangka kerja pihak ketiga untuk mengambil obyek *service* dan *DAO* dari *layer* lain seperti *service layer* dan *data layer*.

Pendaftaran modul baru akan dilakukan oleh kelas *WebAppInitializer*. Kelas ini mengimplementasi kelas *interface* *WebApplicationInitializer* dari Sping MVC dimana kelas *interface* tersebut memiliki fungsi *onStartup(ServletContext)* yang dipanggil ketika kerangka kerja web pertama kali dijalankan.

*Layer* ini juga mengatur keamaanan dari kerangka kerja terutama pada fitur admin. Keamanan pada *layer* ini dilakukan pada bagian *Controller* pada *Spring MVC* dengan menggunakan kerangka kerja *Spring Security*. Gambar 3.17 menggambarkan dependensi dari layer ini terhadap *Spring Security*.

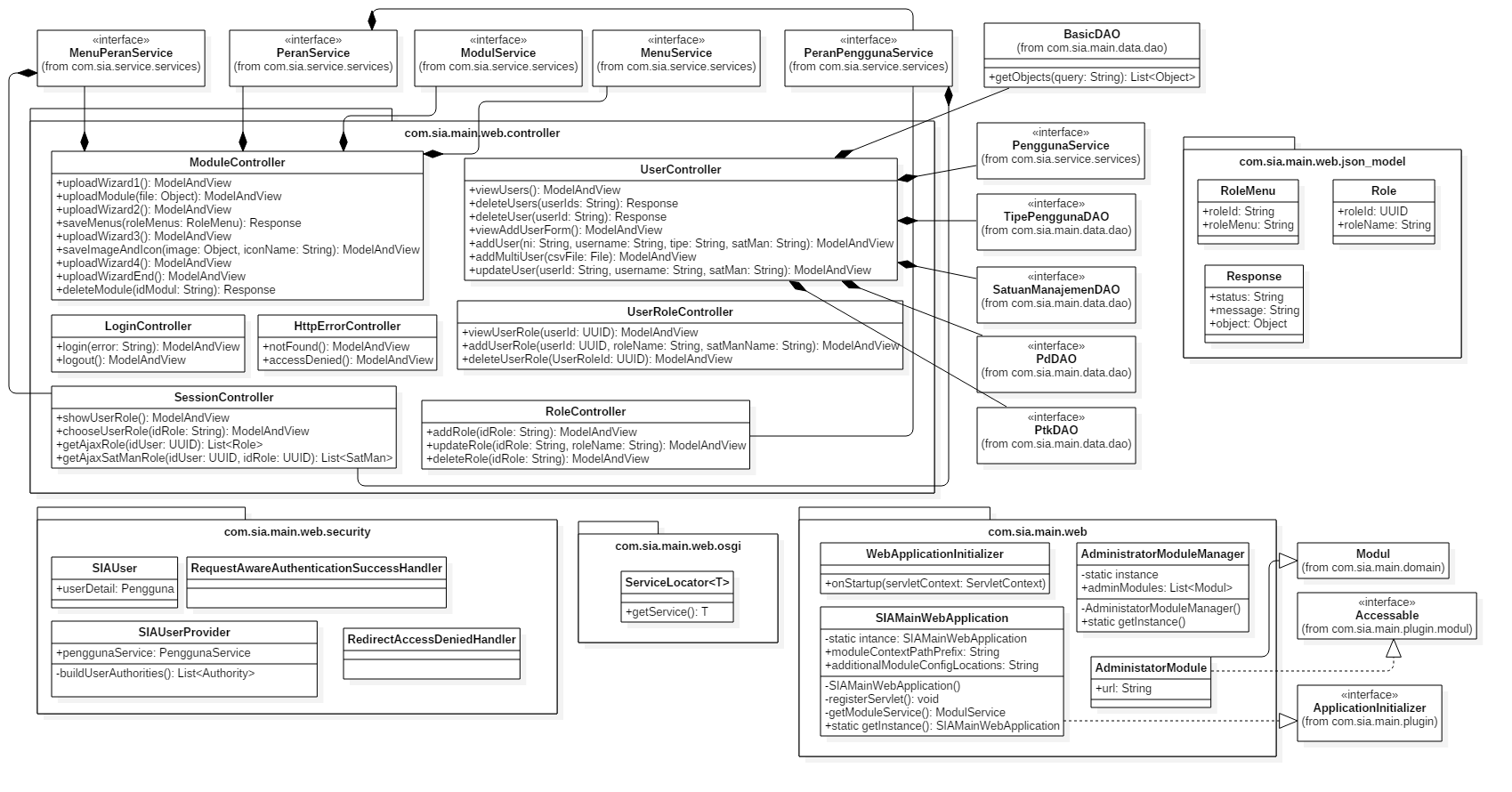
#### *Service layer*

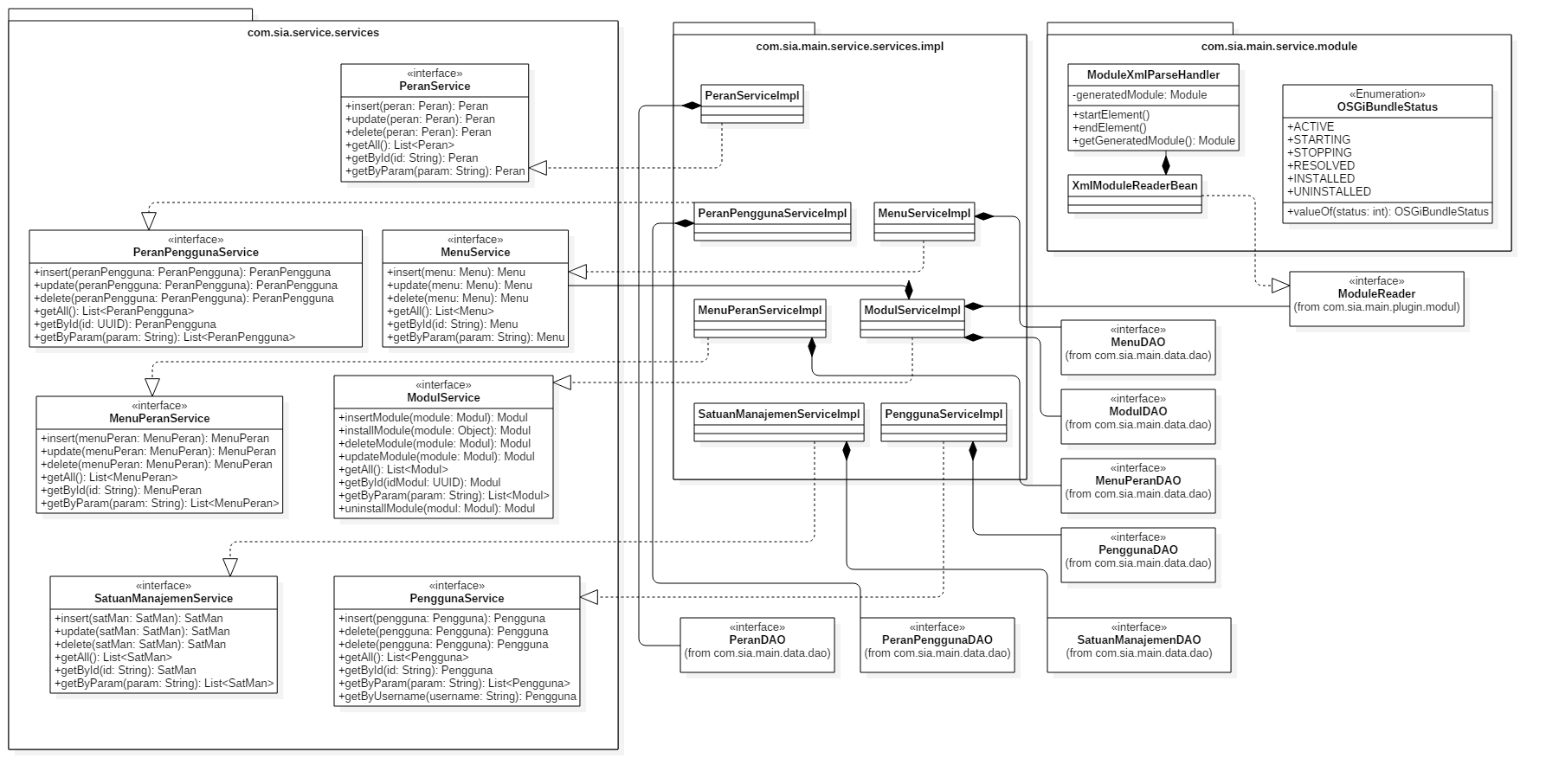
*Service layer* pada kerangka kerja ini berperan sebagai penyedia logika bisnis terutama untuk fitur-fitur admin dari kerangka kerja ini.

Seperti yang digambarkan pada Gambar 3.16, *layer* ini memiliki kelas *interface* dan implementasinya yang memiliki fungsi-fungsi bisnis berdasarkan kelas *domain* yang ada.

Kelas implementasi dari *interface* disediakan *layer* ini akan didaftarkan ke dalam *OSGi Service Registry* berdasarkan kelas *interface* nya sehingga logika bisnis yang disediakan dapat digunakan layer lain (Gambar 3.18).

Gambar 3.15 Diagram kelas *web layer*





Gambar 3.16 Diagram kelas *service layer*

#### *Data layer*

*Data layer* merupakan *layer* yang bertugas mengurus pertukaran data yang diperlukan perangkat lunak terutama oleh fitur admin ke dalam basis data relasional.

Seperti yang digambarkan pada Gambar 3.20, pada *layer* ini terdapat kelas-kelas *interface DAO* dan implementasinya yang saling berhubugan untuk mengurus transaksi yang terlibat pada tiap kelas entitas dalam kerangka kerja ini.

Kelas-kelas *interface DAO* yang bertugas mengurus akses ke basis data memiliki fungsi yang sama namun berbeda pada kelas entitas nya. Oleh karena itu, dapat dibuat kelas *interface* *generic DAO* yang berisi fungsi-fungsi umum yang ada pada *DAO* untuk diturunkan oleh kelas *interface DAO* lain sehingga pendifinisian fungsi pada kelas *interface DAO* lain tidak lagi diperlukan.

Selain adanya *generic DAO*, diperlukan kelas *DAO* yang tidak berhubungan dengan kelas domain manapun. kelas tersebut diberi nama *basicDAO*. Kelas ini digunakan untuk transaksi dan *query* yang langsung berhubungan dengan banyak tabel dan mendukung *query* natif *SQL*.

Kelas-kelas implementasi *DAO* pada *layer* ini akan didaftarkan ke dalam *OSGi Service Registry* sesuai kelas *interface* nya masing-masing sehingga layer lain dapat menggunakannya secara *loosely-coupled* seperti yang digambarkan pada Gambar 3.19.

#### *Plugin layer*

*Plugin layer* pada kerangka kerja ini berisi kelas *interface* dan beberapa implementasinya yang dapat digunakan pengembang modul untuk membuat modul. Modul yang akan dikembangkan wajib menggunakan *Spring MVC* dan mengikuti aturan-aturan yang berupa konvensi.

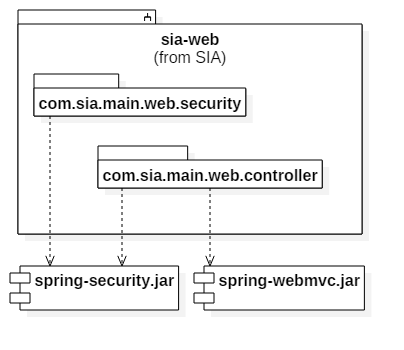
Modul akan didaftarkan melalui fitur penambahan modul pada fitur admin. Modul yang telah didaftarkan akan dipasang melalui konfigurasi *servlet* pada *web layer*.

Modul yang didaftarkan perlu mendefinisikan properti modulnya seperti nama modul, menu, URL menu, dan lain-lain yang akan dibaca pada *service layer* dengan menggunakan kelas *interface* pada layer ini.

Kelas *interface* utama pada *layer* ini yaitu *interface* *Menu* yang berisi nama menu dan URL menu untuk mengakses nya dari *web client* dan *Module* yang dapat berisi nama modul dengan data keterangan *servlet* nya ataupun dengan daftar *base package*-nya.

#### *Domain layer*

*Domain layer* pada kerangka kerja ini berisi kelas-kelas entitas yang merepresentasikan tabel pada basis data. Kelas-kelas ini digunakan sebagai tipe data pada pertukaran data antar layer dan basis data. Selain itu kelas-kelas ini juga merupakan pemetaan tabel pada basis data ke dalam kelas *Java* atau disebut *Object Relational Mapping* (*ORM*). *ORM* pada *layer* ini didukung oleh kerangka kerja *Hibernate*.

Struktur kelas diagram pada *layer* ini mengikuti *Physical Data Model* yang akan dijelaskan pada Bab Perancangan Basis Data

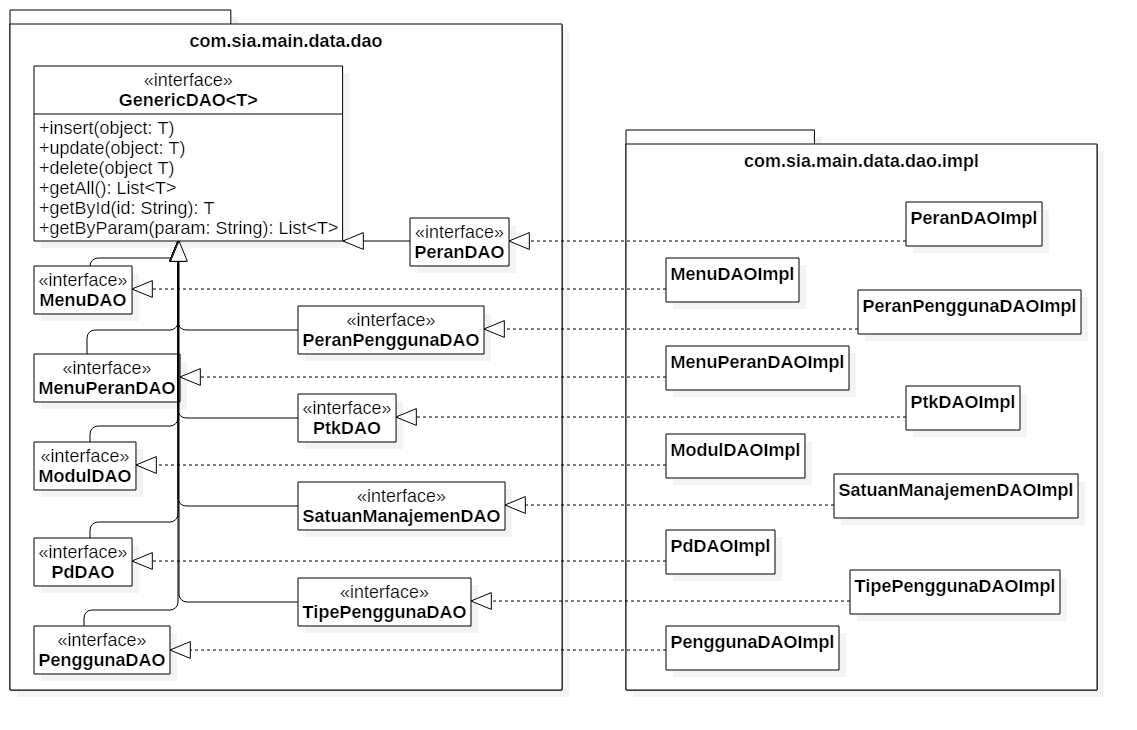
Gambar 3.17 Dependensi *web layer*



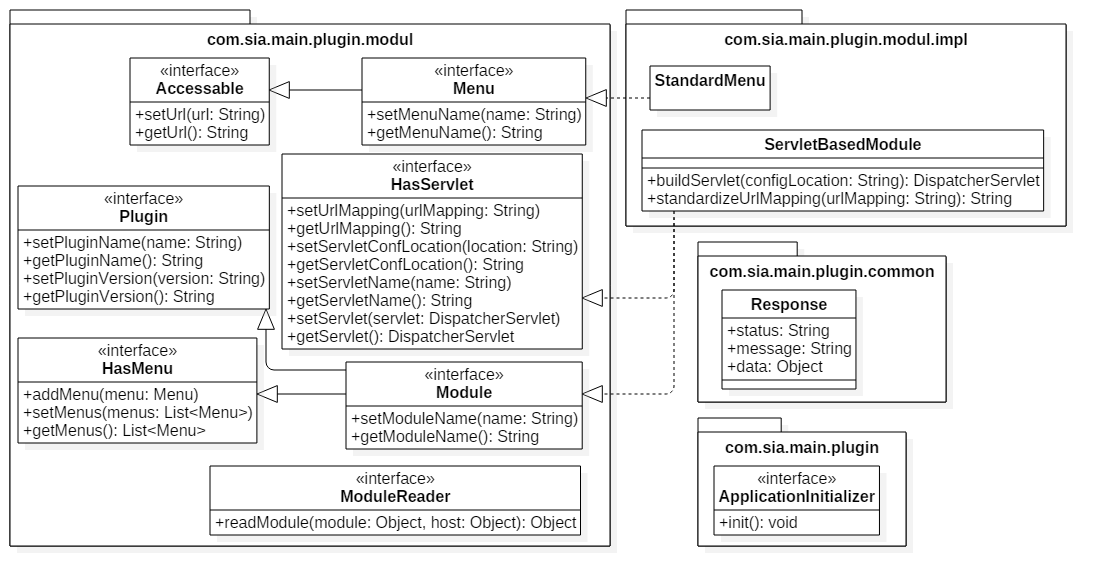
Gambar 3.18 Pengambilan *OSGi service* yang didaftarkan *service layer* oleh *web layer*



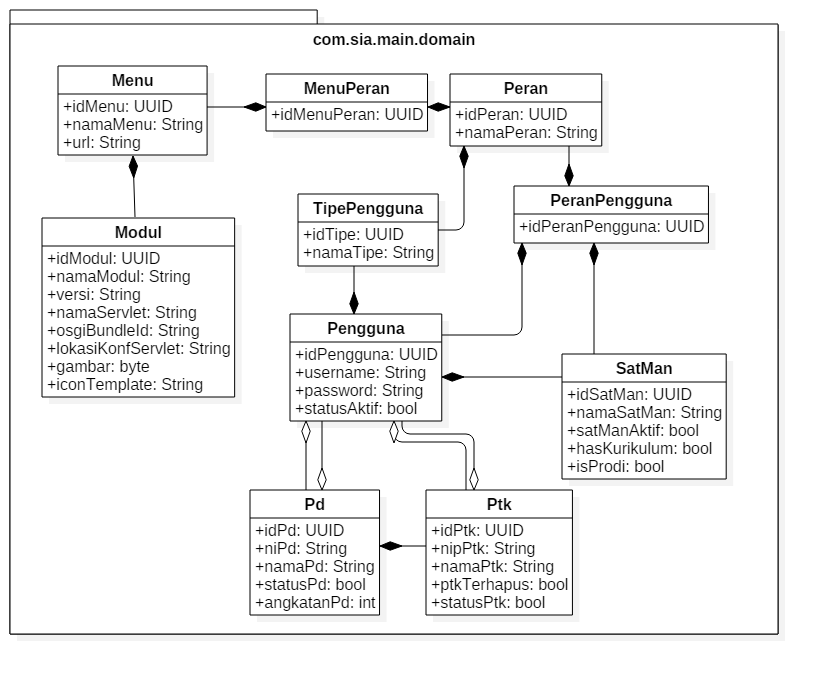
Gambar 3.19 Pengambilan *OSGi service* yang didaftarkan *data layer* oleh *web layer* dan *service layer*



Gambar 3.20 Diagram kelas *data layer*



Gambar 3.21 Diagram kelas *plugin layer*



Gambar 3.22 Diagram kelas *domain layer*

### Perancangan modularitas

Pada bagian ini dibahas mengenai arsitektur modul, dependensi modul terhadap komponen lain seperti Spring MVC dan lainnya, dan proses pendaftaran MVC modul ke dalam perangkat lunak utama yang membentuk pola arsitektur HMVC. Adapun *workflow* pengembangan sebuah modul yang meliputi pengembangannya hingga pendaftaran MVC modul ke dalam perangkat lunak utama ditunjukkan pada Gambar 3.23.



Gambar 3.23 *Workflow* pengembangan modul

#### Lingkungan pengembangan modul

Modul yang dikembangkan wajib memiliki komponen yang mendefinisikan properti modul dan komponen Spring MVC dari modul itu sendiri. Pada komponen Spring MVC, modul kemungkinan menggunakan beberapa kelas entitas dari perangkat lunak utama untuk mengakses pengguna yang terautentikasi. Karena itu, modul memiliki ketergantungan/dependensi dengan komponen lain dari perangkat lunak utama. Komponen yang menjadi dependensi yaitu

* + - 1. sia-plugin

Modul yang dibuat wajib memiliki suatu kelas yang merealisasikan kelas interface Module dan Menu.

* + - 1. sia-domain

Komponen ini diperlukan untuk mengakses kelas entitas dari perangkat lunak utama.

* + - 1. Spring MVC

Komponen ini diperlukan untuk membuat komponen MVC pada modul seperti DispatcherServlet dan @Controller.

Diagram komponen dari suatu modul digambarkan Gambar 3.24.



Gambar 3.24 Diagram komponen modul dan dependensinya dengan komponen lain

#### Arsitektur modul

Modul yang dibangun wajib menggunakan pola perancangan *Model-View-Controller (MVC)* dengan kerangka kerja *Spring MVC* dan dengan penambahan komponen modul yang diperlukan perangkat lunak utama. Gambar 3.25 menggambarkan diagram komponen dari suatu modul.



Gambar 3.25 Diagram komponen modul

#### Dependensi antar modul

Fungsionalitas pada modul memungkinkan penggunaan akses data dan kelas entitas yang sama antara satu modul dengan modul lain. Hal tersebut dapat mengakibatkan adanya penyalinan kode pada modul-modul yang saling beririsan. Kode yang beririsan dapat mengakibatkan perubahan yang banyak jika ada perubahan kode di salah satu modul. Untuk menangani hal tersebut, salah satu modul harus dirancang sedemikian rupa sehingga memiliki dependensi terhadap modul lain yang beririsan dengan modul tersebut. Dependensi antar modul diperbolehkan selama tidak ada dependensi *cyclic*. Contoh perancangan dependensi antar modul dan dependensi terhadap komponen kelas-kelas entitas pada modul ditunjukkan pada Gambar 3.26.

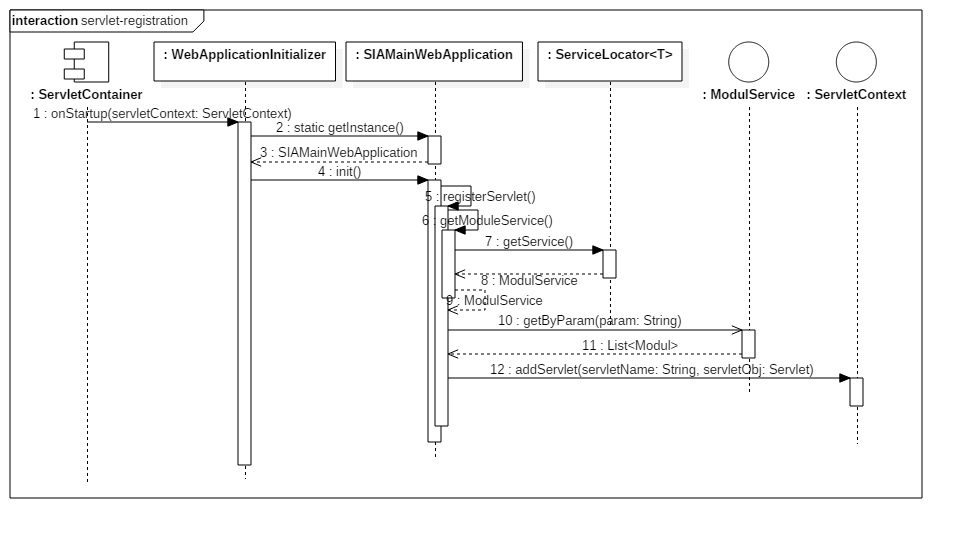
Untuk menangani dependensi modul terhadap kelas entitas dari modul lain, kelas-kelas entitas akan dijadikan satu komponen sehingga modul-modul memiliki dependensi terhadap komponen tersebut.



Gambar 3.26 Dependensi antar modul

#### Pendaftaran *servlet* MVC modul

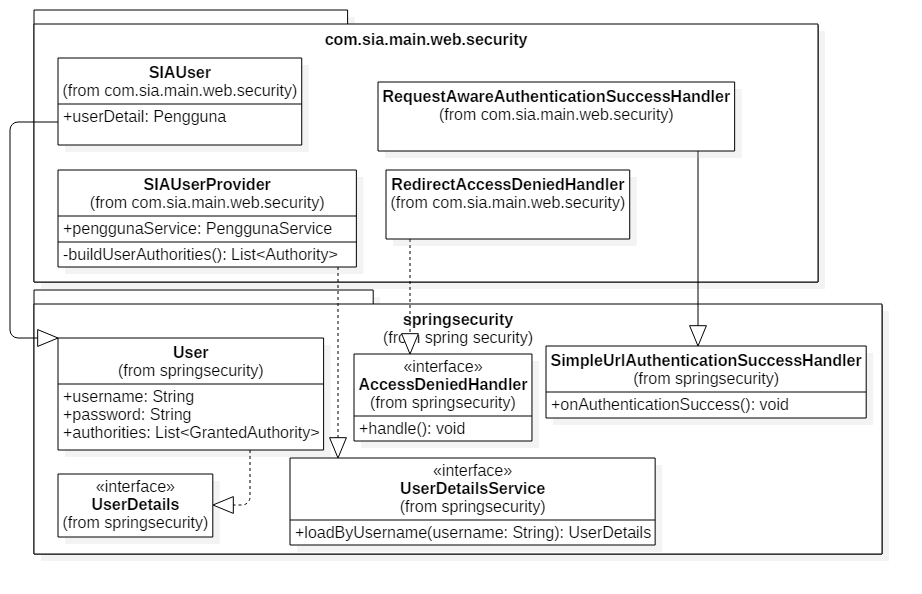
Pendaftaran *servlet* pada modul dilakukan pada *Web Laye*r perangkat lunak utama. Modul yang diunggah diwajibkan memiliki properti yang diperlukan pada saat pendaftaran *servlet* seperti nama *servlet* dan konfigurasi *servlet*. Properti modul tersebut akan disimpan pada basis data saat proses pengunggahan dan dibaca saat proses pendaftaran *servlet*. Berikut proses pendaftaran *servlet* dalam bentuk diagram *sequence* yang digambarkan pada Gambar 3.27.



Gambar 3.27 Diagram *sequence* pendaftaran modul

### Perancangan keamanan

Keamanan pada perangkat lunak ini meliputi proses autentikasi dan otorisasi yang didukung oleh kerangka kerja *Spring Security* dan *HttpSession*. Proses autentikasi dimulai dari halaman masuk hingga halaman pemilihan peran pengguna untuk menentukan modul apa saja yang akan ditampilkan kepada pengguna tersebut. Pendaftaran peran yang dimiliki pengguna akan dibahas pada fitur administratif pada subbab 3.1.3.8 3.1.7.1.7. Kelas-kelas yang diperlukan saat proses autentikasi digambarkan pada Gambar 3.28. Proses otorisasi dilakukan dengan pemeriksaan hak akses pada tiap *URL request*. Proses otorisasi yang gagal akan ditangani oleh kelas *AccessDeniedHandler* yang digambarkan Gambar 3.29. Alur proses otorisasi digambarkan pada diagram alur pada Gambar 3.28.



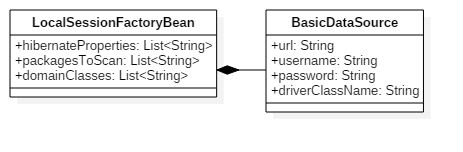
Gambar 3.28 Diagram kelas keamanan perangkat lunak



Gambar 3.29 Diagram alur otorisasi perangkat lunak

### Perancangan akses data

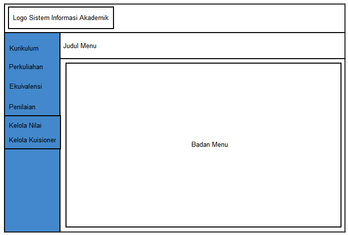
Seluruh modul perangkat lunak Sistem Informasi Akademik menggunakan satu akses data yang sama yang disediakan oleh perangkat lunak utama yang berupa sebuah konfigurasi. Konfigurasi ini didaftarkan bersama dengan konfigurasi *servlet* pada saat melakukan pendaftaran *servlet* modul ke dalam perangkat lunak utama. Gambar 3.30 merupakan gambar kelas yang digunakan untuk konfigurasi akses data modul perangkat lunak ini.



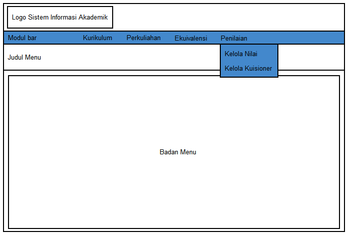
Gambar 3.30 Diagram kelas akses data

### Perancangan *template* antarmuka pengguna

Antarmuka pengguna menggunakan *template* untuk fungsionalitas admin dan fungsionalitas modul seperti yang digambarkan pada Gambar 3.31 dan Gambar 3.32.



Gambar 3.31 Halaman *template* fungsi administratif



Gambar 3.32 Halaman *template* modul

### Perancangan fitur administratif

Fitur administratif merupakan fitur utama yang dapat dilakukan administrator. Perancangan fitur ini meliputi penggambaran diagram sequence untuk tiap kasus penggunaan, perancangan basis data, dan perancangan antarmuka pengguna.

#### Hubungan antar kelas

Pada bagian ini dijelaskan interaksi antar obyek dari kelas-kelas pada sistem melalui fungsi pada obyek-obyek tersebut untuk menjalankan fungsionalitas sistem. Penjelasan tersebut digambarkan dengan *sequence diagram* untuk tiap kasus penggunaan.

##### Menambah modul

Seperti yang digambarkan Gambar 3.34 penambahan modul dilakukan dengan mengunggah berkas pada *web client*, kemudian *web client* melakukan *Http Request* ke *ModulController* untuk mengunggah berkas. *ModulController* melanjutkan berkas tersebut ke fungsi *uploadModule* pada *ModulService* yang kemudian dibaca oleh *OsgiModuleReader*. Modul yang telah dibaca memiliki properti modul dan menu. Kemudian modul disimpan pada basis data. Penyimpanan pada basis data dilakukan dengan memanggil fungsi *insert* pada *ModulService* kemudian diteruskan dengan memanggil fungsi *insert* pada *ModulDAO*. Setelah modul disimpan, menu pada modul disimpan. Penyimpanan dilakukan dengan memanggil fungsi *insert* pada *MenuService* dan kemudian diteruskan ke fungsi *insert* pada *MenuDAO*. Adapun pendaftaran modul dan pembacaan rincian modul telah ditunjukkan pada Gambar 3.23 dan rincian alur pemeriksaan URL *mapping* modul ditunjukkan pada Gambar 3.33.

##### Menghapus modul

Penghapusan modul digambarkan pada *sequence* *diagram* Gambar 3.35. Modul yang penghapusan dilakukan pada *ModulService* dengan fungsi *delete*. Selain pada basis data, modul pada *ModuleManager* juga ikut dihapus.

##### Menambah akses menu

Menambah akses menu yang dimaksud adalah memetakan peran apa saja yang bisa mengakses suatu menu. *Web client* melakukan *request* ke *ModuleController* dengan parameter id menu dan id peran yang ingin ditambah. Setelah id menu dan id peran didapat oleh *controller*, *controller* mengambil obyek menu dan peran melalui fungsi *getById* pada *MenuDAO* dan *PeranDAO*. Setelah obyek didapat, *controller* membuat obyek akses menu. Kemudian obyek tersebut disimpan pada basis data melalui *MenuPeranService* dan diteruskan ke *MenuPeranDAO*.

##### Menghapus akses menu

Akses menu dihapus dengan melakukan *request* ke *deleteRoleMenu* pada *ModuleController* dengan parameter id akses menu yang ingin dihapus. Setelah itu *controller* mengambil obyek dari basis data melalui *MenuPeranDAO*. Setelah obyek didapat, *controller* memanggil fungsi *delete* pada *MenuPeranService* dan diteruskan ke fungsi *delete* pada *MenuPeranDAO*.

