PERANCANGAN DAN IMPLEMENTASI SISTEM INFORMASI SEMESTER PENDEK PTIIK MENGGUNAKAN INTEGRASI *WEB SERVICE* SIAKAD UNIVERSITAS BRAWIJAYA

SKRIPSI

KEMINATAN REKAYASA PERANGKAT LUNAK

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun oleh :

NURWANDA OKTARIA

NIM : 105060800111083

PROGRAM STUDI INFORMATIKA / ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2014

LEMBAR PERSETUJUAN

PERANCANGAN DAN IMPLEMENTASI SISTEM INFORMASI SEMESTER PENDEK PTIIK MENGGUNAKAN INTEGRASI *WEB SERVICE* SIAKAD UNIVERSITAS BRAWIJAYA

SKRIPSI

KEMINATAN REKAYASA PERANGKAT LUNAK

Diajukan untuk memenuhi persyaratan
Memperoleh gelar Sarjana Komputer

Disusun oleh :
NURWANDA OKTARIA
NIM : 105060800111083

Skripsi ini telah disetujui oleh dosen pembimbing pada
Tanggal 20 Juni 2014

Dosen Pembimbing I	**Dosen Pembimbing II**
Satrio Agung W., S.Kom., M.Kom	**Denny Sagita R., S.Kom., M.Kom**
NIP. 19860521 201212 1 001	NIK. 851124 06 1 1 0250

LEMBAR PENGESAHAN

PERANCANGAN DAN IMPLEMENTASI SISTEM INFORMASI SEMESTER PENDEK PTIIK MENGGUNAKAN INTEGRASI *WEB SERVICE* SIAKAD UNIVERSITAS BRAWIJAYA

SKRIPSI

KEMINATAN REKAYASA PERANGKAT LUNAK

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun oleh :

NURWANDA OKTARIA

NIM : 105060800111083

Skripsi ini telah diuji dan dinyatakan lulus pada tanggal 11 Juli 2014

Penguji 1	**Penguji 2**
Aryo Pinandito, ST., M.MT.	**Issa Arwani, S.Kom., M.Sc.**
NIK. 83051916110374	**NIP. 198309222012121003**

Penguji 3

Aswin Suharsono, ST., MT.

NIK. 840919 06 1 1 0251

Mengetahui

Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji, M.T.

NIP. 19670801 199203 1 001

PERNYATAAN
ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata didalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, Juli 2014
Mahasiswa,

Nurwanda Oktaria
NIM. 105060800111083

KATA PENGANTAR

Dengan nama Allah SWT Yang Maha Pengasih dan Penyayang. Segala puji bagi Allah SWT karena atas rahmat dan hidayah-Nya-lah penulis dapat menyelesaikan skripsi yang berjudul **"Perancangan Dan Implementasi Sistem Informasi Semester Pendek PTIIK Menggunakan Integrasi *Web Service* SIAKAD Universitas Brawijaya".** Shalawat serta salam atas junjungan besar kita Nabi Muhammad S.A.W. beserta keluarga dan para sahabat sekalian. Skripsi ini disusun untuk memenuhi sebagai persyaratan memperoleh gelar Sarjana Komputer di Program Studi Informatika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya Malang.

Melalui kesempatan ini, penulis menyampaikan rasa hormat dan terima kasih penulis yang sebesar-besarnya kepada pihak yang telah memberikan bantuan lahir maupun batin selama penulisan skripsi ini. Oleh karena itu, pada kesempatan ini penulis ingin menyampaikan rasa hormat dan terima kasih penulis kepada :

1. Satrio Agung W., S.Kom., M.Kom. selaku Dosen Pembimbing I yang telah memberikan bimbingan, arahan dan motivasi hingga proposal skripsi ini dapat terselesaikan.
2. Denny Sagita R., S.Kom., M.Kom. selaku Dosen Pembimbing II yang telah memberikan pengetahuan, bimbingan, motivasi dan arahan untuk kesempurnaan penulisan proposal skripsi ini.
3. Orang tua penulis dan seluruh keluarga yang senantiasa tiada henti hentinya memberikan do'a demi terselesainya skripsi ini.
4. Bapak Ir. Sutrisno, M.T, Bapak Ir. Heru Nurwasito, M.Kom, Bapak Himawat Aryadita, S.T, M.Sc, dan Bapak Eddy Santoso, S.Kom selaku Ketua, Wakil Ketua 1, Wakil Ketua 2 dan Wakil Ketua 3 Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
5. Bapak Drs. Marji, MT dan Bapak Issa Arwani, S.Kom., M.Sc selaku Ketua dan Sekretaris Program Studi Informatika serta segenap Bapak / Ibu Dosen, Staff Administrasi dan Perpustakaan Program Studi Informatika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.

6. Seluruh dosen Program Studi Informatika atas kesediaan membaginya ilmunya kepada penulis.
7. Rizal Setya Perdana selaku karyawan PPTI Universitas Brawijaya yang memberikan kelancaran serta ilmunya dalam penulisan skripsi ini.
8. Dharma, Nena, Sari, Citra, Depru, Naila, Edo, Ayunita, Zul, Gopi dan Uly selaku sahabat yang tidak pernah lelah memberikan motivasi, saran dan bantuannya yang selalu membangun dalam penulisan skripsi ini.
9. Teman-teman Program Studi Informatika Program Teknologi Informasi dan Ilmu Komputer angkatan 2010 yang selalu memberikan motivasinya.
10. Seluruh pihak yang telah membantu kelancaran penulisan skripsi yang tidak dapat penulisan sebutkan satu per satu.

Penulis menyadari bahwa skripsi ini masih banyak kekurangan dan masih jauh dari sempurna. Untuk itu, saran dan kritik yang membangun, sangat penulis harapkan. Semoga skripsi ini membawa manfaat bagi penyusun maupun pihak lain yang menggunakannya.

Malang, Juli 2014

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Sistem Informasi Semester Pendek PTIIK merupakan suatu kegiatan PTIIK yang memberikan fasilitas kepada mahasiswa agar dapat mengikuti kuliah perbaikan nilai pada waktu libur semester genap, tentunya dengan batasan jumlah SKS dan matakuliah tertentu yang dapat diambil. Implementasi sistem informasi ini dibangun untuk mengoptimalkan kinerja PTIIK dalam mengelola semester pendek yang telah ada sebelumnya. Dalam pengembangannya sistem informasi semester pendek PTIIK dibangun menggunakan integrasi web service SIAKAD UB, penerapan integrasi web service dipilih penulis , karena perancangan dan implementasi sistem informasi ini dapat dijadikan sebagai solusi integrasi data antara aplikasi pusat Universitas Brawijaya dengan sistem informasi semester pendek PTIIK. Selain memudahkan dalam integrasi data, keleluasaan dalam mengolah data juga menjadi topik utama penerapan web service, dengan menjaga keamanan data pada aplikasi asalnya walaupun data yang dikirimkan sedang digunakan dan diolah pada aplikasi lain. Hal ini tentu dapat meningkatkan efektifitas dan originalitas data dalam proses perancangan dan implementasi sistem informasi semester pendek ini.

Kata Kunci : Web Service, Sistem Informasi, Semester Pendek, RPL

ABSTRACT

PTIIK's Short Term Information Systems is an PTIIK's activity that provide facilities to the students in order to improve their grades through attend classes at holiday even semester, of course, with a limited number of credits and particular courses that can be taken. Implementation of the information system is built to optimize the performance of PTIIK in managing short-term which already exists before. In the development of short-term PTIIK's information systems constructed using the web service integration from SIAKAD UB, this integration of web service chosen by authors, because the design and implementation of information systems can be used as a data integration solutions between UB's main applications with a PTIIK's short term information systems. In addition to facilitate the integration of data, flexibility in data processing has also become a major topic of the application of web service, the security of data in its original application despite the transmitted data being used and processed in other applications. This can certainly improve the effectiveness and originality of the data in the design and implementation of short term information systems.

Keywords : Web Service, Information Systems, Short Term, RPL, Integration

BAB I
PENDAHULUAN

1.1 Latar Belakang

Semester pendek merupakan suatu kegiatan PTIIK yang memberikan fasilitas kepada mahasiswa agar dapat mengikuti kuliah perbaikan nilai pada waktu libur semester genap, tentunya dengan batasan jumlah SKS dan matakuliah tertentu yang dapat diambil. Pada proses yang masih dilakukan secara manual sekarang ini, mengharuskan staff akademik untuk meng-*input*-kan kembali data secara satu per satu ke dalam *database* [KTU-12]. Selain itu penambahan waktu juga diperlukan dalam pengadaan semester pendek terutama untuk proses pendataan matakuliah dan pendaftaran bagi mahasiswa PTIIK karena prosesnya masih dilakukan secara kerjasama dengan tim mahasiswa [KTU-12].

Melihat dari prosesnya, pengadaan semester pendek yang telah ada di PTIIK membutuhkan waktu yang cukup banyak, bagi mahasiswa yang ingin mendaftar atau membatalkan semester pendek diharuskan untuk terus berkomunikasi atau *keep-in-touch* dengan tim mahasiswa yang bertugas karena pendataan dan perubahan terus dilakukan secara manual, sedangkan proses ini dilakukan saat liburan semester yang tentunya tidak semua mahasiswa dapat terus berkomunikasi dengan tim mahasiswa untuk mengetahui bagaimana kelanjutan informasi pendaftarannya di semester pendek.

Untuk membantu kinerja PTIIK dalam mengelola semester pendek salah satu caranya dengan mengimplementasikan sistem informasi atau perangkat lunak. Sistem informasi yang dibangun dapat menjadi sebuah solusi karena beberapa faktor, salah satunya yaitu karena pemanfaatan aplikasi atau teknologi perangkat lunak yang berguna untuk mempermudah dan meningkatkan pengembangan sistem informasi tersebut.

Dalam kasus ini, sistem semester pendek PTIIK membutuhkan beberapa data yang menjadi kunci utama dalam proses pengelolaan semester pendek PTIIK seperti

data KHS mahasiswa. Dalam perkembangannya, sebuah sistem informasi kita sadari tidak hanya memiliki beberapa sumber data yang didapat dari *input*-an *user*, tetapi data yang berasal dari beberapa aplikasi tentunya tanpa diperlukan hak akses untuk mengetahui *backhand* dari *database* aplikasi tersebut.

Pengimplementasian ini merupakan salah satu contoh teknologi perangkat lunak khususnya adalah cara kerja *web service* yang merupakan teknologi perangkat lunak untuk merelasikan data dari integrasi aplikasi yang ada [PUR - 08]. Format pertukaran data w*eb service* yang umum diketahui sekarang yaitu XML dan JSON, pemilihan antara keduanya tergantung dengan kebutuhan pemakaian *programmer*. Sedangkan yang dimaksud dengan integrasi data adalah suatu proses menggabungkan, menyatukan data yang berasal dari sumber yang berbeda dan mendukung pengguna untuk melihat kesatuan data [LEN - 02].

Penerapan *web service* selain memudahkan dalam integrasi data, keleluasaan dalam mengolah data juga menjadi topik utama penerapan *web service*, dengan menjaga keamanan data pada aplikasi asalnya walaupun data yang dikirimkan sedang digunakan dan diolah pada aplikasi lain. Hal ini tentu dapat meningkatkan efektifitas dan *originalitas* data dalam proses perancangan dan implementasi sistem informasi semester pendek ini.

Karena pengimplementasian ini tidak terlepas dari aplikasi pusat SIAKAD Universitas Brawijaya, maka pembuatannya disesuaikan dengan kondisi yang ada pada SIAKAD UB termasuk SOA yang menjadi arsitektur penyedia *web service*, dengan format yang digunakan serta teknologi REST *web service* yang diterapkan.

Dengan adanya sistem informasi ini diharapkan dapat membuat sebuah sistem semester pendek yang dapat membantu proses pengelolaan semester pendek, dimulai dari pengajuan mata kuliah sampai pembayaran. Selain itu diharapkan penerapan *web service* dengan aplikasi SIAKAD UB dapat menjawab kebutuhan integrasi data antar aplikasi yang digunakan dalam sistem ini sehingga dapat membantu mengontrol kinerja *staff* akademik karena proses sudah tersistem dan data yang masuk ke dalam *database* dilakukan otomatis secara *real time* dan cepat sehingga lebih akurat dan efisien dalam mengaksesnya.

1.2 Rumusan Masalah

Berdasarkan paparan latar belakang tersebut, maka rumusan masalah adalah sebagai berikut:

1. Bagaimana menganalisis, merancang dan mengimplementasikan sistem informasi semester pendek sesuai dengan ketentuan dan prosedur di PTIIK?
2. Bagaimana mendapat kebutuhan data utama (data KHS) yang diperlukan sistem informasi semester pendek tanpa perlu hak akses khusus ke dalam *database* sumbernya (SIAKAD Universitas Brawijaya) ?
3. Bagaimana mengimplementasikan sistem informasi semester pendek yang memanfaatkan *web service* SIAKAD UB agar dapat diintegrasikan dengan sistem yang dibangun?

1.3 Batasan Masalah

Karena permasalahan keterbatasan waktu dan agar pembahasan tidak menyimpang dari tujuan maka dilakukan pembatasan masalah sebagai berikut :

1. Sistem informasi semester pendek diimplementasikan dengan menggunakan integrasi *web service* SIAKAD Universitas Brawijaya.
2. Sistem ini menggunakan SOA (*Service Oriented Architecture*) yang telah ada di SIAKAD Universitas Brawijaya dan menyesuaikan dengan format data pemrograman *web service* yang digunakan SIAKAD Universitas Brawijaya.
3. Sistem ini hanya mengambil data menggunakan integrasi *web service* yang sesuai dengan kebutuhan fungsional sistem.
4. Sistem informasi ini mengadopsi *rule-rule* atau syarat dan ketentuan yang telah ada sebelumnya pada pengelolaan semester pendek PTIIK.

1.4 Tujuan

Tujuan dari pengimplementasian sistem informasi semester pendek PTIIK ini secara umum adalah untuk menganalisis, merancang, membangun dan menguji sistem informasi yang berhubungan dengan pengelolaan semester pendek PTIIK dengan menggunakan integrasi data dari aplikasi SIAKAD Universitas Brawijaya melalui *web service.*

1.5 Manfaat

Diharapkan dengan adanya perancangan sistem informasi ini dapat bermanfaat bagi :

a. Bagi Penulis
    1. Dapat lebih memahami dan bisa menerapakan pengembangan sistem informasi manajemen semester pendek PTIIK.
    2. Dapat menjadi pembelajaran dan dapat menambah pengalaman di bidang studi keilmuan yang terkait.

b. Bagi Mahasiswa
    1. Sebagai media informasi untuk membantu mahasiswa PTIIK dalam melakukan proses pendaftaran semester pendek.
    2. Dapat memberikan kejelasan dan kemudahan pada mahasiswa sehingga proses pendaftaran semester pendek dapat dilakukan dengan lebih optimal.

c. Bagi PTIIK
    1. Dapat meningkatkan pelayanan terhadap mahasiswa.
    2. Dapat memberikan kemudahan pada PTIIK dalam menyediakan serta mengawasi pengadaan semester pendek.

1.6 Sistematika Penulisan

Penyusunan skripsi ini menggunakan kerangka pembahasan yang tersusun sebagai berikut :

BAB I Pendahuluan

Bab ini berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, sistematika penulisan, dan waktu pengerjaan.

BAB II Dasar Teori

Membahas teori dasar dan teori penunjang yang berhubungan dengan teknologi *web service*, semester pendek, rekayasa perangkat lunak, dan pengujian.

BAB III Metodologi Penelitian

Membahas tentang metode yang digunakan dalam penulisan yang terdiri dari studi literatur, perancangan perangkat lunak berbasis *website*, implementasi perangkat lunak berbasis *website*, pengujian dan analisis.

BAB IV Perancangan dan Implementasi

Membahas tentang analisa kebutuhan dan perancangan dari sistem informasi manajemen semester pendek PTIIK sesuai dengan teori yang ada serta pembahasan tentang implementasi dari sistem.

BAB V Pengujian dan Analisis

Membahas tentang hasil pengujian dan analisis terhadap sistem informasi yang telah dibuat.