##### Menambah pengguna

Penambahan pengguna dapat dilakukan dengan dua cara yaitu manual / satu per satu atau dengan mengunggah berkas *CSV*. Pada cara manual, *web client* melakukan *request* ke *addUser* pada *PenggunaController* dengan parameter data pengguna yang diperlukan. Data pengguna yang diperlukan antara lain *username*, satuan manajemen dan tipe pengguna apakah peserta didik atau pendidik dan tenaga kependidikan. Obyek dari parameter-parameter tersebut diambil dari basis data dengan kelas *DAO* yang bersangkutan. Setelah obyek didapat, obyek pengguna dibuat dan disimpan dengan fungsi *insert* pada *PenggunService* dan kemudian diteruskan ke *PenggunaDAO*. Penambahan pengguna dengan berkas *CSV* dilakukan dengan pembacaan berkas *CSV* pada *controller* kemudian dilakukan mekanisme yang sama seperti cara manual tiap baris pada berkas *CSV*.

##### Menonaktifkan pengguna

Penonaktifan pengguna dilakukan dengan dua cara yaitu menonaktifan pengguna satu per satu atau menonaktifkan lebih dari satu pengguna sekaligus. Pengguna dinonaktifkan dengan memaggil fungsi *delete* pada *controller* dengan parameter id pengguna. Kemudian obyek pengguna diambil dari basis data menggunakan *PenggunaDAO*. Setelah obyek didapat, *controller* memanggil fungsi hapus pada *PenggunaService* dengan parameter pengguna obyek dan kemudian diteruskan ke fungsi *delete* pada *PenggunaDAO*. Menonaktifan lebih dari satu pengguna sekaligus dilakukan dengan memanggil fungsi *deleteUsers* pada *UserController* dengan daftar id pengguna sebagai parameter kemudian dilakukan mekanisme yang sama dengan cara menonaktifkan pengguna satu per satu untuk tiap id pengguna dalam daftar.

##### Menambah hak akses pengguna

Hak akses pengguna ditambahkan dengan melakukan *request* *addUserRole* pada *UserRoleController* dengan parameter id pengguna, id peran, dan satuan manajemen. Obyek dari tiap parameter diambil dari basis data melalui kelas-kelas *DAO* obyek yang bersangkutan. Setelah obyek didapat, *controller* membuat obyek MenuPeran yang kemudian disimpan melalui fungsi *insert* pada *PeranPenggunaService* dan diteruskan ke fungsi *insert* pada *PeranPenggunaDAO*.

##### Menghapus hak akses pengguna

Penghapusan hak akses dilakukan dengan melakukan *request* ke *deleteUserRole* pada *UserRoleController* dengan parameter id user role. Kemudian obyek user role diambil dari basis data melalui *PeranPengguna*. Setelah obyek didapat, *controller* memanggil fungsi *delete* pada *PeranPenggunaService* dan diteruskan ke fungsi *delete* pada *PeranPenggunaDAO*.

##### Menambah peran

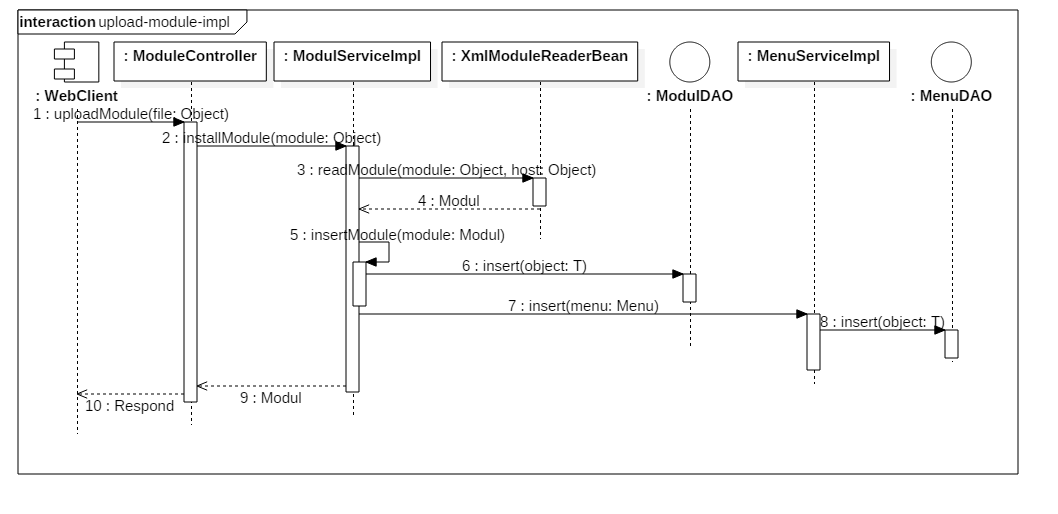
Penambahan peran dilakukan melaui *request* yang bertipe *POST* ke fungsi *addRole()* dengan parameter *request* berupa nama peran. Selanjutnya *request* tersebut diolah oleh kelas implementasi dari *PeranService* yaitu *PeranServiceImpl* dan obyek peran disimpan ke dalam basis data oleh *PeranDAO.* Proses ini digambarkan pada diagram *sequence* pada Gambar 3.44.

##### Menghapus peran

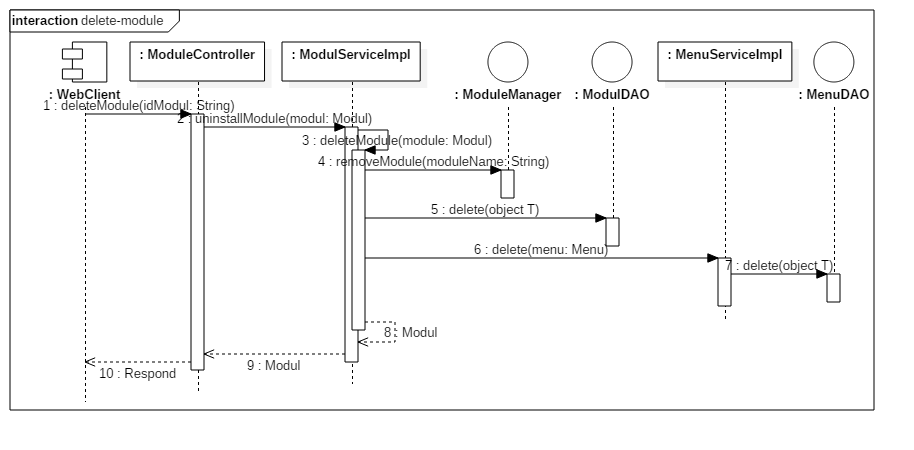
Penghapusan peran dilakukan melaui *request* yang ke fungsi *deleteRole()* dengan parameter *request* berupa id peran yang dipilih. Selanjutnya *request* tersebut diolah oleh kelas implementasi dari *PeranService* yaitu *PeranServiceImpl* dan obyek peran pada basis data dihapus oleh *PeranDAO.* Proses ini digambarkan pada diagram *sequence* pada Gambar 3.45.



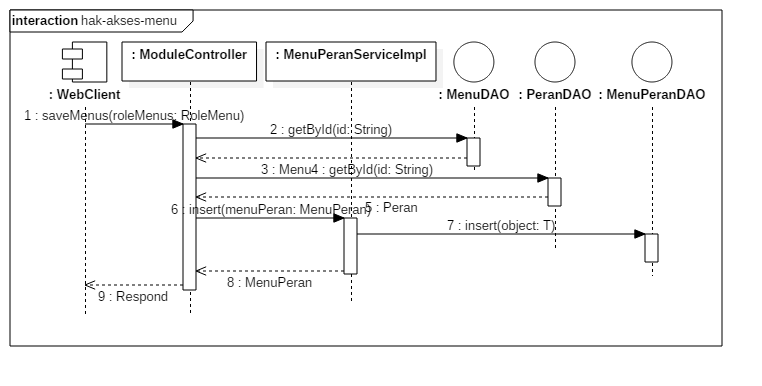
Gambar 3.33 *Flow Chart* penambahan modul ke basis data

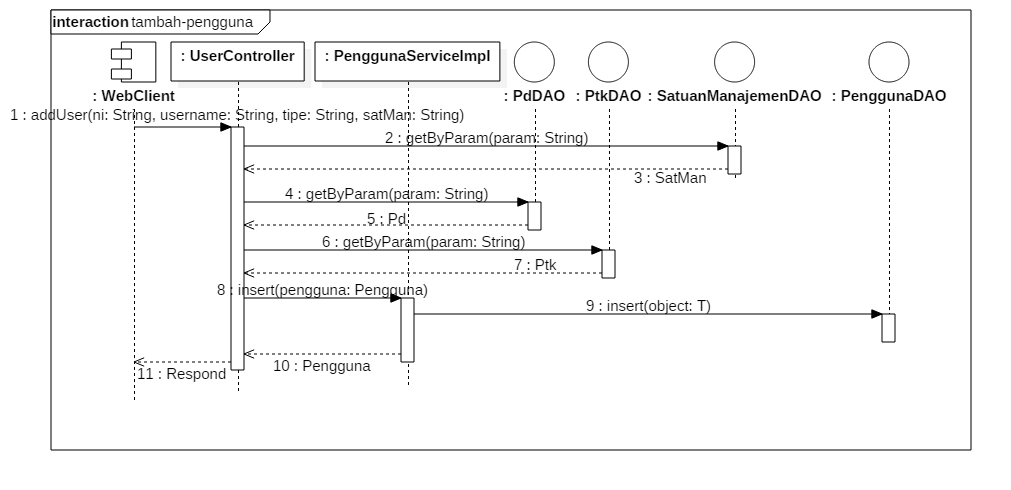


Gambar 3.34 sequence diagram Menambah modul

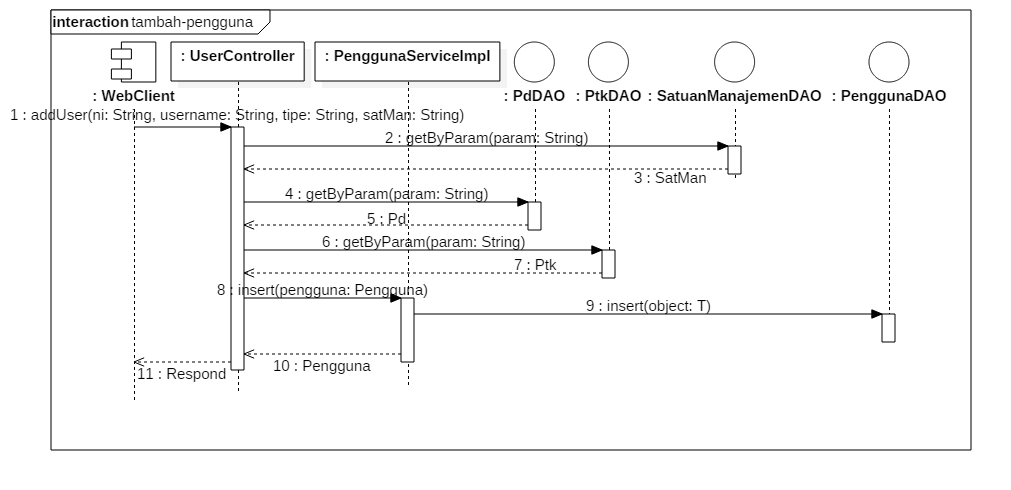


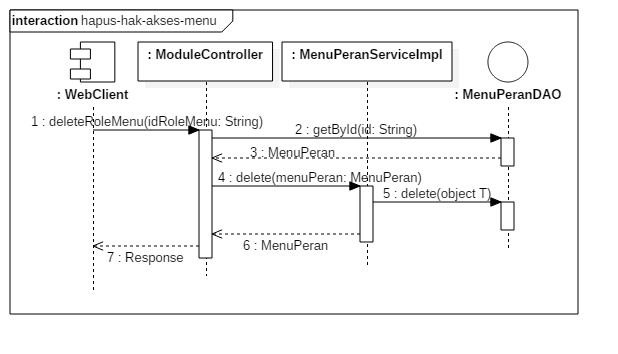
Gambar 3.35 sequence diagram Menghapus modul



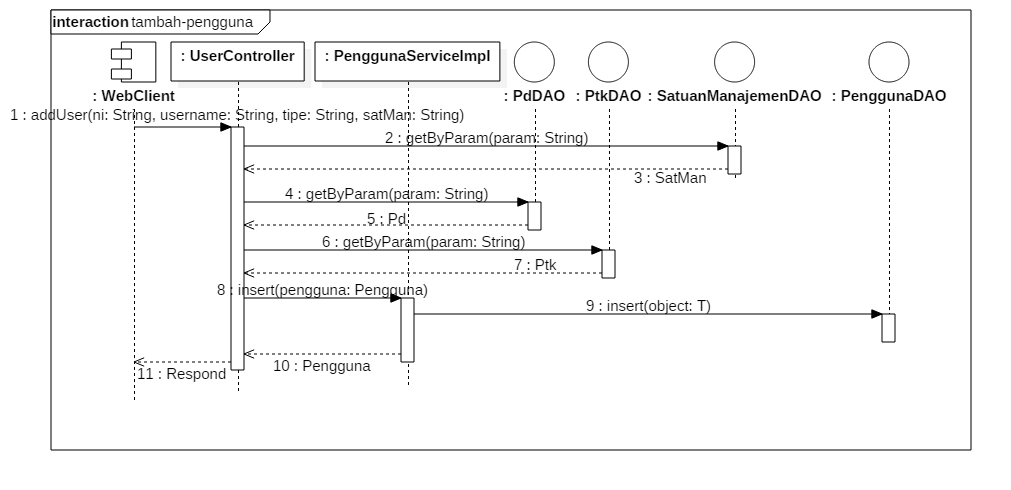


Gambar 3.36 sequence diagram Menambah akses menu

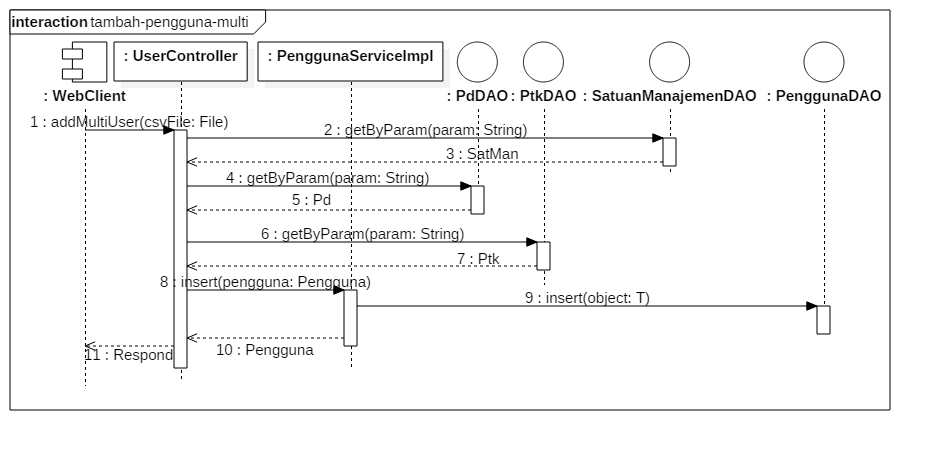




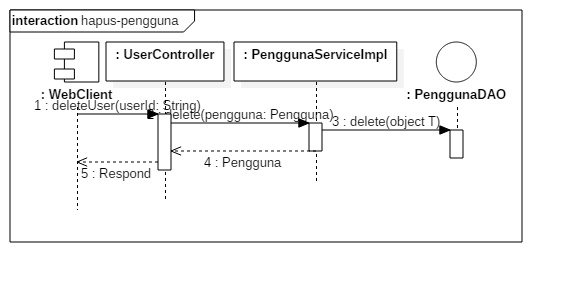
Gambar 3.37 sequence diagram Menghapus akses menu



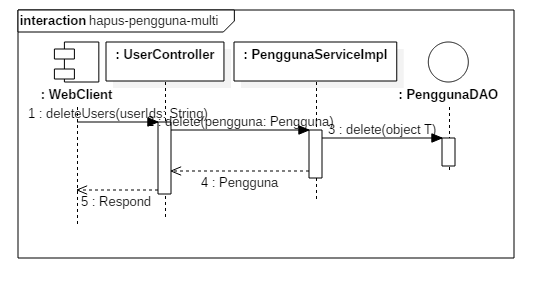
Gambar 3.38 sequence diagram Menambah pengguna



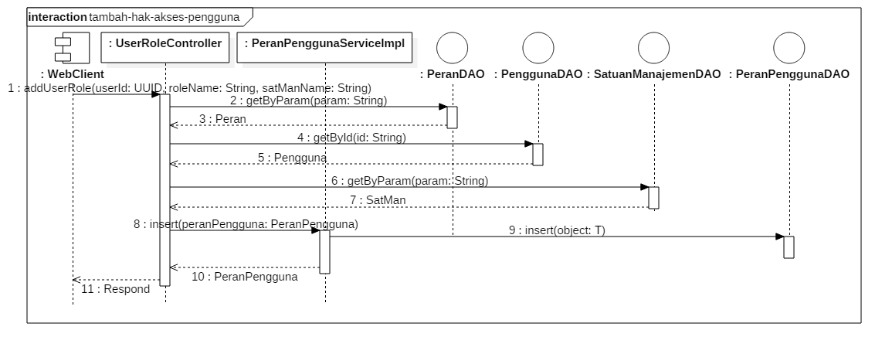
Gambar 3.39 sequence diagram Menambah pengguna (dengan *CSV*)



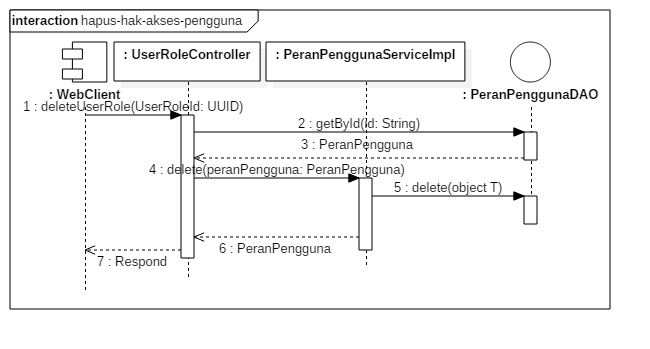
Gambar 3.40 sequence diagram Menonaktifkan pengguna



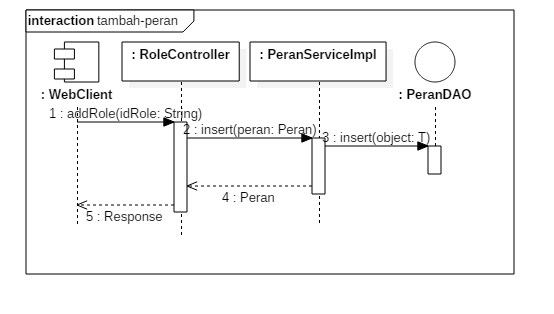
Gambar 3.41 sequence diagram Menonaktifkan pengguna (*multiple*)



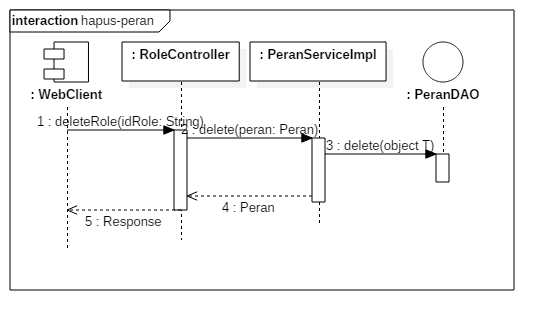
Gambar 3.42 sequence diagram Menambah hak akses pengguna



Gambar 3.43 sequence diagram Menghapus hak akses pengguna

****

Gambar 3.44 sequence diagram Menambah peran

****

Gambar 3.45 sequence diagram Menghapus peran

#### Perancangan basis data

Perancangan basis data pada perangkat lunak ini dijelaskan dalam Model Data Konseptual (CDM), yang digambarkan pada Gambar 3.46, dan Model Data Fisik (PDM), yang digambarkan pada Gambar 3.47. Pada CDM, model data menggambarkan spesifikasi tabel-tabel dan hubungannya dengan tabel lain sacara logis sementara PDM menggambarkan spesifikasi tabel-tabel yang mendekati implementasinya ke dalam Sistem Manajemen Basis Data Relasional.

##### Tabel Modul

Tabel ini merupakan tabel utama pada fitur modularitas dari perangkat lunak. Tabel ini berisi informasi yang dibutuhkan perangkat lunak pada proses pendaftaran modul. Pendaftaran modul disesuaikan dengan kerangka kerja Spring MVC yaitu dengan menggunakan satu *servlet* untuk tiap modul yang kemudian membentuk *Hierarchical* MVC sehingga kolom pada tabel Modul juga disesuaikan dengan atribut sebuah *servlet* yaitu nama *servlet*, konfigurasi *servlet*, dan *URL mapping servlet*. Tabel ini memiliki hubungan dengan tabel menu dimana suatu modul diwajibkan memiliki setidaknya satu menu.

##### Tabel Menu

Tabel ini merupakan tabel yang menyimpan informasi menu yang disediakan oleh modul. Tabel ini memiliki kolom nama menu dan *URL* menu yang digunakan sebagai komponen navigasi pada antarmuka pengguna. Tabel ini memiliki hubungan dengan tabel Modul dan tabel Menu Peran.

##### Tabel Peran

Tabel ini menyimpan data macam-macam peran yang dapat dimiliki seorang pengguna perangkat lunak *web*. Peran dalam perangkat lunak ini digunakan sebagai pembatas akses seorang pengguna ke dalam menu pada modul perangkat lunak. Tabel ini berhubungan dengan tabel Peran Pengguna yang menyatakan hak akses pengguna dan dengan tabel Menu Peran yang membatasi akses dari suatu menu.

##### Tabel Menu Peran

Tabel ini merupakan tabel yang menyimpan data peran apa saja yang bisa mengkases suatu menu pada modul.

##### Tabel Pengguna

Tabel ini menyimpan data yang digunakan oleh seorang pengguna untuk melakukan proses autentikasi ke dalam perangkat lunak. Tabel ini memiliki hubungan dengan tabel Peran Pengguna yang berfungsi sebagai hak akses pengguna, tabel Tipe Pengguna yang membedakan pengguna sebagai peserta didik atau pendidik dan tenaga kependidikan, tabel Satuan Manajemen sebagai identitas satuan manajemen dari pengguna, tabel Peserta Didik sebagai identitas pengguna jika pengguna tersebut adalah peserta



Gambar 3.46 Model Data Konseptual



Gambar 3.47 Model Data Fisik

didik, dan tabel Pendidik dan Tenaga Kependidikan sebagai identitas pengguna jika pengguna tersebut adalah pendidik atau tenaga kependidikan.

##### Tabel Tipe Pengguna

Tabel ini digunakan untuk membantu perangkat lunak membedakan apakah pengguna adalah peserta didik atau pendidik dan tenaga kependidikan.

##### Tabel Peran Pengguna

Tabel ini berisi informasi peran untuk satuan manajemen apa saja yang dimiliki seorang pengguna. Tabel ini digunakan sebagai hak akses seorang pengguna ke dalam perangkat lunak.

##### Tabel Pendidik dan Tenaga Kependidikan

Tabel ini merupakan tabel referensi data pendidik dan tenaga kependidikan yang berhubungan dengan tabel pengguna. Tabel ini berisi informasi pendidik dan tenaga kependidikan sebagai pengguna perangkat lunak.

##### Tabel Peserta Didik

Tabel ini merupakan tabel referensi data peserta didik yang berhubungan dengan tabel pengguna. Tabel ini berisi informasi peserta didik sebagai pengguna perangkat lunak.

#### Perancangan antarmuka pengguna

Subbab ini akan membahas perancangan antarmuka pengguna untuk memenuhi semua kasus penggunaan yang telah dirancang pada subbab sebelumnya. Antarmuka pengguna pada fitur admin menggunakan *template* halaman admin yang telah dibahas pada subbab 3.1.6.

##### Halaman pengunggahan modul

Halaman ini digunakan pengguna untuk melakukan pengunggahan modul. Pemilihan modul dilakukan dengan menekan tombol cari modul. Ketika tombol tersebut ditekan, perangkat lunak akan mengeluarkan dialog untuk mencari berkas modul. Pencarian berkas modul dibatasi hanya berkas jar. Setelah berkas modul dipilih, pengguna diwajibkan menekan tombol unggah yang akan mengunggah modul yang telah dipilih ke perangkat lunak. Setelah modul diunggah, modul sekaligus diaktifkan dan pengguna akan diarahkan ke halaman penambahan akses menu. Rancangan antarmuka pengguna halaman ini dapat dilihat pada Gambar 3.48 dengan komponen antarmuka pengguna yang tertera pada Tabel 3.12.

##### Halaman penambahan akses menu

Halaman penambahan akses menu merupakan halaman kemana pengguna diarahkan setealah mengunggah modul. Halaman ini digunakan untuk memetakan menu dengan peran. Menu yang dimaksud adalah menu pada modul yang baru saja diunggah. Pengguna memilih peran yang ingin dipetakan dengan menu kemudian memilih menu apa saja yang ingin dipetakan ke peran tersebut dan melakukan hal yang sama untuk peran lain. Setelah pemetaan dilakukan pengguna wajib menekan tombol simpan untuk menyimpan hasil pemetaan. Rancangan antarmuka pengguna halaman ini dapat dilihat pada Gambar 3.49 dengan komponen antarmuka pengguna yang tertera pada Tabel 3.13.

##### Halaman daftar modul

Halaman daftar modul berisi daftar modul pada perangkat lunak. Daftar modul tersebut digambarkan dalam sebuah tabel. Pada tiap barus modul terdapat tombol untuk menghapus modul pada baris yang sama. Pengguna dapat mengakses halaman pengunggahan modul melalui tombol tambah modul pada halaman ini. Rancangan antarmuka pengguna halaman ini dapat dilihat pada Gambar 3.50 dengan komponen antarmuka pengguna yang tertera pada Tabel 3.14.

##### Halaman daftar pengguna

Halaman daftar pengguna berisi daftar pengguna yang terdaftar pada sistem. Daftar tersebut ditampilkan dalam bentuk tabel. Pada tiap baris tabel terdapat informasi pengguna, tombol hapus yang digunakan untuk menghapus pengguna pada baris yang sama, tombol ubah untuk mengubah informasi pengguna pada baris yang sama, tombol hak akses yang akan mengarahkan administrator ke halaman penambahan hak akses pengguna, dan checkbox yang digunakan untuk fitur penghapusan banyak pengguna.

Untuk melakukan penghapusan banyak pengguna, administrator perlu mengaktifkan *checkbox* untuk menandakan pengguna untuk dihapus kemudian menekan tombol hapus yang ditandai.

Administrator dapat melakukan pencarian pengguna melalui kotak teks yang disediakan. Pencarian dilakukan berdasarkan semua kolom pada tabel daftar pengguna.

Tombol tambah pengguna akan mengarahkan administrator ke halaman penambahan pengguna. Rancangan antarmuka pengguna halaman ini dapat dilihat pada Gambar 3.51 dengan komponen antarmuka pengguna yang tertera pada Tabel 3.15.

##### Halaman penambahan pengguna

Pada halaman penambahan pengguna, administrator diberi dua pilihan cara untuk menambah pengguna. Cara pertama adalah dengan menambah pengguna manual dengan mengisi formulir dan cara kedua adalah dengan mengunggah berkas CSV seperti yang dijelaskan pada Gambar 3.52 dan Gambar 3.53.

Penambahan melalui formulir dilakukan dengan menekan tombol tambah manual. Saat administrator menekan tombol tersebut, formulir akan ditampilkan dengan beberapa kotak teks yang diperlukan dan tombol simpan untuk menyimpan pengguna baru. Komponen formulir pada halaman ini dijelaskan pada Tabel 3.16.

Penambahan melalui berkas CSV dilakukan dengan menekan tombol unggah CSV. Saat administrator menekan tombol tersebut, formulir unggah akan ditampilkan dengan tombol untuk mencari berkas CSV yang diinginkan dan tombol unggah untuk mengunggah berkas yang telah dipilih ke *server*. Komponen formulir pengunggahan pada halaman ini dijelaskan pada Tabel 3.17.

##### Halaman pengubahan pengguna

Halaman ini digunakan untuk mengubah informasi pengguna. Pengubahan informasi yang dapat diubah adalah *username* dan satuan manajemen dari pengguna. Setelah perubahan dilakukan administrator menekan tombol simpan untuk menyimpan perubahan. Rancangan antarmuka pengguna halaman ini dapat dilihat pada Gambar 3.54 dengan komponen antarmuka pengguna yang tertera pada Tabel 3.18.

##### Halaman pengelolaan hak akses pengguna

Halaman ini digunakan administrator untuk menambah atau menghapus hak akses dari suatu pengguna. Pengguna yang hak aksesnya ingin dikelola, dipilih melalui halaman daftar pengguna. Pada halaman ini terdapat formulir untuk menambah hak akses, tombol untuk menyimpan dan daftar hak akses yang dimiliki pengguna yang dipilih. Rancangan antarmuka pengguna halaman ini dapat dilihat pada Gambar 3.55 dengan komponen antarmuka pengguna yang tertera pada Tabel 3.19.

##### Halaman daftar peran

Halaman ini berisi daftar peran pada sistem. Daftar tersebut berupa tabel dengan tombol untuk menghapus dan mengubah informasi peran pada tiap barisnya. Pada halaman ini terdapat tombol untuk menambah peran. Rancangan antarmuka pengguna halaman ini dapat dilihat pada Gambar 3.56 dengan komponen antarmuka pengguna yang tertera pada Tabel 3.20.

##### Halaman penambahan peran

Halaman ini digunakan untuk menambah peran pada sistem. Pada halaman ini terdapat satu kotak teks untuk mengisi nama peran dan satu tombol untuk menyimpannya. Halaman ini serupa dengan halaman pengubahan nama peran. Rancangan antarmuka pengguna halaman ini dapat dilihat pada Gambar 3.57 dengan komponen antarmuka pengguna yang tertera pada Tabel 3.21.