Bab VI Penutup

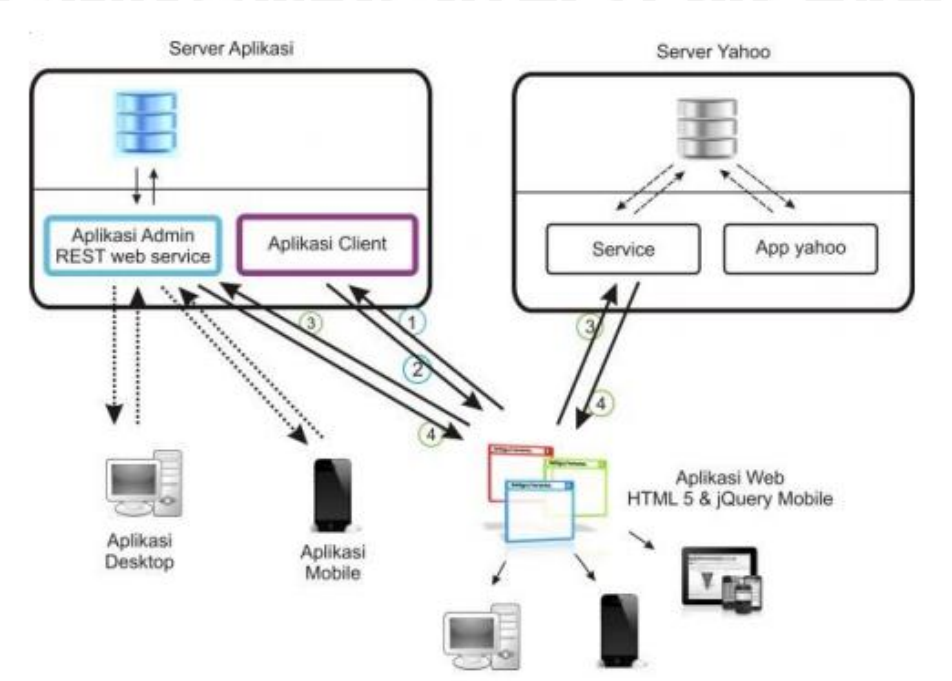
Memuat kesimpulan yang diperoleh dari pembuatan dan pengujian perangkat lunak berbasis website yang dikembangkan dalam skripsi ini serta saran–saran untuk pengembangan lebih lanjut.

BAB II
DASAR TEORI

Pada bab dua, terdiri dari kajian pustaka dan dasar teori. Kajian pustaka adalah membahas penelitian yang telah ada dan yang diusulkan. Dasar teori membahas teori yang diperlukan untuk menyusun penelitian yang diusulkan. Kajian pustaka pada penelitian ini adalah membandingkan penelitian ini dengan penelitian sebelumnya yang berjudul 'Rancang Bangun REST *Web Service* pada Aplikasi Penentuan Portofolio Saham Menggunakan Model Markowitz dan JQuery Mobile'. Pada penelitian ini, dasar teori yang diperlukan berdasarkan latar belakang dan rumusan masalah adalah: konsep dasar sistem pengadaan semester pendek PTIIK, konsep dasar *web service*, konsep dasar JSON, konsep dasar REST, pengembangan dan rekayasa perangkat lunak, konsep *Incremental*, konsep dasar *Unified Modelling Language* yang dipakai pada rekayasa perangkat lunak, konsep dasar teknik dan strategi pengujian perangkat lunak dan penjelasan tentang konsep dasar *Basis data*.

2.1 Kajian Pustaka

Kajian pustaka pada penelitian ini adalah membandingkan penelitian ini dengan penelitian sebelumnya yang berjudul 'Rancang Bangun REST *Web Service* pada Aplikasi Penentuan Portofolio Saham Menggunakan Model Markowitz dan JQuery Mobile'. Pada penelitian ini membahas tentang pengimplementasian REST *web service* pada aplikasi penentuan portofolio saham dengan ,menggunakan model markowitz dan dapat dijalankan secara lintas *platform*. *Web services* yang digunakan dengan arsitektur REST dalam penelitian ini dapat dilakukan untuk pengiriman data dalam berbagai format, sehingga dapat berjalan di berbagai *platform*. Perbedaan antara implementasi sebelumnya dan yang diusulkan digambarkan pada Gambar 2.1 dan Gambar 2.2 berikut.



Gambar 2.1 Arsitektur Sistem Aplikasi Penentuan Portofolio Saham

Sumber : [YSR-14]

Untuk gambar arsitektur sistem informasi semester pendek PTIIK sesuai dengan gambar 4.6 pada sub bab 4.1.2.1.

Dari kedua gambar, arsitektur hampir sama hanya saja perbedaannya dari segi *platform* perangkat lunak yang dapat diimplementasikan dan sumber data pengambilan *web service*. Pada literatur atau gambar 2.1, sistem dapat diimplementasikan secara lintas *platform* dan pada sumber pengambilan data, sistem melakukan *service* ke *server* yahoo. Perbedaan ini tentunya mempengaruhi struktur pengiriman data yang akan diimplementasikan serta kebutuhan fungsional dari tiap sistem. Pada Gambar 2.1 pengimplementasian menggunakan HTML5 sehingga dapat dilakukan secara lintas *platform* serta pengintegrasian *web service* nya dilakukan dan diimplementasikan melalui berbagai *device* ke *server* yahoo. Sedangkan pada Gambar 2.2 sistem akan bekerja dengan bantuan SOA dari SIAKAD UB dan menggunakan JSON sebagai sebuah format rancangan data yang dibutuhkan sesuai dengan fungsionalitas sistem.

2.2 Dasar Teori

2.2.1 Konsep Semester Pendek PTIIK

Berdasarkan hasil observasi secara diskusi dengan dosen yang mengelola pengadaan semester pendek PTIIK diperoleh beberapa informasi yang akan dijelaskan pada bab ini.

2.2.1.1 Pengertian Semester Pendek

Seperti yang telah kita ketahui sebelumnya, dalam sistem pembelajaran di PTIIK selain kegiatan perkuliahan seperti biasanya juga terdapat kegiatan perkuliahan semester pendek. Semester pendek merupakan kegiatan perkuliahan yang diadakan selama masa liburan semester genap yaitu diantara 2 (dua) semester reguler yang sama sesuai dengan sistem satuan kredit (SKS). Pengadaan semester pendek di PTIIK merupakan program yang berorientasi pada pemenuhan kebutuhan siswa dalam mengembangkan kemampuan yang dimiliki, terutama ditujukan untuk perbaikan nilai mata kuliah mahasiswa pada semester reguler sebelumnya [PDM-12].

2.2.1.2 Syarat dan Peraturan Semester Pendek PTIIK

Program Teknologi Informasi dan Ilmu Komputer telah membuat persyaratan dan peraturan bagi mahasiswa yang akan mengikuti semester pendek, yaitu sebagai berikut [PDM-12] :

a. Pengadaan semester pendek dilakukan setiap tahun dan dilaksanakan di antara semester genap dengan semester ganjil (waktu liburan semester genap) selama kurang lebih satu bulan.
b. Dibukanya mata kuliah pada semester pendek terbatas pada peminat yang mendaftarkan mata kuliah tersebut dengan batasan minimal 10 orang peminat.
c. Mata kuliah yang dapat diambil pada semester pendek adalah mata kuliah yang pernah diambil mahasiswa bersangkutan pada semester sebelumnya sehingga tidak dapat melakukan fasilitas lintas jurusan seperti KRS biasa.

d. Mahasiswa hanya diperbolehkan mengambil mata kuliah yang bernilai C+ atau kurang dalam KHS.
e. Jumlah mata kuliah yang diambil setiap mahasiswa maksimum berjumlah 12 SKS.
f. Mata kuliah yang tidak dapat dibuka dikarenakan berbagai kendala, maka biaya semester pendek yang telah dibayarkan dapat diambil kembali di bagian keuangan PTIIK.

2.2.1.3 Pembayaran Semester Pendek

Biaya semester pendek PTIIK dihitung per-SKS dengan besar 1 SKS adalah Rp. 100.000,00 [PDM-12].

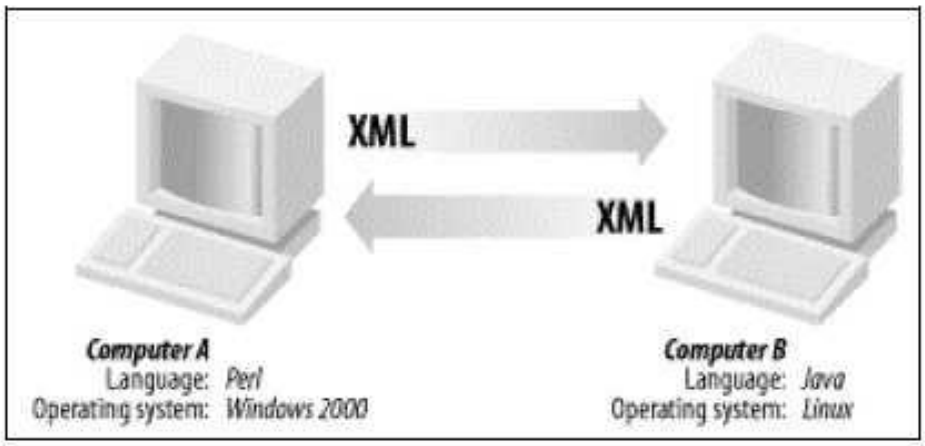
2.2.1.4 Perubahan dan Pengembalian Biaya Semester Pendek

Apabila mahasiswa yang telah mendaftar dan membayar mata kuliah yang didaftarkannya pada semester pendek namun mata kuliah tersebut ternyata tidak dapat dibuka dikarenakan beberapa kendala atau peminat yang kurang, maka mahasiswa tersebut dapat mengubahnya dengan mata kuliah lain atau membatalkan mata kuliah tersebut.

Jika pilihan pembatalan mata kuliah yang dipilih, maka mahasiswa berhak mengambil biaya kembalian semester pendek yang telah ia bayarkan ke bagian keuangan PTIIK tentunya dengan menyerahkan bukti pembayaran yang sah (slip pembayaran asli).

2.2.2 Konsep Dasar *Web Service*

Web service merupakan salah satu metode teknologi antar *device* elektronik yang berbeda melalui *world wide web* (www). *Web services* adalah layanan yang tersedia di internet yang menggunakan format standar XML untuk pengiriman pesannya, tidak terikat kepada bahasa pemrograman atau sistem operasi tertentu Gambar 2.2 menunjukkan konsep dasar *web services* [PUR-08].



Gambar 2.2 Dasar *Web Services*

Sumber : [PUR-08]

Web service secara teknis memiliki mekanisme interaksi antar sistem sebagai penunjang interoperabilitas, baik berupa agregasi (pengumpulan) maupun sindikasi (penyatuan). *Web service* memiliki layanan terbuka untuk kepentingan integrasi data dan kolaborasi informasi yang bisa diakses melalui internet oleh berbagai pihak menggunakan teknologi yang dimiliki oleh masing-masing pengguna. Sekalipun mirip dengan *Application Programming Interface (*API) berbasis web, *web service* lebih unggul karena dapat dipanggil dari jarak jauh melalui internet. Pemanggilan *web service* bisa menggunakan bahasa pemrograman apa saja dan dalam *platform* apa saja, sementara API hanya bisa digunakan dalam *platform* tertentu [LCK-08].

Kelebihan *web service* adalah: 1) lintas *platform*, 2) *language independent*, 3) jembatan penghubung dengan database tanpa perlu *driver database* dan tidak harus mengetahui jenis DBMS, 4) mempermudah proses pertukaran data, serta 5) penggunaan kembali komponen aplikasi [LCK-08].

Dalam perkembangannya, model *web service* memiliki dua metode yang berorientasi pada layanan dan sumberdaya informasi, yaitu: SOAP (*Simple Object Access Protocol*) dan REST (*REpresentational State Transfer*).

Berikut adalah teknologi pendukung *web service* :

1. *Simple Object Access Protocol* (SOAP) yang merupakan teknologi transportasi dan pertukaran dokumen XML yang telah terstruktur dalam pelaksanaan layanan *web* di jaringan komputer.

   Peran SOAP di dalam teknologi *web service* adalah sebagai protokol pemaketan untuk pesan-pesan yang digunakan secara bersama oleh aplikasi-aplikasi penggunanya. SOAP dirancang dengan menggunakan protokol komunikasi HTTP karena HTTP didukung oleh semua *browser* dan *server*, maka SOAP dapat berkomunikasi dengan berbagai aplikasi meskipun terdapat perbedaan sistem operasi, teknologi, dan bahasa pemrogramannya.

   Impementasi model SOA telah banyak dilakukan dan dikembangkan oleh banyak vendor (misal: Microsoft, Sun dan IBM, melalui dukungan platform infrastruktur .Net dan Java). Proses layanan dengan arsitektur SOAP memiliki tiga komponen utama, yaitu : 1) *service provider*, 2) *service requester*, dan 3) *service broker*, serta komponen pendukung yaitu XML, SOAP-XML (terdiri atas header dan body), WSDL, serta UDDI [DSI-08] yang akan dijelaskan selanjutnya.
2. Selain SOAP juga dikembangkan metode REST yang dikembangkan oleh empat prinsip utama teknologi yaitu 1) *Resource identifier through Uniform Resource Identifier* (URI), 2) *Uniform Interface* (sumber CRUD menggunakan operasi PUT, GET, POST, dan DELETE), 3) *self-descriptive messages* (sumberdaya tidak terikat sehingga dapat mengakses konten HTML, XML, PDF, JPEG, plain text, meta data, dll), serta 4) *Stateful Interactions Through Hyperlinks* (bersifat *stateless*) [PTS-08].

   Metode REST lebih sederhana karena menggunakan format standar (HTTP, HTML, XML, URI, MIME), namun jika diperlukan proses pertukaran data, maka konten berupa teks dari hasil eksekusi web service dapat diolah dalam format teks (seperti XML atau HTML) dengan menggunakan utilitas komunikasi data berupa koneksi socket protokol HTTP. Utilitas ini umumnya tersedia dalam pustaka komunikasi pada bahasa pemrograman (seperti Java, Visual Basic, Delphi, PHP, ASP, dan JSP) [SKY-09].

3. *Web Service Definition Language* (WSDL) merupakan antar muka *web service* yang menyatakan parameter masukan dan keluaran untuk pemanggilan servis secara eksternal, struktur penanda fungsi yakni cara pemanggilan.
4. *Universal Description, Discover, and Integration* (UDDI) merupakan direktori yang menampilkan daftar layanan disediakan.

2.2.3 Konsep JSON

JSON dirancang untuk menjadi format pertukaran data yang dapat dibaca manusia dan mudah bagi komputer untuk proses mengurai dan menggunakannya. JSON secara langsung didukung dalam JavaScript dan yang paling cocok untuk JavaScript aplikasi, sehingga memberikan kinerja yang signifikan [NUR-09].

JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai format pertukaran-data [JSN-14].

JSON terbuat dari dua struktur, yaitu :

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

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini [JSN-14].

Bentuk JSON :

- **Objek** adalah sepasang nama atau nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama atau nilai dipisahkan oleh , (koma)
- **Larik** adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma)
- **Nilai (*value*)** dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau true atau false atau null, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat
- **String** adalah kumpulan dari nol atau lebih karakter Unicode, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan backslash escapes "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java
- **Angka** adalah sangat mirip dengan angka di C atau Java, kecuali format oktal dan heksadesimal tidak digunakan [JSN-14].

2.2.4 REST

Istilah REST yang merupakan singkatan dari Representational State Transfer pertama kali digunakan oleh Roy Thomas Fielding, salah seorang pelopor proyek *web server* Apache, pada disertasi doktornya yang berjudul Arc*hitectural Styles and the Design of Network-based Software Architectures di University of California* pada tahun 2000 [WYN-12].

1. WADL

WADL merupakan singkatan dari *Web Application Definition Language* yang digunakan sebagai kontrak layanan yang dijelajahi melalui kode. REST tidak

memiliki penyedia ataupun *partner* yang menangani WSDL, tapi mempunyai pengganti yang berupa WADL.

2. HTTP *Method*

HTTP method diantaranya terdiri dari OPTION, GET, HEAD, POST, PUT, DELETE, dan TRACE. Method-method ini mempunyai peranan yang sangat penting dalam penggunaan HTTP *methods* [ALL-10].

3. URI

URI merupakan singkatan dari *Uniform Resource Identifier*. Dalam banyak kasus, client tidak terkait dengan bagaimana server akan mendesain URI-nya. Desain URI hanya salah satu aspek dalam mengimplementasikan aplikasi REST [ALL-10].

2.2.5 Konsep Rekayasa Perangkat Lunak

2.2.5.1 Pengertian Perangkat Lunak

Perangkat lunak adalah program komputer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi kebutuhan, model desain, dan cara penggunaan (*user manual*). Sebuah program komputer tanpa terasosiasi dengan dokumentasinya maka belum dapat disebut perangkat lunak (*software*). Sebuah perangkat lunak juga sering disebut dengan sistem perangkat lunak. Sistem berarti kumpulan komponen yang saling terkait dan mempunyai satu tujuan yang ingin dicapai [ROS-13].

Sistem perangkat lunak berarti sebuah sistem yang memiliki komponen berupa perangkat lunak yang memiliki hubungan satu sama lain untuk memenuhi kebutuhan pelanggan (*costumer*). Pelanggan (costumer) adalah orang atau organisasi yang memesan atau memebeli perangkat lunak (software) dari pengembang perangkat lunak atau bisa dianggap bahwa pelanggan atau *costumer* adalah orang atau organisasi yang dengan sukarela mengeluarkan uang untuk memesan atau membeli perangkat lunak [ROS-13].

Karakter perangkat lunak adalah sebagai berikut [ROS-13] :

1. Perangkat lunak dibangun dengan rekayasa (*software engineering*) bukan diproduksi secara manufaktur atau pabrikan.

2. Perangkat lunak tidak pernah usang (*wear out*) karena kecacatan dalam perangkat lunak dapat diperbaiki.
3. Perangkat lunak biasanya terus diperbaiki seiring bertambahnya kebutuhan.

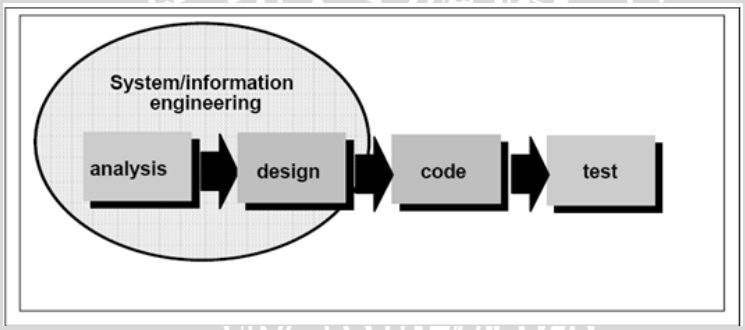
2.2.5.2 Pengertian Rekayasa Perangkat Lunak

Rekayasa perangkat lunak (*software engineering*) merupakan pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomi dan dipercaya dan bekerja secara efisien menggunakan mesin [ROS-13].

Rekayasa perangkat lunak lebih fokus pada praktik pengembangan perangkat lunak dan mengirimkan perangkat lunak yang bermanfaat kepada pelanggan (*costumer*) [ROS-13].

2.2.5.3 Proses Rekayasa Perangkat Lunak

Proses rekayasa perangkat lunak yang dilakukan selama pembangunan perangkat lunak secara garis besar adalah :



Gambar 2.3 Tahapan Umum Rekayasa Perangkat Lunak

Sumber : [ROS-13].

Proses-proses pada gambar diatas dapat dilakukan berulang kali sampai perangkat lunak memenuhi kebutuhan pelanggan atau *user*. Proses perangkat lunak (*software process*) adalah sekumpulan aktifitas yang memiliki tujuan untuk mengembangkan atau mengubah perangkat lunak. Secara umum proses perangkat lunak terdiri dari :

1. Pengumpulan Spesifikasi (*Specification*)
2. Pengembangan (*Development*)

3. Validasi (*Validation*)
4. Evolusi (*Evolution*)

2.2.6 Konsep *Waterfall*

2.2.6.1 Pengertian SDLC

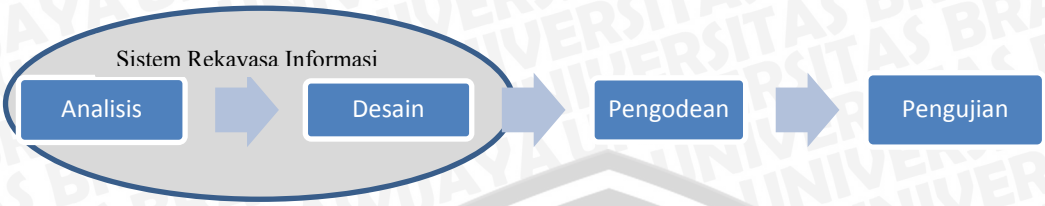
SDLC atau *Software Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem - sistem perangkat lunak sebelumnya [ROS-13].

Tahapan - tahapan yang ada pada SDLC secara global adalah sebagai berikut [ROS-13] :

1. Inisiasi (*initiation*)
2. Pengembangan Konsep Sistem (*system concept development*)
3. Perencanaan (*planning*)
4. Analisis Kebutuhan (*requirements analysis*)
5. Desain (*design*)
6. Pengembangan (*development*)
7. Integrasi dan Pengujian (*integration and test*)
8. Implementasi (*implementation*)
9. Operasi dan Pemeliharaan (*operation and maintenance*)
10. Disposisi (*disposition*)

2.2.6.2 Model *Waterfall*

Model SDLC air terjun (*waterfall*) sering juga disebut model sekuensial linier (*sequential linier*) atau alur hidup klasik. Berikut adalah gambar model air terjun :



Gambar 2.4 Model *Waterfall*

Sumber : [ROS-13]

1. Analisis Kebutuhan Perangkat Lunak

   Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

2. Desain

   Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

3. Pembuatan Kode Program

   Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

4. Pengujian

   Pengujian fokus pada perangkat lunak secara dari segi logika dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

5. Pendukung (*support*) atau Pemeliharaan (*maintenance*)

   Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

Kelemahan yang ada pada model *Waterfall Process* yaitu perubahan yang sangat sulit dilakukan karena tahapan pada proses pengembangan ini bersifat statis. Karena sifat kakunya, model ini tepat jika kebutuhan (*requirements*) telah dikumpulkan secara lengkap sehingga perubahan dapat dikurangi, tetapi kenyataannya konsumen sering memberikan *requirements* yang kurang lengkap sehingga perubahan kebutuhan adalah sesuatu yang wajar.

Dengan kelemahan yang telah disebutkan, mengapa model *waterfall* masih populer untuk saat ini, hal tersebut dikarenakan pengaplikasian model ini mudah. Ketika semua kebutuhan sistem dapat didefinisikan secara utuh, eksplisit dan tidak ada kesalahan di awal *project*, maka proses selanjutnya dapat berjalan dengan baik tanpa masalah.

2.2.7 *Unified Modelling Language* (UML)

2.2.7.1 Pengenalan *Unified Modelling Language* (UML)

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang yang berada di belahan negara manapun dapat mengerti pemodelan perangkat lunak yang telah dibuat. Banyak orang yang telah membuat bahasa pemodelan perangkat lunak dan yang berkembang saat itu bahasa pemrograman *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunakan bahasa terstruktur. Kemudian muncul *State Transition Diagram* (STD) yang digunakan untuk memodelkan sistem *real time*.

Pada perkembangan teknik pemrograman berorientasi objek, muncul pemodelan *Unified Modelling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan diagram dan teks pendukung.

UML adalah kependekan dari *Unified Modeling Language* yang merupakan suatu cara untuk menyelesaikan suatu masalah dengan mendeskripsikannya yang telah menjadi standar dalam dunia industri untuk memvisualisasikan, merancang dan mendokumentasikan sistem perangkat lunak. Dengan menggunakan UML kita dapat membangun model untuk segala bentuk dan jenis aplikasi perangkat lunak, yang mana aplikasi yang dibangun dapat berjalan pada perangkat lunak dengan sistem operasi dan jaringan apapun [PRY-05].

Dalam membangun suatu model perangkat lunak dengan UML, digunakan bentuk-bentuk diagram atau symbol untuk merepresentasikan elemen-elemen dalam sistem. Bentuk diagram yang digunakan untuk merepresentasikannya adalah sebagai berikut [PRY-05] :

- *Use case Diagram*
- *Class Diagram*
- *State Diagram*
- *Sequence Diagram*
- *Collaboration Diagram*
- *Activity Diagram*
- *Component Diagram*
- *Deployment Diagram*

2.2.7.2 *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Dalam setiap kelas memiliki tiga area pokok :

- Nama (dan *stereotype*)
- Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
- Metode atau operasi adalah fungsi – fungsi yang dimiliki oleh suatu kelas.

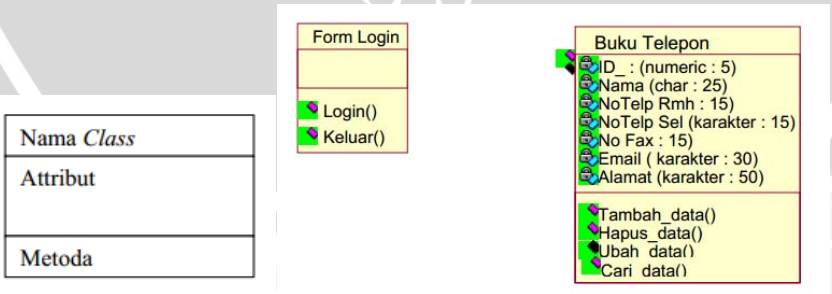
Atribut dan metode dalam *class diagram* dapat memiliki salah satu sifat seperti berikut di bawah ini [PRY-05] :

- *Private*, hanya dapat diakses oleh *class* itu sendiri
- *Protected,* hanya dapat diakses oleh *class* itu sendiri dan turunan dari *class* tersebut
- *Public*, dapat diakses oleh *class* selain dari *class* yang bersangkutan

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Kelas – kelas yang ada pada struktur sistem harus dapat melakukan fungsi – fungsi sesuai dengan kebutuhan sistem sehingga *programmer* dapat membuat kelas – kelas di dalam program perangkat lunak sesuai dengan perancangan diagram kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis – jenis kelas berikut :

- Kelas main
- Kelas yang menangani tampilan (*view*)
- Kelas yang diambil dari pendefinisian *use case* (*controller*)
- Kelas yang diambil dari pendefinisan data (*model*)

Berikut ini merupakan bentuk *class diagram* secara umum [PRY-05] :



Gambar 2.5 Contoh *Class Diagram*

Sumber : [PRY-05]

2.2.7.3 *Use Case Diagram*

Use case atau diagram *use case* merupakan pemodelan untuk perlakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dalam *Use-case diagram* penekanannya adalah "apa" yang diperbuat oleh sistem, dan bukan "bagaimana".

Syarat pemakaian *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian :

- Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri.
- *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau aktor.

Beberapa simbol garis hubungan pada diagram *use case* yang sering digunakan adalah :

Table 2.1 Keterangan Hubungan Pada Use Case Diagram

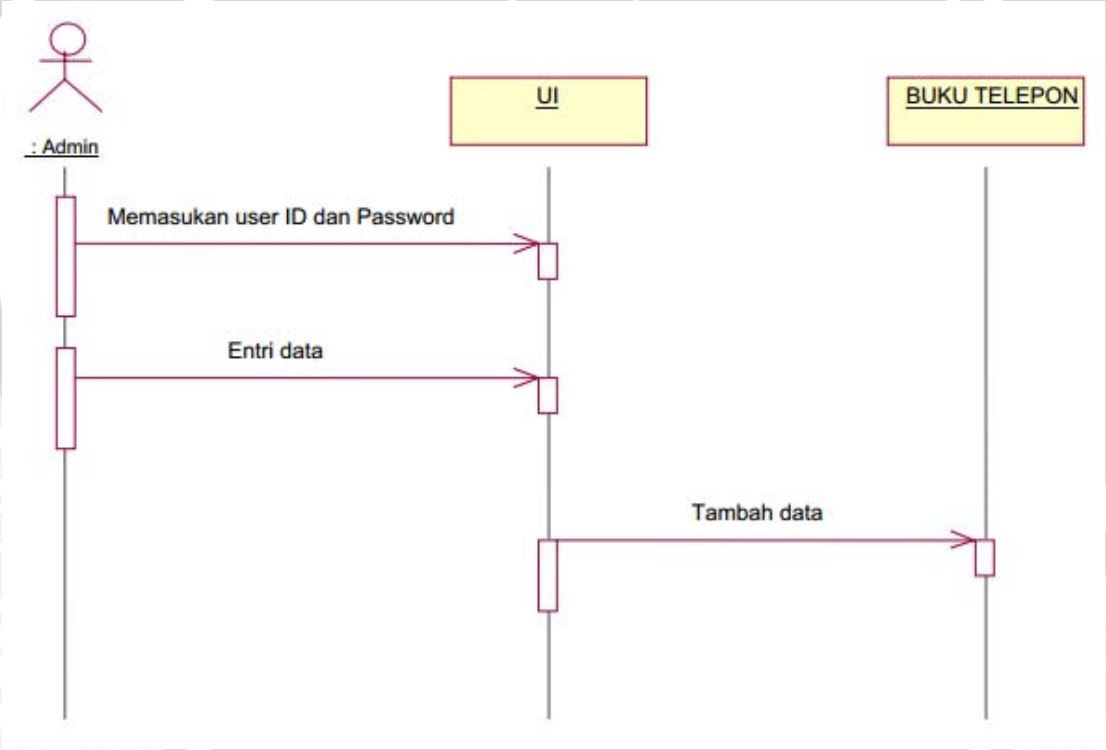
Gambar Garis Hubungan	Keterangan
	Communicate adalah garis yang menghubungkan antara aktor dengan *use case*
«extends»	*Extend* digunakan untuk menetapkan bahwa *use case* yang dituju memperpanjang perilaku dari sumber *use case* .
<<include>>	*Include* digunakan untuk menetapkan bahwa *use case* awal secara eksplisit menggabungkan perilaku dari *use case* yang ditunjuk oleh *use case* awal.

Sumber : [ROS-13]

2.2.7.4 *Sequence Diagram*

Diagram sekuen menggambarkan kolaborasi yang dinamis antara obyek satu dengan yang lain. Kolaborasi ini ditunjukkan dengan adanya interaksi antar obyek di dalam dan di sekitar sistem yang berupa pesan atau instruksi yang berurutan. *Sequence diagram* umumnya digunakan untuk menggambarkan suatu skenario atau urutan langkah-langkah yang dilakukan baik oleh *actor* maupun sistem yang merupakan respon dari sebuah kejadian untuk mendapatkan hasil atau *output* [PRY-05].

Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri, jadi semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.



Gambar 2.6 Contoh *Sequence Diagram*

Sumber : [PRY-05]

2.2.7.5 *Activity Diagram*

Activity diagram atau aktifitas diagram ini menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis yang digunakan untuk mendeskripsikan alur kegiatan yang lainnya seperti *use case* atau interaksi.

Activity diagram juga banyak digunakan untuk mendefinisikan hal-hal berikut [ROS-13] :

- Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
- Urutan atau pengelompokkan tampilan dari sistem (*user interface*) dimana setiap aktivitas dianggap memiliki sebuah rancangan tampilan antarmuka.
- Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
- Rancangan menu yang ditampilkan pada perangkat lunak.

2.2.8 Konsep Basis Data

2.2.8.1 Pengertian Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat.

Sistem informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apapun bentuknya, entah berupa *file* teks maupun *Database Management System* (DBMS). Kebutuhan basis data dalam sistem informasi meliputi [ROS-13] :

- Memasukkan, menyimpan, dan mengambil data.
- Membuat laporan berdasarkan data yang telah disimpan.

Tujuan dari dibuatnya tabel adalah untuk menyimpan data ke dalam tabel agar mudah diakses. Oleh karena itu dalam merancang tabel diperlukan pemikiran yang dalam karena data yang akan disimpan nantinya dalam bentuk baris data (*record*).

2.2.8.2 *Database Management System* (DBMS)

DBMS (*Database Management System*) merupakan suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola dan menampilkan data. Karena pentingnya data bagi suatu perusahaan, maka hampir sebagian perusahaan memanfaatkan DBMS. Pengelolaan DBMS sendiri biasanya ditangani oleh tenaga ahli yang spesialis menangani DBMS yang disebut sebagai DBA (*Database Administrator*).

Berikut ini adalah 4 macam DBMS versi komersial yang paling banyak digunakan dunia saat ini yaitu *Oracle*, *Microsoft SQL Server*, IBM DB2 dan *Microsoft Access*. Sedangkan DBMS versi *open source* yang cukup berkembang dan paling banyak digunakan saat ini adalah MySQL, *PostgreSQL, Firebird,* dan SQLite.

2.2.8.3 *Structured Query Language* (SQL)

SQL atau kepanjangannya adalah *Structured Query Language* adalah bahasa yang digunakan untuk mengelola data pada RDBMS. Produk DBMS nonrelasional menggunakan SQL untuk memanipulasi data. SQL sangat tepat digunakan dalam wilayah akses data relasional karena SQL menyediakan bahasa tingkat tinggi yang meyediakan tingkat abstraksi daripada bahasa prosedural tradisional.

SQL sebaiknya dirancang sedemikian rupa sehingga *programmer* dapat menentukan data apa yang dibutuhkan. DBMS menganalisa setiap pernyataan SQL dan merumuskan instruksi data navigasi di belakang layar, instruksi data ini biasanya disebut jalur akses menggunakan DBMS dapat menentukan optimal jalan akses ke data. Karena DBMS dapat menghasilkan jalur yang lebih efisien dan dinamis akses ke data. Selain itu, karena karakteristik data dan pola akses berubah, DBMS dapat mengubah jalur akses *query* SQL tanpa membutuhkan SQL yang sebenarnya. Contoh *query* SQL adalah :

```
SELECT deptnum, deptname
FROM dept
WHERE supervisornum = '903';
```

SQL juga menggunakan struktur bentuk bebas yang membuatnya sangat fleksibel. Programmer SQL memiliki kemampuan untuk mengembangkan pernyataan SQL dengan cara paling cocok untuk pengguna tertentu. Setiap permintaan SQL diuraikan oleh DBMS sebelum mengeksekusi untuk memeriksa sintaks yang tepat dan mengoptimalkan permintaan. Oleh karena itu, pernyataan SQL tidak dibutuhkan untuk memulai di setiap kolom yang diberikan dan dapat dirangkai dalam satu baris atau pada beberapa baris. Sebagai contoh pernyataan SQL berikut [MUL-02] :

```
SELECT deptnum, deptname FROM dept WHERE supervisornum =
'903';
```

Setiap operasi yang dilakukan pada *database* relasional bertindak atas tabel (atau kumpulan tabel) dan hasil dalam tabel lain. Semua data operasi manipulasi SQL yaitu, SELECT, INSERT, UPDATE, dan DELETE dilakukan pada tingkat yang ditetapkan.

Sebagian besar aplikasi basis data membutuhkan bahasa pemrograman untuk menggunakan SQL unutk berkomunikasi dengan *database.* Berbagai macam *database* yang dapat digunakan dengan SQL dari berbagai bahasa seperti COBOL, FORTRAN, Assembler, C/C++, Java dan Visual Basic.