Tabel 3.11 Spesifikasi atribut antarmuka halaman *template* admin dan modul

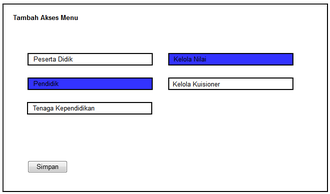
|  |  |  |
| --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan |
| modulList | *List* | Menampilkan daftar modul |
| menuList | *Dropdown list* | Menampilkan daftar menu pada modul |
| menuBody |  | Menampilkan halaman menu |



Gambar 3.48 Halaman penambahan modul

Tabel 3.12 Spesifikasi atribut antarmuka penambahan modul

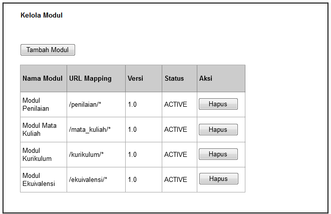
|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| modulInput | *File Input* | Input *berkas* untuk mengunggah modul | *File* |
| uploadBtn | *Button Submit* | Mengunggah berkas ke sistem | *POST Request* |



Gambar 3.49 Halaman penambahan akses menu

Tabel 3.13 Spesifikasi atribut antarmuka penambahan akses menu

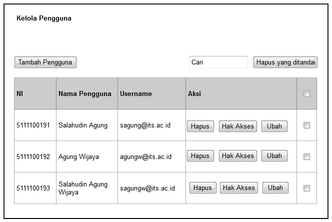
|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| roleItem | *Button* | Memilih peran yang ingin dipetakan ke menu | *String* |
| menuItem | *Button* | Memilih menu yang ingin dipetakan ke peran yang telah dipilih | *String* |
| saveBtn | *Button*  *Submit* | Menyimpan akses menu ke sistem | *POST Request* |



Gambar 3.50 Halaman daftar modul

Tabel 3.14 Spesifikasi atribut antarmuka daftar modul

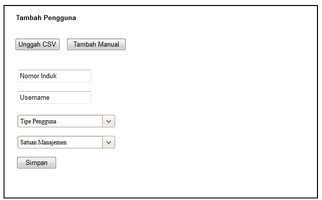
|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| moduleTbl | *Table* | Menampilkan daftar informasi modul |  |
| deleteBtn | *Button*  *Submit* | Menghapus modul | *POST Request* |
| addBtn | *Button* | Menampilkan halaman penambahan modul | *GET Request* |



Gambar 3.51 Halaman daftar pengguna

Tabel 3.15 Spesifikasi atribut antarmuka daftar pengguna

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| userTbl | *Table* | Menampilkan daftar informasi pengguna |  |
| deleteBtn | *Button*  *Submit* | Menonaktifkan pengguna | *POST Request* |
| addBtn | *Button* | Menampilkan halaman penambahan pengguna | *GET Request* |
| roleBtn | *Button* | Menampilkan halaman pengelolaan hak akses pengguna | *GET Request* |
| editBtn | *Button* | Menampilkan halaman pengubahan informasi pengguna | *GET Request* |
| searchBox | *TextBox* | Mencari pengguna | *String* |
| checkBox | *CheckBox* | Menandai pengguna untuk proses penghapusan | *Bool* |
| massDeleteBtn | *Button Submir* | Menghapus pengguna yang ditandai | *POST Request* |



Gambar 3.52 Halaman penambahan pengguna manual

Tabel 3.16 Spesifikasi atribut antarmuka penambahan pengguna manual

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| niBox | *TextBox* | Mengisi nomor induk pengguna | *String* |
| usernameBox | *TextBox* | Mengisi username | *String* |
| typeBox | *ComboBox* | Mengisi tipe pengguna | *String* |
| satManBox | *ComboBox* | Mengisi satuan manajemen pengguna | *String* |
| saveBtn | *Button Submit* | Menyimpan pengguna | *POST Request* |



Gambar 3.53 Halaman penambahan pengguna dengan unggah CSV

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| saveBtn | *Button Submit* | Mengunggah berkas CSV | *POST Request* |
| fileInput | *Input File* | Mencari berkas yang ingin diunggah | *File* |

Tabel 3.17 Spesifikasi atribut antarmuka penambahan pengguna dengan unggah CSV



Gambar 3.54 Halaman pengubahan informasi pengguna

Tabel 3.18 Spesifikasi atribut antarmuka pengubahan informasi pengguna

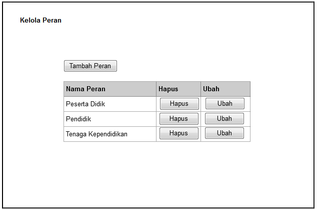
|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| saveBtn | *Button Submit* | Menyimpan perubahan informasi pengguna | *POST Request* |
| usernameBox | *TextBox* | Mengisi perubahan username | *String* |
| satManBox | *ComboBox* | Mengisi perubahan satuan manajemen pengguna | *String* |



Gambar 3.55 Halaman pengelolaan hak akses pengguna

Tabel 3.19 Spesifikasi atribut antarmuka pengelolaan hak akses pengguna

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| saveBtn | *Button Submit* | Menyimpan penambahan hak akses pengguna | *POST Request* |
| usernameBox | *TextBox* | Menampilkan username dari pengguna. Isi *Textbox* ini tidak dapat diubah. | *String* |
| peranBox | *ComboBox* | Mengisi peran | *String* |
| satManBox | *ComboBox* | Mengisi satuan manajemen | *String* |
| userRoleTbl | *Table* | Menampilkan hak akses yang dimiliki pengguna |  |
| deleteBtn | *Button Submit* | Menghapus hak akses pengguna | *POST Request* |



Gambar 3.56 Halaman daftar peran

Tabel 3.20 Spesifikasi atribut antarmuka daftar peran

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| addBtn | *Button* | Menampilkan halaman penambahan peran | *GET Request* |
| roleTbl | *Table* | Menampilkan daftar peran |  |
| deleteBtn | *Button Submit* | Menghapus peran | *POST Request* |
| editBtn | *Button* | Menampilkan halaman penggantian nama peran | *GET Request* |



Gambar 3.57 Halaman penambahan peran

Tabel 3.21 Spesifikasi atribut antarmuka penambahan peran

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| roleBox | *TextBox* | Mengisi nama peran | *String* |
| saveBtn | *Button Submit* | Menyimpan peran | *POST Request* |

# BAB IV IMPLEMENTASI

Pada bab ini akan dijelaskan tahap-tahap implementasi dari perangkat lunak utama berdasarkan perancangan yang telah dibahas pada bab sebelumnya dan aturan pembuatan modul perangkat lunak Sistem Informasi Akademik. Penjelasan aturan pembuatan modul akan disertai langkah-langkah pembuatan modul. Selain itu juga akan dibahas mengenai persiapan lingkungan pengembangan yang mendukung OSGi, langkah-langkah pengembangan perangkat lunak web dengan Spring MVC, dan langkah-langkah *refactoring* dari suatu perangkat lunak *web* standard dengan Spring MVC agar mendukung lingkungan OSGi.

Pada bab ini, implemetasi akan dijelaskan berdasarkan lingkup utama dari perangkat lunak, yaitu modularitas, akses data, *template* antarmuka pengguna, dan keamanan, dan fitur administratif.



## Lingkungan Pengembangan

Berikut kakas bantu yang digunakan pada proses implementasi perangkat lunak ini:

* + - 1. Windows 7 Ultimate 64 bit sebagai sistem operasi
      2. Eclipse Luna 4.4.2 dengan plugin Spring Tool Suite sebagai *Integrated Development Environment* (IDE)
      3. Java 7 sebagai pustaka Java dan lingkungan *Runtime*.
      4. Maven sebagai *dependency* *manager*
      5. PostgreSQL 9.3 sebagai sistem manajemen basis data
      6. Eclipse Virgo 3.6.3.RELEASE sebagai OSGi *Runtime* dengan beberapa kakas bantu didalam nya sebagai berikut:
* Eclipse Equinox sebagai implementasi spesifikasi OSGi
* Apache Tomcat 7.0 sebagai *Web Server*

### Persiapan lingkungan dan implementasi kerangka kerja OSGi

Di antara semua persiapan kakas bantu yang digunakan, persiapan lingkungan OSGi merupakan yang paling rumit. Persiapan lingkungan OSGi yang dilakukan yaitu pemasangan web server yang mendukung OSGi dan pemasangan dependensi dari perangkat lunak beserta masalah yang sering terjadi pada kedua proses tersebut.

Pada awal pengembangan, kerangka kerja OSGi yang dipilih adalah Apache Felix. Alasan utama yang membuat penulis pada awalnya memilih kerangka kerja tersebut adalah karena kurangnya pengetahuan penulis tentang OSGi dan adanya fitur pengelolaan lingkup OSGi dalam bentuk *web*. Setelah mengalami beberapa masalah dan kerumitan pada proses pemenuhan dependensi, lingkungan OSGi diubah ke Eclipse Virgo.

#### Apache Tomcat dalam lingkungan OSGi

Karena perangkat lunak merupakan *bundle* OSGi yang dijalankan di dalam lingkup OSGi, Apache Tomcat perlu dijalankan di dalam OSGi. Untuk menjalankan Tomcat di dalam OSGi diperlukan Tomcat yang berupa berkas jar. Terdapat beberapa berkas jar yang diperlukan dan berkas jar tersebut sangat sulit dicari. Penulis hanya dapat menemukan berkas jar Tomcat versi 6 yang merupakan versi yang sudah lama tertinggal sehingga digunakanlah Tomcat versi 7 yang mendukung OSGi yang sudah terintegrasi dengan Eclipse Virgo.

#### Pemasangan dependensi

Dependensi perangkat lunak merupakan pustaka yang diperlukan suatu perangkat lunak baik pada saat proses kompilasi, *testing*, ataupun *runtime*. Dependensi pada OSGi berupa berkas bundle. Dependensi awal yang perlu disediakan pada lingkungan OSGi adalah Spring MVC.

Salah satu tantangan dalam pemenuhan dependensi pada OSGi adalah bagaimana mengubah status suatu *bundle* dari *INSTALLED* menjadi *RESOLVED*. Setiap *bundle* yang memerlukan sebuah pustaka dari bundle lain (dependensi) perlu menambahkan properti *Import*-*Package* pada berkas *manifest* nya. Pada saat *bundle* dipasang ke dalam sistem OSGi, sistem akan mencari *bundle* lain yang memiliki *package* yang diperlukan. *Bundle* yang menyediakan *package* yang dibutuhkan tersebut perlu menambahkan properti *Export*-*Package* yang bernilai *package* yang diperlukan oleh *bundle* sebelumnya. Jika tidak ada *bundle* yang mengekspor *package* tersebut, maka *bundle* yang mengimpor *package* tersebut tidak dapat bernilai *RESOLVED*. Adapun kasus yang terjadi setidaknya pada saat pemenuhan dependensi untuk tugas akhir ini yaitu konflik dependensi antar *bundle*.

Berikut contoh kasus konflik dependensi. Ada dua buah *bundle* dengan properti sebagai berikut:

*Bundle-SymbolicName: bundleA*

*Export-Package: org.pck.foo;version=”1.0”*

*Import-Package: org.pck.bar;version=”1.0”*

*Bundle-SymbolicName: bundleB*

*Export-Package: org.pck.bar;version=”1.0”*

Pada kondisi seperti ini *bundleA* dan *bundleB* dapat dipasang. Namun ketika ada *bundle* lain dengan properti berikut:

*Bundle-SymbolicName: bundleC*

*Import-Package:* *org.pck.foo;version=”1.0”, org.pck.bar;version=”2.0”*

*Bundle-SymbolicName: bundleD*

*Export-Package: org.pck.bar;version=”2.0”*

*bundleA* dan *bundleC* tidak dapat dipasang karena adanya konflik dimana *bundleA* menggunakan *package* *org.pck.bar* versi 1.0 dari *bundleB* dan *bundleC* menggunakan *package* *org.pck.bar* versi 2.0 dari *bundleD*. Masalah ini dapat diselesaikan dengan memaksakan agar *bundleC* mengimpor *package* dari *bundleB* agar dependensi konsisten. Namun, dengan solusi ini pengembang perlu mengubah berkas MANIFEST.MF pada bundle dan jika pohon dependensi terlalu besar maka solusi ini akan banyak memakan waktu.

Konflik dependensi yang terjadi pada pengembangan perangkat lunak ini antara lain:

* Komponen *servlet* pada *bundle javax.servlet* dan Spring MVC. Solusi yang dilakukan penulis adalah mengganti lingkungan dan kerangka kerja OSGi ke Eclipse Virgo yang sudah menyediakan Spring MVC dan *javax*.*servlet* dengan versi yang sesuai.
* Pustaka untuk membaca berkas xml yaitu *javax*.*xml*.*parsers* yang konflik dengan pustaka yang sudah disediakan Java. Solusi yang dilakukan adalah dengan mengubah versi *javax.xml.parsers* ke versi yang sudah disediakan Java dan menghapus bundle dengan versi yang berbeda.

Tantangan lain agar dependensi proyek terpenuhi adalah mencari pustaka yang berbentuk *bundle*. Seperti yang telah dijelaskan pada subbab 2.5.1, *bundle* merupakan berkas jar atau war yang memiliki properti khusus pada berkas MANIFEST.MF-nya. Tanpa properti tersebut berkas jar tidak mendukung OSGi. Pustaka dapat dicari melalui *Maven Repository* [19], namun banyak pustaka pada repository tersebut yang belum mendukung OSGi. Selain *Maven Repository* [19], beberapa berkas pustaka yang mendukung OSGi dapat ditemukan pada *Spring Enterprise Bundle Repository* [20] walaupun terbatas. Jika pustaka tidak ditemukan pada *repository* tersebut, penulis terpaksa mencari kode sumber dari pustaka tersebut dan melakukan *refactoring* agar pustaka tersebut mendukung OSGi. Beberapa pustaka yang di-*refactor* yaitu pustaka *org.jadira* dan *libphonenumber* yang digunakan pada beberapa kelas entitas.

Masalah terakhir pada pemasangan *bundle* untuk pemenuhan dependensi yaitu pemenuhan dependensi opsional dari suatu *bundle* yang sudah berstatus *RESOLVED.* Pemenuhan dependensi opsional tidak dapat dilakukan jika *bundle* sudah berstatus *RESOLVED*. Solusi yang dapat dilakukan adalah memasang *bundle* tersebut bersamaan dengan dependensinya atau dengan memasang dependensinya terlebih dahulu sebelum memasang *bundle*. Kasus ini terjadi pada saat pemasangan *hibernate* yang merupakan dependensi opsional dari Spring ORM dan pustaka *Jackson* yang merupakan dependensi opsional dari Spring MVC. Untuk dapat melakukan solusi tersebut, tentunya kedua *bundle* Spring harus dihapus dari lingkungan OSGi, namun hal itu tidak dapat dilakukan karena pada Eclipse Virgo, kedua *bundle* Spring tersebut merupakan *bundle* *default* yang akan selalu ada dalam lingkungan OSGi. Untuk itu pemasangan *hibernate* dan *Jackson* dilakukan bersamaan pada saat lingkungan OSGi pertama kali dijalankan. Cara tersebut dilakukan dengan membuat berkas XML dengan ekstensi “.plan” untuk *hibernate* dan *Jackson*. Kode sumber XML tersebut ditunjukkan pada Kode Sumber 4.1 dan Kode Sumber 4.2.

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<plan** name="hibernate.plan" version="4.2.18.Final" scoped="false" atomic="false"
3. xmlns="http://www.eclipse.org/virgo/schema/plan"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5. xsi:schemaLocation="
6. http://www.eclipse.org/virgo/schema/plan
7. http://www.eclipse.org/virgo/schema/plan/eclipse-virgo-plan.xsd"**>**
9. **<artifact** type="bundle" name="com.springsource.org.dom4j" version="1.6.1"**/>**
10. **<artifact** type="bundle" name="com.springsource.antlr" version="2.7.7"**/>**
11. **<artifact** type="bundle" name="com.springsource.org.apache.tools.ant" version="1.8.3"**/>**
12. **<artifact** type="bundle" name="com.springsource.org.jboss.jandex" version="1.0.3.Final"**/>**
13. **<artifact** type="bundle" name="classmate" version="0.5.4"**/>**
14. **<artifact** type="bundle" name="org.jboss.logging.jboss-logging" version="3.1.0.GA"**/>**
15. **<artifact** type="bundle" name="org.hibernate.common.hibernate-commons-annotations" version="4.0.2.Final"**/>**
16. **<artifact** type="bundle" name="javassist" version="3.18.1.GA"**/>**
17. **<artifact** type="bundle" name="javax.persistence" version="2.0.0.v200911271158"**/>**
18. **<artifact** type="bundle" name="org.jboss.spec.javax.transaction.jboss-transaction-api\_1.1\_spec" version="1.0.1.Final"**/>**
19. **<artifact** type="bundle" name="org.hibernate.core" version="4.2.18.Final"**/>**
20. **</plan>**

Kode Sumber 4.1 Berkas XML hibernate.plan

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<plan** name="jackson.plan" version="1.9.10" scoped="false" atomic="false"
3. xmlns="http://www.eclipse.org/virgo/schema/plan"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5. xsi:schemaLocation="
6. http://www.eclipse.org/virgo/schema/plan
7. http://www.eclipse.org/virgo/schema/plan/eclipse-virgo-plan.xsd"**>**
9. **<artifact** type="bundle" name="jackson-core-asl" version="1.9.10"**/>**
10. **<artifact** type="bundle" name="jackson-mapper-asl" version="1.9.10"**/>**
12. **</plan>**

Kode Sumber 4.2 Berkas XML jackson.plan

Kedua berkas *plan* yang sudah dibuat, diletakkan pada *folder* “/repository/usr/” pada *folder* instalasi Eclipse Virgo. Kedua berkas tersebut berisi *bundle* yang ingin dipasang beserta dependensinya ke dalam lingkungan OSGi. Daftar *bundle* pada berkas tersebut sudah diurutkan sehingga pengembang dapat merujuk ke berkas tersebut untuk menentukan *bundle* apa yang perlu dipasang terlebih dahulu jika pengembang tidak menggunakan fitur berkas plan. Setelah berkas tersebut diletakkan pada *folder* “/repository/usr/”, perlu dibuat sebuah berkas *plan* lain yang diberi nama “sia.plan”. berkas *plan* ini berisi *plan* apa saja yang ingin dibaca sekaligus semua dependensi dari perangkat lunak Sistem Informasi Akademik. Berkas tersebut kemudian diletakkan pada *folder* “/pickup” pada *folder* instalasi Eclipse Virgo. Kode sumber berkas XML ditunjukkan pada Kode Sumber 4.3.

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<plan** name="sia.plan" version="1.0.0" scoped="false" atomic="false"
3. xmlns="http://www.eclipse.org/virgo/schema/plan"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5. xsi:schemaLocation="
6. http://www.eclipse.org/virgo/schema/plan
7. http://www.eclipse.org/virgo/schema/plan/eclipse-virgo-plan.xsd"**>**
9. **<artifact** type="plan" name="commons-dbcp.plan" version="2.1.0"**/>**
10. **<artifact** type="plan" name="hibernate.plan" version="4.2.18.Final"**/>**
11. **<artifact** type="plan" name="jackson.plan" version="1.9.10"**/>**
12. **<artifact** type="plan" name="spring-osgi.plan" version="1.2.1"**/>**
13. **<artifact** type="plan" name="spring-security.plan" version="3.1.7.RELEASE"**/>**
14. **<artifact** type="bundle" name="com.springsource.com.opensymphony.sitemesh" version="2.4.1"**/>**
15. **<artifact** type="bundle" name="org.postgresql.jdbc4" version="9.4.0.build-1201"**/>**
16. **<artifact** type="bundle" name="org.apache.commons.logging" version="1.2.0"**/>**
17. **<artifact** type="bundle" name="com.springsource.net.sf.cglib" version="2.2.0"**/>**
18. **<artifact** type="bundle" name="com.springsource.org.apache.xmlcommons" version="1.3.4"**/>**
19. **<artifact** type="plan" name="jasperreports.plan" version="5.0.1"**/>**
20. **<artifact** type="plan" name="sia-modul.plan" version="1.0"**/>**
22. **</plan>**

Kode Sumber 4.3 Berkas XML sia.plan

## Implementasi Fungsionalitas Administratif

Fungsionalitas administratif diimplementasi menggunakan pola arsitektur *multilayer* yang memisahkan perangkat lunak menjadi beberapa *layer* secara fisik yaitu *layer* *web*, *service*, *DAO*, *domain*, dan *plugin*. *Layer*-*layer* tersebut di-*deploy* ke dalam lingkungan OSGi untuk dapat saling berhubungan.

Kode sumber yang disertakan pada bab ini diberi nomor baris yang sesuai dengan kode sumber asli.

### *Web* *layer*

*Web* *layer* diimplementasi menggunakan Spring MVC. Pada *layer* ini terdapat kelas-kelas dan komponen yang membentuk pola arsitektur MVC dan kelas-kelas yang berhubungan dengan pertukaran data dari *client* ke perangkat lunak ataupun sebaliknya. Selain itu, terdapat kelas *ServiceLocator<T>* dan berkas xml yang diberi nama “sia-web-osgi.xml” untuk mengambil *service* yang disediakan *layer* lain. Kelas-kelas tersebut dibungkus ke dalam tigas *package* yaitu *Controllers*, JSON *Model* dan OSGi.

#### *Package* *Controllers*

*Package* ini, tentunya, berisi kelas-kelas *controller* yang membentuk perangkat lunak Spring MVC. Pada perangkat lunak ini, *package* diberi nama “com.sia.main.web.controllers”. Berikut kelas-kelas *controller* tersebut. Kelas-kelas *controller* difokuskan hanya berfungsi untuk pertukaran data dari *web* *client* ke perangkat lunak dan sebaliknya.

##### Kelas *ModuleController*

Kelas ini merupakan kelas yang bertugas menangani *Http* *Request* yang berhubungan dengan pengelolaan modul. Pengelolaan modul meliputi pengunggahan modul, penghapusan modul, dan pengelolaan akses menu modul. Kode sumber fungsi-fungsi pada kelas *ModuleController* ditunjukkan pada Kode Sumber 4.4, Kode Sumber 4.5, dan Kode Sumber 4.6.

1. @RequestMapping(value = "/uploadWizard/1/upload", method = RequestMethod.POST)
2. **public** ModelAndView uploadModule(@RequestParam("file") Object file, HttpSession session) {
3. ModelAndView modelAndView = **new** ModelAndView();
4. Bundle hostBundle = FrameworkUtil.getBundle(**this**.getClass());
5. Response response = **this**.modulService.installModule(file, hostBundle);
6. **if**(response.getData() != **null**) {
7. session.setAttribute("moduleOnWizard", (Modul)response.getData());
8. modelAndView.setViewName("redirect:/admin/module/uploadWizard/2");
9. } **else** {
10. session.setAttribute("uploadFailed", response);
11. modelAndView.setViewName("redirect:/admin/module/uploadWizard/1");
12. }
13. **return** modelAndView;
14. }

Kode Sumber 4.4 Fungsi pengunggahan berkas modul

1. @RequestMapping(value = "/uploadWizard/2/submit",  method = RequestMethod.POST)
2. @ResponseBody
3. **public** Response saveMenus(@RequestBody RoleMenu[] roleMenus, HttpSession session) {
4. Modul modul = (Modul)session.getAttribute("moduleOnWizard");
5. **for**(Menu menu: modul.getMenus()) {
6. **for**(MenuPeran menuPeran: menu.getDaftarMenuPeran()) {
7. **this**.menuPeranService.delete(menuPeran);
8. }
9. }
10. List<MenuPeran> failures = **new** ArrayList<MenuPeran>();
11. **for**(RoleMenu roleMenu: roleMenus) {
12. Peran peran = **this**.peranService.getById(UUID.fromString(roleMenu.getRoleId()));
13. **for** (String menuId : roleMenu.getRoleMenus()) {
14. Menu menu = **this**.menuService.getById(UUID.fromString(menuId));
15. MenuPeran menuPeran = **new** MenuPeran(**null**, peran, menu);
16. MenuPeran result = (MenuPeran)**this**.menuPeranService.insertInto(menuPeran).getData();
17. **if**(result == **null**)
18. failures.add(menuPeran);
19. }
20. }
21. session.setAttribute("moduleOnWizard", **this**.modulService.getById(modul.getIdModul()));
22. **if**(failures.size() > 0) {
23. StringBuilder sb = **new** StringBuilder();
24. sb.append("Penambahan gagal pada: \n");
25. **for**(MenuPeran menuPeran: failures) {
26. sb.append("- Hak akses peran " + menuPeran.getPeran() + " untuk menu " + menuPeran.getMenu() + ".\n");
27. }
28. **return** **new** Response(Response.error, sb.toString(), **null**);
29. } **else** {
30. **return** **new** Response(Response.ok, "Penambahan hak akses menu pada peran berhasil.", **null**);
31. }
32. }

**Kode Sumber 4.5 Fungsi penambahan akses menu modul**

1. @RequestMapping(value = "/delete", method = RequestMethod.POST)
2. **public** @ResponseBody Response deleteModule(@RequestParam("idModul") String idModul) {
3. Modul modul = modulService.getById(UUID.fromString(idModul));
4. Response response = **this**.modulService.uninstallModule(modul);
5. **return** response;
6. }

**Kode Sumber 4.6 Fungsi penghapusan modul**

Dapat dilihat fungsi *uploadModule* dan *saveMenus* memiliki *RequestMapping* “uploadWizard”. Fungsionalitas pengunggahan modul dan penambahan akses menu memang sengaja dibuat menjadi sebuah w*i*zard dengan tujuan memaksa pengguna, dalam kasus ini admin, agar segera menambah akses menunya agar modul dapat segera digunakan.

##### Kelas *UserController*

Kelas ini merupakan kelas yang bertugas menangani *Http* *Request* yang berhubungan dengan pengelolaan pengguna. Pengelolaan pengguna meluputi penambahan pengguna, menonaktifkan pengguna, dan menyunting informasi pengguna. Penambahan pengguna dibagi menjadi dua fungsi yaitu fungsi *submitUser* untuk menambah pengguna satu per satu dan *uploadCSV* untuk menambah pengguna secara masal dengan pengunggahan berkas CSV. Hal ini diperlukan untuk memudahkan penambahan pengguna. Kode sumber fungsi-fungsi pada kelas ini ditunjukkan pada Kode Sumber 4.7.

1. @RequestMapping(value = "/add/", method = RequestMethod.POST)
2. **public** ModelAndView submitUser(@RequestParam("ni") String ni, @RequestParam("username") String username, @RequestParam("tipe") String tipe, @RequestParam("satMan") String satMan) {
3. ModelAndView modelAndView = **new** ModelAndView();
4. SatMan newSatMan = **this**.satuanManajemenDAO.getByParam("where nmSatMan = '" + satMan + "'").get(0);
5. Pd pd = **null**;
6. Ptk ptk = **null**;
7. **if**(tipe.toLowerCase().equals("pd")) {
8. pd = **this**.pdDAO.getByParam("where niPd = '" + ni + "'").get(0);
9. } **else** {
10. ptk = **this**.ptkDAO.getByParam("where niPtk = '" + ni + "'").get(0);
11. }
12. TipePengguna tp = **this**.tipePenggunaDAO.getByParam("where namaTipe = '" + tipe + "'").get(0);
13. Pengguna user = **new** Pengguna(UUID.randomUUID(), pd, ptk, newSatMan, username, ni, **true**, "", **null**, tp);
14. **this**.penggunaService.insertInto(user);
15. modelAndView.setViewName("redirect:/admin/user/");
16. **return** modelAndView;
17. }
19. @RequestMapping(value = "/add/upload", method = RequestMethod.POST)
20. **public** ModelAndView uploadCSV(@RequestParam("file") Object file) {
21. ModelAndView modelAndView = **new** ModelAndView();
22. MultipartFile multipartFile = (MultipartFile) file;
23. BufferedReader  br = **null**;
24. **try** {
25. File newFile = **new** File(multipartFile.getOriginalFilename());
26. multipartFile.transferTo(newFile);
27. String line = "";
28. br = **new** BufferedReader(**new** FileReader(newFile));
29. **while** ((line = br.readLine()) != **null**) {
30. String[] userData = line.split(",");
31. SatMan satMan = **this**.satuanManajemenDAO.getByParam("where nmSatMan = '" + userData[3] + "'").get(0);
32. Pd pd = **null**;
33. Ptk ptk = **null**;
34. **if**(userData[2].toLowerCase().equals("pd")) {
35. pd = **this**.pdDAO.getByParam("where niPd = '" + userData[0] + "'").get(0);
36. } **else** {
37. ptk = **this**.ptkDAO.getByParam("where niPtk = '" + userData[0] + "'").get(0);
38. }
39. TipePengguna tp = **this**.tipePenggunaDAO.getByParam("where namaTipe = '" + userData[2] + "'").get(0);
40. Pengguna user = **new** Pengguna(UUID.randomUUID(), pd, ptk, satMan, userData[1], userData[0], **true**, "", **null**, tp);
41. **this**.penggunaService.insertInto(user);
42. }
44. } **catch** (FileNotFoundException e) {
45. e.printStackTrace();
46. } **catch** (IllegalStateException e1) {
47. e1.printStackTrace();
48. } **catch** (IOException e) {
49. e.printStackTrace();
50. } **finally** {
51. **if** (br != **null**) {
52. **try** {
53. br.close();
54. } **catch** (IOException e) {
55. e.printStackTrace();
56. }
57. }
58. }
59. modelAndView.setViewName("redirect:/admin/user/");
60. **return** modelAndView;
61. }

**Kode Sumber 4.7 Fungsi penambahan pengguna**

##### Kelas *UserRoleController*

Kelas ini merupakan kelas yang bertugas menangani *Http* *Request* yang berhubungan dengan pengelolaan hak akses pengguna. Hak akses pengguna dikelola per pengguna. Pengelolaan ini meliputi penambahan dan penghapusan hak akses pengguna. Kode sumber fungsi-fungsi pada kelas ini ditunjukkan pada

Kode Sumber **4.8** dan Kode Sumber 4.9.

1. @RequestMapping(value = "/add", method = RequestMethod.POST)
2. **public** ModelAndView add(@RequestParam("id") UUID userId, @RequestParam("peran") String namaPeran, @RequestParam("satMan") String namaSatMan, HttpSession session) {
3. ModelAndView modelAndView = **new** ModelAndView();
4. Pengguna pengguna = **this**.penggunaDAO.getById(userId);
5. Peran peran = **this**.peranDAO.getByParam("where namaPeran = '" + namaPeran + "'").get(0);
6. SatMan satMan = **this**.satManDAO.getByParam("where nmSatMan = '" + namaSatMan + "'").get(0);
7. PeranPengguna peranPengguna = **new** PeranPengguna(**null**, pengguna, peran, satMan);
8. **if**(peranPenggunaService.insertInto(peranPengguna) != **null**) {
9. Response response = **new** Response("success", "peran pengguna berhasil ditambah", **null**);
10. session.setAttribute("addResponse", response);
11. } **else** {
12. Response response = **new** Response("error", "peran pengguna gagal ditambah", **null**);
13. session.setAttribute("addResponse", response);
14. }
15. modelAndView.setViewName("redirect:/admin/userRole?userId=" + userId);
16. **return** modelAndView;
17. }

**Kode Sumber 4.8 Fungsi penambahan hak akses pengguna**

1. @RequestMapping(value = "/delete", method = RequestMethod.POST)
2. **public** ModelAndView delete(@RequestParam("id") UUID peranPenggunaId, HttpSession session) {
3. ModelAndView modelAndView = **new** ModelAndView();
4. PeranPengguna peranPengguna = **this**.peranPenggunaService.getById(peranPenggunaId);
5. **if**(**this**.peranPenggunaService.delete(peranPengguna) != **null**) {
6. Response response = **new** Response("success", "peran pengguna berhasil dihapus", **null**);
7. session.setAttribute("addResponse", response);
8. } **else** {
9. Response response = **new** Response("error", "peran pengguna gagal dihapus", **null**);
10. session.setAttribute("addResponse", response);
11. }
12. modelAndView.setViewName("redirect:/admin/userRole?userId=" + peranPengguna.getPengguna().getIdPengguna());
13. **return** modelAndView;
14. }

**Kode Sumber 4.9 Fungsi penghapusan hak akses pengguna**

##### Kelas *RoleController*

Kelas ini merupakan kelas yang bertugas menangani *Http* *Request* yang berhubungan dengan pengelolaan peran. Peran dapat ditambah dan dihapus. Kode sumber kedua fungsi tersebut ditunjukkan pada

Kode Sumber **4.10** dan Kode Sumber 4.11.

1. @RequestMapping(value = "/add", method = RequestMethod.POST)
2. **public** ModelAndView tambahPeran(@RequestParam("namaPeran") String namaPeran, HttpSession session) {
3. ModelAndView modelAndView = **new** ModelAndView();
4. Response response;
5. Peran peran = **new** Peran();
6. peran.setNamaPeran(namaPeran);
7. Peran res = peranService.insertInto(peran);
8. **if**(res != **null**) {
9. response = **new** Response("ok", "peran berhasil ditambahkan", res);
10. } **else** {
11. response = **new** Response("error", "peran gagal ditambahkan", **null**);
12. }
13. session.setAttribute("addResponse", response);
14. modelAndView.setViewName("redirect:/admin/role/");
15. **return** modelAndView;
16. }

**Kode Sumber 4.10 Fungsi penambahan peran**

1. @RequestMapping(value = "/delete", method = RequestMethod.POST)
2. **public** ModelAndView hapusPeran(@RequestParam("idPeran") String idPeran, HttpSession session) {
3. ModelAndView modelAndView = **new** ModelAndView();
4. Response response;
5. Peran peran = peranService.getById(UUID.fromString(idPeran));
6. Peran res = peranService.delete(peran);
7. **if**(res != **null**) {
8. response = **new** Response("ok", "peran " + peran.getNamaPeran() + " berhasil dihapus", res);
9. } **else** {
10. response = **new** Response("error","peran " + peran.getNamaPeran() + " gagal dihapus", **null** );
11. }
12. session.setAttribute("addResponse", response);
13. modelAndView.setViewName("redirect:/admin/role/");
14. **return** modelAndView;
15. }

**Kode Sumber 4.11 Fungsi penghapusan peran**

#### Package JSON *Model*

*Package* ini berisi kelas-kelas yang digunakan sebagai tipe data dari pertukaran data melalui JSON pada beberapa fungsi pada *controller*. *Package* ini diberi nama “com.sia.main.web.json\_model”. Untuk mendukung JSON pada Java, diperlukan pustaka *Jackson*. Pemasangan pustaka *Jackson* sudah disertakan pada subbab 4.1.1.2.

##### Kelas *Role*

Kelas ini merepresentasikan peran yang digunakan pada fungsi pengelolaan peran. Kode sumber kelas ini ditunjukka pada Kode Sumber 4.12.

1. **public** **class** Role {
3. **private** UUID roleId;
5. **private** String roleName;
7. // setters, getters, and constructor
9. }

**Kode Sumber 4.12 Kelas *Role***

##### Kelas *RoleMenu*

Kelas ini merepresentasikan akses menu yang digunakan pada fungsi penambahan akses menu modul. Kode sumber kelas ini ditunjukkan pada Kode Sumber 4.13.