Meskipun SQL merupakan bahasa tunggal, SQL terdiri dari jenis-jenis yang berbeda, kategori pembagian SQL berdasarkan jenis eksekusi, kebutuhan program, dan dinamisme. SQL dapat direncanakan atau tidak terencana, tertanam dalam program atau berdiri sendiri, dinamis atau statis [MUL-02].

Tabel 2.2 SQL

Situasi	**Tipe Eksekusi**	**Kebutuhan Program**	**Dinamisme**
Kolom dan predikat dari pernyataan SQL dapat berubah selama eksekusi	Terencana	*embedded*	Dinamis
Formula SQL tidak berubah	Terencana	*embedded*	Statis
Dilakukan bersamaan, kinerja transaksi tinggi	Terencana	*embedded*	Dinamis atau statis

Ad-hoc satu *query*	Tidak terencana	Berdiri sendiri	Dinamis
Pertanyaan berulang	Terencana	*Embedded* atau berdiri sendiri	Dinamis atau statis
Quick one-time "fix" programs	Tidak terencana	*Embedded* atau berdiri sendiri	Dinamis atau statis

Sumber : [MUL-02]

2.2.9 Konsep Teknik Dan Strategi Pengujian

Ada beberapa jenis dan strategi dalam pengujian perangkat lunak, yang semuanya itu memiliki satu tujuan yang sama, yaitu meningkatkan kepercayaan diri pengembang perangkat lunak terhadap fungsi-fungsi perangkat lunaknya. Beranjak dari tujuan ini, suatu perangkat lunak dapat diuji untuk menerima berbagai perlakuan dalam rangka pengujian. Diantaranya adalah dengan melakukan tindakan-tindakan yang mungkin dilakukan pengguna perangkat lunak sesuai kebutuhan pengguna dan karakteristik perangkat lunak. Juga, perhitungan terhadap keandalan perangkat lunak selama operasi dilakukan [SMT-10].

Pengujian perangkat lunak adalah elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain dan pengkodean. Pengujian menyajikan anomali yang menarik bagi perekayasa perangkat lunak. Pada proses perangkat lunak, perekayasa pertama-tama berusaha membangun perangkat lunak dari konsep abstrak ke implementasi yang dapat dilihat, baru dilakukan pengujian. Perekayasa menciptakan sederetan *test case* yang dimaksudkan untuk "membongkar" perangkat lunak yang sudah dibangun [JSN-14].

Apapun keterbatasan yang dihadapi, pengujian merupakan bagian yang integral dalam pengembangan perangkat lunak. Pengujian secara luas digunakan di setiap tahapan dalam siklus pengembangan perangkat lunak. Biasanya, lebih dari

50% waktu pengembangan dihabiskan untuk pengujian. Pengujian biasanya dilakukan untuk tujuan berikut [SMT-10]:

1. Untuk meningkatkan kualitas
2. Untuk verifikasi dan validasi (V&V)
3. Untuk keandalan estimasi

2.2.9.1 Klasifikasi Validasi

Pengujian sistem dilakukan dengan melakukan pengujian validasi dengan metode *blackbox*. Pengujian metode *blackbox* merupakan metode yang digunakan untuk mengetahui apakah sistem berfungsi dengan baik dan sesuai dengan harapan tanpa memperhatikan struktur logika internal perangkat lunak. Sedangkan pengujian validasi yaitu untuk mengetahui apakah sistem yang sudah dibangun telah benar sesuai dengan kebutuhan. Jadi, tahapan pengujian validasi pada sistem dengan metode *blackbox* karena tidak memerlukan konsentrasi khusus terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan konformitas antara kinerja sistem dengan daftar kebutuhan.

Software yang berintegrasi diuji berdasarkan pada kebutuhan untuk memastikan bahwa kita memiliki sistem yang benar. Fokus-nya adalah untuk meng-*uncover error* pada:

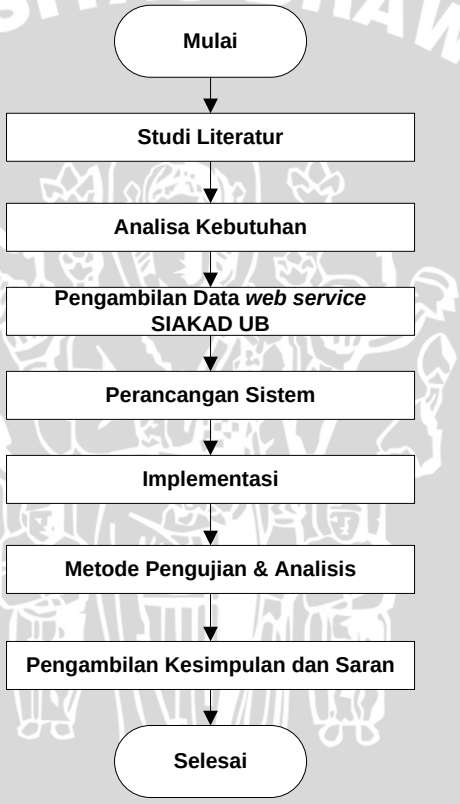
- *Input/output* sistem
- Informasi fungsi sistem dan data
- *Interface* sistem dengan bagian eksternal
- *User interface*
- Perilaku dan *performance* sistem

2.2.9.2 Pengujian REST *Web Service*

Pengujian yang dirancang untuk menguji *software* ketika bekerja dalam konteks *web service* REST dengan bahasa pemrograman JSON. Pengujian ini dilakukan dengan metode validasi untuk mengetahui apakah REST pada *web service* dapat berjalan sesuai dengan yang diinginkan [YSR-14].

BAB III
METODOLOGI PENELITIAN

Bab ini membahas metode dan tahapan pengerjaan yang digunakan dalam penelitian yang terdiri dari studi literatur, pengambilan data *web service* SIAKAD UB, analisa kebutuhan, perancangan sistem, implementasi, metode pengujian dan analisis serta pengambilan kesimpulan dan saran. Berikut adalah diagram alir dari metodologi penelitian yang dilakukan :



Gambar 3.1 Diagram Alir Metodologi Penelitian

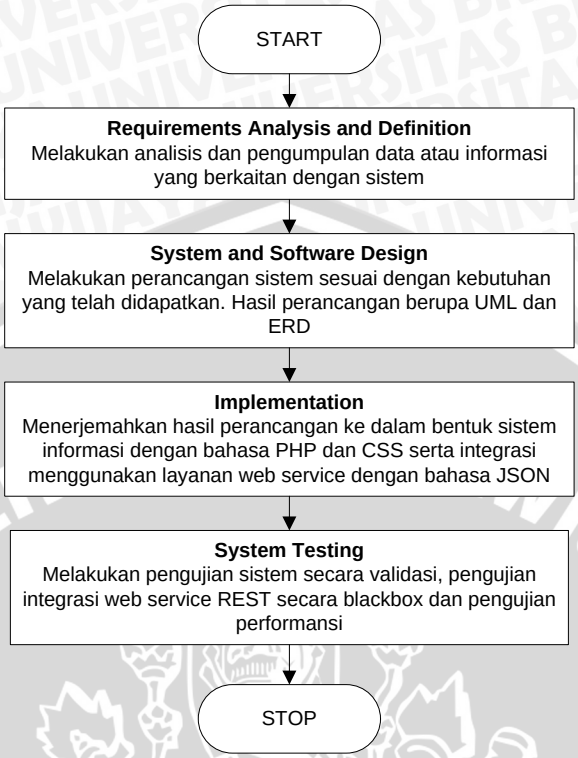
3.1 Studi Literatur

Metode ini digunakan untuk mendapatkan dasar teori sebagai sumber acuan untuk mempelajari ilmu pengetahuan dan perancangan yang berhubungan dengan penulisan skripsi dan perancangan serta pengembangan sistem informasi ini. Teori dan pustaka yang berkaitan dengan skripsi ini meliputi :

1. Semester Pendek PTIIK
    a. Syarat dan Peraturan Semester Pendek
    b. Pengajuan Mata Kuliah
    c. Proses Pembayaran
2. *Web Service* dengan Teknologi REST
3. Pemodelan Rekayasa Perangkat Lunak Secara *Waterfall*
4. UML (*Unified Modeling Language*)
    a. *Use case Diagram*
    b. *Class Diagram*
    c. *Sequence Diagram*
5. Basis Data (*Database*)
6. Pemrograman Berbasis Web Menggunakan PHP dan CSS
7. Pengujian Perangkat Lunak
    a. Pengujian Validasi
    b. Pengujian REST *Web Service*

Studi literatur menjelaskan dasar teori yang digunakan sebagai penunjang dan pendukung penulisan skripsi. Teori penunjang dan pendukung skripsi ini meliputi *web service* dengan teknologi REST berbahasa JSON, sistem semester pendek PTIIK, sistem pembukaan mata kuliah, sistem pembayaran semester pendek dan pemrograman PHP. Sumber atau referensi yang digunakan antara lain buku, jurnal, laporan penelitian, dan bantuan mesin pencari (*search engine*) internet.

Model proses perangkat lunak yang diimplementasikan adalah model sekuensial linier (*classic life cycle* atau *waterfall model*) merupakan paradigma rekayasa perangkat lunak yang paling tua dan paling banyak dipakai. Model ini mengusulkan sebuah pendekatan perkembangan perangkat lunak yang sistematik dan sekunsial yang dimulai pada tingkat dan kemajuan sistem pada seluruh analisis, desain, kode, pengujian, dan pemeliharaan. Adapun tahapan-tahapan dalam pembuatan Model Sekuensial Linier dapat dijelaskan pada Gambar 3.2.



Gambar 3.2 Diagram Alir Pemodelan *Waterfall*

Penjelasan mengenai gambar pemodelan waterfall yang dilakukan sebagai berikut :

1. *Requirements analysis and definition*

   Pada tahapan ini dilakukan pengumpulan kebutuhan pada sistem informasi semester pendek yaitu kebutuhan perangkat keras, perangkat lunak, pengguna dan basis data. Analisis kebutuhan ini penting dilakukan karena sistem informasi (perangkat lunak) yang akan dibangun merupakan bagian dari sistem komputer.

   Kebutuhan data yang harus didapatkan adalah data mata kuliah, data mahasiswa, data KHS serta data syarat dan peraturan pengadaan semester pendek di PTIIK. Pencarian data dilakukan melalui cara *observasi* dan berdiskusi.

2. *System and software design*

   Pada tahap ini mulai dilakukan perancangan sistem dan perangkat lunak sesuai dengan kebutuhan untuk sistem informasi semester pendek

yang berupa data input, proses yang terjadi dan output yang diharapkan melalui tahapan sebelumnya. Hasilnya berupa diagram yang dapat berupa diagram UML seperti *use case*, diagram *sequential*, *class diagram* serta diagram keterhubungan entitas (ERD) yang telah dijelaskan pada bab dasar teori.

3. *Implementation*

   Pada tahap ini, menerjemahkan analisa kebutuhan yang telah dibuat ke dalam bentuk rancangan berbentuk program. Tahap ini dilakukan dengan membuat sistem informasi berbasis *website* dengan menggunakan bahasa pemrograman PHP dan CSS serta melakukan integrasi *database* dengan *web service* menggunakan format pertukaran data JSON dan teknologi *service* REST.

4. *System Testing*

   Sebelum sistem informasi semester pendek ini dapat dipublikasikan, maka harus dilakukan pengujian terlebih dahulu. Pengujian difokuskan pada logika internal (*code*), fungsi eksternal dan mencari semua kemungkinan kesalahan, serta memeriksa apakah *input* maupun *output*-nya sesuai dengan hasil yang diinginkan. Pada skripsi ini melakukan pengujian dengan cara validasi dan pengujian REST *web service* secara *blackbox*.

3.2 Analisa Kebutuhan

Analisis kebutuhan bertujuan untuk mendapatkan semua kebutuhan dalam membangun sistem informasi semester pendek PTIIK. Metode analisis yang digunakan dengan menggunakan bahasa pemodelan UML *(Unified Modeling Language).* Diagram *Use Case* digunakan untuk mendeskripsikan kebutuhan-kebutuhan dan fungsionalitas sistem dari perspektif *user*. Analisis kebutuhan dilakukan dengan mengidentifikasi semua kebutuhan (*requirements*) sistem.

Kebutuhan fungsional yang nantinya akan disediakan oleh sistem informasi ini antara lain adalah :

1. Sistem pada komputer ini harus menyediakan fasilitas untuk login sebagai mahasiswa, sebagai *administrator* dan sebagai *administrator* keuangan. Sehingga *administrator* yang terdaftar, dapat menggunakan layanan sistem untuk memilih mata kuliah yang dapat dibuka untuk mahasiswa, *admin* keuangan dapat melakukan pemasukan pembayaran ataupun pencatatan pengembalian biaya semester pendek mahasiswa. Sedangkan mahasiswa dapat menggunakan layanannya untuk pendaftaran, memilih mata kuliah yang akan diambil serta mengetahui status pembayaran.
2. Pada sistem *user* terdapat fasilitas untuk fungsi registrasi semester pendek dimana *user* dapat melakukan pemilihan mata kuliah semester pendek yang akan diajukan sesuai dengan data pada KHS-nya.

   Tentunya pada saat fungsi ini dieksekusi terdapat beberapa kondisi yang telah disesuaikan dengan syarat dan ketentuan pengadaan semester pendek PTIIK.
3. Sistem juga menyediakan fungsi pembatalan mata kuliah ajuan untuk *user*. Fungsi ini dapat menghapus daftar mata kuliah yang telah diajukan oleh *user* pada semester pendek.
4. Sistem dapat menyediakan fungsi lihat tagihan pembayaran semester pendek untuk *user*.
5. Pada sistem *administrator* akademik dapat melakukan penambahan mata kuliah yang telah diajukan oleh *user* selaku mahasiswa menjadi mata kuliah aktif yang berarti mata kuliah tersebut dibuka di semester pendek.
6. *Administrator* akademik juga dapat melakukan pembatalan mata kuliah aktif yang berarti daftar mata kuliah tersebut dihapus dan tidak dibuka pada semester pendek.
7. Untuk sistem *administrator* keuangan, sistem dapat menampilkan tagihan pembayaran dan tagihan pengembalian mahasiswa.
8. Sistem *administrator* keuangan juga dapat melakukan penambahan pembayaran semester pendek dan pengembalian pembayaran. Ketika *user* melakukan pembayaran, maka secara otomatis sistem dapat mengubah

daftar mata kuliah yang diajukan mahasiswa menjadi mata kuliah yang terdaftar dalam KRS semester pendeknya.

3.3 Pengambilan Data *Web Service* SIAKAD UB

Metode ini digunakan untuk mendapatkan data akses *web service* SIAKAD UB sebagai solusi untuk pengembangan sistem informasi semester pendek ini. Metode yang digunakan salah satunya yaitu berdiskusi dengan dosen dan karyawan PPTI selaku penanggung jawab data *web service* SIAKAD UB. Adapun data yang dimaksud adalah data mahasiswa, data mata kuliah, data SKS per mata kuliah, data KHS sebagai data yang digunakan untuk implementasi. Data-data tersebut digunakan untuk mengintegrasikan kebutuhan sistem semester pendek dengan data yang ada pada aplikasi SIAKAD UB.

3.4 Perancangan Sistem

Perancangan arsitektur sistem adalah tahap dimana penulis mulai merancang suatu sistem yang mampu memenuhi semua kebutuhan fungsional aplikasi dalam skripsi ini. Teori-teori dari pustaka dan data yang dibutuhkan digabungkan dengan ilmu yang didapat diimplementasikan untuk merancang serta mengembangkan suatu *web service* untuk sistem informasi semester pendek di PTIIK. Perancangan sistem berdasarkan *Object Oriented Analysis* dan *Object Oriented Design* yaitu menggunakan pemodelan UML (*Unified Modeling Language*).

Basis data yang dimiliki oleh sistem hanya satu, yang terhubung dengan komputer administrator, komputer administrator keuangan, dan komputer mahasiswa. Sedangkan sistem informasi semester pendek juga melakukan pengambilan data melalui *service* dari aplikasi SIAKAD Universitas Brawijaya dengan menggunakan SOA yang telah disediakan oleh SIAKAD UB. Jaringan komputer dan basis data yang digunakan akan ditunjukkan pada Gambar 4.6.

Alur proses yang terjadi pada sistem informasi dijelaskan melalui gambar diagram UML sesuai dengan aktor yang bersangkutan. Penjelasan akan ditunjukkan pada bab perancangan dan implementasi sub bab perancangan arsitektural.

3.5 Implementasi

Implementasi aplikasi dilakukan dengan mengacu kepada perancangan aplikasi. Implementasi perangkat lunak dilakukan dengan menggunakan implementasi *basis data* MySQL dengan *software* XAMPP 1.7.3 dan bahasa pemrograman PHP serta dihubungkan dengan implementasi *web service* menggunakan SOA yang telah disediakan aplikasi SIAKAD Universitas Brawijaya melalui format JSON.

JSON adalah singkatan dari *JavaScript Object Notation* yaitu format untuk pertukaran data seperti halnya XML. JSON salah satu contoh *web service* untuk mengintegrasikan data antar *platform*. Tipe media internet resmi untuk JSON adalah application/json. Sedangkan ekstensi *file* untuk JSON adalah .json.