1. **public** **class** RoleMenu {
3. **private** String roleId;
5. **private** String[] roleMenus;
7. // setters, getters, and constructor
9. }

**Kode Sumber 4.13 Kelas *RoleMenu***

##### Kelas *SatMan*

Kelas ini merepresentasikan data satuan manajemen yang digunakan pada fungsi pengelolaan pengguna dan hak akses pengguna. Kode sumber kelas ini ditunjukkan pada

Kode Sumber **4.14**.

1. **public** **class** SatMan {
3. **private** String idSatMan;
5. **private** String namaSatMan;
7. // setters, getters, and constructor
9. }

**Kode Sumber 4.14 Kelas SatMan**

#### *Package* OSGi dan OSGi *Service Reference*

Subbab ini menjelaskan *package* yang berisi kelas *ServiceLocator<T>* yang berfungsi untuk mengambil *service* dari *layer* lain melalui OSGi *Service Registry*. *Package* ini diberi nama “com.sia.main.web.osgi”. Selain itu pengambilan *service* melalui OSGi *Service Registry* juga dapat dilakukan dengan Gemini Blueprint. Gemini Blueprint merupakan kakas bantu berbentuk berkas XML untuk mengambil *service* pada OSGi *Service Registry* dengan metode *Dependency Injection*. Kedua cara tersebut memang terlihat redundan. Namun, Kelas *ServiceLocator<T>* tetap diperlukan saat pendaftaran *servlet* modul pada saat perangkat lunak sedang mulai dijalankan karena Blueprint hanya dapat dijalankan ketika perangkat lunak sudah berjalan. *Layer* ini hanya mengambil OSGi *service* dari *layer* lain tanpa mendaftarkan OSGi *service* apapun. Kode sumber kelas *ServiceLocator<T>* ditunjukkan pada

Kode Sumber **4.15**. Contoh berkas XML ditunjukkan pada Kode Sumber 4.16.

1. @SuppressWarnings({ "unchecked", "rawtypes" })
2. **public** **class** ServiceLocator<T> {
4. **private** ServiceTracker serviceTracker;
6. **private** Class<T> serviceClass;
8. **public** ServiceLocator() {
9. }
11. **public** ServiceLocator(BundleContext bundleContext, Class<T> serviceClass) {
12. **this**.serviceClass = serviceClass;
13. **this**.serviceTracker = **new** ServiceTracker(bundleContext, **this**.serviceClass.getName(), **null**);
14. **this**.serviceTracker.open();
15. }
17. **public** T getService(**long** timeout) {
18. T service = **null**;
19. **try** {
20. service = **this**.serviceClass.cast(timeout == -1 ? **this**.serviceTracker.getService() : **this**.serviceTracker.waitForService(timeout));
21. } **catch**(Exception e) {
22. e.printStackTrace();
23. }
24. **return** service;
25. }
27. **public** **void** close() {
28. **this**.serviceTracker.close();
29. }
31. }

**Kode Sumber 4.15 Kelas *ServiceLocator<T>***

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<beans** xmlns="http://www.springframework.org/schema/beans"
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4. xmlns:context="http://www.springframework.org/schema/context"
5. xmlns:osgi="http://www.eclipse.org/gemini/blueprint/schema/blueprint"
6. xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
7. http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd
8. http://www.eclipse.org/gemini/blueprint/schema/blueprint http://www.eclipse.org/gemini/blueprint/schema/blueprint/gemini-blueprint.xsd"**>**
10. **<osgi:reference** id="modulServiceRef" interface="com.sia.main.service.services.ModulService" availability="mandatory"**>**
11. **</osgi:reference>**
13. **<osgi:reference** id="penggunaDAO" interface="com.sia.main.data.dao.PenggunaDAO" availability="mandatory"**>**
14. **</osgi:reference>**
16. **</beans>**

**Kode Sumber 4.16 berkas XML Blueprint**

#### Antarmuka pengguna

Implementasi antarmuka pengguna dilakukan dengan menggunakan JSP sebagai tipe berkas *view* dan pustaka JSTL untuk mengakses *model* yang dikirim dari *controller*.

##### Halaman pengunggahan modul

Halaman ini merupakan halaman yang digunakan untuk pengunggahan berkas modul. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.1.

##### Halaman penambahan akses menu

Halaman ini merupakan halaman untuk memetakan hak akses dengan menu-menu yang ada pada modul yang sedang dipasang. Halaman ini merupakan halaman lanjutan dari halaman pengunggahan berkas modul. Kedua halaman tersebut dibuat menjadi suatu *wizard*. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.2.

##### Halaman daftar modul

Halaman ini merupakan halaman yang menampilkan daftar modul yang terpasang pada Sistem Informasi Akademik. Pada halaman ini terdapat tombol yang mengarahkan pengguna ke halaman pengunggahan modul. Tiap baris pada daftar modul, terdapat tombol untuk menghapus modul dan tombol untuk mengakses halaman penambahan akses menu modul yang dipilih dari daftar modul. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.3.

##### Halaman daftar pengguna

Halaman ini menampilkan daftar pengguna yang masih aktif pada Sistem Informasi Akademik. Pada halaman ini terdapat beberapa komponen yaitu tombol untuk mengakses penambahan pengguna, tombol untuk menghapus pengguna satu per satu ataupun banyak pengguna, penanda pengguna untuk fitur penghapusan banyak pengguna, *filter* pengguna berdasarkan satuan manajemen dan tipe pengguna, Kotak teks untuk mencari pengguna, dan tabel daftar pengguna. Implementasi ini dibantu dengan pustaka jQuery datatable. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.4.

##### Halaman penambahan pengguna

Halaman penambahan pengguna dibagi menjadi dua yaitu penambahan secara manual ataupun dengan berkas CSV. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.5 dan Gambar 4.6.

##### Halaman penyuntingan informasi pengguna

Halaman ini sebenarnya diimplementasi bukan sebagai halaman sendiri melainkan dalam bentuk *modal*. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.7.

##### Halaman pengelolaan hak akses pengguna

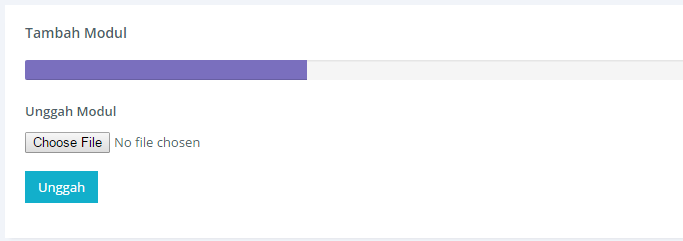
Halaman ini digunakan untuk melihat, menambah, dan menghapus hak akses yang dimiliki seorang pengguna. Halaman ini diakses melalui halaman daftar pengguna pada tiap baris dari tabel pengguna. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.8.

##### Halaman daftar peran

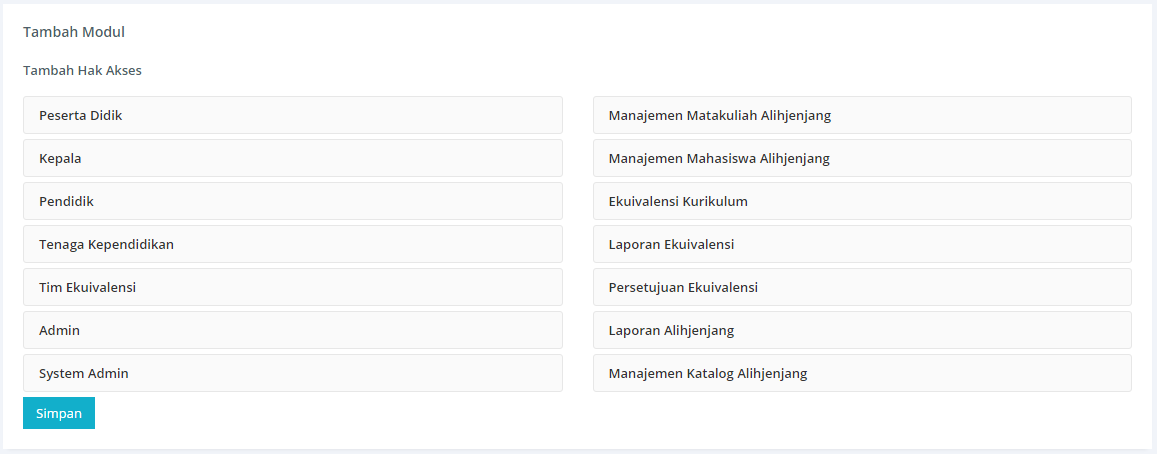
Halaman ini berisi tabel daftar peran pada sistem. Tiap baris pada tabel terdapat informasi peran dan tombol untuk menghapus peran dan menyunting informasi peran. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.10.

##### Halaman penambahan peran

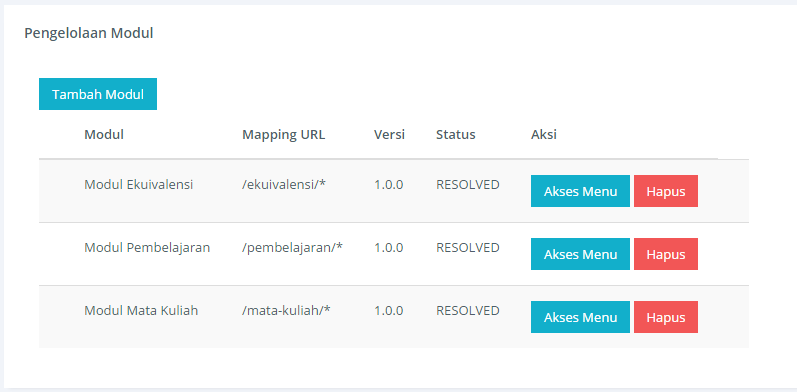
Halaman ini berbentuk *modal* yang berfungsi untuk menambah peran pada sistem. Hasil implementasi halaman ini ditunjukkan pada Gambar 4.9.



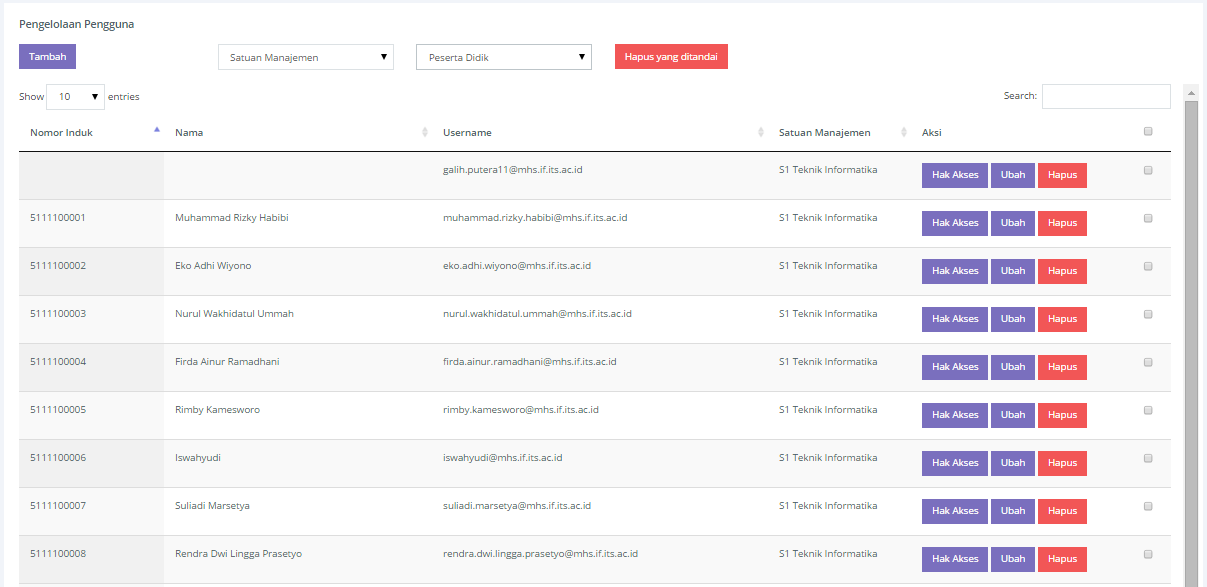
Gambar 4.1 Halaman penambahan modul



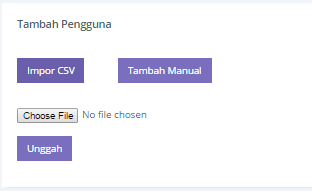
Gambar 4.2 Halaman penambahan akses menu



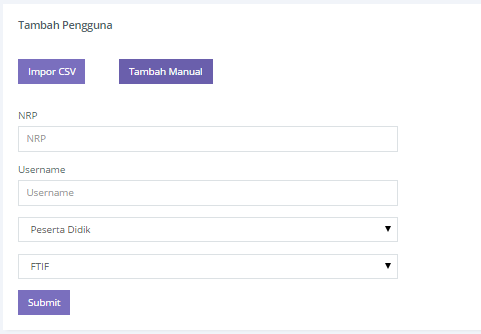
Gambar 4.3 Halaman daftar modul



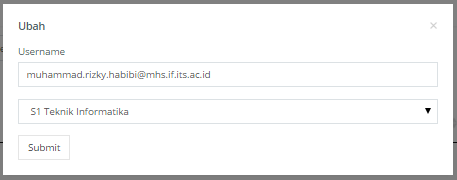
Gambar 4.4 Halaman daftar pengguna



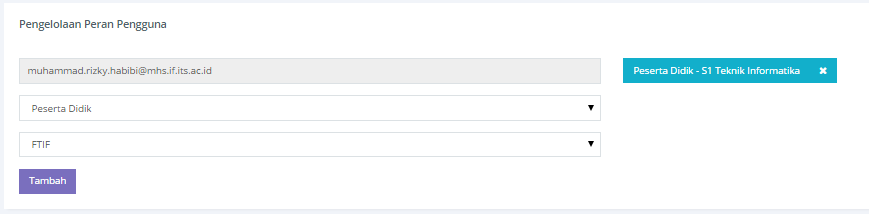
Gambar 4.5 Halaman pengunggahan berkas CSV



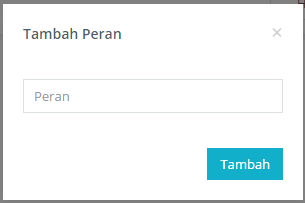
Gambar 4.6 Halaman penambahan pengguna



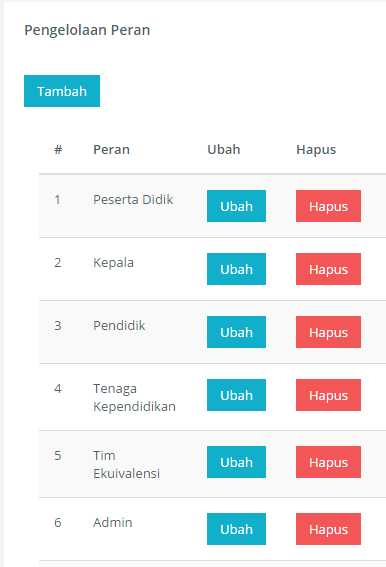
Gambar 4.7 Halaman penyuntingan informasi pengguna



Gambar 4.8 Halaman pengelolaan hak akses pengguna



Gambar 4.9 Halaman penambahan peran



Gambar 4.10 Halaman daftar peran

### *Service* *layer*

*Service* *layer* merupakan *layer* yang berisi kelas-kelas yang mengimplementasi logika proses bisnis yang dibagi berdasarkan kelas entitas. Selain itu juga terdapat kelas-kelas yang menangani pembacaan berkas modul. Kelas-kelas tersebut dikelompokkan menjadi tiga *package*. Yaitu *package* yang berisi kelas *interface* service, *package* yang berisi kelas realisasi dari kelas *interface* pada package sebelumnya, dan *package* *module* yang berisi kelas untuk membaca informasi modul.

Selain *package*, pada layer ini juga terdapat berkas Blueprint untuk mendaftarkan dan mengambil *service* ke dan dari dalam OSGi *Service Registry*. *Service* yang didaftarkan merupakan kelas-kelas realisasi dari kelas interface *service*. *Service* yang diambil merupakan *service* yang didaftarkan dari layer DAO.

#### *Package* *Service*

*Package* ini diberi nama “com.sia.main.service.services”. *Package* ini berisi kelas interface *service* yang dibagi berdasarkan kelas entitas. Kelas-kelas tersebut adalah:

* *ModulService*
* *MenuService*
* *MenuPeranService*
* *PenggunaService*
* *PeranService*
* *PeranPenggunaService*
* *SatuanManajemenService*

Berikut adalah

Kode Sumber **4.17** yang merupakan salah satu contoh kelas *interface* dari layer ini.

1. **public** **interface** ModulService {
2. **public** Response installModule(Object modulFile, Object hostBundle);
4. **public** Response uninstallModule(Modul modul);
6. **public** Modul update(Modul modul);
8. **public** List<Modul> getAll();
10. **public** Modul getById(UUID idModul);
12. **public** List<Modul> getByParam(String queryParam);
13. }

**Kode Sumber 4.17 Kelas *interface ModulService***

#### Package Service Implementation

*Package* ini diberi nama “com.sia.main.service.services.impl”. *Package* ini berisi kelas-kelas realisasi dari kelas *interface* *service* pada package “com.sia.main.service.services” yang dibagi berdasarkan kelas entitas. Kelas-kelas tersebut adalah:

* *ModulServiceImpl*
* *MenuServiceImpl*
* *MenuPeranServiceImpl*
* *PenggunaServiceImpl*
* *PeranServiceImpl*
* *PeranPenggunaServiceImpl*
* *SatuanManajemenServiceImpl*

Berikut adalah

Kode Sumber **4.18** yang merupakan fungsi-fungsi penambahan modul pada kelas *ModulServiceImpl* yang direalisasikan dari kelas *ModulService*.

1. @Override
2. **public** Response installModule(Object modulFile, Object hostBundleObj) {
3. **try** {
4. MultipartFile multipartFile = (MultipartFile) modulFile;
5. ModuleWriter moduleToDiskWritter = **new** ModuleWriter();
6. moduleToDiskWritter.writeToDisk(multipartFile, **this**.temporaryModuleLocation);
7. File tempFile = moduleToDiskWritter.getFile(**this**.temporaryModuleLocation, multipartFile.getOriginalFilename());
8. Bundle hostBundle = (Bundle) hostBundleObj;
9. Modul modul = **this**.saveModule(tempFile.getAbsolutePath(), hostBundle);
10. moduleToDiskWritter.writeToDisk(multipartFile, **this**.installedModuleLocation);
11. moduleToDiskWritter.deleteFromDisk(tempFile.toPath());
12. **return** **new** Response(Response.ok, "Modul berhasil ditambah", modul);
13. } **catch** (Exception e) {
14. e.printStackTrace();
15. **return** **new** Response(Response.error, "Modul gagal ditambah. " + e.getMessage(), **null**);
16. }
17. }
19. **private** Modul saveModule(String filePath, Bundle hostBundle) **throws** Exception {
20. Modul res = **null**;
21. BundleContext bundleContext = FrameworkUtil.getBundle(**this**.getClass()).getBundleContext();
22. filePath = "file:" + filePath;
23. Bundle bundle = **null**;
24. **try** {
25. bundle = bundleContext.installBundle(filePath);
26. Bundle systemBundle = bundleContext.getBundle(0);
27. FrameworkWiring fw = systemBundle.adapt(FrameworkWiring.**class**);
28. ArrayList<Bundle> bundles = **new** ArrayList<Bundle>();
29. bundles.add(bundle);
30. fw.resolveBundles(bundles);
31. Modul modul = (Modul) **this**.moduleReader.readModule(bundle, hostBundle);
32. modul.setStatus(OSGiBundleStatus.valueOf(bundle.getState()).name());
33. modul.setOsgiBundleId(String.valueOf(bundle.getBundleId()));
34. modul.setNamaServlet("");
35. modul.setVersi(bundle.getVersion().toString());
36. res = **this**.insertInto(modul);
37. **if** (res != **null**) {
38. **for**(Menu menu : modul.getMenus()) {
39. menu.setModul(res);
40. **this**.menuService.insertInto(menu);
41. }
42. **for**(Menu menuOld : **this**.menuService.getByParam("where modul.idModul = '" + res.getIdModul() + "'")){
43. **boolean** found = **false**;
44. **for**(Menu menuNew : modul.getMenus()) {
45. **if**(menuOld.getNamaMenu().equals(menuNew.getNamaMenu())) {
46. found = **true**;
47. }
48. }
49. **if**(!found) {
50. **this**.menuService.delete(menuOld);
51. }
52. }
53. } **else** {
54. bundle.uninstall();
55. **throw** **new** Exception("Modul dengan nama " + modul.getNamaModul() + " sudah ada dan pemetaan URL " + modul.getURLMapping() + " sudah digunakan");
56. }
57. } **catch** (BundleException e) {
58. bundle.uninstall();
59. **throw** **new** BundleException("Instalasi modul ke dalam container OSGi gagal. Pesan Exception: " + e.getMessage());
60. } **catch** (Exception e) {
61. bundle.uninstall();
62. **throw** e;
63. }
64. **return** res;
65. }
67. **private** Modul insertInto(Modul modul) {
68. List<Modul> usedNameModules = **this**.getByParam("where namaModul = '" + modul.getNamaModul() + "'");
69. List<Modul> usedURLModules = **this**.getByParam("where URLMapping = '" + modul.getURLMapping() + "'");
70. **if** ((usedNameModules != **null** && usedNameModules.size() > 0) && (!usedNameModules.get(0).getURLMapping().equals(modul.getURLMapping())) && (usedURLModules != **null** && usedURLModules.size() > 0)) {
71. **return** **null**;
72. }
73. **if** ((usedNameModules == **null** || usedNameModules.size() == 0) && (usedURLModules == **null** || usedURLModules.size() == 0)) {
74. **this**.modulDAO.insert(modul);
75. **return** **this**.getByParam("where namaModul = '" + modul.getNamaModul() + "'").get(0);
76. } **else** {
77. **return** **this**.update(modul);
78. }
79. }

**Kode Sumber 4.18 Kelas *ModulServiceImpl***

#### *Package* *Module*

*Package* ini diberi nama “com.sia.main.service.module”. *Package* ini berisi kelas yang menangani pembacaan berkas detil modul. Kelas tersebut adalah *XmlModuleReaderBean* dan *ModuleXmlParseHandler*.

Kelas *XmlModuleReaderBean* bertugas untuk mengambil hasil dari pembacaan modul yang dilakukan oleh kelas *ModuleXmlParseHandler*. Detil modul yang didapat diperiksa kelengkapannya. Jika tidak lengkap, Kelas ini mengirim *Java* *Excpetion*.

Kelas *ModuleXmlParseHandler* bertugas membaca berkas detil modul. Pembacaan dilakukan tiap elemen pada xml yang ditemukan. Elemen pada xml pada berkas juga diperiksa ke-*valid*-an nya. Ke-*valid*-an yang dimaksud adalah peletakkan suatu elemen terhadap elemen lain.

Contoh fungsi pembacaan modul pada kelas *XmlModuleReaderBean* ditunjukkan pada

Kode Sumber **4.19**.

1. @Override
2. **public** Modul readModule(Object module, Object host) **throws** Exception {
3. Bundle moduleBundle = (Bundle) module;
4. Bundle hostBundle = (Bundle) host;
5. Modul modul = **null**;
6. **try** {
7. SAXParserFactory factory = SAXParserFactory.newInstance();
8. SAXParser saxParser = factory.newSAXParser();
9. String moduleDetailLocation = **this**.moduleDetailXmlPath + **this**.prefix + moduleBundle.getSymbolicName() + ".xml";
10. URL fileURL = hostBundle.getResource(moduleDetailLocation);
11. **if**(fileURL == **null**) {
12. **throw** **new** NullPointerException("Berkas xml tidak ditemukan");
13. }
14. ModuleXmlParseHandler parseHandler = **new** ModuleXmlParseHandler();
15. saxParser.parse(fileURL.openStream(), parseHandler);
16. modul = parseHandler.getGeneratedModule();
17. **if**(modul.getNamaModul() == **null** || modul.getNamaModul().equals("")) **throw** **new** NullPointerException("Nama modul tidak boleh kosong");
18. **if**(modul.getURLMapping() == **null** || modul.getURLMapping().equals("")) **throw** **new** NullPointerException("Pemetaan URL modul tidak boleh kosong");
19. **if**(modul.getLokasiKonfigServlet() == **null** || modul.getLokasiKonfigServlet().equals("")) **throw** **new** NullPointerException("Lokasi konfigurasi servlet modul tidak boleh kosong");
20. **if**(modul.getMenus() == **null** || modul.getMenus().size() == 0) **throw** **new** NullPointerException("Tidak ada menu terdaftar pada modul");
21. **else** {
22. **for**(Menu menu: modul.getMenus()) {
23. **if**(menu.getNamaMenu() == **null** || menu.getNamaMenu().equals("")) **throw** **new** NullPointerException("Nama menu tidak boleh kosong");
24. **if**(menu.getURLMenu() == **null** || menu.getURLMenu().equals("")) **throw** **new** NullPointerException("URL menu tidak boleh kosong");
25. }
26. }
27. **for**(String configLocation: modul.getServletConfigLocationList()) {
28. URL configLocURL = hostBundle.getResource(configLocation);
29. **if**(configLocURL == **null**) {
30. **throw** **new** NullPointerException("Konfigurasi servlet tidak ditemukan");
31. }
32. }
33. } **catch** (Exception e) {
34. **throw** e;
35. }
36. **return** modul;
37. }

**Kode Sumber 4.19 Kelas *XmlModuleReaderBean***

#### OSGi *Service Reference*

Sama seperti pada *layer* *web*, Pendaftaran dan pengambilan OSGi *service* dilakukan dengan menggunakan Blueprint. OSGi *service* yang didaftakan adalah obyek-obyek dari kelas realisasi *interface* dari *package* com.sia.main.service.services. OSGi *service* yang diambil merupakan OSGi *service* yang didaftarkan oleh *layer* DAO. Potongan kode sumber XML ini ditunjukka pada Kode Sumber 4.20.

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<beans** xmlns="http://www.springframework.org/schema/beans"
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4. xmlns:osgi="http://www.eclipse.org/gemini/blueprint/schema/blueprint"
5. xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
6. http://www.eclipse.org/gemini/blueprint/schema/blueprint http://www.eclipse.org/gemini/blueprint/schema/blueprint/gemini-blueprint.xsd"**>**
8. **<osgi:service** id="modulServiceServ" interface="com.sia.main.service.services.ModulService" ref="modulServiceBean"**/>**
10. **<osgi:reference** id="modulDAORef" interface="com.sia.main.data.dao.ModulDAO" availability="mandatory"**/>**
12. **</beans>**

**Kode Sumber 4.20 Berkas XML Blueprint untuk mengambil *service*** **dari komponen DAO *layer***

### DAO *layer*

Layer ini berisi dua package yang berisi kelas interface DAO dan kelas realisasinya. Kelas-kelas DAO bertugas menyimpan perubahan data dari perangkat lunak ke basis data ataupun sebaliknya.

#### *Package* DAO

*Package* ini diberi nama “com.sia.main.data.dao”. *Package* ini berisi kelas interface *DAO* yang dibagi berdasarkan kelas entitas. Kelas-kelas tersebut adalah:

* *GenericDAO*
* *BasicDAO*
* *ModulDAO*
* *MenuDAO*
* *MenuPeranDAO*
* *PenggunaDAO*
* *PeranDAO*
* *PeranPenggunaDAO*
* *SatuanManajemenDAO*

Berikut adalah

Kode Sumber **4.21** yang menunjukkan kelas *interface* *GenericDAO<T>.*

1. **public** **interface** GenericDAO<T> {
3. **public** **void** insert(T object);
5. **public** **void** update(T object);
7. **public** **void** delete(T object);
9. **public** List<T> getAll();
11. **public** T getById(UUID idObject);
13. **public** List<T> getByParam(String queryParam);
15. }

**Kode Sumber 4.21 Kelas *interface GenericDAO***

#### *Package* DAO *Implementation*

*Package* ini berisi kelas realisasi dari *package* sebelumnya. Kelas-kelas pada *package* ini yaitu:

* *ModulDAOImpl*
* *MenuDAOImpl*
* *MenuPeranDAOImpl*
* *PenggunaDAOImpl*
* *PeranDAOImpl*
* *PeranPenggunaDAOImpl*
* *SatuanManajemenDAOImpl*

Implementasi pada kelas-kelas ini pada dasarnya hampir sama yaitu menambah, mengubah, menghapus, dan membaca data dari basis data namun berbeda pada kelas entitas yang ditangani. Implementasi pada kelas ini menggunakan Hibernate.

#### OSGi *Service Reference*

Pendaftaran OSGi *service* pada *layer* ini juga menggunakan Blueprint namun *layer* ini tidak mengambil OSGi *service* melainkan hanya mendaftarkan.

Kode Sumber **4.22** menunjukkan XML Blueprint untuk mendaftarkan *service.*

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<beans** xmlns="http://www.springframework.org/schema/beans"
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4. xmlns:osgi="http://www.eclipse.org/gemini/blueprint/schema/blueprint"
5. xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
6. http://www.eclipse.org/gemini/blueprint/schema/blueprint http://www.eclipse.org/gemini/blueprint/schema/blueprint/gemini-blueprint.xsd"**>**
8. **<osgi:service** id="modulDAOServ" interface="com.sia.main.data.dao.ModulDAO" ref="modulDAO"**/>**
10. **</beans>**

**Kode Sumber 4.22 Berkas XML Blueprint pada DAO *layer***

### *Domain* *layer*

*Layer* ini berisi satu *package* yang diberi nama “com.sia.main.domain”. Dalam *package* tersebut terdapat kelas-kelas entitas yang atributnya dicocokan dengan kolom-kolom pada tabel pada basis data. Kelas-kelas tersebut dipetakan dengan tabel-tabel pada basis data dengan menggunakan Hibernate. Pemetaan dilakukan dengan menambah anotasi pada kelas dan atributnya. Contoh salah satu kelas entitas ditunjukkan oleh Kode Sumber 4.23.

1. @Entity
2. @Table(name = "modul")
3. **public** **class** Modul {
5. @Id
6. @GeneratedValue(generator = "uuid")
7. @GenericGenerator(name = "uuid", strategy = "uuid2")
8. @Type(type = "pg-uuid")
9. @Column(name = "id\_modul")
10. **private** UUID idModul;
12. @Column(name = "nama\_modul", nullable = **false**)
13. **private** String namaModul;
15. @Column(name = "URL\_mapping", nullable = **true**)
16. **private** String URLMapping;
18. @Column(name = "versi", nullable = **false**)
19. **private** String versi;
21. @Column(name = "status", nullable = **false**)
22. **private** String status;
24. @Column(name = "osgi\_bundle\_id", nullable = **false**)
25. **private** String osgiBundleId;
27. @Column(name = "lokasi\_konf\_servlet", nullable = **false**)
28. **private** String lokasiKonfigServlet;
30. @Column(name = "nama\_servlet", nullable = **false**)
31. **private** String namaServlet;
33. @Column(name = "gambar", nullable = **true**)
34. **private** **byte**[] gambar;
36. @Column(name = "icon", nullable = **true**)
37. **private** String namaIconTemplate;
39. @OneToMany(fetch = FetchType.LAZY, mappedBy = "modul")
40. **private** List<Menu> menus;
42. //constructor, setters, and getters
44. }

**Kode Sumber 4.23 Kelas entitas Modul**

## Pembuatan Perangkat Lunak Web Standard dengan Spring MVC

Pada subbab ini akan dibahas langkah-langkah yang mengembangkan perangkat lunak Spring MVC. Subbab ini diperlukan sebagai panduan dasar kepada pengembang lain yang akan melanjutkan pengembangan perangkat lunak ini.

Beberapa hal yang perlu disiapkan sebelum memulai panduan ini antara lain

* 1. Eclipse IDE dengan versi minimal 4.4.2
  2. Maven
  3. Apache Tomcat
  4. Koneksi Internet

### Pembuatan proyek baru

Pembuatan proyek baru dilakukan dengan membuat proyek maven. Hal tersebut dilakukan dengan cara berikut:

1. Klik File -> New -> Project -> Maven Project
2. Pilih maven-archetype-webapp. Perlu dicatat, pemilihan archetype maven diperlukan koneksi internet untuk pengunduhan struktur proyek. Jika terjadi kegagalan dalam mengunduh, hapus isi dari *folder* “C:\Users\{nama user}\.m2\repository\org\apache\maven\archetypes” dan ulangi langkah nomor 1.
3. Isi kolom Group Id, Artifact Id, dan Version. Biarkan kolom packaging kosong atau isi dengan nilai “war”

### Pom.xml

Berkas pom.xml merupakan berkas yang merepresentasikan proyek dalam bentuk xml. Pada tahap ini pengembang perlu menambahkan dependensi proyek. Penambahan dependensi dapat dilakukan dengan menamabah <dependency></dependency> untuk tiap pustaka dependensi ke dalam <dependencies></dependencies>. Dependensi-dependensi yang diperlukan dapat ditemukan pada web Maven Repository. Pada kasus perangkat lunak Spring MVC sederhana, dependensi yang diperlukan adalah Spring webmvc, Javax Servlet API, JSP API dan JSTL. Berikut dependensi tersebut dalam <dependency></dependency>.

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>3.1.1.RELEASE</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.0.1</version>

</dependency>

<dependency>

<groupId>javax.servlet.jsp</groupId>

<artifactId>jsp-api</artifactId>

<version>2.1</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

Setelah dependensi tersebut ditambah, pengembang perlu menentukan versi java yang digunakan pada proses kompilasi proyek. Untuk melakukan hal tersebut pengembang dapat menggunakan maven plugin.

Maven plugin dapat ditambahkan dengan menambah maven-compiler-plugin dengan properti versi Java yang diinginkan. beserta isinya ke dalam <plugins></plugins> yang berada pada <build></build>. Kode Sumber 4.24 menunjukkan kode untuk menambah *maven compiler plugin*.

1. **<plugin>**
2. **<groupId>**org.apache.maven.plugins**</groupId>**
3. **<artifactId>**maven-compiler-plugin**</artifactId>**
4. **<configuration>**
5. **<source>**1.7**</source>**
6. **<target>**1.7**</target>**
7. **</configuration>**
8. **</plugin>**

**Kode Sumber 4.24 *Maven compiler plugin***

Berikut adalah Kode Sumber 4.25 merupakan hasil akhir berkas pom.xml.

1. **<?xml** version="1.0"**?>**
2. **<project** xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"
3. xmlns="http://maven.apache.org/POM/4.0.0"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"**>**
5. **<modelVersion>**4.0.0**</modelVersion>**
6. **<groupId>**com.sia.main**</groupId>**
7. **<artifactId>**SpringWeb**</artifactId>**
8. **<version>**1.0**</version>**
9. **<artifactId>**sia-web**</artifactId>**
10. **<packaging>**war**</packaging>**
12. **<build>**
13. **<finalName>**sia-web**</finalName>**
15. **<plugins>**
16. **<plugin>**
17. **<groupId>**org.apache.maven.plugins**</groupId>**
18. **<artifactId>**maven-compiler-plugin**</artifactId>**
19. **<configuration>**
20. **<source>**1.7**</source>**
21. **<target>**1.7**</target>**
22. **</configuration>**
23. **</plugin>**
25. **<**/plugins**>**
27. **</build>**
29. **<dependencies>**
31. **<dependency>**
32. **<groupId>**org.springframework**</groupId>**
33. **<artifactId>**spring-webmvc**</artifactId>**
34. **<version>**3.1.1.RELEASE**</version>**
35. **</dependency>**
36. **<dependency>**
37. **<groupId>**javax.servlet**</groupId>**
38. **<artifactId>**javax.servlet-api**</artifactId>**
39. **<version>**3.0.1**</version>**
40. **</dependency>**
41. **<dependency>**
42. **<groupId>**javax.servlet.jsp**</groupId>**
43. **<artifactId>**jsp-api**</artifactId>**
44. **<version>**2.1**</version>**
45. **</dependency>**
46. **<dependency>**
47. **<groupId>**javax.servlet**</groupId>**
48. **<artifactId>**jstl**</artifactId>**
49. **<version>**1.2**</version>**
50. **</dependency>**
52. **</dependencies>**
54. **</project>**

**Kode Sumber 4.25 Berkas pom.xml**

### *Servlet* dan *Web Application Context*

*Servlet* merupakan komponen utama dalam perangkat lunak Java berbasis web. *Servlet* dibuat dan kemudian didaftarkan kedalam sebuah berkas xml bernama web.xml. Pada berkas web.xml, *servlet* didaftarkan dan dipetakan ke dalam suatu URL agar dapat diakses.

#### Konfigurasi *Servlet*

Pada Spring MVC *servlet* berbentuk obyek dengan kelas *DispatcherServlet*. Obyek *DispatcherServlet* dibuat dengan konfigurasi yang berbentuk berkas xml. Selain konfigurasi *servlet*, berkas tersebut juga akan mendaftakan konfigurasi untuk *root application context* pada perangkat lunak Spring MVC. Berikut langkah-langkah membuat konfigurasi *servlet*.

* 1. Buat *folder* “WEB-INF” pada *folder* src/main/webapp/ jika belum ada.
  2. Buat berkas xml pada *folder* WEB-INF yang telah dibuat. Pengembang dapat membuat *folder* tambahan pada WEB-INF untuk meletakkan berkas xml. Namai berkas dengan nama apapun.
  3. Berikut adalah Kode Sumber 4.26 yang merupakan contoh isi konfigurasi *servlet*.

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<beans:beans** xmlns="http://www.springframework.org/schema/mvc"
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4. xmlns:beans="http://www.springframework.org/schema/beans"
5. xmlns:context="http://www.springframework.org/schema/context"
6. xsi:schemaLocation="http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd
7. http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
8. http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd"**>**
10. <!-- DispatcherServlet Context: defines this servlet's request-processing infrastructure -->
12. <!-- Enables the Spring MVC @Controller programming model -->
13. **<annotation-driven** **/>**
15. <!-- Handles HTTP GET requests for /resources/\*\* by efficiently serving up static resources in the ${webappRoot}/resources directory -->
16. **<resources** mapping="/resources/\*\*" location="/resources/" **/>**
18. <!-- Resolves views selected for rendering by @Controllers to .jsp resources in the /WEB-INF/views directory -->
19. **<beans:bean** class="org.springframework.web.servlet.view.InternalResourceViewResolver"**>**
20. **<beans:property** name="prefix" value="/WEB-INF/views/" **/>**
21. **<beans:property** name="suffix" value=".jsp" **/>**
22. **</beans:bean>**
24. **<context:component-scan** base-package="com.sia.main.web.controllers" **/>**
26. **</beans:beans>**

**Kode Sumber 4.26 Contoh konfigurasi *servlet***

Baris 2 sampai 8 berguna untuk mendefiniskan lokasi pustaka xml. Baris 13 digunakan agar *servlet* dapat membaca kelas-kelas dengan anotasi @Controller. Baris 16 digunakan untuk memetakan *resource* pada perangkat lunak ke dalam URI sehingga dapat diakses oleh *web client*. Beris 19 sampai 22 adalah *bean* yang merepresentasikan obyek *InternalResourceViewResolver*. *Bean* ini adalah komponen yang menangani pengambilan berkas *view* yang dipanggil melalui @Controller. Baris 24 digunakan *servlet* untuk mencari kelas-kelas komponen untuk didaftarkan pada *servlet* pada *package* yang didefinisikan. Package yang berisi kelas *controller*, *service*, DAO wajib ditambahkan ke dalam berkas ini dengan elemen <context:component-scan>

#### Web.xml

Berkas web.xml digunakan untuk mendaftarkan *servlet* dan mendefinisikan *root application context*. Berkas web.xml dapat diletakkan pada *folder* WEB-INF. Contoh kode sumber berkas ini ditunjukkan pada Kode Sumber 4.27.

1. **<web-app** xmlns="http://java.sun.com/xml/ns/javaee"
2. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3. xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
4. http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"
5. version="2.5"**>**
7. **<listener>**
8. **<listener-class>**
9. org.springframework.web.context.ContextLoaderListener
10. **</listener-class>**
11. **</listener>**
12. **<context-param>**
13. **<param-name>**contextClass**</param-name>**
14. **<param-value>**org.springframework.web.servlet.DispatcherServlet**</param-value>**
15. **</context-param>**
16. **<context-param>**
17. **<param-name>**contextConfigLocation**</param-name>**
18. **<param-value>**
19. /WEB-INF/root-context.xml
20. **</param-value>**
21. **</context-param>**
22. **<servlet>**
23. **<servlet-name>**spring-web**</servlet-name>**
24. **<servlet-class>**
25. org.springframework.web.servlet.DispatcherServlet
26. **</servlet-class>**
27. **<init-param>**
28. **<param-name>**contextConfigLocation**</param-name>**
29. **<param-value>**/WEB-INF/spring-mvc-config.xml**</param-value>**
30. **</init-param>**
31. **<load-on-startup>**1**</load-on-startup>**
32. **</servlet>**
34. **<servlet-mapping>**
35. **<servlet-name>**spring-web**</servlet-name>**
36. **<url-pattern>**/**</url-pattern>**
37. **</servlet-mapping>**
39. **</web-app>**

**Kode Sumber 4.27 Contoh berkas web.xml**

Baris 7 sampai 21 digunakan untuk mendefinisikan *root application context*. Baris 23 sampai 33 merupakan definisi *servlet*. Baris 24 mendefiiniskan nama *servlet*. Baris 30 mendefinisikan lokasi dan nama berkas konfigurasi *servlet* yang telah dijelaskan pada subbab sebelumnya. Baris 35 sampai 47 mendefinisikan pemetaan *servlet* ke dalam URI.

### Controllers

Pada Spring MVC, *controller* adalah sebuah kelas dengan anotasi @Controller. Pada tiap kelas *controller*, terdapat fungsi yang dipetakan dengan alamat *request*. Fungsi pemetaan tersebut memiliki anotasi @RequestMapping. Contoh kelas *controller* ditunjukkan pada Kode Sumber 4.28.

1. **package** com.sia.main.web.controllers;
3. **import** org.springframework.stereotype.Controller;
4. **import** org.springframework.web.bind.annotation.RequestMapping;
5. **import** org.springframework.web.bind.annotation.RequestMethod;
6. **import** org.springframework.web.servlet.ModelAndView;
8. @Controller
9. **public** **class** ExampleController {
11. @RequestMapping(value = "/", method = RequestMethod.GET)
12. **public** ModelAndView test() {
13. ModelAndView mav = **new** ModelAndView();
14. mav.setViewName("page");
15. **return** mav;
16. }
18. }

**Kode Sumber 4.28 Contoh kelas *controller***

### View

View pada Spring MVC adalah berkas JSP. Berkas JSP dibuat seperti berkas HTML. Berkas JSP diletakkan pada WEB-INF/views/. Contoh berkas JSP dapat dilihat pada Kode Sumber 4.29.

1. **<**%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%**>**
3. <!DOCTYPE html**>**
4. **<html>**
5. **<head>**
6. **<title></title>**
7. **</head>**
8. **<body>**
9. **<form** action="/user" method="post"**>**
10. **<input** type="text" name="username"**>**
11. **<input** type="submit"**>**
12. **</form>**
13. **</body>**
15. **</html>**

**Kode Sumber 4.29 Berkas JSP**

### Pertukaran Data

Pertukaran data yang dimaksud pada langkah ini adalah pengiriman data dari *view* ke *controller* untuk diolah dan sebaliknya untuk ditampilkan. Pengiriman data dari *view* dapat berupa *request* paramenter, URL *parameter*, atau JSON. Pengiriman data dari *controller* ke *view* dapat dilakukan dengan *ModelAndView*.*addObject*() atau @ResponseBody.

#### @RequestParam

@RequestParam adalah anotasi yang menandakan suatu parameter yang dikirim *view* melalui HTTP *Request*. Data tersebut biasanya berasal dari *form* *submittion* atau melalui URL *variable*. Contoh penggunaan @RequestParam dapat dilihat pada Kode Sumber 4.30.

1. @RequestMapping(value = "/user", method = RequestMethod.GET)
2. **public** **void** showUser(@RequestParam(value = "username", required = **true**) String username) {
4. }

**Kode Sumber 4.30 Contoh penggunaan @RequestParam**

Pengiriman data dapat dilakukan dari *form* atau dari *request* *variable*. Contoh penggunaan @RequestParam dengan form ditunjukkan pada Kode Sumber 4.31.

1. **<form** action="/user" method="get"**>**
2. **<input** type="text" name="username"**>**
3. **<input** type="submit"**>**
4. **</form>**

**Kode Sumber 4.31 Contoh penggunaan @RequestParam dengan *form***

Parameter *value* dari @RequestParam diisi dengan properti name dari elemen *form*.

Contoh pengiriman data dengan *request* *variable* adalah dengan mengarahkan URI ke “/user?username=username@ac.id”.

#### @RequestBody dan JSON

@RequestBody adalah anotasi yang digunakan untuk mengirim data yang berupa JSON. Penggunaan @RequestBody harus disertakan @ResponseBody sebagai HTTP *Response*. Untuk menggunakan @RequestBody diperlukan pustaka Jackson. Tambahkan Jackson sebagai dependensi pada berkas pom.xml spserti pada Kode Sumber 4.32. Contoh penggunaan @RequestBody ditunjukkan pada Kode Sumber 4.33.

1. **<dependency>**
2. **<groupId>**org.codehaus.jackson**</groupId>**
3. **<artifactId>**jackson-mapper-asl**</artifactId>**
4. **<version>**1.9.10**</version>**
5. **</dependency>**

**Kode Sumber 4.32 Dependensi Jackson pada pom.xml**

1. @RequestMapping(value = "/user",  method = RequestMethod.POST)
2. @ResponseBody
3. **public** Response showUser(@RequestBody User[] users) {
4. **return** **new** Response();
5. }

**Kode Sumber 4.33 Contoh penggunaan @RequestBody**

#### @PathVariable

@PathVariable adalah anotasi untuk mengambil nilai pada URI. Contoh penggunaan @PathVariable ditunjukkan pada Kode Sumber 4.34.

1. @RequestMapping(value = "/user/{username}", method = RequestMethod.GET)
2. **public** **void** showUser(@PathVariable("username") String username) {
4. }

**Kode Sumber 4.34 Contoh penggunaan @PathVariable**

#### @ResponseBody

@ResponseBody adalah anotasi untuk mengirim data dari *controller* ke *view* melalui HTTP *Response*. Contoh penggunaan @ResponseBody ditunjukkan pada Kode Sumber 4.35.

1. @RequestMapping(value = "/user", method = RequestMethod.POST)
2. **public** @ResponseBody Response showUser(@RequestParam("username") String username) {
3. **return** response;
4. }

**Kode Sumber 4.35 Contoh penggunaan @ResponseBody**

#### *ModelAndView*

Pengiriman data dari *controller* ke *view* dapat dilakukan melalui fungis *addObject*() dari kelas *ModelAndView*. Fungsi ini menerima dua parameter yaitu nama data dan obyek data. Contoh penggunaan fungsi *addObject*() ditunjukkan pada Kode Sumber 4.36.

1. @RequestMapping(value = "/user", method = RequestMethod.GET)
2. **public** ModelAndView showUser(@RequestParam(value = "username", required = **true**) String username) {
3. ModelAndView modelAndView = **new** ModelAndView();
4. modelAndView.addObject("username", username);
5. **return** modelAndView;
6. }

**Kode Sumber 4.36 Contoh penggunaan fungsi *addObject*()**

#### JSP *Expression* *Language*

JSP *Expression* *Language* digunakan untuk mengakses data yang ada pada HTTP *Response*. Contoh penggunaan JSP *Expression* *Language* ditunjukkan pada Kode Sumber 4.37 dan Kode Sumber 4.38.

1. **public** ModelAndView showUser(@RequestParam(value = "username", required = **true**) String username) {
2. ModelAndView modelAndView = **new** ModelAndView();
3. modelAndView.addObject("username", username);
4. **return** modelAndView;
5. }

**Kode Sumber 4.37 Contoh penggunaan fungsi *addObject*() untuk diakses pada JSP**

1. **<**%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%**>**
3. <!DOCTYPE html**>**
5. **<html>**
6. **<head>**
7. **<title></title>**
8. **</head>**
9. **<body>**
10. **<h1>**${username}**</h1>**
11. **</body>**
13. **</html>**

**Kode Sumber 4.38 Contoh pengambilan data dengan *expression* *language***

#### JSTL

JSTL digunakan untuk memanipulasi data pada HTTP Response. Berikut adalah Kode Sumber 4.39 dan Kode Sumber 4.40 yang menunjukkan contoh penggunaan JSTL.

1. @RequestMapping(value = "/user", method = RequestMethod.GET)
2. **public** ModelAndView showUser() {
3. ModelAndView modelAndView = **new** ModelAndView();
4. List<String> usernames = **new** ArrayList<String>();
5. usernames.add("salahudin@if.its.ac.id");
6. usernames.add("agung@if.its.ac.id");
7. usernames.add("wijaya@if.its.ac.id");
8. modelAndView.addObject("username", usernames);
9. **return** modelAndView;
10. }

**Kode Sumber 4.39 Contoh penggunaan fungsi *addObject*() untuk ditampilkan dengan JSTL**

1. **<**%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%**>**
2. **<**%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%**>**
4. <!DOCTYPE html**>**
6. **<html>**
7. **<head>**
8. **<title></title>**
9. **</head>**
10. **<body>**
11. **<h1>**
12. **<c:forEach** items="${usernames}" var="username"**>**
13. **<h1>**${username}**</h1>**
14. **</c:forEach>**
15. **</h1>**
16. **</body>**
18. **</html>**

**Kode Sumber 4.40 Contoh penggunaan JSTL**

## *Refactoring* Perangkat Lunak Web Menjadi OSGi *Bundle*

Subbab ini menjelaskan *refactoring* yang diperlukan untuk membuat suatu perangkat lunak Spring MVC dapat di-*deploy* ke dalam OSGi. Tahap ini diperlukan sebagai panduan jika perangkat lunak ingin dikembangkan lebih lanjut.

### Perubahan dependensi pada pom.xml

Pada tiap dependensi pada berkas pom.xml ditambahkan elemen <scope>provided</scope>. Langkah ini opsional, namun sebaiknya dilakukan karena semua dependensi yang disertakan dalam berkas war tidak digunakan. Dependensi akan dipenuhi melalui OSGi. Dengan perubahan ini berkas *war* akan berukuran lebih kecil.

### Penambahan maven-bundle-plugin pada pom.xml

Untuk membuat aplikasi Spring MVC yang mendukung OSGi, aplikasi web standard perlu diubah menjadi *web* *bundle*. Pembuatan *web* *bundle* dilakukan dengan cara berikut:

* Menambahkan maven-bundle-plugin pada pom.xml seperti pada Kode Sumber 4.41

1. **<plugin>**
2. **<groupId>**org.apache.felix**</groupId>**
3. **<artifactId>**maven-bundle-plugin**</artifactId>**
4. **<version>**2.1.0**</version>**
5. **<extensions>**true**</extensions>**
6. **<configuration>**
7. **<supportedProjectTypes>**
8. **<supportedProjectType>**war**</supportedProjectType>**
9. **<supportedProjectType>**bundle**</supportedProjectType>**
10. **</supportedProjectTypes>**
11. **<instructions>**
12. **<Bundle-Classpath>**WEB-INF/classes**</Bundle-Classpath>**
13. **<Web-ContextPath>**springmvcosgi**</Web-ContextPath>**
14. **<Import-Library>**org.springframework.spring;version="3.1.0.RELEASE"**</Import-Library>**
15. **<Import-Bundle>**com.springsource.javax.servlet.jsp.jstl;version="1.2.0"**</Import-Bundle>**
16. **<Import-Package>**
17. org.eclipse.virgo.web.dm;version="3.6.3.RELEASE"
18. **</Import-Package>**
19. **</instructions>**
20. **</configuration>**
21. **<executions>**
22. **<execution>**
23. **<id>**bundle-manifest**</id>**
24. **<phase>**process-classes**</phase>**
25. **<goals>**
26. **<goal>**manifest**</goal>**
27. **</goals>**
28. **</execution>**
29. **</executions>**
30. **</plugin>**

**Kode Sumber 4.41 maven-bundle-plugin pada berkas pom.xml**

Nilai pada elemen<Web-ContextPath perlu disesuaikan dengan perangkat lunak web yang dibuat.

<Bundle-Classpath> berisi *classpath* dari bundle aplikasi spring mvc ini, dimana pada *classpath* terdapat *controller* yang sudah dibuat. Jika elemen ini tidak disertakan, *controller* tidak akan dipetakan oleh *servlet* dan tidak bisa dipanggil melalui *web* *browser*.

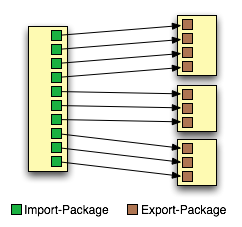
<Web-ContextPath> merupakan elemen yang berisi URL *mapping* untuk *web* *bundle* ini. Pada aplikasi ini, elemen <Web-ContextPath> diisi “springmvcosgi” sehingga untuk membuka web bundle ini url yang diperlukan adalah “[ip\_address]:[port\_tomcat]/springmvcosgi/”.

<Import-Package> merupakan elemen standard dari spesifikasi OSGi untuk menentukan dependensi *package* yang diperlukan dari *bundle* lain. Pada contoh ini, penulis menambahkan org.eclipse.virgo.web.dm versi 3.6.3.RELEASE dari OSGi *container* sebagai dependensi *package* yang diperlukan proyek ini. Package ini diperlukan untuk mengambil kelas *ServerOsgiBundleXmlWebApplicationContext* yang akan dibahas pada langkah Menambahkan OSGi *Application Context* pada web.xml.

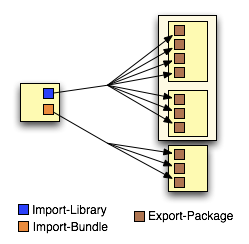
<Import-Bundle> merupakan elemen khusus spring *dynamic* *module* / gemiini blueprint untuk menentukan dependensi *bundle* yang diperlukan oleh *web* *bundle* ini. Pada dasarnya elemen ini mirip dengan elemen <Import-Package> namun elemen ini mengambil seluruh *package* pada *bundle* yang diperlukan. Pada kasus ini, penulis menambahkan *bundle* com.springsource.javax.servlet.jsp.jstl versi 1.2.0 dari OSGi *container* sebagai *bundle* yang diperlukan proyek .

<Import-Library> merupakan elemen khusus spring *dynamic* *module* / gemiini blueprint untuk menentukan kumpulan b*u*ndle yang diperlukan oleh *bundle*. Satu *library* menyatakan beberapa *bundle* yang termasuk dalam *library* tersebut. Pada contoh ini, penulis menambahkan *library* org.springframework.spring versi 3.1.0.RELEASE dari OSGi *container* sebagai *library* yang diperlukan proyek.

Berikut ilustrasi perbedaan antara Import-Package, Import-Bundle, dan Import-Library.



Gambar 4.11 Perbedaan Import-Package dan Export-Package



Gambar 4.12 Perbedaan Import-Library, Import-Bundle, dan Export-Package

* Mendefinisikan berkas manifest yang akan dimanipulasi oleh maven-bundle-plugin.

Langkah ini dilakukan dengan cara mendefinisikan ulang maven-war-plugin dan menambahkan lokasi berkas manifest proyek setelah di-*build* pada pom.xml. Pendefinisian maven-war-plugin dapat dilihat pada Kode Sumber 4.42.

1. **<plugin>**
2. **<groupId>**org.apache.maven.plugins**</groupId>**
3. **<artifactId>**maven-war-plugin**</artifactId>**
4. **<version>**2.1.1**</version>**
5. **<configuration>**
6. **<archive>**
7. **<manifestFile>**
8. ${project.build.outputDirectory}/META-INF/MANIFEST.MF
9. **</manifestFile>**
10. **</archive>**
11. **<failOnMissingWebXml>**false**</failOnMissingWebXml>**
12. **</configuration>**
13. **</plugin>**

**Kode Sumber 4.42 maven-war-plugin pada pom.xml**

### *ContextClass* pada *DispactherServlet*

Hal ini dilakukan agar aplikasi *web* dapat mengakses OSGi *BundleContext*. *BundleContext* diperlukan untuk mendaftarkan ataupun mengambil *Service* dari *Service* *Registry*.

Tambahkan kelas *ServerOsgiBundleWebApplicationContext* pada *ContextLoaderListener* atau pada *DispatcherServlet* seperti yang ditunjukkan pada Kode Sumber 4.43.

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<web-app** version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4. xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"**>**
6. **<context-param>**
7. **<param-name>**contextClass**</param-name>**
8. **<param-value>**org.eclipse.virgo.web.dm.ServerOsgiBundleXmlWebApplicationContext**</param-value>**
9. **</context-param>**
10. **<context-param>**
11. **<param-name>**contextConfigLocation**</param-name>**
12. **<param-value>**
13. /WEB-INF/spring/springmvcosgi/applicationContext.xml
14. **</param-value>**
15. **</context-param>**
17. **<listener>**
18. **<listener-class>**org.springframework.web.context.ContextLoaderListener**</listener-class>**
19. **</listener>**
21. **<servlet>**
22. **<servlet-name>**springmvcosgi**</servlet-name>**
23. **<servlet-class>**org.springframework.web.servlet.DispatcherServlet**</servlet-class>**
24. **<init-param>**
25. **<param-name>**contextClass**</param-name>**
26. **<param-value>**org.eclipse.virgo.web.dm.ServerOsgiBundleXmlWebApplicationContext**</param-value>**
27. **</init-param>**
28. **<init-param>**
29. **<param-name>**contextConfigLocation**</param-name>**
30. **<param-value>**
31. /WEB-INF/spring/springmvcosgi/appServlet-servlet.xml
32. **</param-value>**
33. **</init-param>**
34. **<load-on-startup>**1**</load-on-startup>**
35. **</servlet>**
37. **<servlet-mapping>**
38. **<servlet-name>**springmvcosgi**</servlet-name>**
39. **<url-pattern>**/**</url-pattern>**
40. **</servlet-mapping>**
42. **</web-app>**

**Kode Sumber 4.43 Berkas web.xml yang mendukung OSGi**

## Implementasi Modularitas

Pada bagian ini implementasi yang dijelaskan meliputi pembuatan modul beserta aturan yang perlu diimplementasi, pembacaan modul pada perangkat lunak utama, dan pendaftaran servlet dari modul sehingga MVC dari modul dapat dijalankan pada perangkat lunak utama.

### Pembuatan modul

Penjelasan cara pembuatan modul meliputi aturan apa saja yang diperlukan sebuah modul agar dapat digabungkan dengan perangkat lunak utama dan langkah-langkah pembuatan modul yang mengikuti aturan-aturan yang telah dijelaskan. Langkah-langkah pembuatan peragkat lunak dengan Spring MVC akan dijelaskan pada subbab lain. Berikut aturan-aturan yang perlu diimplementasikan pada modul.

Modul wajib di-*build* sebagai berkas jar.

Modul wajib di-*build* sebagai OSGi *fragment* *bundle* dari perangkat lunak utama dengan nama *symbolic* yang berbeda dengan modul lain. Alasan modul dibuat menjadi OSGi *fragment* *bundle* dan tidak standard OSGi *bundle* adalah karena adanya masalah yang ditemukan pada proses memanggil berkas *view* pada *controller*. Kelas *InternalResourceViewResolver* tidak bisa memanggil berkas *view* selain dari perangkat lunak utama. Dengan dibuatnya modul menjadi *fragment* *bundle*, berkas *view* modul akan seolah-olah berada pada perangkat lunak utama.

Modul wajib mengimplementasikan keamanan pada menunya menggunakan anotasi @Secured.

Jika ingin mengakses obyek *security*, modul dapat mengakses obyek *HttpSession* dan menambahkan dependensi sia-domain ke dalam proyeknya. Sia-domain merupakan domain *layer* dari perangkat lunak utama yang berisi kelas-kelas entitas dari fitur admin.

Modul diwajibkan memiliki konfigurasi *DispatcherServlet* yang berupa berkas xml yang setidaknya berisi lokasi komponen Spring MVC yang berupa *controller*, *service*, atau *repository*/DAO, komponen *ViewResolver*, dan komponen *resource* *mapping*.

Antarmuka pengguna menu hanya berisi badan menu. Komponen seperti menubar, header, dan footer disediakan oleh *template* pada perangkat lunak utama.

Berkas konfigurasi *DispatcherServlet* dan JSP tidak diletakkan pada *folder* “/WEB-INF/”.

Modul wajib memiliki berkas xml yang berisi properti detil dari modul. Berkas tersebut wajib diletakkan pada *folder* “/module-detail/” dengan nama berkas “module-” + nama *symbolic* modul sebagai OSGi *bundle* + “.xml”. Properti yang wajib dimiliki modul antara lain.

* Nama modul
* URL *mapping* yang berbeda dari modul lain
* Lokasi berkas konfigurasi *servlet*
* Daftar menu. Tiap menu memiliki properti:
* Nama menu
* URL menu

#### Pembuatan Proyek Maven Baru

Pembuatan proyek maven pada Eclipse dilakukan dengan cara berikut:

1. Kilk File -> New -> Project -> Maven Project
2. Centang “Create simple project (skip archetype selection)” karena proyek tidak memerlukan maven archetype apapun
3. Isi *group* id, *artifact* id, versi, dan *packaging*. Form lain opsional.

Disarankan untuk mengisi *Group* Id dengan “com.sia.modul” dan *Artifact* Id dengan nama modul agar seragam. *Packaging* dari proyek diwajibkan untuk diisi “jar”.

1. Klik “Finish”

#### Pengaturan POM pada maven project

Pengaturan dependensi dilakukan pada berkas pom.xml. POM merupakan kepanjangan dari “Project Object Model”. Berkas pom.xml merupakan berkas khusus proyek maven yang merepresentasikan proyek ke dalam berkas xml. Pengaturan ini dilakukan agar pustaka java yang diperlukan dapat digunakan saat proses implementasi. Pengaturan dependensi yang diperlukan juga disesuaikan dengan OSGi *container* sehingga dapat di-*deploy* dan dijalankan dalam lingkup OSGi. Pengaturan dilakukan dengan langkah-langkah sebagai berikut:

1. Tambahkan dependensi proyek yang diperlukan modul. Dependensi standard antara lain:
2. Spring MVC versi 3.1.1.RELEASE. Semua modul dikembangkan menggunakan kerangka kerja Spring MVC.
3. Javax Servlet API versi 3.0 sebagai pustaka pendukung dari Spring MVC
4. JSP API versi 2.1 dan JSTL API versi 1.2 untuk dapat menggunakan JSP dan pustaka JSTL.
5. Spring Security versi 3.1.7.RELEASE sebagai pustaka keamanan.
6. SIA modul domain yang berisi kelas-kelas domain yang dibutuhkan modul.

Perlu ditekankan bahwa dependensi diatas merupakan dependensi standard yang harus ada. Dependensi lain mungkin saja diperlukan tergantung kebutuhan pustaka pada pengembangan modul.

1. Tambahkan elemen **<**scope**>**provided**</**scope**>**  untuk setiap elemen <dependency></dependency>. Penambahan elemen ini dilakukan agar maven tidak menyertakan berkas pustaka dependensi, yang berupa berkas jar, ke dalam modul saat modul di-*build*. Berkas pustaka tersebut akan disediakan OSGi *container*. Kode Sumber 4.44 menunjukkan dependensi minimal yang diperlukan modul.
2. **<dependency>**
3. **<groupId>**org.springframework**</groupId>**
4. **<artifactId>**spring-webmvc**</artifactId>**
5. **<version>**${spring-version}**</version>**
6. **<scope>**provided**</scope>**
7. **</dependency>**
8. **<dependency>**
9. **<groupId>**javax.servlet**</groupId>**
10. **<artifactId>**javax.servlet-api**</artifactId>**
11. **<version>**${servlet-version}**</version>**
12. **<scope>**provided**</scope>**
13. **</dependency>**
14. **<dependency>**
15. **<groupId>**javax.servlet.jsp**</groupId>**
16. **<artifactId>**jsp-api**</artifactId>**
17. **<version>**${jsp-version}**</version>**
18. **<scope>**provided**</scope>**
19. **</dependency>**
20. **<dependency>**
21. **<groupId>**javax.servlet**</groupId>**
22. **<artifactId>**jstl**</artifactId>**
23. **<version>**${jstl-version}**</version>**
24. **<scope>**provided**</scope>**
25. **</dependency>**
26. **<dependency>**
27. **<groupId>**org.springframework.security**</groupId>**
28. **<artifactId>**spring-security-core**</artifactId>**
29. **<version>**${spring-security-version}**</version>**
30. **<scope>**provided**</scope>**
31. **</dependency>**
32. **<dependency>**
33. **<groupId>**org.springframework.security**</groupId>**
34. **<artifactId>**spring-security-web**</artifactId>**
35. **<version>**${spring-security-version}**</version>**
36. **<scope>**provided**</scope>**
37. **</dependency>**
38. **<dependency>**
39. **<groupId>**org.springframework.security**</groupId>**
40. **<artifactId>**spring-security-config**</artifactId>**
41. **<version>**${spring-security-version}**</version>**
42. **<scope>**provided**</scope>**
43. **</dependency>**
44. **<dependency>**
45. **<groupId>**org.springframework.security**</groupId>**
46. **<artifactId>**spring-security-taglibs**</artifactId>**
47. **<version>**${spring-security-version}**</version>**
48. **<scope>**provided**</scope>**
49. **</dependency>**
50. **<dependency>**
51. **<groupId>**com.sia.modul**</groupId>**
52. **<artifactId>**sia-modul-domain**</artifactId>**
53. **<version>**${sia-modul-domain-version}**</version>**
54. **<scope>**provided**</scope>**
55. **</dependency>**

**Kode Sumber 4.44 Dependensi minimal modul pada berkas pom.xml**

1. Tambahkan maven-bundle-plugin, maven-jar-plugin, dan maven-compiler-plugin. Untuk dapat men-*deploy* modul ke dalam OSGi container, modul harus di-*build* menjadi b*u*ndle. Dengan maven-bundle-plugin, modul dapat di-build menjadi *bundle* dan properti-properti *bundle* diisi jika properti tersebut tidak dinyatakan. Properti *bundle* yang diperlukan dapat dinyatakan langsung pada berkas pom.xml atau dipisah pada berkas lain. Pada kasus ini properti bundle akan dipisah pada berkas “osgi.properties”. Untuk dapat membaca berkas tersebut, tambahkan <\_include>-osgi.properties</\_include>  ke dalam elemen <instructions> </instructions>. Properti *bundle* yang diperlukan akan dijelaskan pada langkah lain.