Dreamweaver adalah aplikasi *desain* dan pengembangan *website* yang menyediakan editor visual (*Design view*) dan kode editor. Dreamweaver merupakan salah satu *software* untuk bahasa pemrograman PHP dan CSS yang memiliki fitur *browser* yang terintegrasi untuk melihat halaman *web* yang sedang dikembangkan di jendela program sendiri agar konten memungkinkan untuk terbuka di *web browser* yang telah terinstall.

MySQL adalah *software basis data* untuk menyimpan data mahasiswa, mata kuliah, KRS, pembayaran serta administrator yang akan ditampilkan dalam *output website* sistem informasi semester pendek. Mahasiswa yang tidak terdaftar dalam basis data, tidak dapat melakukan pendaftaran dan pengajuan semester pendek melalui sistem informasi ini.

3.6 Pengujian dan Analisis

Pengujian dari perangkat lunak ini berkaitan dengan pengujian sistem (*black box*). Proses pengujian dilakukan melalui tahapan pengujian validasi. Pada pengujian validasi akan digunakan teknik pengujian *black box* (*Black Box Testing*) karena lebih mengutamakan hasil kerja sistem. Pengujian validasi digunakan untuk menguji

fungsional kerja dari sistem. Skenario pada pengujian ini dilakukan dengan menjalankan sistem sesuai dengan fungsi-fungsi internal yang akan dijabarkan pada bab perancangan dan implementasi sub bab diagram use case. Selain itu dilakukan pencatatan dalam bentuk tabel apakah hasil kebutuhan fungsional berhasil terpenuhi oleh sistem (valid) atau tidak (tidak valid).

Pengujian *web service* dalah hal performa aplikasi juga akan dilakukan untuk mengetahui kinerja sistem informasi dalam melayani *request* dari *user* dan mengirimkan *response* ke *user* (mahasiswa) atau dari *admin* ke *user*. Dalam skenarionya, akan dilakukan perhitungan waktu akses yang didapatkan dari fungsi microtime PHP. Pengujian dilakukan selama 10 kali percobaan dan selanjutnya diambil rata-rata untuk di analisis.

Sesuai dengan pengujian *blackbox*, pengujian REST *web service* yang menggunakan strategi *blackbox testing* bertujuan untuk mengetahui apakah sistem dapat berjalan dengan baik (dalam hal ini pengambilan data melalui *web service*), sesuai dengan kebutuhan pengguna dan menganalisa kesalahan-kesalahan yang ada. Skenario yang akan dijalankan sama dengan pengujian validasi yang telah dijelaskan sebelumnya. Hanya saja perbedaan pada objek pengujian yaitu *request* REST *web service*.

Analisis sistem dilakukan dengan melakukan evaluasi mengenai kesulitan yang dialami selama proses pengerjaan dan membandingkan dengan hasil pengujian data sehingga didapatkan suatu kesimpulan.

3.7 Pengambilan Kesimpulan

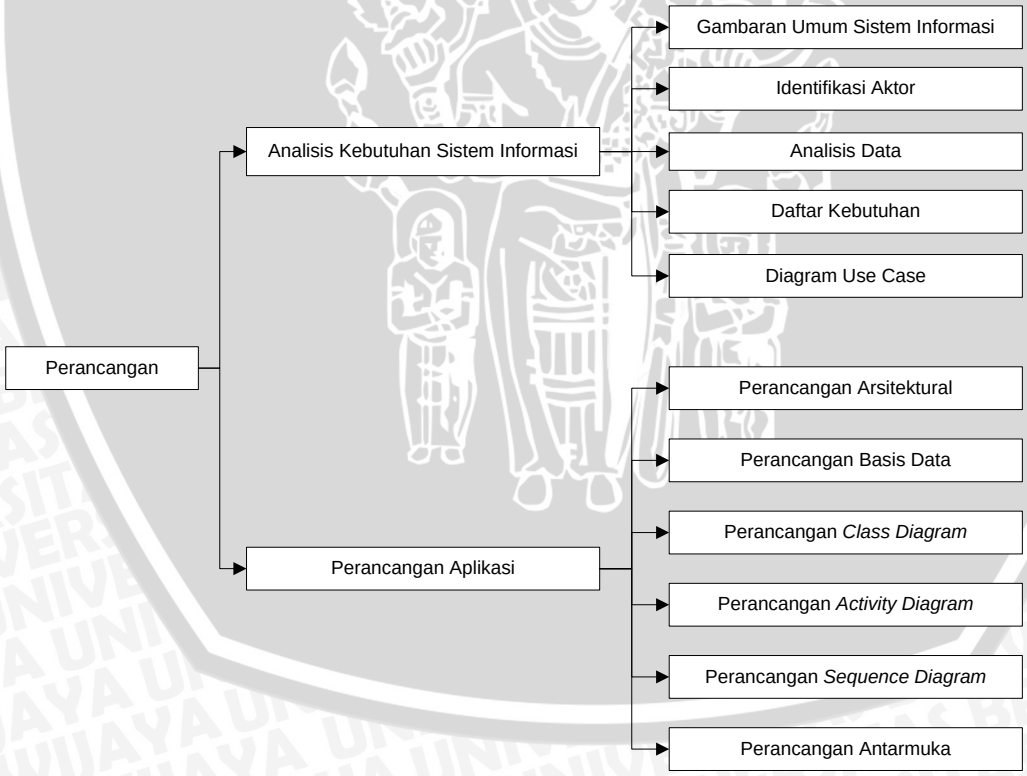
Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan, implementasi dan pengujian sistem aplikasi telah selesai dilakukan dan didasarkan pada kesesuaian antara teori dan praktik. Kesimpulan diambil untuk menjawab rumusan masalah yang telah ditetapkan sebelumnya. Tahap terakhir dari penulisan adalah saran yang dimaksudkan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan sistem informasi selanjutnya.

BAB IV
PERANCANGAN DAN IMPLEMENTASI

Bab ini membahas mengenai perancangan dan implementasi aplikasi. Perancangan yang dilakukan meliputi dua tahap yaitu analisis kebutuhan dan perancangan aplikasi. Sedangkan pembahasan implementasi terdiri atas penjelasan tentang spesifikasi lingkungan implementasi, batasan-batasan dalam implementasi, implementasi tiap *class* pada *file* program, dan implementasi sistem.

4.1 Perancangan

Pada sub bab ini dibahas mengenai perancangan yang dibutuhkan dalam sistem sesuai bab metodologi penelitian. Struktur sub bab perancangan ditunjukkan pada Gambar 4.1.



Gambar 4.1 Struktur Sub Bab Perancangan

Proses analisis kebutuhan dilakukan pada tahap pertama dan tahap kedua proses perancangan sistem informasi. Tahap analisis kebutuhan terdiri atas lima langkah yaitu melakukan penjabaran tentang gambaran umum aplikasi, melakukan proses identifikasi aktor yang telibat dalam aplikasi, melakukan proses analisis data yang diperlukan, membuat daftar kebutuhan pengguna menggunakan pemodelan diagram *use case*. Proses perancangan basis data, pemodelan diagram *class*, pemodelan *activity diagram*, pemodelan diagram *sequence*, dan perancangan antarmuka pengguna dari aplikasi.

4.1.1 Analisis Kebutuhan Sistem Informasi

Proses analisis kebutuhan ini diawali dengan penjabaran gambaran umum aplikasi, identifikasi aktor yang terlibat, analisis data yang akan disimpan, penjabaran tentang daftar kebutuhan dan kemudian memodelkannya ke dalam diagram *use case*, analisis kebutuhan ini bertujuan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh sistem agar dapat memenuhi kebutuhan pengguna.

4.1.1.1 Gambaran Umum Sistem Informasi

Pembahasan gambaran umum sistem informasi semester pendek PTIIK terdiri atas dua bagian, yaitu deskripsi umum sistem informasi semester pendek dan lingkungan sistem informasi semester pendek.

1. **Deskripsi Sistem Informasi Semester Pendek**

Sistem informasi semester pendek PTIIK dapat digunakan untuk membantu pengguna dalam hal ini mahasiswa, *staff* akademik, dan lembaga pendidikan PTIIK dalam mengelola pengadaan semester pendek PTIIK sekaligus memudahkan proses pembayaran, pendataan dan pendaftarannya.

Sistem informasi semester pendek PTIIK memiliki tiga bagian utama yaitu, subsistem untuk mahasiswa (*user*), subsistem untuk *staff* akademik (*administrator*), subsistem *staff* keuangan.

a. Sistem Informasi Semester Pendek Bagi Mahasiswa (*user*)

Sistem informasi semester pendek untuk mahasiswa adalah sistem informasi yang berfungsi sebagai alat bantu mahasiswa dalam pendaftaran semester pendek dan menentukan matakuliah yang dapat diambil dalam semester pendek, sekaligus mengetahui pembayaran yang harus dibayarkan.

b. Sistem Informasi Semester Pendek Bagi *Staff* Akademik (*administrator*)

Sistem informasi semester pendek *administrator* adalah sistem informasi yang digunakan untuk mengelola data mata kuliah yang dapat diambil pada semester pendek, data mahasiswa yang mengajukan matakuliah semester pendek sehingga dapat dikelola sesuai ketentuan dan syarat tertentu yang tersimpan pada *database server*.

c. Sistem Informasi Semester Pendek Bagi Staff Keuangan (*administrator* keuangan)

Sistem informasi semester pendek *administrator* keuangan adalah sistem informasi yang digunakan untuk membantu *admin* keuangan dalam mengelola dan mengelola transaksi mahasiswa yang akan mengikuti semester pendek sekaligus memudahkan proses pembayaran untuk membantu efektifitas secara optimal.

2. Lingkungan Sistem Informasi

Sistem informasi semester pendek PTIIK ini membutuhkan suatu lingkungan yang digunakan sebagai tempat berjalannya sistem. Secara keseluruhan sistem informasi ini berbasis *web application* yang diintegrasikan dengan *web service* melalui SOA milik SIAKAD Universitas Brawijaya dengan teknologi REST dalam format JSON, sehingga membutuhkan sebuah *browser* untuk menjalankan sistem informasi tersebut.

4.1.1.2 Identifikasi Aktor

Tahap ini adalah untuk melakukan identifikasi terhadap aktor-aktor yang akan berinteraksi dengan aplikasi. Tabel 4.1 memperlihatkan aktor-aktor yang terlibat beserta penjelasannya.

Tabel 4.1 Identifikasi Aktor

Aktor	Deskripsi
User / Mahasiswa	*User* / Mahasiswa adalah pengguna yang memiliki hak untuk memilih, mengubah dan membatalkan ajuan mata kuliah untuk semester pendek yang akan diambil
Administrator / Staff Akademik	Administrator adalah pengguna sistem informasi yang memiliki hak untuk melihat mata kuliah ajuan dan dapat memilih mata kuliah ajuan terbanyak agar dapat dibuka
Administrator Keuangan / Staff Keuangan	Administrator Keuangan adalah pengguna sistem informasi keuangan semester pendek PTIIK yang memiliki hak untuk memasukkan pembayaran mahasiswa dan mengembalikan pembayaran mahasiswa yang ada pada *database*.

4.1.1.3 Analisis Data

Analisis data bertujuan untuk mendapatkan struktur penyimpanan data yang dibutuhkan. Struktur penyimpanan data pada aplikasi disusun berdasarkan analisis data sebagai berikut :

1. Data mahasiswa yang terdiri dari NIM dan password mahasiswa.
2. Data *administrator* baik bagian akademik maupun keuangan yang terdiri dari *username administrator* dan *password administrator*.
3. Data mata kuliah yang terdiri dari kode mata kuliah, tahun mata kuliah, nama mata kuliah dan jumlah SKS tiap mata kuliah.

4. Data pembayaran yang terdiri dari id_transaksi, NIM mahasiswa, K_MK, total tagihan, total pengembalian, status tagihan, status pengembalian, waktu pembayaran dan waktu pengembalian.
5. Data KRS yang terdiri dari NIM mahasiswa, kode mata kuliah, kode nilai, jumlah SKS, tahun mata kuliah, status semester pendek tahun ganjil, status ajuan dan status mata kuliah yang aktif dibuka pada semester pendek.
6. Data Berita yang terdiri dari judul, content, gambar, dan waktu.
7. Data m_user yang terdiri dari nama atau *username* seluruh pengguna sistem, *password*, serta rule yang membedakan antara pengguna sistem informasi sebagai *user*, *administrator* akademik, maupun *administrator* keuangan.

4.1.1.4 Analisis *Web Service* SIAKAD UB

Analisis *web service server* pusat SIAKAD UB bertujuan untuk mendapatkan informasi mengenai perancangan kebutuhan data yang dibutuhkan sistem dengan data yang disediakan oleh SIAKAD UB serta mendeskripsikan bagaimana proses integrasi data antara sistem dengan aplikasi SIAKAD berlangsung melalui *web service*. Pendeskripsian proses integrasi data *web service server* SIAKAD UB berdasarkan analisis data sebagai berikut :

1. Format pengiriman data dengan menggunakan JSON dan REST *web service* yang disesuaikan dengan *web service* aplikasi SIAKAD UB.
2. Data masukan yang dibutuhkan dalam *request web service* adalah NIM mahasiswa.
3. Menuliskan *service endpoint* dari *server* di *file request* data *web service* **(http://devel184.ub.ac.id/siakad_api/api/akademik/CallGetKhsMhs)**.
4. Menjalankan *browser*
5. Sistem *user* pada sistem informasi semester pendek mengirim parameter berbentuk url yang merujuk *file request* dengan menambahkan parameter *query* berupa NIM mahasiswa, semester mahasiswa (parameter ini tidak berhubungan dengan data yang akan diambil), is_pendek dan is_*current* yang telah ditentukan **(http://localhost/ws-semester-**

pendek/test.nilai.khs.php?nim=$nim&semester=2&is_pendek=0&is_current=0).

6. Data yang dikembalikan berstruktur JSON. Dalam hal ini, penggunaan struktur JSON yang diperlukan hanya data KHS yang telah ada pada aplikasi SIAKAD UB. Struktur Data KHS Mahasiswa yang dikirimkan adalah sebagai berikut :

{ "SEMESTER":integer,
"K_MK":"varchar(9)",
"THN_MK":integer,
"NAMA":"varchar(50)",
"K_NILAI":"varchar(9)",
"JML_SKS":"varchar",
"IS_PENDEK":"char",
"K_TIPE_MK":"varchar(10)"
}

4.1.1.5 Daftar Kebutuhan

Daftar Kebutuhan terdiri dari kebutuhan fungsional. Pada daftar kebutuhan fungsional akan dispesifikasikan menjadi tiga yaitu spesifikasi kebutuhan fungsional mahasiswa sebagai *user*, klasifikasi kebutuhan fungsional staff akademik sebagai *administrator*, serta klasifikasi kebutuhan fungsional staff keuangan sebagai *administrator* keuangan.

Tabel 4.2 Spesifikasi kebutuhan fungsional *user*

Nomor SRS	Kebutuhan	*Use Case*
SRS_1_01	Perangkat lunak harus mampu menyediakan fasilitas untuk menangani proses manajemen mata kuliah semester pendek untuk mahasiswa.	Manajemen Mata Kuliah Semester Pendek

SRS_1_02	Sistem informasi harus mampu menyediakan fasilitas untuk mengambil data mata kuliah semester pendek yang telah diambil.	Lihat Mata Kuliah KRS Semester Pendek
SRS_1_03	Sistem informasi harus mampu menyediakan fasilitas untuk mengambil data mata kuliah semester pendek yang berstatus aktif atau telah dibuka oleh *administrator*.	Lihat Mata Kuliah Aktif Semester Pendek
SRS_1_04	Sistem informasi harus mampu menyediakan fasilitas untuk mengambil data tagihan pembayaran mahasiswa yang mendaftar semester pendek sesuai kalkulasi dan ketentuan semester pendek.	Lihat Tagihan Biaya Semester Pendek

Tabel 4.3 Spesifikasi kebutuhan fungsional *admin* akademik

Nomor SRS	**Kebutuhan**	***Use Case***
SRS_2_01	Sistem informasi harus mampu menyediakan fasilitas untuk mengambil data mata kuliah ajuan yang telah dipilih mahasiswa serta jumlah peminatnya untuk kemudian diaktifkan (dibuka) pada semester pendek.	Lihat Mata Kuliah Ajuan
SRS_2_02	Perangkat lunak harus mampu menyediakan fasilitas untuk menangani proses pengelolaan mata kuliah aktif yang dapat dibuka pada semester pendek.	Manajemen Mata Kuliah Aktif
SRS_2_03	Perangkat lunak harus mampu menyediakan fasilitas untuk mengelola berita atau informasi mengenai semester	Manajemen Informasi atau Pengumuman

	pendek	
SRS_2_04	Perangkat lunak harus mampu menyediakan fungsi untuk keluar dari sistem informasi *administrator* semester pendek PTIIK.	*Log Out*

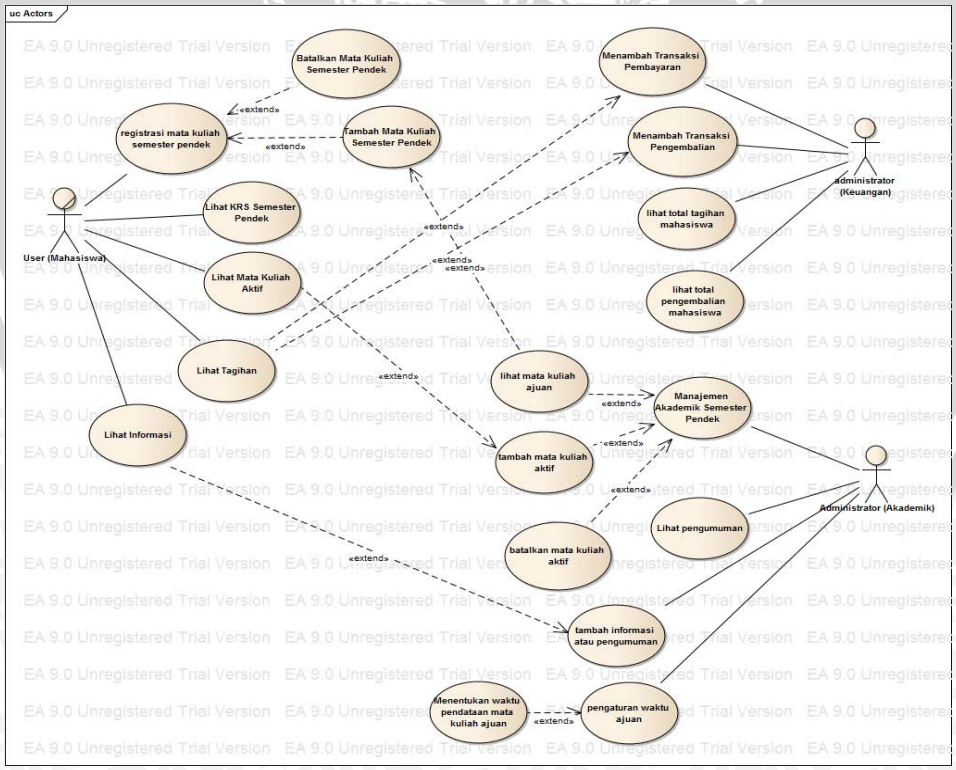
Tabel 4.4 Spesifikasi kebutuhan fungsional *admin* keuangan

Nomor SRS	Kebutuhan	*Use Case*
SRS_3_01	Sistem informasi semester pendek harus mampu menyediakan fasilitas *sign in* sehingga hanya administrator keuangan saja yang dapat mengelola sistem pembayaran semester pendek.	*Log In*
SRS_3_02	Sistem informasi harus mampu menyediakan fasilitas penambahan transaksi pembayaran untuk data mahasiswa yang akan mendaftarkan mata kuliah ajuan semester pendek sesuai dengan ketentuan dan syarat yang diberlakukan.	Transaksi Pembayaran
SRS_3_03	Sistem informasi harus mampu menyediakan fasilitas penambahan transaksi pengembalian untuk mahasiswa yang memiliki sisa saldo di keuangan.	Transaksi Pengembalian
SRS_3_04	Perangkat lunak harus mampu menyediakan fasilitas untuk menampilkan tagihan pembayaran mahasiswa yang mengikuti semester pendek.	Lihat Tagihan Pembayaran Mahasiswa
SRS_3_05	Perangkat lunak harus mampu	Lihat Tagihan

	menyediakan fasilitas untuk menampilkan tagihan pengembalian pembayaran mahasiswa yang mengikuti semester pendek.	Pengembalian Mahasiswa
SRS_3_06	Perangkat lunak harus mampu menyediakan fungsi untuk keluar dari sistem informasi *administrator* keuangan semester pendek PTIIK.	*Log Out*

4.1.1.6 Diagram *Use Case*

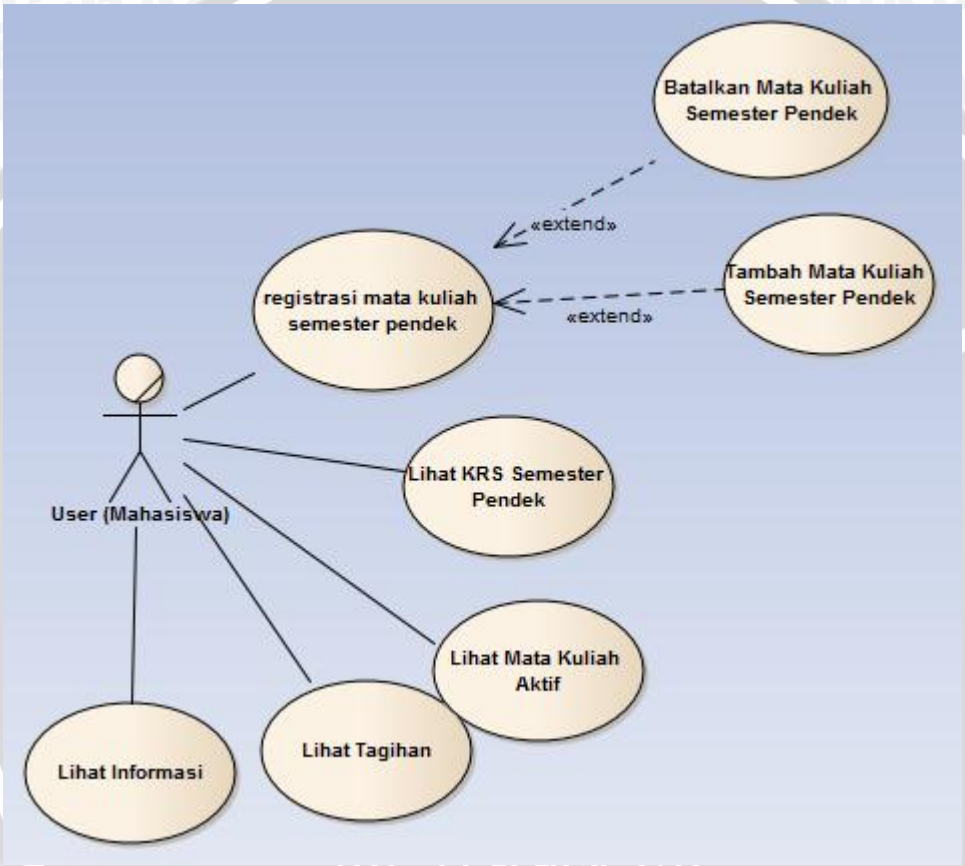
Diagram *use case* adalah salah satu diagram untuk memodelkan aspek perilaku sistem. Diagram *use case* menunjukkan sekumpulan *use case*, aktor, dan hubungannya. *Use case* merupakan fungsionalitas dari sistem yang diinisialisasi oleh aktor. Berikut ini merupakan diagram *use case* sistem secara keseluruhan ditunjukkan pada Gambar 4.2.



Gambar 4.2 Diagram *Use Case* Sistem

1. **Diagram *Use Case* Mahasiswa *(User)***

Diagram *use case* ini melibatkan *user* atau mahasiswa sebagai aktor dan 7 buah *use case*. 7 buah *use case* ini juga akan disertai dengan skenario *use case* untuk menjelaskan rangkaian aktivitas yang terjadi di *use case* tersebut. Diagram *use case* untuk mahasiswa (*user*) ditunjukkan pada Gambar 4.3 berikut ini :



Gambar 4.3 Diagram *Use Case* Mahasiswa (*User*)

Tabel 4.5 *Use case* tambah mata kuliah *user*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_1_01_01
Nama	Tambah mata kuliah semester pendek
Tujuan	Agar mahasiswa (*user*) dapat menambah mata kuliah yang akan diambil pada semester pendek
Deskripsi	*Use Case* ini menjelaskan bagaimana proses memasukkan atau menambah mata

	kuliah yang akan dipilih *user* di semester pendek
Aktor	Mahasiswa (*user*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman registrasi semester pendek
Aksi Aktor	Reaksi Sistem
1. *User* menekan tombol ‘registrasi semester pendek’ pada sistem	2. Sistem menampilkan halaman data mata kuliah yang telah diambil yang terdapat pada KHS
3. *User* memilih mata kuliah yang akan didaftarkan di semester pendek dengan menekan tombol ‘tambahkan’	4. Sistem menerima dan memasukkan data masukan *user* kemudian menyimpannya di *database*
Skenario Alternatif 1: Jika jumlah mata kuliah yang dipilih melebihi 12 SKS	
	5. Sistem akan menampilkan pesan *error* untuk menambah mata kuliah maksimal 12 SKS
Skenario Alternatif 2 : Jika jumlah mata kuliah yang dipilih nilai > C+	
	6. Sistem akan menampilkan pesan *error* untuk menambah mata kuliah maksimal nilai C+
Kondisi Akhir	Mata kuliah yang dipilih oleh user akan disimpan di *database server* untuk diproses selanjutnya.

Tabel 4.6 *Use case* batalkan mata kuliah *user*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_1_01_02
Nama	Batal Mata Kuliah Semester Pendek
Tujuan	Agar mahasiswa (*user*) dapat membatalkan data mata kuliah semester pendek yang telah diajukan
Deskripsi	*Use Case* ini menjelaskan bagaimana *user* dapat melakukan proses pembatalan terhadap mata kuliah yang telah didaftarkannya.
Aktor	Mahasiswa (*user*)

Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman batalkan mata kuliah
Aksi Aktor	Reaksi Sistem
1. *User* memilih tab ‘batalkan mata kuliah’ 2. Sistem menampilkan daftar mata kuliah yang telah terdaftar dalam KRS semester pendek 3. *User* menekan tombol ‘batalkan’ pada mata kuliah yang akan dihapus dari daftar ajuan	4. Sistem menerima data masukan user kemudian mengirimkan request untuk menghapus data mata kuliah yang didaftarkan ke server
Kondisi Akhir	Data mata kuliah ajuan *user* berhasil dihapus

Tabel 4.7 *Use case* lihat KRS semester pendek *user*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_1_02
Nama	Lihat KRS semester pendek
Tujuan	Agar mahasiswa (*user*) dapat melihat mata kuliah semester pendek yang telah didaftarkan sebelumnya
Deskripsi	*Use Case* ini menjelaskan bagaimana *user* melakukan proses untuk melihat tabel data mata kuliah semester pendek mahasiswa yang telah diajukan
Aktor	Mahasiswa (*user*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman utama *user*
Aksi Aktor	Reaksi Sistem
1. *User* menekan tombol ‘lihat KRS semester pendek’ pada sistem	2. Sistem menampilkan halaman data informasi mengenai mata kuliah yang telah dipilih dan akan ataupun memiliki status dibuka di semester pendek
Kondisi Akhir	Halaman daftar tabel KRS semester

	pendek berisi mata kuliah yang telah dipilih *user* dapat ditampilkan

Tabel 4.8 *Use case* lihat mata kuliah aktif *user*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_1_03
Nama	Lihat mata kuliah aktif
Tujuan	Agar mahasiswa (*user*) dapat melihat mata kuliah yang telah dibuka (berstatus aktif) di semester pendek
Deskripsi	*Use Case* ini menjelaskan bagaimana *user* melakukan proses untuk melihat tabel data mata kuliah aktif semester pendek
Aktor	Mahasiswa (*user*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman utama *user*
Aksi Aktor	Reaksi Sistem
1. *User* menekan tombol 'lihat mata kuliah aktif' pada sistem	2. Sistem menampilkan halaman data informasi mengenai mata kuliah terbuka (berstatus aktif) yang dapat kembali dipilih di semester pendek oleh mahasiswa
Kondisi Akhir	Halaman daftar tabel mata kuliah aktif semester pendek untuk *user* dapat ditampilkan

Tabel 4.9 *Use case* lihat tagihan *user*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_1_04
Nama	Lihat tagihan pembayaran semester pendek
Tujuan	Agar mahasiswa (*user*) dapat melihat tagihan pembayaran semester pendek
Deskripsi	*Use Case* ini menjelaskan bagaimana *user* melakukan proses untuk melihat tagihan pembayaran atau pengembalian semester pendek
Aktor	Mahasiswa (*user*)
Skenario Utama	

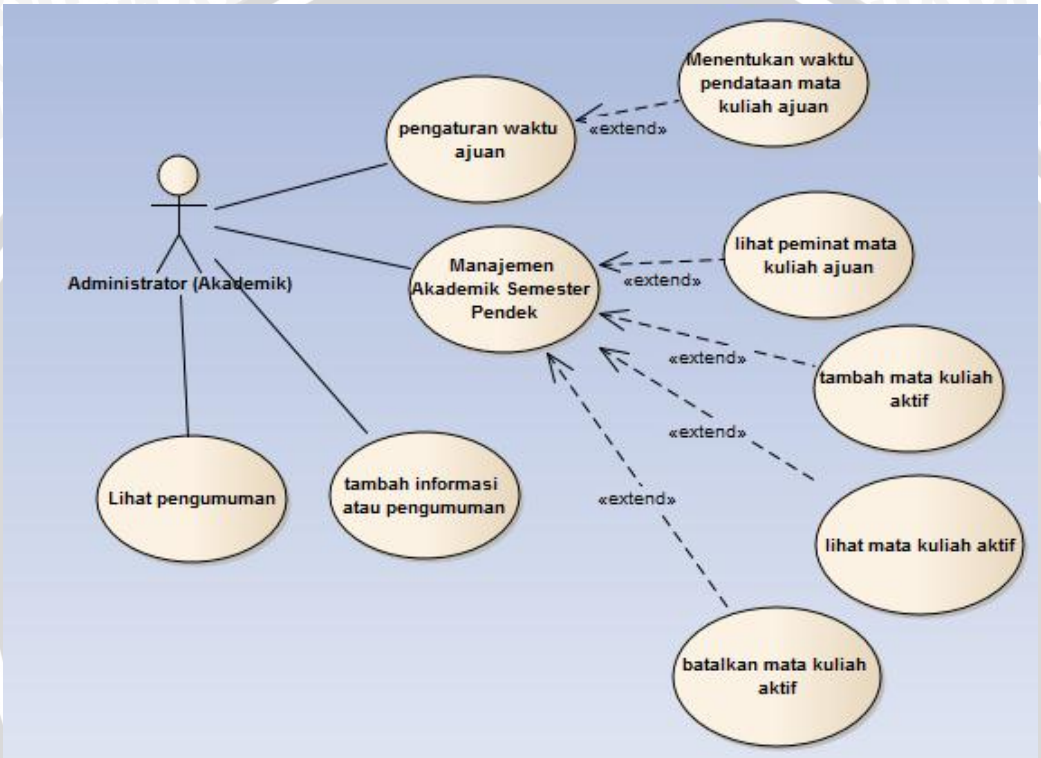
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman utama *user*
Aksi Aktor	Reaksi Sistem
1. *User* menekan tombol 'lihat tagihan' pada sistem	2. Sistem menampilkan halaman data informasi mengenai tagihan pembayaran semester pendek yang telah dikalkulasi sesuai ketentuan semester pendek PTIIK
Kondisi Akhir	Halaman daftar tabel tagihan pembayaran semester pendek untuk *user* dapat ditampilkan

Tabel 4.10 *Use case* lihat informasi *user*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_1_05
Nama	Lihat informasi semester pendek
Tujuan	Agar mahasiswa (*user*) dapat melihat informasi maupun berita mengenai semester pendek
Deskripsi	*Use Case* ini menjelaskan bagaimana *user* melakukan proses untuk melihat informasi maupun berita mengenai semester pendek
Aktor	Mahasiswa (*user*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman utama *user*
Aksi Aktor	Reaksi Sistem
1. *User* menekan tombol 'lihat informasi' pada sistem	2. Sistem menampilkan halaman data informasi mengenai pengumuman semester pendek PTIIK
Kondisi Akhir	Halaman daftar informasi atau pengumuman untuk *user* dapat ditampilkan

2. Diagram *Use Case* Akademik *(Administrator)*

Diagram *use case* ini melibatkan *administrator* atau *staff* akademik sebagai aktor dan 7 buah *use case*. 7 buah *use case* ini juga akan disertai dengan skenario *use case* untuk menjelaskan rangkaian aktivitas yang terjadi di *use case* tersebut. Diagram *use case* untuk akademik (*administrator*) ditunjukkan pada Gambar 4.4 berikut ini :



Gambar 4.4 Diagram *Use Case* Akademik (*Administrator*)

Tabel 4.11 *Use case* pengaturan waktu ajuan *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_2_01_01
Nama	Menentukan waktu pendataan mata kuliah ajuan
Tujuan	Agar akademik (*administrator*) dapat menentukan tenggang waktu proses pengadaan semester pendek
Deskripsi	*Use Case* ini menjelaskan bagaimana proses menambah keterangan batas waktu selama pengadaan semester pendek PTIIK

Aktor	Akademik (*administrator*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal sistem informasi semester pendek untuk *admin*
Aksi Aktor	Reaksi Sistem
1. *Administrator* menekan tombol 'aktifkan semester pendek mata' pada sistem	2. Sistem menampilkan halaman yang berisi form input tanggal
3. *Administrator* memasukkan batas tanggal proses pengadaan semester pendek	4. Sistem memasukkan inputan data tersebut untuk digabung dengan database lainnya
Kondisi Akhir	Halaman inputan ling semester pendek untuk *akademik* dapat ditampilkan

Tabel 4.12 *Use case* lihat mata kuliah ajuan *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_2_02_01
Nama	Lihat mata kuliah ajuan
Tujuan	Agar staff akademik (*admin*) dapat melihat mata kuliah ajuan yang telah dipilih sesuai daftar dan jumlah peminat mata kuliah tersebut
Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* melakukan proses untuk melihat tabel data mata kuliah ajuan semester pendek
Aktor	Staff akademik (*admin*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal admin
Aksi Aktor	Reaksi Sistem
1. *Admin* menekan tombol 'mata kuliah ajuan' pada sistem	2. Sistem menampilkan halaman data informasi mengenai mata kuliah ajuan yang jumlah peminatnya paling banyak sampai paling sedikit.
Kondisi Akhir	Halaman daftar tabel mata kuliah ajuan semester pendek untuk *admin* dapat ditampilkan

Tabel 4.13 *Use case* tambah mata kuliah aktif *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_2_02_02
Nama	Tambah mata kuliah aktif
Tujuan	Agar staff akademik (*administrator*) dapat menambah mata kuliah yang dapat diaktifkan atau dibuka pada semester pendek
Deskripsi	*Use Case* ini menjelaskan bagaimana *user* melakukan proses untuk menambah mata kuliah aktif pada semester pendek
Aktor	Akademik (*administrator*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal mata kuliah ajuan semester pendek
Aksi Aktor	Reaksi Sistem
1. *Admin* menekan tombol ‘mata kuliah ajuan’ pada sistem semester pendek.	2. Sistem menampilkan halaman daftar tabel mata kuliah ajuan yang akan dipilih untuk di ajukan pada semester pendek
3. *Admin* memilih mata kuliah yang akan dibuka di semester pendek 4. *Admin* menekan tombol ‘aktifkan’	5. Sistem menerima dan memasukkan data masukan *admin* kemudian menyimpannya di *database server*
Skenario Alternatif 1: Jika jumlah peminat mata kuliah < 10 mahasiswa	
	6. Sistem akan menampilkan pesan eror, mata kuliah tidak dapat diaktifkan
Kondisi Akhir	Mata kuliah yang dipilih oleh *admin* akan disimpan di *database server* untuk diproses selanjutnya.