Maven-jar-plugin pada dasarnya merupakan plugin pada maven untuk membuat proyek dapat di-*build* menjadi berkas jar. Namun, pada kasus ini, selain digunakan untuk hal tersebut, *plugin* ini dikombinasikan dengan maven-bundle-plugin untuk memanipulasi berkas MANIFEST.MF pada berkas jar agar menyerupai berkas *bundle*. Untuk memanipulasi berkas MANIFEST.MF, tambahkan <manifestFile>${project.build.outputDirectory}/META-INF/MANIFEST.MF</manifestFile> ke dalam elemen <archive> </archive> pada elemen <configuration> </configuration>.

Maven-compiler-plugin digunakan untuk mengganti versi java yang digunakan pada proses kompilasi. Versi java yang digunakan adalah versi 1.7. Penambahan maven *plugin* ditunjukkan pada Kode Sumber 4.45.

1. **<build>**
2. **<plugins>**
3. **<plugin>**
4. **<groupId>**org.apache.felix**</groupId>**
5. **<artifactId>**maven-bundle-plugin**</artifactId>**
6. **<version>**2.1.0**</version>**
7. **<extensions>**true**</extensions>**
8. **<configuration>**
9. **<supportedProjectTypes>**
10. **<supportedProjectType>**bundle**</supportedProjectType>**
11. **<supportedProjectType>**jar**</supportedProjectType>**
12. **</supportedProjectTypes>**
13. **<instructions>**
14. **<\_include>**-osgi.properties**</\_include>**
15. **</instructions>**
16. **</configuration>**
17. **<executions>**
18. **<execution>**
19. **<id>**bundle-manifest**</id>**
20. **<phase>**process-classes**</phase>**
21. **<goals>**
22. **<goal>**manifest**</goal>**
23. **</goals>**
24. **</execution>**
25. **</executions>**
26. **</plugin>**
27. **<plugin>**
28. **<artifactId>**maven-jar-plugin**</artifactId>**
29. **<configuration>**
30. **<archive>**
31. **<manifestFile>**                ${project.build.outputDirectory}/META-INF/MANIFEST.MF
32. **</manifestFile>**
33. **</archive>**
34. **</configuration>**
35. **</plugin>**
36. **<plugin>**
37. **<groupId>**org.apache.maven.plugins**</groupId>**
38. **<artifactId>**maven-compiler-plugin**</artifactId>**
39. **<configuration>**
40. **<source>**1.7**</source>**
41. **<target>**1.7**</target>**
42. **</configuration>**
43. **</plugin>**
44. **</plugins>**
45. **</build>**

**Kode Sumber 4.45 Maven *plugin* modul pada berkas pom.xml**

Jika terjadi *error* pada elemen <executions> atau elemen lain pada maven-bundle-plugin, klik pada tanda *error* dan pilih *install* m2e *connector*. Proses intallasi m2e *connector* memerlukan koneksi internet.

#### Pengisian OSGi *bundle* properties

*Bundle* properti digunakan untuk menyatakan identitas dari suatu *bundle*. Dengan maven-bundle-plugin, properti tertentu otomatis ditentukan jika properti tersebut tidak dinyatakan. Properti-properti *bundle* dinyatakan di dalam berkas osgi.properties. pembuatan dan pengisian properti dilakukan dengan langkah-langkah berikut:

1. Buat berkas osgi.properti pada *root* *project* dengan File -> New -> File pada Eclipse. Isi nama berkas dengan “osgi.properties”.
2. Isi berkas tersebut seperti pada Kode Sumber 4.46
3. Fragment-Host: com.sia.main.sia-web;bundle-version="0.0.1.SNAPSHOT"
4. Import-Package:
5. Export-Package:

**Kode Sumber 4.46 Berkas osgi.properties pada modul**

*Fragment* *host* merupakan properti yang menyatakan bahwa *bundle* tersebut merupakan pecahan dari *bundle* lain. Pada kasus ini adalah *bundle* sia-web yang merupakan perangkat lunak utama Sitem Informasi Akademik. Import-Package dan Export-Package diisi kosong karena dependensi *bundle* sudah dinyatakan pada *host* *bundle*.

#### Implementasi Modul

Proses implementasi pada subbab ini hanya meliputi penyesuaian agar modul dapat ditambah ke dalam perangkat lunak utama. Modul diimplementasi dengan menggunakan kerangka kerja Spring MVC. Langkah-langkah Implementasi yang diperlukan adalah sebagai berikut:

1. Penambahan keamanan pada *controller*

Keamanan merupakan pembatasan akses terhadap suatu fungsi atau *request* ke dalam modul Sistem Informasi Akademik. keamanan pada modul didukung oleh kerangka kerja Spring Security. Pengaturan utama dari keamanan sudah diimplementasi pada perangkat lunak utama Sistem Informasi Akademik sehingga modul hanya perlu menentukan *request* apa saja yang ingin dibatasi.

Penambahan keamanan dilakukan dengan menambahkan anotasi @Secured pada fungsi *controller* atau pada kelas *controller*, pada Spring MVC fungsi *controller* merupakan fungsi pada java yang memiliki anotasi @RequestMapping dan kelas *controller* merupakan kelas yang memiliki anotasi @Controller. Penambahan anotasi @Secured pada suatu fungsi *controller* akan membatasi akses pada fungsi *controller* itu saja sementara penambahan anotasi @Secured pada kelas *controller* akan membatasi semua fungsi *controller* pada kelas *controller*.

Anotasi @Secured memiliki satu parameter yang merupakan *role* apa saja yang bisa mengakses *controller* tersebut. Pengaturan pada perangkat lunak utama Sistem Informasi Akademik menetapkan aturan penamaan *role* pada @Secured dengan menambahkan “ROLE\_” + nama *role*. Contoh: Untuk menetapkan @Secured untuk *role* “Administrator” pengembang menambahkan @Secured(“ROLE\_Administrator”) pada *controller*.

* Contoh penambahan anotasi @Secured pada fungsi controller ditunjukkan pada Kode Sumber 4.47

1. @Secured(value = { "ROLE\_Administrator” })
2. @RequestMapping(value = {"/home", "/"}, method = RequestMethod.GET)
3. **public** ModelAndView home() {
4. ModelAndView modelAndView = **new** ModelAndView();
6. **return** modelAndView;
7. }

**Kode Sumber 4.47 Penggunaan @Secured pada @RequestMapping**

* Contoh penambahan anotasi @Secured pada kelas controller ditunjukkan pada Kode Sumber 4.48

1. @Controller
2. @Secured(value = { "ROLE\_Administrator" })
3. @RequestMapping(value = "/admin/module")
4. **public** **class** ModuleController {
6. @RequestMapping(value = {"/home", "/"}, method = RequestMethod.GET)
7. **public** ModelAndView home() {
8. ModelAndView modelAndView = **new** ModelAndView();
10. **return** modelAndView;
11. }
13. }

**Kode Sumber 4.48 Penggunaan @Secured pada @Controller**

Untuk dapat mengakses obyek yang berisi identitas pengguna dan hak aksesnya, dapat dilakukan dengan mengambil atribut dengan nama “userRoleSession” dari obyek *HttpSession* dan disimpan ke dalam obyek PeranPengguna. Contoh penggunaannya ditunjukkan pada Kode Sumber 4.49.

1. @RequestMapping(value = {"/home", "/"}, method = RequestMethod.GET)
2. **public** ModelAndView home(HttpSession session) {
3. ModelAndView modelAndView = **new** ModelAndView();
4. PeranPengguna peranPengguna = (PeranPengguna)session.getAttribute("userRoleSession");
6. **return** modelAndView;
7. }

**Kode Sumber 4.49 Pengambilan obyek *session* pengguna yang terautentikasi**

Untuk dapat menggunakan kelas PeranPengguna diperlukan dependensi pustaka sia-domain dari perangkat lunak utama. Tambahkan dependensi maven sia-domain ke dalam berkas pom.xml seperti pada Kode Sumber 4.50.

1. **<dependency>**
2. **<groupId>**com.sia.main**</groupId>**
3. **<artifactId>**sia-domain**</artifactId>**
4. **<version>**${sia-version}**</version>**
5. **<scope>**provided**</scope>**
6. **</dependency>**

**Kode Sumber 4.50 Penambahan dependensi sia-domain pada berkas pom.xml**

1. Implementasi akses data

Konfigurasi akses data yang disediakan perangkat lunak utama menggunakan Hibernate Transaction Manager. Oleh karena itu perlu ditambahkan anotasi @Repository pada kelas DAO dan @Transactional pada fungsi kelas tersebut.

Penambahan anotasi @Repository bertujuan untuk menginjeksi obyek Hibernate SessionFactory ke dalam atribut SessionFactory pada kelas DAO.

Anotasi @Transactional ditambah agar fungsi DAO secara otomatis menggunakan Transaction pada saat melakukan pertukaran data dari basis data. Transaction berguna ketika terjadi kesalahan pada proses pertukaran data, basis data akan dikembalikan ke keadaan sebelum pertukaran data dilakukan (*Rollback*)

Contoh penggunaan anotasi-anotasi tersebut ditunjukkan pada Kode Sumber 4.51.

1. **package** com.sia.main.data.dao.impl;
3. **import** org.hibernate.SessionFactory;
4. **import** org.springframework.stereotype.Repository;
5. **import** org.springframework.transaction.annotation.Transactional;
7. @Repository
8. @Transactional
9. **public** **class** ADAOClass {
11. **private** SessionFactory sessionFactory;
13. **public** SessionFactory getSessionFactory() {
14. **return** sessionFactory;
15. }
17. **public** **void** setSessionFactory(SessionFactory sessionFactory) {
18. **this**.sessionFactory = sessionFactory;
19. }
21. **public** **void** insert(Object object) {
22. **this**.sessionFactory.getCurrentSession().save(object);
23. }
25. }

**Kode Sumber 4.51 Contoh kelas DAO**

1. Peniadaan berkas web.xml

Berkas web.xml merupakan berkas yang berisi meta-data dari perangkat lunak berbasis web pada java seperti definisi *servlet*, *filter*,dan lain-lain. Berkas web.xml ini tidak diperlukan karena berkas web.xml dan pendaftaran konfigurasi *servlet* akan ditangani oleh perangkat lunak utama.

1. Penambahan dan aturan penempatan definisi konfigurasi *servlet*

Komponen *servlet* yang diperlukan pada dasarnya hanya:

1. <resources mapping="” location="" />  yang menyatakan lokasi *resource* dari modul. *Resource* yang dimaksud adalah berkas *stylesheet*, *javascript*, atau gambar untuk keperluan *front end*.
2. <beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver"> dengan properti *prefix* dan *suffix* yang menentukan lokasi penempatan berkas JSP.
3. <context:component-scan base-package="" />  untuk menetukan package apa saja yang dapat dibaca sebagai komponen Spring Bean seperti *controller*, *service*, atau DAO.

Peletakkan berkas harus berbeda antara satu modul dengan modul lain. Untuk membedakan peletakkan berkas ini. Pengembang dapat membuat *folder* “module-resource” pada *folder* src/main/resources/ dan membuat *folder* dengan nama modul sebagai nama *folder* pada *folder* module-resource.

Langkah-langkah pembuatan konfigurasi *servlet* adalah sebagai berikut:

1. Buatlah *folder* dengan nama “resource” pada src/main/ jika belum ada.
2. Pada *folder* “resource”, Klik kanan dan pilih File -> New -> File. Pilih XML File pada window New File.
3. Isi nama berkas xml.
4. Isi berkas xml dengan elemen konfigurasi *servlet* Spring MVC. Contoh isi berkas XML ditunjukkan pada Kode Sumber 4.52 seperti berikut ini:
5. **<?xml** version="1.0" encoding="UTF-8"**?>**
6. **<beans:beans** xmlns="http://www.springframework.org/schema/mvc"
7. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
8. xmlns:beans="http://www.springframework.org/schema/beans"
9. xmlns:context="http://www.springframework.org/schema/context"
10. xmlns:tx="http://www.springframework.org/schema/tx"
11. xsi:schemaLocation="http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd
12. http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
13. http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd
14. http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.1.xsd "**>**
16. **<resources** mapping="/resources/\*\*" location="/resources/" **/>**
18. **<beans:bean** class="org.springframework.web.servlet.view.InternalResourceViewResolver"**>**
19. **<beans:property** name="prefix" value="/module-resource/penilaian/views/" **/>**
20. **<beans:property** name="suffix" value=".jsp" **/>**
21. **</beans:bean>**
23. **<context:component-scan** base-package="com.siakad.modul\_penilaian.controller" **/>**
24. **<context:component-scan** base-package="com.siakad.modul\_penilaian.repository" **/>**
25. **<context:component-scan** base-package="com.siakad.modul\_penilaian.service" **/>**
27. **</beans:beans>**

**Kode Sumber 4.52 Konigurasi *servlet* modul**

1. Simpan berkas.

Pada bean dengan kelas org.springframework.web.servlet.view.InternalResourceViewResolver terdapat properti dengan nama “prefix” yang bernilai “/module-resource/views/”. Artinya, penempatan berkas JSP harus pada *folder* “/module-resource/penilaian/views/”.

1. Antarmuka pengguna
2. Antarmuka pengguna dapat dibuat dengan seluruh elemen standard html lengkap.
3. Penambahan <title> dapat dilakukan di dalam <head></head>.
4. Beberapa *script* dan *stylesheet* di dalam <head></head> dari tema yang sudah disepakati tidak perlu dinyatakan ulang karena sudah dinyatakan pada template. *Script*  dan *stylesheet* tersebut ditunjukkan pada Kode Sumber 4.53:
5. **<link** rel="shotcut icon" href="${pageContext.servletContext.contextPath}/resources/images/icon.ico"**>**
6. **<link** href='http://fonts.googleapis.com/css?family=Open+Sans:400,300,600' rel='stylesheet' type='text/css'**>**
7. **<link** href="${pageContext.servletContext.contextPath}/resources/plugins/pace-master/themes/blue/pace-theme-flash.css" rel="stylesheet"**/>**
8. **<link** href="${pageContext.servletContext.contextPath}/resources/plugins/bootstrap/css/bootstrap.min.css" rel="stylesheet" type="text/css"**/>**
9. **<link** href="${pageContext.servletContext.contextPath}/resources/plugins/fontawesome/css/font-awesome.css" rel="stylesheet" type="text/css"**/>**
10. **<link** href="${pageContext.servletContext.contextPath}/resources/plugins/waves/waves.min.css" rel="stylesheet" type="text/css"**/>**
11. **<link** href="${pageContext.servletContext.contextPath}/resources/plugins/toastr/toastr.min.css" rel="stylesheet" type="text/css"**/>**
12. **<link** href="${pageContext.servletContext.contextPath}/resources/css/modern.min.css" rel="stylesheet" type="text/css"**/>**
13. **<link** href="${pageContext.servletContext.contextPath}/resources/css/themes/white.css" class="theme-color" rel="stylesheet" type="text/css"**/>**

**Kode Sumber 4.53 Daftar *stylesheet* yang telah disediakan template**

Elemen <head></head> dari JSP modul akan otomatis ditambahkan ke akhir <head></head> dari *template.*

1. Penambahan *Script* pada <body> </body> dilakukan di dalam <content tag=”scripts”></content>. <content></content> merupakan elemen *custom* dari mesin *template*. Beberapa *script* dari tema yang sudah disepakati tidak perlu dinyatakan ulang karena sudah dinyatakan pada template. Berikut adalah Kode Sumber 4.54 yang menunjukkan *script* tersebut:
2. **<script** src="${pageContext.servletContext.contextPath}/resources/plugins/jquery/jquery-2.1.3.min.js"**></script>**
3. **<script** src="${pageContext.servletContext.contextPath}/resources/plugins/jquery-ui/jquery-ui.min.js"**></script>**
4. **<script** src="${pageContext.servletContext.contextPath}/resources/plugins/jquery-blockui/jquery.blockui.js"**></script>**
5. **<script** src="${pageContext.servletContext.contextPath}/resources/plugins/bootstrap/js/bootstrap.min.js"**></script>**
6. **<script** src="${pageContext.servletContext.contextPath}/resources/plugins/pace-master/pace.min.js"**></script>**
7. **<script** src="${pageContext.servletContext.contextPath}/resources/plugins/waves/waves.min.js"**></script>**
8. **<script** src="${pageContext.servletContext.contextPath}/resources/plugins/jquery-slimscroll/jquery.slimscroll.min.js"**></script>**
9. **<script** src="${pageContext.servletContext.contextPath}/resources/plugins/toastr/toastr.min.js"**></script>**
10. **<script** src="${pageContext.servletContext.contextPath}/resources/js/pages/notifications.js"**></script>**
11. **<script** src="${pageContext.servletContext.contextPath}/resources/js/modern.min.js"**></script>**
12. **<script>**
13. toastr.options = {
14. "closeButton": true,
15. "debug": false,
16. "newestOnTop": false,
17. "progressBar": false,
18. "positionClass": "toast-top-right",
19. "preventDuplicates": false,
20. "onclick": null,
21. "showDuration": "300",
22. "hideDuration": "1000",
23. "timeOut": "5000",
24. "extendedTimeOut": "1000",
25. "showEasing": "swing",
26. "hideEasing": "linear",
27. "showMethod": "fadeIn",
28. "hideMethod": "fadeOut"
29. }
31. var contextPath = "${pageContext.servletContext.contextPath}";
32. **</script>**

**Kode Sumber 4.54 Daftar *script* yang telah disediakan template**

*Script* yang ditambahkan ke dalamelemen <content></content> pada JSP modul akan otomatis ditambahkan ke sebelum *closing* *tag* </body> pada *template.*

1. Jika menggunakan tema yang sama dengan *template*. Isi <body> pada JSP modul hanya isi dari <div id=”main-wrapper”></div>.
2. Contoh JSP dengan *element* tambahan ditunjukkan pada Kode Sumber 4.55
3. **<**%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%**>**
4. **<**%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %**>**
5. **<**%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %**>**
7. <!DOCTYPE html**>**
8. **<html>**
9. **<head>**
10. **<title>**Insert title here**</title>**
11. **<link** href="${pageContext.servletContext.contextPath}/resources/aCSSPlugin/aCSSPlugin.css"**></link**
12. **</head>**
13. **<body>**
14. **<div** class="row"**>**
15. **</div>**
16. **<content** tag="scripts"**>**
17. **<script** src="${pageContext.servletContext.contextPath}/resources/aJSPlugin/aJSPlugin.js"**></script>**
18. **<script>**
19. <!-- js / jquery here -->
20. **</script>**
21. **</content>**
22. **</body>**
23. **</html>**

**Kode Sumber 4.55 Contoh struktur berkas JSP**

1. Penempatan berkas JSP wajib disesuaikan dengan nilai *prefix* dari kelas *InternalResourceViewResolver* pada berkas konfigurasi yang telah dijelaskan.
2. Penempatan berkas resources seperti stylesheet dan javascript dapat diletakkan pada *folder* resources pada *folder* src/main/resources.
3. Tiap menu pada modul dapat menampilkan judul menu ke *template* antar muka pengguna dengan cara mendaftarkan obyek Menu ke dalam obyek *HttpSession* dengan nama atribut “menuActive” seperti yang ditunjukkan pada Kode Sumber 4.56.
4. @RequestMapping(value = {"/home", "/"}, method = RequestMethod.GET)
5. **public** ModelAndView home(HttpSession session) {
6. ModelAndView modelAndView = **new** ModelAndView();
7. String menu;
8. //isi object menu yang sesuai dengan request mapping
9. session.setAttribute("menuActive", menu);
11. **return** modelAndView;
12. }

**Kode Sumber 4.56 Penambahan menu yang sedang diakses**

1. Penambahan Rincian Modul

Walaupun modul dapat dibaca oleh OSGi sebagai *fragment* *bundle* dari perangkat lunak utama, perangkat lunak utama tidak mengetahui properti - properti dari modul tersebut. Untuk itu diperlukan suatu berkas yang berisi properti modul. Properti-properti tersebut harus dimiliki modul. Jika tidak, modul tidak bisa ditambahkan ke dalam perangkat lunak utama. Rincian modul dibuat sebagai berkas xml. Properti tersebut antara lain adalah:

1. Nama modul dengan elemen dalam xml <name></name>. Elemen ini harus ada di dalam elemen <module></module>.
2. URL *mapping* modul. URL *mapping* modul merupakan URL *mapping* yang akan ditambahkan pada *request* *mapping* menu pada modul untuk membedakan URL suatu modul dengan modul lain. URL *mapping* harus berbeda antara suatu modul dengan modul lain. Jika tidak, modul tidak bisa ditambahkan. Elemen properti ini adalah <url-maping></url-mapping>. Elemen ini harus ada di dalam elemen <module></module>.
3. Lokasi konfigurasi *servlet*. Konfigurasi *servlet* yang telah dibuat pada langkah sebelumnya perlu didaftarkan ke dalam berkas rincian ini. Perangkat lunak utama akan memeriksa apakah konfigurasi *servlet* ada pada lokasi yang tertulis pada properti ini. Jika konfigurasi *servlet* tidak ditemukan, penambahan modul ditolak. Elemen properti ini adalah <config-locations></config-locations>. Elemen ini harus ada di dalam elemen <module></module>.
4. Daftar menu dengan elemen <menues></menues>. Properti ini diisi oleh elemen <menu></menu>. Elemen ini harus ada di dalam elemen <module></module>.
5. Menu dengan elemen <menu></menu>. Properti ini digunakan untuk pembatasan akses peran terhadap menu dan agar menu bisa diakses melalui menubar pada antarmuka pengguna. Elemen ini harus ada di dalam elemen <menues></menues>.
6. Nama menu dengan elemen yang sama dengan nama modul yaitu <name></name>. Elemen ini harus ada di dalam elemen <menu></menu>.
7. URL menu dengan elemen <url></URL>. Elemen ini harus ada di dalam elemen <menu></menu>.

Langkah-langkah membuat berkas rincian ini adalah sebagai berikut:

1. Buatlah *folder* “module-detail” pada *folder* “resource” yang telah dibuat pada langkah sebelumnya.
2. Pada *folder* tersebut buatlah berkas xml. Isi nama berkas xml dengan “module-” + *group* id proyek maven + “.” + *artifact* id proyek maven + “.xml”. Contoh: “module-com.sia.modul.penilaian.xml”.
3. Isi berkas xml tersebut dengan elemen properti yang telah dijelaskan sebelumnya dan simpan. Berkas XML ini ditunjukkan pada Kode Sumber 4.57 yaitu sebagai berikut:
4. **<?xml** version="1.0" encoding="UTF-8"**?>**
5. **<module>**
6. **<name>**Modul Penilaian**</name>**
7. **<url-mapping>**/penilaian/\***</url-mapping>**
8. **<config-locations>**/module-resource/spring/appServlet/servlet-context.xml**</config-locations>**
9. **<menues>**
10. **<menu>**
11. **<name>**Lihat IPS**</name>**
12. **<url>**/lihat\_ips/**</URL>**
13. **</menu>**
14. **</menues>**
15. **</module>**

**Kode Sumber 4.57 Contoh struktur berkas properti modul**

1. *Build* modul

Langkah ini merupakan langkah terakhir sebelum modul ditambahkan. Langkah build modul dilakukan dengan maven build. Langkah melakukan maven *build* adalah sebagai berikut:

1. Klik kanan pada maven proyek modul
2. Pilih Run As -> Maven build
3. Isi Goals dengan “clean install”
4. Klik Run

Jika terjadi *error*, Beberapa cara dibawah ini dapat dilakukan:

* Pastikan proyek bisa mengakses *repository* utama maven.
* Pastikan semua dependensi yang dibutuhkan sudah diatur pada berkas pom.xml
* Ubah Goals dengan “install” melalui Run As -> Run Configuration.

1. Berkas modul yang berupa jar dapat ditemukan pada *folder* “target”
2. Modul siap ditambahkan ke dalam perangkat lunak utama

### Pembacaan modul

Pembacaan berkas detail modul yang berupa berkas xml dilakukan di dalam fungsi *installModule*() dari kelas implementasi *interface* *ModulService*. Pada implementasi fungsi tersebut, fungsi *readModule*() dari kelas *XmlModuleReaderBean* dipanggil. Pada fungsi tersebut, berkas detail modul dibaca pada lokasi, *prefix*, *suffix* dari berkas yang telah ditentukan yaitu pada lokasi adalah “/module-detail/”, *prefix* adalah “module-”, dan *suffix* adalah “.xml”. Jika ada properti yang kurang, maka fungsi tersebut menghasilkan error. Jika properti modul lengkap, obyek modul beserta menu didalamnya ditambahkan ke dalam basis data melalui kelas implementasi dari *ModulDAO*. Fungsi *readModule*() ditunjukkan Kode Sumber 4.58.

1. @Override
2. **public** Modul readModule(Object module, Object host) **throws** Exception {
3. Bundle moduleBundle = (Bundle) module;
4. Bundle hostBundle = (Bundle) host;
5. Modul modul = **null**;
6. **try** {
7. SAXParserFactory factory = SAXParserFactory.newInstance();
8. SAXParser saxParser = factory.newSAXParser();
9. String moduleDetailLocation = **this**.moduleDetailXmlPath + **this**.prefix + moduleBundle.getSymbolicName() + ".xml";
10. URL fileURL = hostBundle.getResource(moduleDetailLocation);
11. **if**(fileURL == **null**) {
12. **throw** **new** NullPointerException("Berkas xml tidak ditemukan");
13. }
14. ModuleXmlParseHandler parseHandler = **new** ModuleXmlParseHandler();
15. saxParser.parse(fileURL.openStream(), parseHandler);
16. modul = parseHandler.getGeneratedModule();
17. **if**(modul.getNamaModul() == **null** || modul.getNamaModul().equals("")) **throw** **new** NullPointerException("Nama modul tidak boleh kosong");
18. **if**(modul.getURLMapping() == **null** || modul.getURLMapping().equals("")) **throw** **new** NullPointerException("Pemetaan URL modul tidak boleh kosong");
19. **if**(modul.getLokasiKonfigServlet() == **null** || modul.getLokasiKonfigServlet().equals("")) **throw** **new** NullPointerException("Lokasi konfigurasi servlet modul tidak boleh kosong");
20. **if**(modul.getMenus() == **null** || modul.getMenus().size() == 0) **throw** **new** NullPointerException("Tidak ada menu terdaftar pada modul");
21. **else** {
22. **for**(Menu menu: modul.getMenus()) {
23. **if**(menu.getNamaMenu() == **null** || menu.getNamaMenu().equals("")) **throw** **new** NullPointerException("Nama menu tidak boleh kosong");
24. **if**(menu.getURLMenu() == **null** || menu.getURLMenu().equals("")) **throw** **new** NullPointerException("URL menu tidak boleh kosong");
25. }
26. }
27. **for**(String configLocation: modul.getServletConfigLocationList()) {
28. URL configLocURL = hostBundle.getResource(configLocation);
29. **if**(configLocURL == **null**) {
30. **throw** **new** NullPointerException("Konfigurasi servlet tidak ditemukan");
31. }
32. }
33. } **catch** (ParserConfigurationException | SAXException | IOException | NullPointerException e) {
34. **throw** e.getClass().getDeclaredConstructor(String.**class**).newInstance("Pembacaan detail modul pada berkas xml gagal. Pesan Exception: " + e.getMessage());
35. }
36. **return** modul;
37. }

Kode Sumber 4.58 Kode sumber fungsi *readModule*()

### Pendaftaran *servlet* modul

Modul yang telah dibaca dan disimpan dalam basis data didaftarkan ke dalam perangkat lunak utama sebagai obyek dari kelas *DispatcherServlet* yang disediakan oleh pustaka Spring MVC. Sampai saat buku ini dibuat, penulis hanya dapat mendaftarkan *servlet* pada saat aplikasi pertama kali dijalankan. Untuk dapat mendaftarkan *servlet*, perlu dibuat kelas yang mengimplementasi fungsi *onStartup*(*ServletContext* *servletContext*) dari interface *WebApplicationInitialzer*. Pada implementasi ini kelas implementasi tersebut dinamai *WebAppInitializer*. Fungsi implementasi dari *onStartup*(*ServletContext* *servletContext*) memanggil *instance* dari kelas *SIAMainWebApplication* dan memanggil fungsi *init*() dari kelas tersebut seperti yang ditunjukkan pada Kode Sumber 4.60. Pada fungsi *init*(), fungsi *registerServlet*() pada kelas yang sama dipanggil. Informasi yang diperlukan untuk pendaftaran *servlet* diambil langsung dari basis data melalui *interface* *ModulService*. *Interface* tersebut merupakan *interface* dari layer *service*. *Interface* tersebut diambil melalui kelas *ServiceTracker* yang disediakan OSGi untuk mengambil *service* pada OSGi *Service* *Registry*. Pengambilan *service* dari OSGi *Service* *Registry* sebenarnya dapat lebih mudah dilakukan yaitu dengan *blueprint* namun *blueprint* tidak bisa dilakukan sebelum fungsi *onStartup*() selesai. Kode sumber bagian tersebut ditunjukkan pada Kode Sumber 4.59. Pada saat pendaftaran *servlet* dilakukan, konfigurasi tambahan yang meliputi implementasi hak akses dan keamanan disertakan bersama dengan konfigurasi *servlet* pada masing-masing modul. Selain itu tiap URL *mapping* tiap *servlet* diberikan *prefix* “/modul” untuk berguna sebagai pembeda antara *servlet* perangkat lunak utama dan *servlet* modul dan sebagai pembeda saat penentuan *template* antarmuka pengguna yang digunakan. Kode sumber dari fungsi *registerServlet* ditunjukkan pada Kode Sumber 4.60.