Tabel 4.14 *Use case* lihat mata kuliah aktif *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_2_02_02
Nama	Lihat mata kuliah aktif
Tujuan	Agar staff akademik (*admin*) dapat melihat mata kuliah berstatus aktif dapat

	dibuka akademik.
Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* melakukan proses untuk melihat tabel daftar mata kuliah aktif semester pendek
Aktor	Staff akademik (*admin*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal admin
Aksi Aktor	Reaksi Sistem
1. *Admin* menekan tombol 'mata kuliah aktif' pada sistem	2. Sistem menampilkan halaman data informasi mengenai mata kuliah terbuka dan telah aktif di semester pendek
Kondisi Akhir	Halaman daftar tabel mata kuliah aktif semester pendek untuk *admin* dapat ditampilkan

Tabel 4.15 *Use case* batal mata kuliah aktif *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_2_02_03
Nama	Batal mata kuliah aktif
Tujuan	Agar *staff* akademik (*admin*) dapat menghapus informasi data mata kuliah aktif semester pendek yang diinginkan
Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* dapat melakukan proses penghapusan terhadap mata kuliah yang telah diaktifkan (telah dibuka).
Aktor	Staff akademik (*admin*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal mata kuliah
Aksi Aktor	Reaksi Sistem
1. *Admin* memilih daftar mata kuliah yang akan dihapus dari manajemen mata kuliah ajuan 2. *Admin* menekan tombol 'batalkan' pada mata kuliah yang akan di hapus	3. Sistem menerima data masukan *admin* kemudian mengirimkan *request* untuk menghapus data mata kuliah yang dibuka ke *server database*

Kondisi Akhir	Data mata kuliah ajuan *administrator* berhasil dihapus

Tabel 4.16 *Use case* tambah informasi *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_2_03
Nama	Tambah Informasi
Tujuan	Agar staff akademik (*administrator*) dapat menambahkan informasi.
Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* melakukan proses untuk menambahkan informasi, berita, ataupun pengumuman unutk semester pendek
Aktor	Staff Akademik (*admin*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal admin
Aksi Aktor	Reaksi Sistem
1. *Admin* menekan tombol dropdown 'informasi' selanjutnya menekan tombol 'tambah' pada sistem 2. *Admin* memasukkan isi berita atau pengumuman	3. Sistem menampilkan pesan sukses penambahan berita
Kondisi Akhir	Pesan sukses pada halaman tambah berita dapat ditampilkan

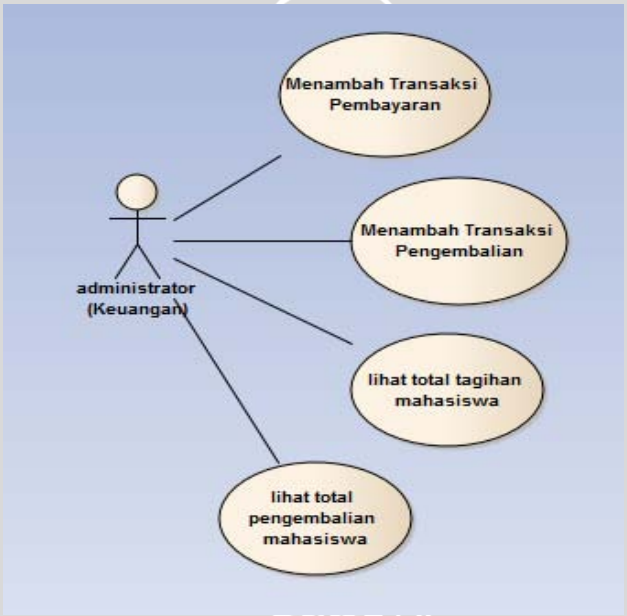
Tabel 4.17 *Use case* lihat informasi *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_2_04
Nama	Lihat Informasi atau Pengumuman
Tujuan	Agar staff akademik (*administrator*) dapat menambah informasi yang berkaitan dengan semester pendek
Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* melakukan proses untuk melihat berita atau pengumuman yang pernah dituliskannya
Aktor	Staff Akademik (*administrator*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal admin
Aksi Aktor	Reaksi Sistem

1. *Admin* menekan tombol 'lihat informasi' pada sistem	2. Sistem menampilkan halaman kumpulan berita dan pengumuman yang pernah dituliskannya
Kondisi Akhir	Pengumuman yang dipilih oleh *admin* dapat ditampilkan secara keseluruhan

1. Diagram *Use Case* Keuangan *(Administrator* Keuangan*)*

Diagram *use case* ini melibatkan *administrator* atau staff keuangan sebagai aktor dan 3 buah *use case*. 3 buah *use case* ini juga akan disertai dengan skenario *use case* untuk menjelaskan rangkaian aktivitas yang terjadi di *use case* tersebut. Diagram *use case* untuk keuangan (*administrator*) ditunjukkan pada Gambar 4.5 berikut ini :



Gambar 4.5 Diagram *Use Case* Keuangan (*Administrator*)

Tabel 4.18 *Use case* tambah transaksi pembayaran *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_3_01
Nama	Tambah Transaksi Pembayaran
Tujuan	Agar staff keuangan (*admin*) dapat menambah daftar transaksi pembayaran mahasiswa terkait semester pendek

Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* melakukan proses untuk menambah data transaksi tagihan pembayaran mahasiswa sesuai dengan mata kuliah yang didaftarkan semester pendek
Aktor	Staff keuangan (*admin*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman lihat pembayaran terdaftar
Aksi Aktor	Reaksi Sistem
1. *Admin* memasukkan NIM dan nama mahasiswa	2. Sistem menampilkan halaman daftar KRS semester pendek mahasiswa yang bersangkutan
3. *Admin* melihat daftar tagihan pembayaran 4. Admin menekan tombol 'bayar' apabila ada yang ingin membayar	5. Sistem menampilkan pesan berhasil, kembali ke halaman daftar tagihan
Kondisi Akhir	Halaman daftar tabel tagihan pembayaran semester pendek untuk *admin* dapat ditampilkan

Tabel 4.19 *Use case* tambah transaksi pengembalian *admin*

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_3_02
Nama	Tambah Transaksi Pengembalian
Tujuan	Agar staff keuangan (*admin*) dapat menambah daftar transaksi pengembalian mahasiswa terkait semester pendek
Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* melakukan proses untuk menambah data transaksi tagihan pengembalian mahasiswa sesuai dengan mata kuliah yang didaftarkan semester pendek
Aktor	Staff keuangan (*admin*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman lihat pembayaran

	terdaftar
Aksi Aktor	Reaksi Sistem
1. *Admin* memasukkan NIM dan nama mahasiswa	2. Sistem menampilkan halaman daftar tagihan semester pendek mahasiswa yang bersangkutan
3. *Admin* melihat daftar tagihan pengembalian 4. Admin menekan tombol 'bayar' apabila ada yang ingin membayar	5. Sistem menampilkan pesan berhasil, kembali ke halaman daftar tagihan
Kondisi Akhir	Halaman daftar tabel pengembalian semester pendek untuk *admin* dapat ditampilkan

Tabel 4.20 *Use case* lihat tagihan mahasiswa

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_3_03
Nama	Lihat Tagihan Mahasiswa
Tujuan	Agar staff akademik (*administrator*) dapat melihat tagihan kuliah mahasiswa
Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* melakukan proses untuk melihat tabel data tagihan pembayaran seta nim seluruh peserta pendaftar semester pendek
Aktor	Staff Akademik (*admin*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal admin
Aksi Aktor	Reaksi Sistem
1. *Admin* menekan tombol menu dropdown 'tagihan' lalu memilih menu 'tagihan mahasiswa' pada sistem	2. Sistem menampilkan halaman data informasi mengenai mahasiswa yang mendaftar semester pendek berserta seluruh jumlah tagihannya.
Kondisi Akhir	Halaman daftar tabel data tagihan pembayaran seluruh mahasiswa pada semester pendek untuk *admin* dapat ditampilkan

Tabel 4.21 *Use case* lihat saldo mahasiswa

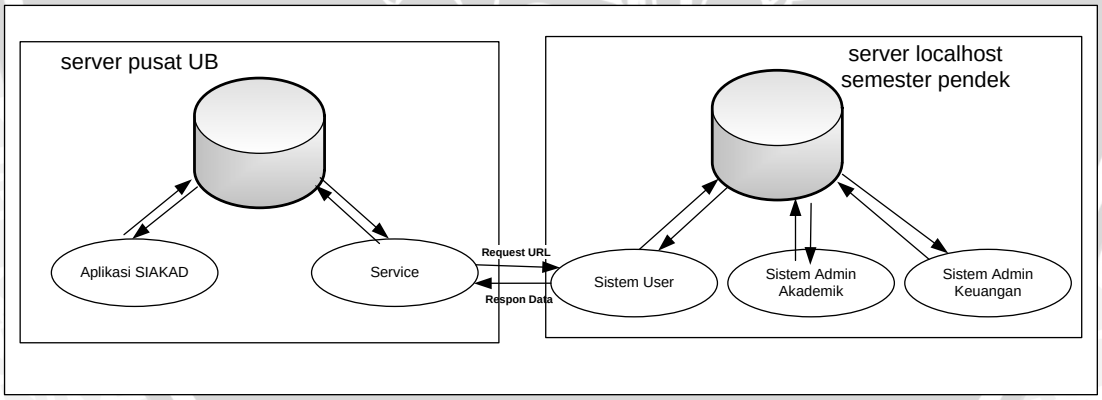
Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_3_04
Nama	Lihat Saldo Mahasiswa
Tujuan	Agar staff akademik (*administrator*) dapat melihat saldo kuliah mahasiswa
Deskripsi	*Use Case* ini menjelaskan bagaimana *admin* melakukan proses untuk melihat tabel data tagihan pengembalian seta nim seluruh peserta pendaftar semester pendek
Aktor	Staff Akademik (*admin*)
Skenario Utama	
Kondisi Awal	Sistem informasi telah berjalan dan menampilkan halaman awal admin
Aksi Aktor	Reaksi Sistem
1. *Admin* menekan tombol menu dropdown ‘tagihan’ lalu memilih menu ‘saldo mahasiswa’ pada sistem	2. Sistem menampilkan halaman data informasi mengenai mahasiswa yang mendaftar semester pendek berserta seluruh jumlah saldonya untuk segera dikembalikan.
Kondisi Akhir	Halaman daftar tabel data saldo pembayaran seluruh mahasiswa pada semester pendek untuk *admin* dapat ditampilkan

4.1.2 Perancangan Aplikasi

Perancangan aplikasi dilakukan dalam enam tahap, yaitu perancangan arsitektural, perancangan basis data, pemodelan *class diagram* untuk menggambarkan perancangan struktur *class-class* yang menyusun sistem informasi, perancangan *activity diagram*, perancangan *sequence diagram* untuk menggambarkan interaksi antar objek atau *class* di dalam aplikasi dan perancangan antarmuka pengguna dari aplikasi. Perancangan aplikasi pada skripsi ini menggunakan pendekatan desain berorientasi objek yang direpresentasikan dengan menggunakan UML (*Unified Modelling Language*).

4.1.2.1 Perancangan Arsitektural

Sistem informasi pengadaan semester pendek di PTIIK dibangun menggunakan teknologi REST *web service* dengan format JSON. Setiap data yang dibutuhkan pada semester pendek tersedia di aplikasi pusat Universitas Brawijaya yaitu SIAKAD UB. Dengan begitu penulis menggunakan *web service* untuk dapat mengambil data yang dibutuhkan dari SIAKAD UB. SOA yang digunakan telah disediakan oleh SIAKAD dengan format JSON dan teknologi RESTFUL, sehingga pada penelitian ini penulis hanya melakukan *request* data ke *web service* SIAKAD UB. Sedangkan pada sistem informasi semester pendek dibangun menggunakan *framework CodeIgniter* dan didampingi *mySQL* sebagai penyimpanan data proses semester pendek. Proses kerja sistem dimulai dari sistem informasi semester pendek berbasis *web* yang diminta oleh *user* dari *browser* ke *server localhost*, kemudian *server localhost* mengirimkan *response file* sistem untuk ditampilkan di *browser user*. Setelah itu, dari aplikasi *browser user*, seluruh *request* yang dilakukan ditujukan ke *service endpoint* baik di *server localhost* maupun *server* SIAKAD. Desain arsitektural hubungan sistem yang dibangun dengan di luar sistem yang dibangun ditunjukkan pada Gambar 4.6.

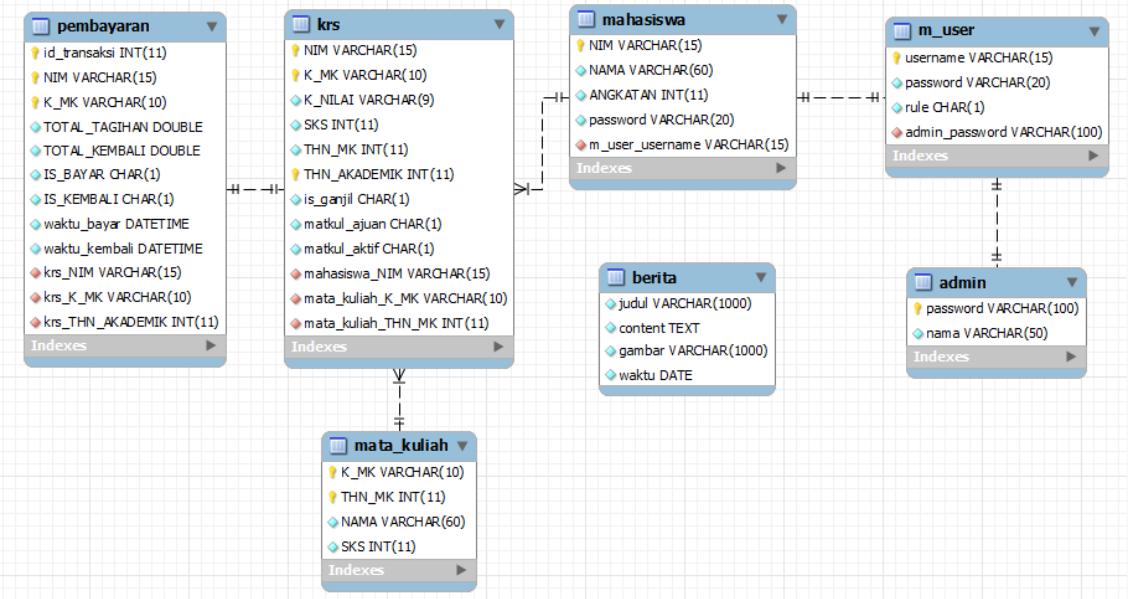


Gambar 4.6 Perancangan Arsitektural Sistem

4.1.2.2 Perancangan Basis Data

Basis data berfungsi sebagai tempat menyimpan data. Sistem ini selain menggunakan data hasil *request* melalui *web service*, juga membutuhkan basis data sebagai tempan menyimpan data yang diperlukan dalam proses semester pendek.

Perancangan basis data digunakan unutk merancang basis data yang akan dibuat agar masukan dan keluaran sistem sesuai dengan apa yang diharapkan. Perancangan basis data mengambil acuan dari proses analisis data yang dilakukan pada tahap analisis kebutuhan. Perancangan basis data secara relasi tabel dapat dilihat pada Gambar 4.7.