1. **package** com.sia.main.web;
3. **import** javax.servlet.ServletContext;
4. **import** javax.servlet.ServletException;
6. **import** org.springframework.web.WebApplicationInitializer;
8. **public** **class** WebAppInitializer **implements** WebApplicationInitializer {
10. @Override
11. **public** **void** onStartup(ServletContext servletContext)**throws** ServletException {
12. SIAMainWebApplication webApplication = SIAMainWebApplication.getInstance();
13. webApplication.setServletContext(servletContext);
14. webApplication.init();
15. }
17. }

Kode Sumber 4.59 Kode sumber kelas WebAppInitializer

1. **package** com.sia.main.web;
3. **import** java.util.List;
5. **import** javax.servlet.ServletContext;
6. **import** javax.servlet.ServletRegistration;
8. **import** org.osgi.framework.BundleContext;
9. **import** org.osgi.framework.FrameworkUtil;
10. **import** org.springframework.web.context.support.XmlWebApplicationContext;
11. **import** org.springframework.web.servlet.DispatcherServlet;
13. **import** com.sia.main.domain.Modul;
14. **import** com.sia.main.plugin.ApplicationInitializer;
15. **import** com.sia.main.service.services.ModulService;
16. **import** com.sia.main.web.osgi.ServiceLocator;
18. **public** **class** SIAMainWebApplication **implements** ApplicationInitializer {
20. **private** **static** SIAMainWebApplication instance;
22. **private** ServletContext servletContext;
24. **private** ServletRegistration.Dynamic dispatcher;
26. **private** String moduleContextPathPrefix;
28. **private** String[] additionalModuleConfigLocations;
30. **public** **static** SIAMainWebApplication getInstance() {
31. **if**(instance == **null**) {
32. instance = **new** SIAMainWebApplication();
33. }
34. **return** instance;
35. }
37. **private** SIAMainWebApplication() {
38. **this**.moduleContextPathPrefix = "/modul";
39. **this**.additionalModuleConfigLocations = **new** String[] {"/WEB-INF/spring-beans/servlet/modul/sia-modul-servlet.xml", "/WEB-INF/spring-beans/servlet/modul/sia-modul-dataaccess.xml"};
40. }
42. // properties setters and getters
44. @Override
45. **public** **void** init() {
46. **this**.registerServlet();
47. }
49. **private** **void** registerServlet() {
50. ModulService modulService = **this**.getModulService();
51. **if**(modulService != **null**) {
52. List<Modul> modules =  modulService.getByParam("where status = 'RESOLVED'");
53. **if**(modules != **null** && modules.size() > 0) {
54. **for**(Modul module: modules) {
55. List<String> servletConfigLocations = module.getServletConfigLocationList();
56. **for**(String loc: **this**.additionalModuleConfigLocations) {
57. servletConfigLocations.add(loc);
58. }
59. XmlWebApplicationContext context = **new** XmlWebApplicationContext();
60. context.setConfigLocations(servletConfigLocations.toArray(**new** String[servletConfigLocations.size()]));
61. **this**.dispatcher = **this**.servletContext.addServlet(module.getNamaServlet(), **new** DispatcherServlet(context));
62. **this**.dispatcher.addMapping(**this**.moduleContextPathPrefix + module.getURLMapping());
63. **this**.dispatcher.setLoadOnStartup(0);
64. }
65. }
66. }
67. }
69. **private** ModulService getModulService() {
70. BundleContext bundleContext = FrameworkUtil.getBundle(**this**.getClass()).getBundleContext();
71. ServiceLocator<ModulService> serviceLocator = **new** ServiceLocator<ModulService>(bundleContext, ModulService.**class**);
72. **return** serviceLocator.getService(0);
73. }
75. }

Kode Sumber 4.60 Kode sumber kelas SIAMainWebApplication sebagai kelas pendaftar servlet

## Implementasi Keamanan

Keamanan pada sistem ini diimplementasi dengan menggunakan kerangka kerja Spring Security. Komponen-komponen keamanan perangkat lunak berupa kelas-kelas yang diletakkan pada *package* “com.sia.main.web.security” dan berkas xml yang merupakan Spring Beans. Kedua komponen tersebut diletakkan pada *layer web*.

### *Package* *Security*

*Package* ini berisi kelas-kelas implementasi dari kerangka kerja Spring Security yang kemudian digunakan sebagai komponen keamanan yang diatur dalam berkas “sia-web-security.xml”.

#### Kelas *RedirectAccessDeniedHandler*

Kelas ini merealisasi kelas *interface* *AccessDeniedHandler*. Kelas ini berfungsi untuk mengarahkan pengguna jika hak akses pengguna terhadap suatu menu tidak terpenuhi.

#### Kelas *SessionSavedRequestAwareAuthenticationSuccessHandler*

Kelas ini menurunkan kelas *SimpleURLAuthenticationSuccessHandler* yang berfungsi untuk menangani kejadian ketika proses authentikasi berhasil. Kelas turunan ini memiliki fungsi tambahan yaitu menyimpan *Request* yang terakhir dilakukan sebelum pengguna diarahkan ke halaman login karena belum melakukan authentikasi dan menyimpan request terakhir tersebut ke dalam *Http* *Session* untuk diproses pada saat pemilihan hak akses pengguna dan pembentukan menu sesuai hak akses yang dipilih.

#### Kelas *SIAUser*

Kelas ini menurunkan kelas *User* yang merupakan kelas yang merepresentasikan identitas pengguna yang terautentikasi. Kelas *SIAUser* menambahkan atribut dengan tipe data Pengguna sehingga memiliki atribut detail sesuai dengan kelas entitas.

#### Kelas *SIAUserProvider*

Kelas ini merupakan kelas realisasi dari kelas *interface* *UserDetailsService*. Interface ini merupakan salah satu komponen utama yang dapat diimplementasi sesuai dengan sumber penyedia identitas pengguna. Pada kelas *SIAUserProvider*, sumber penyedia identitas pengguna adalah basis data dari sistem. Berikut Kode Sumber 4.61 yang merupakan fungsi yang diimplementasi dari kelas *interface* *UserDetailsService*.

1. @Override
2. **public** UserDetails loadUserByUsername(String username) **throws** UsernameNotFoundException {
3. Pengguna user = penggunaService.getByUsername(username);
4. **return** **new** SIAUser(user.getUsername(), user.getPassword(),user.isStatusKeaktifan(),
5. **true**, **true**, **true**, **this**.buildUserAuthorities(user.getPeranPenggunaList()), user);
6. }
8. **private** List<GrantedAuthority> buildUserAuthorities(List<PeranPengguna> userRoles) {
9. List<GrantedAuthority> auths = **new** ArrayList<GrantedAuthority>();
10. **for** (PeranPengguna userRole : userRoles) {
11. auths.add(**new** SimpleGrantedAuthority("ROLE\_" + userRole.getPeran().getNamaPeran()));
12. }
13. **return** auths;
14. }

**Kode Sumber 4.61 Kelas *SIAUserProvider***

#### Sia-web-security.xml

Berkas ini merupakan berkas utama yang diperlukan oleh sistem keamanan. Setiap komponen keamanan didefinisikan sebagai Spring Bean pada berkas tersebut. Isi berkas sia-web-security.xml ditunjukkan pada Kode Sumber 4.62.

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<beans** xmlns="http://www.springframework.org/schema/beans"
3. xmlns:security="http://www.springframework.org/schema/security"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5. xmlns:context="http://www.springframework.org/schema/context"
6. xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
7. http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd
8. http://www.springframework.org/schema/security http://www.springframework.org/schema/security/spring-security.xsd"**>**
10. **<security:http** use-expressions="true" **>**
11. **<security:form-login**
12. login-page="/account/login"
13. authentication-failure-URL="/account/login?error"
14. username-parameter="username"
15. password-parameter="password"
16. authentication-success-handler-ref="loginSuccessHandler"**/>**
17. **<security:logout** logout-success-URL="/account/logout" delete-cookies="JSESSIONID"**/>**
18. **<security:access-denied-handler** ref="accessDeniedHandler"**/>**
19. **</security:http>**
21. **<security:authentication-manager>**
22. **<security:authentication-provider** user-service-ref="userDetailProvider"**>**
23. **<security:password-encoder** hash="md5"**></security:password-encoder>**
24. **</security:authentication-provider>**
25. **</security:authentication-manager>**
27. **<bean** id="userDetailProvider" class="com.sia.main.web.security.SIAUserProvider"**>**
28. **<property** name="penggunaService" ref="penggunaServiceRef"**></property>**
29. **</bean>**
31. **<bean** id="loginSuccessHandler" class="com.sia.main.web.security.SessionSavedRequestAwareAuthenticationSuccessHandler"**>**
32. **<property** name="useReferer" value="false"**></property>**
33. **<property** name="defaultTargetURL" value="/account/login/"**></property>**
34. **</bean>**
36. **<bean** id="accessDeniedHandler" class="com.sia.main.web.security.RedirectAccessDeniedHandler"**>**
37. **<property** name="redirectURL" value="/error/403"**></property>**
38. **</bean>**
40. **</beans>**

**Kode Sumber 4.62 Berkas sia-web-security.xml**

## Implementasi Akses Data

Akses data pada perangkat lunak ini dibagi menjadi dua yaitu akses data untuk fungsionalitas administratif dan modul. Kedua akses data diimplementasi menjadi berkas xml yang serupa dengan Spring context dengan kerangka kerja Hibernate dan Spring ORM. Isi dari berkas ini yaitu lokasi sumber data beserta autentikasinya, sistem manajemen basis data relasional yang digunakan, kelas-kelas pemetaan tabel basis data, dan konfigurasi *Transaction* *Manager*. Kode sumber dari implementasi akses data untuk fungsionalitas admin ditunjukkan pada Kode Sumber 4.63.

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<beans** xmlns="http://www.springframework.org/schema/beans"
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4. xmlns:tx="http://www.springframework.org/schema/tx"
5. xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
6. http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.1.xsd"**>**
8. <!-- JDBC setting -->
9. **<bean** id="dataSource" name="dataSource"
10. class="org.apache.commons.dbcp2.BasicDataSource"**>**
11. **<property** name="driverClassName" value="org.postgresql.Driver" **/>**
12. **<property** name="URL" value="jdbc:postgresql://localhost:5432/sia-security" **/>**
13. **<property** name="username" value="postgres" **/>**
14. **<property** name="password" value="admin" **/>**
15. **<property** name="validationQuery" value="SELECT 1" **/>**
16. **</bean>**
18. <!-- Hibernate session factory -->
19. **<bean** id="sessionFactory" name="sessionFactory"
20. class="org.springframework.orm.hibernate4.LocalSessionFactoryBean"**>**
21. **<property** name="dataSource" ref="dataSource" **/>**
23. **<property** name="hibernateProperties"**>**
24. **<props>**
25. **<prop** key="hibernate.c3p0.min\_size"**>**5**</prop>**
26. **<prop** key="hibernate.c3p0.max\_size"**>**20**</prop>**
27. **<prop** key="hibernate.c3p0.timeout"**>**300**</prop>**
28. **<prop** key="hibernate.c3p0.max\_statements"**>**50**</prop>**
29. **<prop** key="hibernate.c3p0.idle\_test\_period"**>**3000**</prop>**
30. **<prop** key="hibernate.show\_sql"**>**false**</prop>**
31. **<prop** key="hibernate.dialect"**>**org.hibernate.dialect.PostgreSQLDialect**</prop>**
32. **<prop** key="hibernate.connection.charSet"**>**UTF-8**</prop>**
33. **<prop** key="hibernate.enable\_lazy\_load\_no\_trans"**>**true**</prop>**
34. **</props>**
35. **</property>**
37. **<property** name="annotatedClasses"**>**
38. **<array>**
39. **<value>**com.sia.main.domain.Menu**</value>**
40. **<value>**com.sia.main.domain.MenuPeran**</value>**
41. **<value>**com.sia.main.domain.Modul**</value>**
42. **<value>**com.sia.main.domain.Pengguna**</value>**
43. **<value>**com.sia.main.domain.Peran**</value>**
44. **<value>**com.sia.main.domain.PeranPengguna**</value>**
45. **<value>**com.sia.main.domain.SatMan**</value>**
46. **<value>**com.sia.main.domain.Pd**</value>**
47. **<value>**com.sia.main.domain.Ptk**</value>**
48. **<value>**com.sia.main.domain.TipePengguna**</value>**
49. **</array>**
50. **</property>**
52. **</bean>**
54. <!-- Hibernate transaction manager -->
55. **<bean** id="transactionManager" name="transactionManager"
56. class="org.springframework.orm.hibernate4.HibernateTransactionManager"**>**
57. **<property** name="sessionFactory" ref="sessionFactory" **/>**
58. **</bean>**
60. **<tx:annotation-driven** transaction-manager="transactionManager" **/>**
62. **</beans>**

**Kode Sumber 4.63 Berkas konfigurasi akses data fungsionalitas admin**

Berkas xml konfigurasi akses data untuk fungsionalitas admin diletakkan pada *layer* DAO sementara berkas konfigurasi akses data untuk modul diletakkan pada *layer* *web*. Berkas akses data untuk modul akan menjadi salah satu berkas konfigurasi *servlet* modul pada proses pendaftaran *servlet* modul.

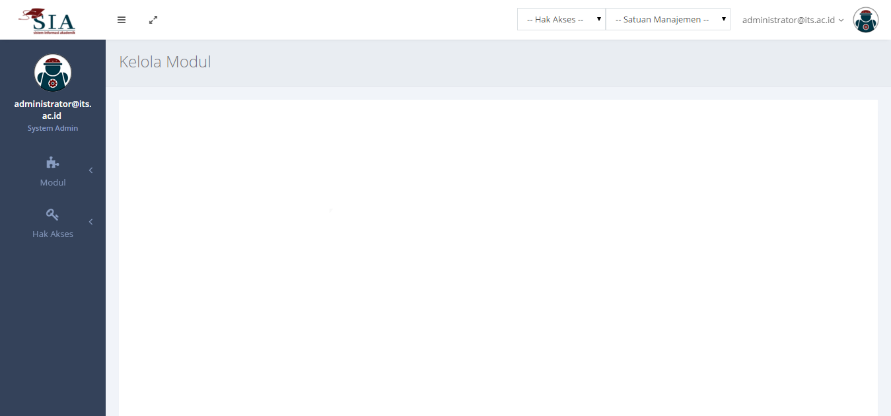
## Implementasi *Template* Antarmuka Pengguna

*Template* antarmuka pengguna dibuat menjadi dua *template* yang berbeda yaitu *template* untuk menu-menu modul dan *template* untuk fungsionalitas administratif. Sistem akan membedakan kedua *template* tersebut melalui pola URI dari *Http* *Request*. Adapun beberapa halaman yang tidak disertakan *template* seperti halaman *login*, halaman pemilihan hak akses pengguna, dan halaman beranda. *Template* dibuat sebagai berkas JSP dengan konfigurasi berbentuk berkas xml. Berkas konfigurasi *template* ditunjukkan pada Kode Sumber 4.64.

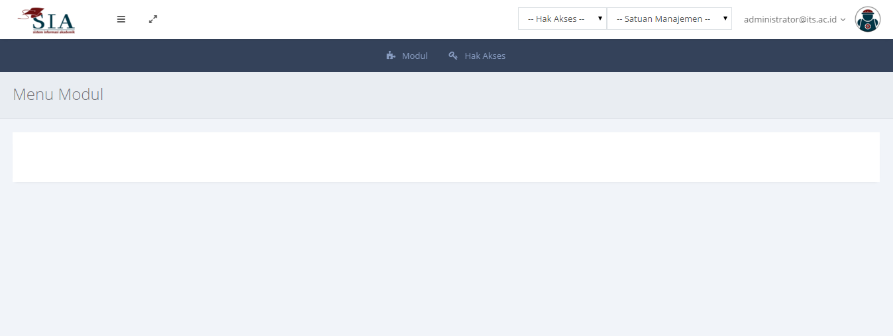
1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<decorators** defaultdir="/WEB-INF/views/master"**>**
4. **<decorator** name="admin-master-page" page="AdminMaster.jsp"**>**
5. **<pattern>**/admin/\***</pattern>**
6. **</decorator>**
8. **<decorator** name="module-master-page" page="ModuleMaster.jsp"**>**
9. **<pattern>**/modul/\***</pattern>**
10. **</decorator>**
12. **<excludes>**
13. **<pattern>**/account/\***</pattern>**
14. **<pattern>**/session/\***</pattern>**
15. **<pattern>**/error/\***</pattern>**
16. **</excludes>**
18. **</decorators>**

**Kode Sumber 4.64 Berkas konfigurasi *template* antarmuka pengguna**

Isi dari halaman menu dipanggil oleh halaman *template* dengan menggunakan elemen *custom* <decorator:body></decorator:body>. Hasil implementasi *template* ditunjukkan pada Gambar 4.13 dan Gambar 4.14.



Gambar 4.13 Halaman *template* fungsionalitas administratif



Gambar 4.14 Halaman *template* modul

# BAB V UJI COBA DAN EVALUASI

Pada bab ini akan dijelaskan pengujian dan evaluasi dari perangkat lunak ini yang meliputi rincian lingkungan uji coba dan pengujian pada semua fungsionalitas administratif dan semua menu pada modul yang terintegrasi. Aspek yang diperhatikan dalam pengujian perangkat lunak ini adalah terpenuhinya fungsionalitas administratif dan integrasi modul ke dalam perangkat lunak.



## Lingkungan Uji Coba

Lingkungan uji coba merupakan perangkat keras dan perangkat lunak yang digunakan selama melakukan pengujian. Pengujian dilakukan dengan menggunakan dua lingkungan pengujian yaitu klien dan *server* yang merangkap sebagai *web* *server* dan *server* basis data. Rincian dari kedua lingkungan pengujian ditunjukkan pada Tabel 5.1 dan Tabel 5.2.

Tabel 5.1 Tabel lingkungan uji coba klien

|  |  |
| --- | --- |
| Spesifikasi | Deskripsi |
| CPU | Intel ® Core ™ i5-2450 CPU @ 2.50 GHz |
| RAM | 4.00 GB |
| Sistem Operasi | Windows 7 Ultimate 64 bit |
| Browser | Google Chrome versi 47.0 |

Tabel 5.2 Tabel lingkungan uji coba server

|  |  |
| --- | --- |
| Spesifikasi | Deskripsi |
| CPU | Intel® Xeon® E5-2600 v3@ 2.4 GHz |
| RAM | 8.00 GB |
| Sistem Operasi | Ubuntu Server 14.04.3 LTS |
| Web Server | Eclipse Virgo Tomcat Server 3.6.3.RELEASE |
| Basis Data | PostgreSQL 9.3 |

## Uji Coba

Uji coba fungsionalitas administratif adalah uji coba yang dilakukan terhadap fungsionalitas yang dapat dilakukan oleh administrator. Fungsionalitas tersebut merupakan kasus penggunaan yang sebelumnya telah dijelaskan pada Subbab 3.1.3. Penjelasan uji coba meliputi skenario uji coba dan hasil uji coba.

Uji coba dilakukan dengan metode *black box* yang artinya fungsionalitas diperiksa apakah terpenuhi atau tidak tanpa melihat struktur internal ataupun metode yang digunakan dalam pengerjaan fungsionalitas tersebut.

### Kasus uji fungsionalitas administratif

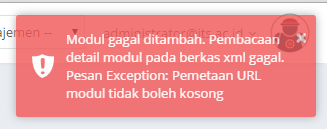
Pengujian fungsionalitas administratif dilakukan dengan menguji kasus-kasus uji tiap kasus penggunaan. Berikut adalah kasus uji dari fungsionalitas administratif.

#### Kasus uji menambah modul

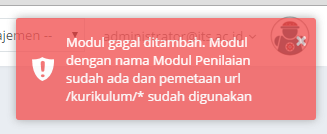
Pada kasus uji ini terdapat empat skenario yaitu jika modul yang ditambahkan belum ada, jika properti modul yang ingin ditambahkan tidak terpenuhi, jika modul yang ingin ditambahkan sudah terdaftar dan URL mapping belum digunakan, dan jika ada modul yang terdaftar memiliki URL mapping yang sama dengan modul yang ingin ditambah. Properti yang dimaksud sudah dijelaskan pada Subbab 4.5.1.4. Properti tidak memenuhi artinya salah satu properti yang diperlukan tidak ada atau berkas konfigurasi *servlet* yang dinyatakan dalam properti tidak ditemukan. Jika properti tidak memenuhi atau terdapat URL *mapping* yang sama pada modul yang sudah terdaftar dengan modul yang ingin ditambah, sistem akan mengeluarkan pesan *error* seperti pada Gambar 5.1 dan Gambar 5.2. Rincian kasus uji ini ditunjukkan pada Tabel 5.3.

Tabel 5.3 Tabel kasus uji menambah modul

|  |  |
| --- | --- |
| ID | TC-001 |
| Kasus Penggunaan | Menambah modul |
| Nama | Pengujian menambah modul |
| Tujuan Pengujian | Menguji apakah modul dapat ditambah ke dalam perangkat lunak |
| *Skenario 1* | *Aktor menambahkan modul dengan semua properti dan URL mapping memenuhi* |
| Kondisi Awal | Modul belum terdaftar |
| Data Uji | * 1. Modul yang didaftarkan dengan properti modul: * Nama modul: Modul Penilaian * URL *mapping*: /penilaian/\* * Lokasi konfigurasi *servlet*: /module-resource/penilaian/servlet-context.xml * Menu: * Nama Menu: Transkrip Nilai * URL Menu: /lihat\_transkrip/ |
| Langkah Pengujian | 1. Aktor memilih menu menambah modul. 2. Aktor memilih berkas modul yang ingin ditambah 3. Aktor memulai ulang perangkat lunak *web* melalui *web* *server* |
| Hasil yang Diharapkan | Modul berhasil didaftarkan |
| Kondisi Akhir | Modul terdaftar |
| *Skenario 2* | *Aktor menambahkan modul dengan properti yang kurang* |
| Kondisi Awal | Modul belum terdaftar atau modul sudah terdaftar |
| Data Uji | * + 1. Modul yang didaftarkan dengan properti modul: * Nama modul: Modul Penilaian * URL *mapping*: * Lokasi konfigurasi *servlet*: /module-resource/penilaian/servlet-context.xml * Menu: * Nama Menu: Transkrip Nilai * URL Menu: /lihat\_transkrip/ |
| Langkah Pengujian | 1. Aktor memilih menu menambah modul. 2. Aktor memilih berkas modul yang ingin ditambah |
| Hasil yang diharapkan | Penambahan modul gagal dan mengeluarkan pesan error |
| Kondisi Akhir | Modul kembali ke kondisi awal |
| *Skenario 3* | *Aktor menambahkan modul yang sudah terdaftar dengan URL mapping yang belum digunakan* |
| Kondisi Awal | Modul sudah terdaftar |
| Data Uji | * + - 1. Modul yang didaftarkan dengan properti modul: * Nama modul: Modul Penilaian * URL *mapping*: /penilaian/\* * Lokasi konfigurasi *servlet*: /module-resource/penilaian/servlet-context.xml * Menu: * Nama Menu: Transkrip Nilai * URL Menu: /lihat\_transkrip/   1. Modul yang sudah terdaftar * Nama modul: Modul Penilaian * URL *mapping*: /modul\_penilaian/\* * Lokasi konfigurasi *servlet*: /module-resource/servlet-context.xml * Menu: * Nama Menu: Transkrip Nilai * URL Menu: /lihat\_transkrip/ |
| Langkah Pengujian | 1. Aktor memilih menu menambah modul. 2. Aktor memilih berkas modul yang ingin ditambah 3. Aktor memulai ulang perangkat lunak *web* melalui *web* *server* |
| Hasil yang diharapkan | Properti modul diperbarui |
| Kondisi Akhir | Properti modul yang terdaftar diperbarui menjadi:   * Nama modul: Modul Penilaian * URL *mapping*: /penilaian/\* * Lokasi konfigurasi *servlet*: /module-resource/penilaian/servlet-context.xml * Menu: * Nama Menu: Transkrip Nilai * URL Menu: /lihat\_transkrip/ |
| *Skenario 4* | *Aktor menambahkan modul yang sudah terdaftar dengan URL mapping yang sama dengan modul lain* |
| Kondisi Awal | Modul sudah terdaftar |
| Data Uji | * + - 1. Modul yang didaftarkan dengan properti modul: * Nama modul: Modul Penilaian * URL *mapping*: /kurikulum/\* * Lokasi konfigurasi *servlet*: /module-resource/penilaian/servlet-context.xml * Menu: * Nama Menu: Transkrip Nilai * URL Menu: /lihat\_transkrip/   + 1. Modul yang sudah terdaftar * Nama modul: Modul Penilaian * URL *mapping*: /penilaian/\* * Lokasi konfigurasi *servlet*: /module-resource/servlet-context.xml * Menu: * Nama Menu: Transkrip Nilai * URL Menu: /lihat\_transkrip/ * Nama modul: Modul Kurikulum * URL *mapping*: /kurikulum/\* * Lokasi konfigurasi *servlet*: /module-resource/kurikulum/servlet-context.xml * Menu: * Nama Menu: Kelola Rencana Pembelajaran * URL Menu: /kelola\_rp/ |
| Langkah Pengujian | 1. Aktor memilih menu menambah modul. 2. Aktor memilih berkas modul yang ingin ditambah |
| Hasil yang diharapkan | Penambahan modul gagal |
| Kondisi Akhir | Modul tidak terdaftar |



Gambar 5.1 *Error* jika properti modul tidak lengkap



Gambar 5.2 *Error* jika URL *mapping* sudah digunakan

#### Kasus uji menghapus modul

Kasus uji menghapus modul memeriksa apakah modul dapat dihapus dari basis data perangkat lunak dan dari OSGi *Environment*. Penghapusan modul sekaligus menghapus menu semua menu pada modul tersebut. Rincian kasus uji menghapus modul ditunjukkan pada Tabel 5.4.

Tabel 5.4 Tabel kasus uji menghapus modul

|  |  |
| --- | --- |
| ID | TC-002 |
| Kasus Penggunaan | Menghapus modul |
| Nama | Pengujian menghapus modul |
| Tujuan Pengujian | Menguji apakah modul dapat dihapus dari perangkat lunak |
| Kondisi Awal | Modul terdaftar |
| Langkah Pengujian | 1. Aktor memilih menu kelola modul. 2. Aktor memilih modul yang ingin dihapus 3. Aktor menekan tombol hapus |
| Hasil yang Diharapkan | Modul berhasil dihapus |
| Kondisi Akhir | Modul tidak terdaftar |

#### Kasus uji menambah akses menu

Kasus uji ini digunakan untuk menguji apakah peran dan menu dapat dipetakan. Rincian kasus uji ditunjukkan pada Tabel 5.5.

Tabel 5.5 Tabel kasus uji menambah akses menu

|  |  |
| --- | --- |
| ID | TC-003 |
| Kasus Penggunaan | Menambah akses menu |
| Nama | Pengujian menambah akses menu |
| Tujuan Pengujian | Menguji apakah akses menu dapat ditambah |
| Kondisi Awal | Menu tidak terpetakan dengan hak akses/peran |
| Langkah Pengujian | 1. Aktor memilih menu kelola modul 2. Aktor memilih modul yang akses menunya ingin dikelola 3. Aktor menekan tombol “Akses Menu” 4. Aktor memilih menu dan peran yang ingin dipetakan 5. Aktor menyimpan perubahan |
| Hasil yang Diharapkan | Menu berhasil dipetakan dengan peran |
| Kondisi Akhir | Menu terpetakan dengan peran |

#### Kasus uji menghapus akses menu

Kasus uji ini digunakan untuk menguji apakah pemetaan menu terhadap peran yang sudah ada dapat dihapus. Rincian kasus uji ditunjukkan pada Tabel 5.6.

Tabel 5.6 Tabel kasus uji menghapus akses menu

|  |  |
| --- | --- |
| ID | TC-004 |
| Kasus Penggunaan | Menghapus akses menu |
| Nama | Pengujian menghapus akses menu |
| Tujuan Pengujian | Menguji apakah akses menu dapat dihapus |
| Kondisi Awal | Menu terpetakan dengan hak akses/peran |
| Langkah Pengujian | 1. Aktor memilih menu kelola modul 2. Aktor memilih modul yang akses menunya ingin dikelola 3. Aktor menekan tombol “Akses Menu” 4. Aktor menghapus menu yang sudah dipetakan untuk dihapus 5. Aktor menyimpan perubahan |
| Hasil yang Diharapkan | Pemetaan menu dan peran dapat terhapus |
| Kondisi Akhir | Pemetaan menu dan peran dapat terhapus |

#### Kasus uji menambah pengguna

Kasus uji ini digunakan untuk menguji apakah pengguna perangkat lunak dapat didaftarkan. Kasus uji ini memiliki dua skenario yaitu penambahan satu per satu atau dengan berkas CSV. Rincian kasus uji ditunjukkan pada Tabel 5.7.

Tabel 5.7 Tabel kasus uji menambah pengguna

|  |  |
| --- | --- |
| ID | TC-005 |
| Kasus Penggunaan | Menambah pengguna |
| Nama | Pengujian menambah pengguna |
| Tujuan Pengujian | Menguji apakah pengguna dapat ditambah |
| *Skenario 1* | *Aktor menambahkan pengguna satu per satu* |
| Kondisi Awal | Pengguna belum terdaftar |
| Langkah Pengujian | 1. Aktor memilih menu kelola pengguna. 2. Aktor memilih pilihan tambah pengguna 3. Aktor memilih pilihan menambah pengguna satu per satu 4. Aktor mengisi isian NRP, username, tipe pengguna, dan satuan manajemen. 5. Aktor menyimpan pengguna |
| Hasil yang Diharapkan | Pengguna berhasil didaftarkan |
| Kondisi Akhir | Pengguna terdaftar |
| *Skenario 2* | *Aktor mengunggah berkas CSV* |
| Kondisi Awal | Pengguna belum terdaftar |
| Langkah Pengujian | 1. Aktor memilih menu kelola pengguna. 2. Aktor memilih pilihan tambah pengguna 3. Aktor memilih pilihan unggah CSV 4. Aktor memilih berkas CSV 5. Aktor mengunggah berkas yang dipilih |
| Hasil yang diharapkan | Pengguna berhasil didaftarkan |
| Kondisi Akhir | Pengguna terdaftar |

#### Kasus uji menonaktifkan pengguna

Kasus uji ini digunakan untuk menguji apakah pengguna perangkat lunak yang sudah terdaftar dapat dinonaktifkan. Rincian kasus uji ditunjukkan pada Tabel 5.8.

Tabel 5.8 Tabel kasus uji menonaktifkan pengguna

|  |  |
| --- | --- |
| ID | TC-006 |
| Kasus Penggunaan | Menonaktifkan pengguna |
| Nama | Pengujian menonaktifkan pengguna |
| Tujuan Pengujian | Menguji apakah pengguna dapat dinonaktifkan |
| Kondisi Awal | Pengguna terdaftar |
| Langkah Pengujian | 1. Aktor memilih menu kelola pengguna. 2. Aktor memilih pengguna yang ingin dihapus 3. Aktor memilih menu hapus |
| Hasil yang Diharapkan | Pengguna berhasil dinonaktifkan |
| Kondisi Akhir | Pengguna terdaftar namun berstatus non aktif |

#### Kasus uji menambah hak akses pengguna

Kasus uji ini digunakan untuk menguji apakah pengguna dapat diberikan hak akses. Rincian kasus uji ditunjukkan pada Tabel 5.9.

Tabel 5.9 Tabel kasus uji menambah hak akses pengguna.

|  |  |
| --- | --- |
| ID | TC-007 |
| Kasus Penggunaan | Menambah hak akses pengguna |
| Nama | Pengujian menambah hak akses pengguna |
| Tujuan Pengujian | Menguji apakah hak akses pengguna dapat ditambah |
| Kondisi Awal | Pengguna belum memiliki hak akses yang ingin ditambahkan |
| Langkah Pengujian | 1. Aktor memilih menu kelola pengguna 2. Aktor memilih pengguna terkait hak akses yang ingin ditambahkan 3. Aktor mengisi isian peran yang ingin ditambahkan ke pengguna sebagai hak akses 4. Aktor memilih pilihan untuk menyimpan perubahan |
| Hasil yang Diharapkan | Penambahan hak akses pengguna berhasil |
| Kondisi Akhir | Pengguna memiliki hak akses yang diinginkan |

#### Kasus uji menghapus hak akses pengguna

Kasus uji ini digunakan untuk menguji apakah hak akses yang sudah dimiliki pengguna dapat dihapus. Rincian kasus uji ditunjukkan pada Tabel 5.10.

Tabel 5.10 Tabel kasus uji menghapus hak akses pengguna.

|  |  |
| --- | --- |
| ID | TC-008 |
| Kasus Penggunaan | Menghapus hak akses pengguna |
| Nama | Pengujian menghapus hak akses pengguna |
| Tujuan Pengujian | Menguji apakah hak akses pengguna dapat dihapus |
| Kondisi Awal | Pengguna memiliki hak akses yang ingin dihapus |
| Langkah Pengujian | 1. Aktor memilih menu kelola pengguna 2. Aktor memilih pengguna terkait hak akses yang ingin dihapus dari pengguna 3. Aktor memilih pilihan hapus hak akses yang ingin dihapus dari pengguna 4. Aktor menyimpan perubahan |
| Hasil yang Diharapkan | Hak akses pengguna berhasil dihapus. |
| Kondisi Akhir | Pengguna tidak memiliki hak akses yang ingin dihapus |

#### Kasus uji menambah peran

Kasus uji ini digunakan untuk menguji apakah peran dapat ditambah ke dalam perangkat lunak untuk digunakan sebagai hak akses. Rincian kasus uji ditunjukkan pada Tabel 5.11.

Tabel 5.11 Tabel kasus uji menambah peran

|  |  |
| --- | --- |
| ID | TC-009 |
| Kasus Penggunaan | Menambah peran |
| Nama | Pengujian menambah peran |
| Tujuan Pengujian | Menguji apakah peran dapat ditambahkan |
| Kondisi Awal | Peran tidak terdaftar |
| Langkah Pengujian | 1. Aktor memilih menu kelola peran 2. Aktor memilih pilihan tambah peran 3. Aktor mengisi isian nama peran 4. Aktor memilih pilihan tambah |
| Hasil yang Diharapkan | Peran berhasil ditambah |
| Kondisi Akhir | Peran terdaftar |

#### Kasus uji menghapus peran

Kasus uji ini digunakan untuk menguji apakah peran yang terdaftar dapat dihapus dari perangkat. Rincian kasus uji ditunjukkan pada Tabel 5.12.

Tabel 5.12 Tabel kasus uji menghapus peran

|  |  |
| --- | --- |
| ID | TC-010 |
| Kasus Penggunaan | Menghapus peran |
| Nama | Pengujian menghapus peran |
| Tujuan Pengujian | Menguji apakah peran dapat dihapus |
| Kondisi Awal | Peran terdaftar |
| Langkah Pengujian | 1. Aktor memilih menu kelola peran 2. Aktor memilih peran yang ingin dihapus 3. Aktor menekan tombol hapus |
| Hasil yang Diharapkan | Peran berhasil dihapus |
| Kondisi Akhir | Peran tidak terdaftar |

### Uji coba integrasi modul

Modul yang diuji coba merupakan modul-modul yang telah dibuat pada tugas akhir lain. Modul tersebut ditunjukkan pada Tabel 5.13. Pengujian pada modul tidak dilakukan dengan kasus uji karena sudah dibahas pada masing-masing buku melainkan dilakukan dengan menjalankan semua menu yang ada pada suatu modul dan memeriksa apakah menu dapat dijalankan dengan normal seperti saat modul masih berdiri sendiri. Menu-menu tiap modul ditunjukkan pada Tabel 5.14, Tabel 5.15, Tabel 5.16, dan Tabel 5.17.

**Tabel 5.13 Tabel daftar modul yang diuji coba**

|  |  |
| --- | --- |
| **No** | **Nama Modul** |
| 1 | Modul Ekivalensi [4] |
| 2 | Modul Kurikulum [3] |
| 3 | Modul Pembelajaran [5] |
| 4 | Modul Penilaian [6] |

Tabel 5.14 Tabel daftar menu pada modul kurikulum [3]

|  |  |
| --- | --- |
| **No** | **Nama Menu** |
| 1 | Kelola Bentuk Penilaian |
| 2 | Kelola Capaian Belajar Mata Kuliah |
| 3 | Kelola Capaian Belajar Satuan Manajemen |
| 4 | Kelola Kurikulum |
| 5 | Kelola Mata Kuliah |
| 6 | Kelola Mata Kuliah Prasyarat |
| 7 | Kelola Mata Kuliah untuk Satuan Manajemen |
| 8 | Kelola Metode Pembelajaran |
| 9 | Kelola Pustaka Mata Kuliah |
| 10 | Kelola Rencana Pembelajaran |
| 11 | Kelola Rumpun Matakuliah |
| 12 | Kelola Silabus |
| 13 | Laporan Rencana Pembelajaran |
| 14 | Laporan Silabus |

Tabel 5.15 Tabel daftar menu pada modul pembelajaran [5]

|  |  |
| --- | --- |
| **No** | **Nama Menu** |
| 1 | Absensi |
| 2 | Berita Acara |
| 3 | Kartu Rencana Studi |
| 4 | Kelola Aturan Pengganti |
| 5 | Kelola Batas Pengambilan KRS |
| 6 | Kelola Pembelajaran |
| 7 | Kelola Pendamping Akademik |
| 8 | Kelola Periode |
| 9 | Kelola Rombongan Belajar |
| 10 | Kelola Semester |
| 11 | Kelola Status Absensi |
| 12 | Kelola Tahun Ajaran |
| 13 | Laporan Pembayaran |
| 14 | Laporan Pertemuan |
| 15 | Monitoring Pengambilan KRS |
| 16 | Persetujuan Kartu Rencana Studi |

Tabel 5.16 Tabel daftar menu pada modul penilaian [6]

|  |  |
| --- | --- |
| **No** | **Nama Menu** |
| 1 | Isi Kuisioner |
| 2 | Kelola Kuisioner |
| 3 | Kelola Nilai |
| 4 | Konversi Nilai |
| 5 | Laporan Kuisioner Per Kelas |
| 6 | Laporan Kuisioner Periodik |
| 7 | Peringkat IPK |
| 8 | Peringkat IPS |
| 9 | Lihat Nilai Per Kelas |
| 10 | Nilai Per Periode |
| 11 | Transkrip Nilai |
| 12 | Perbaharui IP |

Tabel 5.17 Tabel daftar menu pada modul ekuivalensi [4]

|  |  |
| --- | --- |
| **No** | **Nama Menu** |
| 1 | Kelola Katalog Alih Jenjang |
| 2 | Kelola Matakuliah Alih Jenjang |
| 3 | Kelola Katalog Satman |
| 4 | Manajemen Mahasiswa Alih Jenjang |
| 5 | Laporan Alih Jenjang |
| 6 | Laporan Ekuivalensi |
| 7 | Persetujuan Ekuivalensi |
| 8 | Ekuivalensi Kurikulum |
| 9 | Kelola Katalog Satman |
| 10 | Ekuivalensi Alihjenjang |
| 11 | Ekuivalensi Alih Jenjang |

### Uji coba hak akses

Uji coba ini dilakukan untuk menguji apakah hak akses berhasil diterapkan ke dalam menu-menu pada modul. Aspek pengujian yang dilihat pada pengujian ini adalah apakah suatu menu hanya dapat diakses oleh peran tertentu yang sudah dinyatakan saja. Peran-peran yang sudah terdaftar pada basis data ditunjukkan pada Tabel 5.18. Daftar menu-menu pada modul dengan hak aksesnya ditunjukkan pada Tabel 5.19, Tabel 5.20, Tabel 5.21, dan Tabel 5.22. Dapat dilihat pada daftar peran terdapat peran Admin dan System Admin. Kedua peran tersebut berbeda dimana peran Admin hanya memiliki fungsi administrator pada modul sementara SystemAdmin adalah administrator yang mengurus fungsi administratif perangkat lunak ini.

Tabel 5.18 Tabel daftar peran

|  |  |
| --- | --- |
| **No** | **Nama Peran** |
| 1 | Kepala |
| 2 | Pendidik |
| 3 | Tenaga Kependidikan |
| 4 | Tim Ekuivalensi |
| 5 | Admin |
| 6 | SystemAdmin |
| 7 | Peserta Didik |

Tabel 5.19 Tabel daftar akses menu dari modul kurikulum [3]

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Menu** | **Peran** |
| 1 | Kelola Bentuk Penilaian | Admin |
| 2 | Kelola Capaian Belajar Mata Kuliah | Kepala |
| 3 | Kelola Capaian Belajar Satuan Manajemen | Kepala |
| 4 | Kelola Kurikulum | Admin |
| 5 | Kelola Mata Kuliah | Admin |
| 6 | Kelola Mata Kuliah Prasyarat | Admin |
| 7 | Kelola Mata Kuliah untuk Satuan Manajemen | Admin |
| 8 | Kelola Metode Pembelajaran | Admin |
| 9 | Kelola Pustaka Mata Kuliah | Admin |
| 10 | Kelola Rencana Pembelajaran | Pendidik |
| 11 | Kelola Rumpun Matakuliah | Admin |
| 12 | Kelola Silabus | Kepala |
| 13 | Laporan Rencana Pembelajaran | Pendidik, Peserta Didik |
| 14 | Laporan Silabus | Kepala, Pendidik, Peserta Didik |

Tabel 5.20 Tabel daftar akses menu dari modul pembelajaran [5]

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Menu** | **Peran** |
| 1 | Absensi | Tenaga Kependidikan |
| 2 | Berita Acara | Pendidik |
| 3 | Kartu Rencana Studi | Peserta Didik |
| 4 | Kelola Aturan Pengganti | Admin |
| 5 | Kelola Batas Pengambilan KRS | Admin |
| 6 | Kelola Pembelajaran | Tenaga Kependidikan |
| 7 | Kelola Pendamping Akademik | Tenaga Kependidikan |
| 8 | Kelola Periode | Admin |
| 9 | Kelola Rombongan Belajar | Tenaga Kependidikan |
| 10 | Kelola Semester | Admin |
| 11 | Kelola Status Absensi | Admin |
| 12 | Kelola Tahun Ajaran | Admin |
| 13 | Laporan Pembayaran | Kepala |
| 14 | Laporan Pertemuan | Kepala |
| 15 | Monitoring Pengambilan KRS | Admin |
| 16 | Persetujuan Kartu Rencana Studi | Pendidik |

Tabel 5.21 Tabel daftar akses menu dari modul penilaian [6]

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Menu** | **Peran** |
| 1 | Isi Kuisioner | Peserta Didik |
| 2 | Kelola Kuisioner | Admin |
| 3 | Kelola Nilai | Pendidik |
| 4 | Konversi Nilai | Admin |
| 5 | Laporan Kuisioner Per Kelas | Admin, Kepala |
| 6 | Laporan Kuisioner Periodik | Admin, Kepala |
| 7 | Peringkat IPK | Admin, Kepala, Pendidik, Peserta Didik, Tenaga Kependidikan |
| 8 | Peringkat IPS | Admin, Kepala, Pendidik, Peserta Didik, Tenaga Kependidikan |
| 9 | Lihat Nilai Per Kelas | Admin, Kepala, Pendidik, Peserta Didik, Tenaga Kependidikan |
| 10 | Nilai Per Periode | Admin, Kepala, Pendidik, Peserta Didik, Tenaga Kependidikan |
| 11 | Transkrip Nilai | Admin, Kepala, Pendidik, Peserta Didik, Tenaga Kependidikan |
| 12 | Perbaharui IP | Admin |

Tabel 5.22 Tabel daftar akses menu dari modul ekuivalensi [4]

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Menu** | **Peran** |
| 1 | Kelola Katalog Alih Jenjang | Admin |
| 2 | Kelola Matakuliah Alih Jenjang | Admin |
| 3 | Kelola Katalog Satman | Admin |
| 4 | Manajemen Mahasiswa Alih Jenjang | Admin |
| 5 | Laporan Alih Jenjang | Admin |
| 6 | Laporan Ekuivalensi | Peserta Didik, Tenaga Kependidikan |
| 7 | Persetujuan Ekuivalensi | Tim Ekuivalensi |
| 8 | Ekuivalensi Kurikulum | Tim Ekuivalensi |
| 9 | Kelola Katalog Satman | Tim Ekuivalensi |
| 10 | Ekuivalensi Alihjenjang | Tim Ekuivalensi |
| 11 | Ekuivalensi Alih Jenjang | Tim Ekuivalensi |

## Evaluasi

Pada Subbab ini dijelaskan hasil dari pengujian yang dilakukan pada Subbab sebelumnya. Sama seperti Subbab sebelumnya, evaluasi dibagi menjadi tiga yaitu evaluasi pengujian fungsionalitas administratif, evaluasi integrasi modul, dan evaluasi hak akses.

### Evaluasi fungsionalitas administratif

Evaluasi ini adalah hasil dari pengujian kasus uji pada Subbab 5.2.1. Hasil dinyatakan dalam status terpenuhi atau tidak. Hasil tersebut ditunjukkan pada Tabel 5.23.

Tabel 5.23 Tabel evaluasi kasus uji fungsionalitas administratif

|  |  |  |
| --- | --- | --- |
| No. | Kode Kasus Pengujian | Terpenuhi |
| 1 | TC-001 | √ |
| 2 | TC -002 | √ |
| 3 | TC -003 | √ |
| 4 | TC -004 | √ |
| 5 | TC -005 | √ |
| 6 | TC -006 | √ |
| 7 | TC -007 | √ |
| 8 | TC -008 | √ |
| 9 | TC -009 | √ |
| 10 | TC -010 | √ |

Berdasarkan Tabel 5.23 dapat disimpulkan bahwa seluruh fungsionalitas administratif dapat dijalankan oleh aktor dan semua kasus uji coba terpenuhi.

### Evaluasi Integrasi modul

Evaluasi ini adalah hasil dari uji coba integrasi modul. Hasil menyatakan apakah menu-menu pada modul dapat berjalan dan jika tidak, disertakan penyebab kegagalan menu tersebut. Hasil ditunjukkan pada Tabel 5.24, Tabel 5.25, Tabel 5.26, dan Tabel 5.27.

Tabel 5.24 Tabel evaluasi integrasi modul kurikulum [3]

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Menu** | **Terpenuhi** | **Keterangan** |
| 1 | Kelola Bentuk Penilaian | √ |  |
| 2 | Kelola Capaian Belajar Mata Kuliah | X | Masalah pada javascript |
| 3 | Kelola Capaian Belajar Satuan Manajemen | X | Masalah pada javascript |
| 4 | Kelola Kurikulum | √ |  |
| 5 | Kelola Mata Kuliah | √ |  |
| 6 | Kelola Mata Kuliah Prasyarat | √ |  |
| 7 | Kelola Mata Kuliah untuk Satuan Manajemen | √ |  |
| 8 | Kelola Metode Pembelajaran | √ |  |
| 9 | Kelola Pustaka Mata Kuliah | √ |  |
| 10 | Kelola Rencana Pembelajaran | X | Masalah pada javascript |
| 11 | Kelola Rumpun Matakuliah | √ |  |
| 12 | Kelola Silabus | √ |  |
| 13 | Laporan Rencana Pembelajaran | X | Masalah pada javascript dan kemungkinan masalah pada JasperReports |
| 14 | Laporan Silabus | X | Masalah pada javascript dan kemungkinan masalah pada JasperReports |

Tabel 5.25 Tabel evaluasi integrasi modul pembelajaran [5]

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Menu** | **Terpenuhi** | **Keterangan** |
| 1 | Absensi | √ |  |
| 2 | Berita Acara | X | Kemungkinan masalah pada pustaka joda.time yang gagal melakukan deserializing data dari JSON yang berupa tanggal |
| 3 | Kartu Rencana Studi | √ |  |
| 4 | Kelola Aturan Pengganti | X | Kemungkinan masalah pada pustaka joda.time yang gagal melakukan deserializing data dari JSON yang berupa tanggal |
| 5 | Kelola Batas Pengambilan KRS | √ |  |
| 6 | Kelola Pembelajaran | √ |  |
| 7 | Kelola Pendamping Akademik | √ |  |
| 8 | Kelola Periode | X | Kemungkinan masalah pada pustaka joda.time yang gagal melakukan deserializing data dari JSON yang berupa tanggal |
| 9 | Kelola Rombongan Belajar | √ |  |
| 10 | Kelola Semester | √ |  |
| 11 | Kelola Status Absensi | √ |  |
| 12 | Kelola Tahun Ajaran | √ |  |
| 13 | Laporan Pembayaran | √ |  |
| 14 | Laporan Pertemuan | √ |  |
| 15 | Monitoring Pengambilan KRS | √ |  |
| 16 | Persetujuan Kartu Rencana Studi | √ |  |

Tabel 5.26 Tabel evaluasi integrasi modul penilaian [6]

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Menu** | **Terpenuhi** | **Keterangan** |
| 1 | Isi Kuisioner | √ |  |
| 2 | Kelola Kuisioner | √ |  |
| 3 | Kelola Nilai | √ |  |
| 4 | Konversi Nilai | √ |  |
| 5 | Laporan Kuisioner Per Kelas | √ |  |
| 6 | Laporan Kuisioner Periodik | √ |  |
| 7 | Peringkat IPK | √ |  |
| 8 | Peringkat IPS | √ |  |
| 9 | Lihat Nilai Per Kelas | √ |  |
| 10 | Nilai Per Periode | √ |  |
| 11 | Transkrip Nilai | X | Kemungkinan masalah pada versi JasperReporting yang digunakan |
| 12 | Perbaharui IP | √ |  |

Tabel 5.27 Tabel evaluasi integrasi modul ekuivalensi [4]

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Menu** | **Terpenuhi** | **Keterangan** |
| 1 | Kelola Katalog Alihjenjang | √ |  |
| 2 | Kelola Matakuliah Alihjenjang | √ |  |
| 3 | Kelola Katalog Satman | √ |  |
| 4 | Manajemen Mahasiswa Alihjenjang | X | Kemungkinan masalah pada pustaka joda.time yang gagal melakukan deserializing data dari JSON yang berupa tanggal |
| 5 | Laporan Alihjenjang | X | Kemungkinan masalah pada versi JasperReporting yang digunakan |
| 6 | Laporan Ekuivalensi | X | Kemungkinan masalah pada versi JasperReporting yang digunakan |
| 7 | Persetujuan Ekuivalensi | √ |  |
| 8 | Ekuivalensi Kurikulum | √ |  |
| 9 | Kelola Katalog Satman | √ |  |
| 10 | Ekuivalensi Alihjenjang | X | Memerlukan input dari menu Manajemen Mahasiswa Alihjenjang |

Berdasarkan keempat tabel diatas, yaitu Tabel 5.24, Tabel 5.25, Tabel 5.26, dan Tabel 5.27, hampir semua menu pada modul yang sudah diintegrasikan dapat dijalankan dan terdapat beberapa menu yang tidak dapat dijalankan. Hal-hal yang mengakibatkan tidak dapat berjalannya menu-menu tersebut antara lain:

* 1. Versi JasperReport. JasperReport merupakan sebuah pustaka pembuatan laporan dalam bentuk berkas PDF ataupun berkas lain. Versi pustaka yang digunakan pada modul dan versi yang disediakan pada perangkat lunak utama tidak cocok sehingga laporan yang sudah dibuat pada modul perlu diganti menyesuaikan versi pustaka pada perangkat lunak utama. Menu-menu yang tidak dapat dijalankan karena masalah ini antara lain:
* Menu laporan rencana pembelajaran dan menu laporan silabus pada modul kurikulum.
* Menu transkrip nilai pada modul penilaian.
* Menu laporan alihjenjang dan laporan ekivalensi pada modul ekivalensi
  1. Kelas-kelas pustaka joda.time yang tidak dapat terbaca oleh perankat lunak utama pada saat *runtime.* Menu-menu yang terpengaruhi adalah menu-menu yang memerlukan format tanggal pada input nya, yaitu antara lain:
* Menu berita acara, kelola aturan pengganti, dan kelola periode pada modul pembelajaran.
* Menu manajemen mahasiswa alihjenjang pada modul ekivalensi yang mengakibatkan tidak berjalannya menu ekivalensi alihjenjang pada modul yang sama.
  1. Kesalahan pada pembuatan javascript pada menu modul. Menu yang terpengaruh masalah ini adalah menu kelola capaian belajar mata kuliah, capaian belajar satuan manajemen, dan kelola rencana pembelajaran pada modul kurikulum.

### Evaluasi hak akses modul

Evaluasi hak akses modul adalah hasil dari uji coba hak akses. Hasil yang ditampilkan yaitu berupa tabel yang berisi nama menu pada tiap modul dengan rincian apakah suatu peran dapat mengakses menu tersebut. Tabel tersebut ditunjukkan pada Tabel 5.28, Tabel 5.29, Tabel 5.30, dan Tabel 5.31.

Tabel 5.28 Tabel evaluasi hak akses untuk modul kurikulum [3]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Nama Menu** | **Kepala** | **Pendidik** | **Tenaga Kependidikan** | **Tim Ekuivalensi** | **Admin** | **Peserta Didik** |
| 1 | Kelola Bentuk Penilaian | X | X | X | X | √ | X |
| 2 | Kelola Capaian Belajar Mata Kuliah | √ | X | X | X | X | X |
| 3 | Kelola Capaian Belajar Satuan Manajemen | √ | X | X | X | X | X |
| 4 | Kelola Kurikulum | X | X | X | X | √ | X |
| 5 | Kelola Mata Kuliah | X | X | X | X | √ | X |
| 6 | Kelola Mata Kuliah Prasyarat | X | X | X | X | √ | X |
| 7 | Kelola Mata Kuliah untuk Satuan Manajemen | X | X | X | X | √ | X |
| 8 | Kelola Metode Pembelajaran | X | X | X | X | √ | X |
| 9 | Kelola Pustaka Mata Kuliah | X | X | X | X | √ | X |
| 10 | Kelola Rencana Pembelajaran | X | √ | X | X | X | X |
| 11 | Kelola Rumpun Matakuliah | X | X | X | X | √ | X |
| 12 | Kelola Silabus | √ | X | X | X | X | X |
| 13 | Laporan Rencana Pembelajaran | X | √ | X | X | X | √ |
| 14 | Laporan Silabus | √ | √ | X | X | X | √ |

Tabel 5.29 Tabel evaluasi hak akses untuk modul pembelajaran [5]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Nama Menu** | **Kepala** | **Pendidik** | **Tenaga Kependidikan** | **Tim Ekuivalensi** | **Admin** | **Peserta Didik** |
| 1 | Absensi | X | X | √ | X | X | X |
| 2 | Berita Acara | X | √ | X | X | X | X |
| 3 | Kartu Rencana Studi | X | X | X | X | X | √ |
| 4 | Kelola Aturan Pengganti | X | X | X | X | √ | X |
| 5 | Kelola Batas Pengambilan KRS | X | X | X | X | √ | X |
| 6 | Kelola Pembelajaran | X | X | √ | X | X | X |
| 7 | Kelola Pendamping Akademik | X | X | √ | X | X | X |
| 8 | Kelola Periode | X | X | X | X | √ | X |
| 9 | Kelola Rombongan Belajar | X | X | √ | X | X | X |
| 10 | Kelola Semester | X | X | X | X | √ | X |
| 11 | Kelola Status Absensi | X | X | X | X | √ | X |
| 12 | Kelola Tahun Ajaran | X | X | X | X | √ | X |
| 13 | Laporan Pembayaran | √ | X | X | X | X | X |
| 14 | Laporan Pertemuan | √ | X | X | X | X | X |
| 15 | Monitoring Pengambilan KRS | X | X | X | X | √ | X |
| 16 | Persetujuan Kartu Rencana Studi | X | √ | X | X | X | X |

Tabel 5.30 Tabel evaluasi hak akses untuk modul penilaian [6]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Nama Menu** | **Kepala** | **Pendidik** | **Tenaga Kependidikan** | **Tim Ekuivalensi** | **Admin** | **Peserta Didik** |
| 1 | Isi Kuisioner | X | X | X | X | X | √ |
| 2 | Kelola Kuisioner | X | X | X | X | √ | X |
| 3 | Kelola Nilai | X | √ | X | X | X | X |
| 4 | Konversi Nilai | X | X | X | X | √ | X |
| 5 | Laporan Kuisioner Per Kelas | √ | X | X | X | √ | X |
| 6 | Laporan Kuisioner Periodik | √ | X | X | X | √ | X |
| 7 | Peringkat IPK | √ | √ | √ | X | √ | √ |
| 8 | Peringkat IPS | √ | √ | √ | X | √ | √ |
| 9 | Lihat Nilai Per Kelas | √ | √ | √ | X | √ | √ |
| 10 | Nilai Per Periode | √ | √ | √ | X | √ | √ |
| 11 | Transkrip Nilai | √ | √ | √ | X | √ | √ |
| 12 | Perbaharui IP | X | X | X | X | X | X |

Tabel 5.31 Tabel evaluasi hak akses untuk modul ekuivalensi [4]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Nama Menu** | **Kepala** | **Pendidik** | **Tenaga Kependidikan** | **Tim Ekuivalensi** | **Admin** | **Peserta Didik** |
| 1 | Kelola Katalog Alihjenjang | X | X | X | X | √ | X |
| 2 | Kelola Matakuliah Alihjenjang | X | X | X | X | √ | X |
| 3 | Kelola Katalog Satman | X | X | X | X | √ | X |
| 4 | Manajemen Mahasiswa Alihjenjang | X | X | X | X | √ | X |
| 5 | Laporan Alihjenjang | X | X | X | X | √ | X |
| 6 | Laporan Ekuivalensi | X | X | √ | X | X | √ |
| 7 | Persetujuan Ekuivalensi | X | X | X | √ | X | X |
| 8 | Ekuivalensi Kurikulum | X | X | X | √ | X | X |
| 9 | Kelola Katalog Satman | X | X | X | √ | X | X |
| 10 | Ekuivalensi Alihjenjang | X | X | X | √ | X | X |

Berdasarkan tabel evaluasi hak akses modul, yaitu Tabel 5.28, Tabel 5.29, Tabel 5.30, Tabel 5.31, implementasi otorisasi menu untuk membatasi akses menu pada modul yang sudah diintegrasi berhasil.

# BAB VI KESIMPULAN DAN SARAN

Bab ini membahas kesimpulan yang dapat diambil dari hasil uji coba dan perancangan perangkat lunak sebagai jawaban dari rumusan masalah yang telah dikemukakan dan saran yang berisi pengembangan yang dapat dilakukan lebih lanjut untuk menyempurnakan perangkat lunak.



## Kesimpulan

Berikut merupakan kesimpulan yang dapat diambil dari proses pengembangan dan hasil uji coba.

1. Perangkat lunak telah dapat mengintegrasikan fungsionalitas modul. Perangkat lunak telah dapat mengintegrasikan modul-modul generik dari tugas akhir sebelumnya dan menjalankan menu di dalam modul-modul tersebut. Modul yang diintegrasikan dalam perangkat lunak ini antara lain modul kurikulum [3], modul pembelajaran [5], modul penilaian [6], dan modul ekuivalensi [4]. Bagan modul-modul tersebut dapat dilihat pada Gambar 6.1.
2. Terdapat beberapa menu pada modul yang tidak dapat dijalankan yang kemungkinan dikarenakan masalah dependensi proyek terhadap JasperReports dan Joda-Time pada lingkungan OSGi dan implementasi Javascript pada menu-menu tersebut.
3. Modularitas perangkat lunak dapat dicapai dengan menggunakan OSGi.
4. Modul wajib dibuat dalam bentuk OSGi *fragment* *bundle* dan mengimplementasi spesifikasi/aturan yang telah ditetapkan perangkat lunak utama agar dapat dibaca dan dibedakan dengan modul yang lain.
5. Penggunaan *fragment* *bundle* sebagai bentuk modul memiliki kekurangan pada proses OSGi *dependency* *resolving* dimana dependensi OSGi yang diperlukan modul harus dinyatakan pada perangkat lunak utama.

## Saran

Berikut ini merupakan pengembangan lebih lanjut yang dapat dilakukan untuk menyempurnakan perangkat lunak.

* 1. Baik modul yang belum terintegrasi dan sudah terintegrasi dapat ditambah atau diganti untuk menyesuaikan proses bisnis sistem informasi akademik pada perguruan tinggi tertentu. Contoh modul sistem informasi akademik yang mungkin dapat ditambah ditunjukkan sebagai kotak berwarna putih pada Gambar 6.1.
  2. Perangkat lunak utama perlu dikembangkan lebih lanjut untuk membaca modul dalam bentuk OSGi *bundle* mengingat adanya kekurangan pada OSGi *fragment* *bundle*. Pengembangan yang dilakukan untuk masalah ini adalah pengembangan komponen *ViewResolver* pada Spring MVC agar dapat membaca berkas JSP antar OSGi *bundle*.
  3. Perangkat lunak utama perlu dikembangkan lebih lanjut untuk fitur *hot* *deployment* agar perangkat lunak utama tidak perlu di-*restart* ketika ada penambahan modul.
  4. Versi bundle dan dependensinya pada lingkungan OSGi perlu disesuaikan untuk menghindari masalah pustaka seperti JasperReports.



Gambar 6.1 Modul-modul SIAKAD yang sudah terintegrasi dan yang dapat dikembangkan lebih lanjut

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | PDDIKTI, "SRV5 PDDIKTI : Pangkalan Data Pendidikan Tinggi," PDDIKTI, 2013. [Online]. Available: http://forlap.dikti.go.id/perguruantinggi/homegraphpt. [Accessed 29 Desember 2015]. |
| [2] | A. Rouf, "PENGUJIAN PERANGKAT LUNAK DENGAN MENGGUNAKAN METODEWHITE BOX DAN BLACK BOX," *STIMIK HIMSYA JOURNAL,* vol. VIII, no. 1, pp. 3-6, 2012. |
| [3] | A. T. Averousi, Rancang Bangun Perangkat Lunak Sistem Informasi Akademik Generik Pada Modul Kurikulum, Surabaya: Institut Teknologi Sepuluh Nopember, 2015. |
| [4] | B. A. Alfirdaus, Rancang Bangun Perangkat Lunak Sistem Informasi Akademik Generik Berbasis Web Pada Modul Ekuivalensi, Surabaya: Institut Teknologi Sepuluh Nopember, 2015. |
| [5] | G. P. N. Suminto, Rancang Bangun Commercial Off The Shelf (COTS) Sistem Informasi Akademik Berbasis Web Pada Modul Kelola Pembelajaran, Surabaya: Institut Teknologi Sepuluh Nopember, 2015. |
| [6] | H. Rahman, Rancang Bangun Sistem Informasi Akademik Generik Pada Modul Penilaian Menggunakan Pola Perancangan Hierarchical Model-View-Controller, Surabaya: Institut Teknologi Sepuluh Nopember, 2015. |
| [7] | C. Walls, Modular Java - Creating Flexible Applications with OSGi and Spring, The Pragmatic Bookshelf, 2009. |
| [8] | J. Cai, R. Kapila and G. Pal, "HMVC: The layered pattern for developing strong client tiers," JavaWorld, Inc, 21 Juli 2000. [Online]. Available: http://www.javaworld.com/article/2076128/design-patterns/hmvc--the-layered-pattern-for-developing-strong-client-tiers.html. [Accessed 29 Desember 2015]. |
| [9] | "Introduction of Spring Framework," Spring, [Online]. Available: http://docs.spring.io/spring-framework/docs/3.0.x/reference/overview.html. [Accessed 6 April 2015]. |
| [10] | R. Harrop, "OSGi Concepts," Eclipse Virgo, 2011. [Online]. Available: http://www.eclipse.org/virgo/documentation/virgo-documentation-3.6.4.RELEASE/docs/virgo-user-guide/htmlsingle/virgo-user-guide.html#concepts.osgi. [Accessed 29 Desember 2015]. |
| [11] | T. A. S. Foundation, "Apache Tomcat," The Apache Software Foundation, [Online]. Available: http://tomcat.apache.org/. [Accessed 2015 Desember 29]. |
| [12] | T. P. G. D. Group, "PostgreSQL 9.3.10 Documentation," The PostgreSQL Global Development Group, [Online]. Available: http://www.postgresql.org/docs/9.3/static/intro-whatis.html. [Accessed 2015 Desember 29]. |
| [13] | J2EEBrain, "Hibernate Object Relational Mapping," J2EEBrain, [Online]. Available: http://www.j2eebrain.com/java-J2ee-hibernate-object-relational-mapping.html. [Accessed 29 Desember 2015]. |
| [14] | B. Varanasi and S. Belida, "Maven Dependency Management," in *Introducing Maven*, Apress, 2014, p. 15. |
| [15] | B. Varanasi and S. Belida, "Transitive Dependencies," in *Introducing Maven*, Apress, 2014, p. 19. |
| [16] | B. Varanasi and S. Belida, "Dependency Scope," in *Introducing Maven*, Apress, 2014, p. 21. |
| [17] | B. Varanasi and S. Belida, "Maven Life Cycle," in *Introducing Maven*, Apress, 2014, p. 37. |
| [18] | M. Fayad and D. C. Schmidt, "Object-Oriented Application Frameworks," 28 September 2006. [Online]. Available: http://www1.cse.wustl.edu/~schmidt/CACM-frameworks.html. [Accessed 2016 Januari 14]. |
| [19] | MvnRepository, "MavenRepository," Maven, 2006. [Online]. Available: http://mvnrepository.com/. [Accessed 29 Desember 2015]. |
| [20] | SpringSource, "SpringSource Enterprise Bundle Repository," SpringSource, 2008. [Online]. Available: http://ebr.springsource.com/repository/app/. [Accessed 29 Desember 2015]. |
| [21] | M. Deinum and K. Serneels, "The request processing workflow," in *Pro Spring MVC*, New York, Apress, 2012, p. 66. |

***[Halaman ini sengaja dikosongkan]***

# BIODATA PENULIS

Salahudin Agung Wijaya, lahir di Jakarta, pada tanggal 15 April 1993. Penulis menempuh pendidikan dasar di SD Islam Pondok Duta Depok (1999-2005), pendidikan menengah pertama di SMP Negeri 103 Jakarta (2005-2008), pendidikan menengah atas di SMA Negeri 58 Jakarta (2008-2011) dan pendidikan tinggi di S1 Teknik Informatika ITS (2011-2015).

Selama kuliah di Teknik Informatika ITS, penulis mendalami bidang minat Rekayasa Perangkat Lunak (RPL) dan Penulis pernah menjadi asisten dosen pada mata kuliah konstruksi perangkat lunak. Penulis dapat dihubungi melalui surel: **salahudin.agung@yahoo.co.id**.