Gambar 4.7 Diagram *Entity Relationship* Sistem

Rancangan masing – masing detail tabel dalam model e*ntity relationship diagram* adalah sebagai berikut :

1. **Tabel Mahasiswa**

   Nama Tabel : MAHASISWA

   Jumlah *field* : 4

   Fungsi : Untuk menyimpan data mahasiswa

Tabel 4.22 Struktur tabel mahasiswa

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	NIM	Varchar	15	NIM Mahasiswa
2	NAMA	Varchar	60	Nama Mahasiswa
3	ANGKATAN	Integer	-	Angkatan Mahasiswa
4	PASSWORD	Varchar	20	Password Mahasiswa

2. **Tabel Administrator**

Nama Tabel : ADMIN

Jumlah *field* : 2

Fungsi : Untuk menyimpan data administrator

Tabel 4.23 Struktur tabel administrator

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	PASSWORD	Varchar	100	Password Administrator
2	NAMA	Varchar	50	Username Administrator

3. **Tabel Mata Kuliah**

Nama Tabel : MATA_KULIAH

Jumlah *field* : 4

Fungsi : Untuk menyimpan data Mata Kuliah

Tabel 4.24 Struktur tabel mata kuliah

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	K_MK	Varchar	10	Kode Mata Kuliah Mahasiswa
2	THN_MK	Integer	-	Tahun Mata Kuliah
3	NAMA	Varchar	60	Nama Mata Kuliah
4	SKS	Integer	-	Jumlah SKS Mata Kuliah

4. **Tabel Pembayaran**

Nama Tabel : Pembayaran

Jumlah *field* : 7

Fungsi : Untuk menyimpan data tagihan semester pendek mahasiswa PTIIK

Tabel 4.25 Struktur tabel pembayaran

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	Id_transaksi	Integer	-	Nomor ID transaksi
2	NIM	Varchar	15	NIM Mahasiswa
3	K_MK	Varchar	10	Kode Mata Kuliah

				Mahasiswa
3	TOTAL_TAGIHAN	Double	-	Jumlah Tagihan Pembayaran Mahasiswa
4	TOTAL_KEMBALI	Double	-	Jumlah Tagihan Pengembalian Mahasiswa
5	IS_BAYAR	Char	1	Status Pembayaran
6	IS_KEMBALI	Char	1	Status Pengembalian
7	Waktu_bayar	DATETIME	-	Waktu Transaksi pembayaran
8	Waktu_kembali	DATETIME	-	Waktu Transaksi pengembalian

5. Tabel KRS

Nama Tabel : KRS

Jumlah *field* : 8

Fungsi : Untuk menyimpan data KRS semester pendek mahasiswa

Tabel 4.26 Struktur tabel KRS

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	NIM	Varchar	15	NIM Mahasiswa
2	K_MK	Varchar	10	Kode Mata Kuliah Mahasiswa
3	K_NILAI	Varchar	9	Kode nilai mahasiwa
4	SKS	Integer	1	Jumlah SKS
5	THN_MK	Integer	11	Tahun Mata Kuliah
6	THN_AKADEMIK	Integer	11	Tahun Akademik
7	IS_GANJIL	Char	1	Status semester pendek saat semester ganjil
8	MATKUL_AJUAN	Char	1	Status Pengambilan Mata Kuliah Ajuan
9	MATKUL_AKTIF	Char	1	Status Mata Kuliah Aktif

6. Tabel Berita atau Pengumuman

Nama Tabel : Berita

Jumlah *field* : 4

Fungsi : Untuk menyimpan data informasi atau pengumuman yang berkaitan dengan semester pendek mahasiswa

Tabel 4.27 Struktur tabel berita

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	Judul	Varchar	1000	Judul Berita
2	Content	Text	-	Isi Berita atau Informasi
3	Gambar	Varchar	1000	Tempat Gambar
4	Waktu	Date	-	Waktu Pengisian Berita atau Informasi

7. Tabel Master User

Nama Tabel : m_user

Jumlah *field* : 3

Fungsi : Untuk menyimpan data informasi mengenai pengguna sistem, yaitu *user* (mahasiswa) serta *administrator* (akademik dan keuangan)

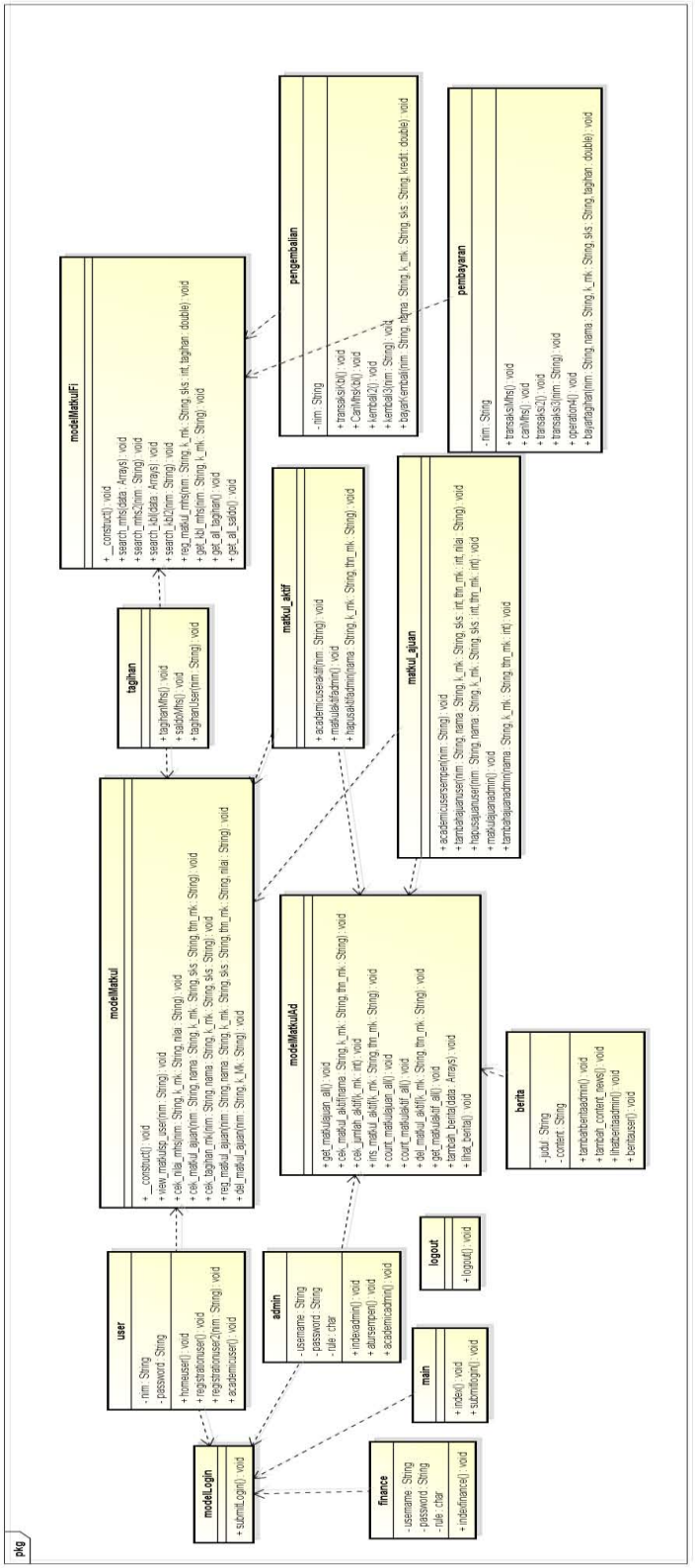
Tabel 4.28 Struktur tabel master user

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	Username	Varchar	15	Username Pengguna Sistem
2	Password	Varchar	20	Password Pengguna Sistem
3	Rule	Char	1	Status Pengguna

4.1.2.3 Perancangan *Class Diagram*

Diagram *class* memberikan gambaran pemodelan elemen-elemen *class* yang membentuk sebuah sistem. *Class* bisa didapatkan dengan menganalisis secara detail terhadap *use case* yang dimodelkan.

Pada perancangan ini, *class diagram* yang dibuat sama dengan class diagram yang digunakan oleh seluruh aktor yang bersangkutan dan dirancang berdasarkan struktur CodeIgniter yaitu terdapat *class-class controllers* yang sama-sama saling menggunakan *class models* yang dapat digambarkan pada Gambar 4.8.



Gambar 4.8 Diagram *Class* Sistem

Pada gambar 4.8 menjelaskan mengenai *class diagram* sistem semester pendek. Dalam perancangannya, sistem ini menggunakan *class-class* yang saling berhubungan dengan *class* model atau fungsi *query*. Sebagai contoh, pada gambar terdapat *class* user, matkul_ajuan, matkul_aktif yang saling berhubungan dengan relasi "*use*" ke *class* model nya yaitu modelMatkul.

4.1.2.4 Perancangan *Activity Diagram*

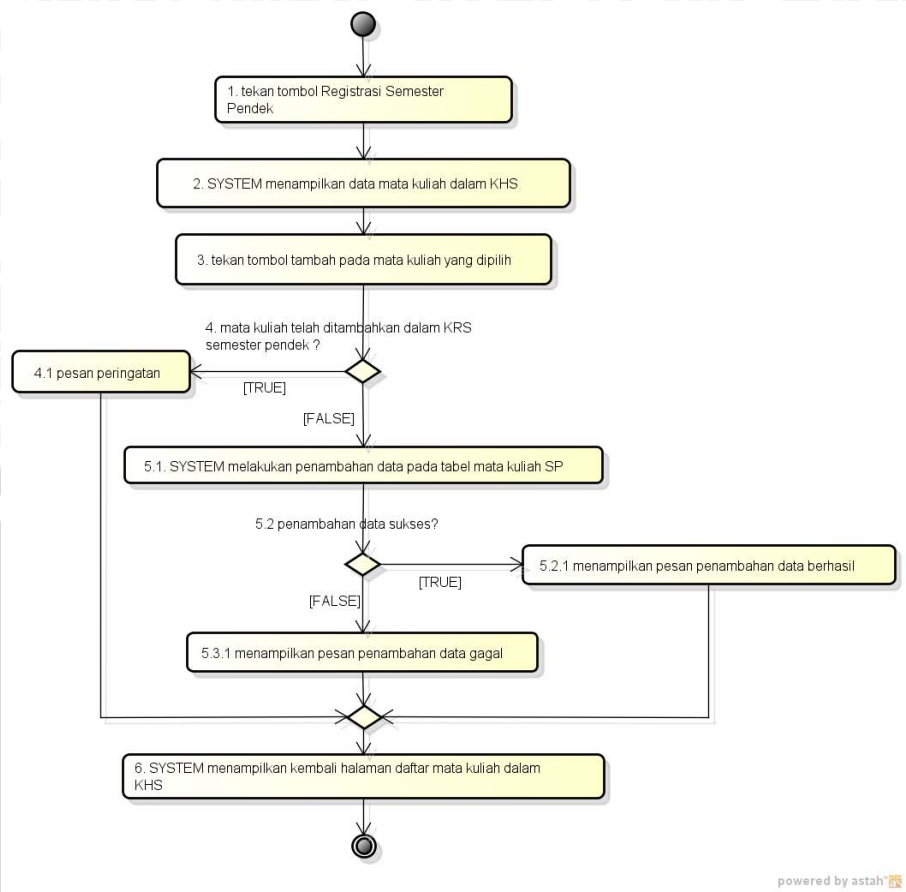
Diagram activity menggambarkan pemodelan proses sebuah interaksi antara *user* dengan sistem.

1. ***Activity Diagram* Sistem *User***

Activity diagram sistem *user* menggambarkan proses interaksi *user* dengan sistem informasi semester pendek *user*. Berikut akan dijelaskan beberapa contoh *activity diagram* utama yang terdapat pada sistem *user*.

a. *Activity Diagram* Tambah Mata Kuliah Semester Pendek

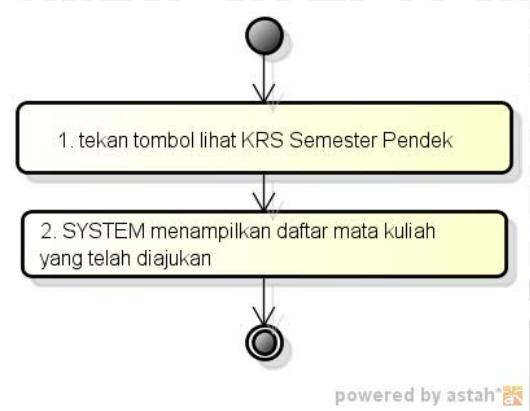
Activity diagram tambah mata kuliah adalah *event* ketika *user* ingin menambah mata kuliah yang akan di daftarkan pada sistem informasi semester pendek. Perancangan *activity diagram* tambah mata kuliah semester pendek dapat dilihat pada Gambar 4.9.



Gambar 4.9 *Activity Diagram* Tambah Mata Kuliah

b. *Activity Diagram* Lihat Mata Kuliah Semester Pendek

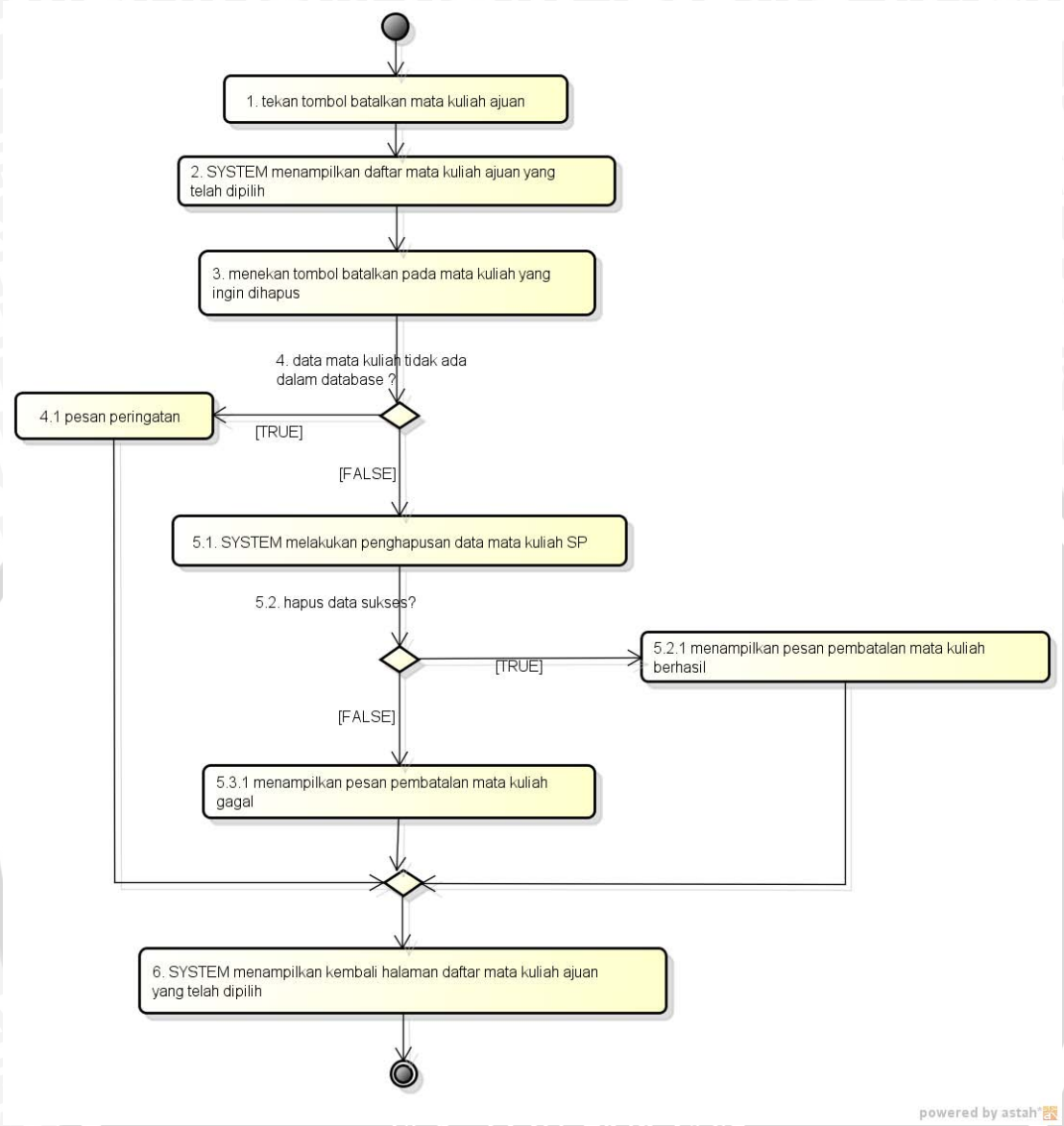
Activity diagram lihat mata kuliah adalah *event* ketika *user* ingin melihat daftar mata kuliah yang telah di daftarkan pada sistem informasi semester pendek. Perancangan *activity diagram* lihat mata kuliah semester pendek dapat dilihat pada Gambar 4.10.



Gambar 4.10 *Activity Diagram* Lihat Mata Kuliah

c. *Activity Diagram* Batalkan Mata Kuliah Semester Pendek

Activity diagram batalkan mata kuliah adalah *event* ketika *user* ingin membatalkan daftar mata kuliah yang telah di diajukan pada sistem informasi semester pendek. Perancangan *activity diagram* batalkan mata kuliah semester pendek dapat dilihat pada Gambar 4.11.



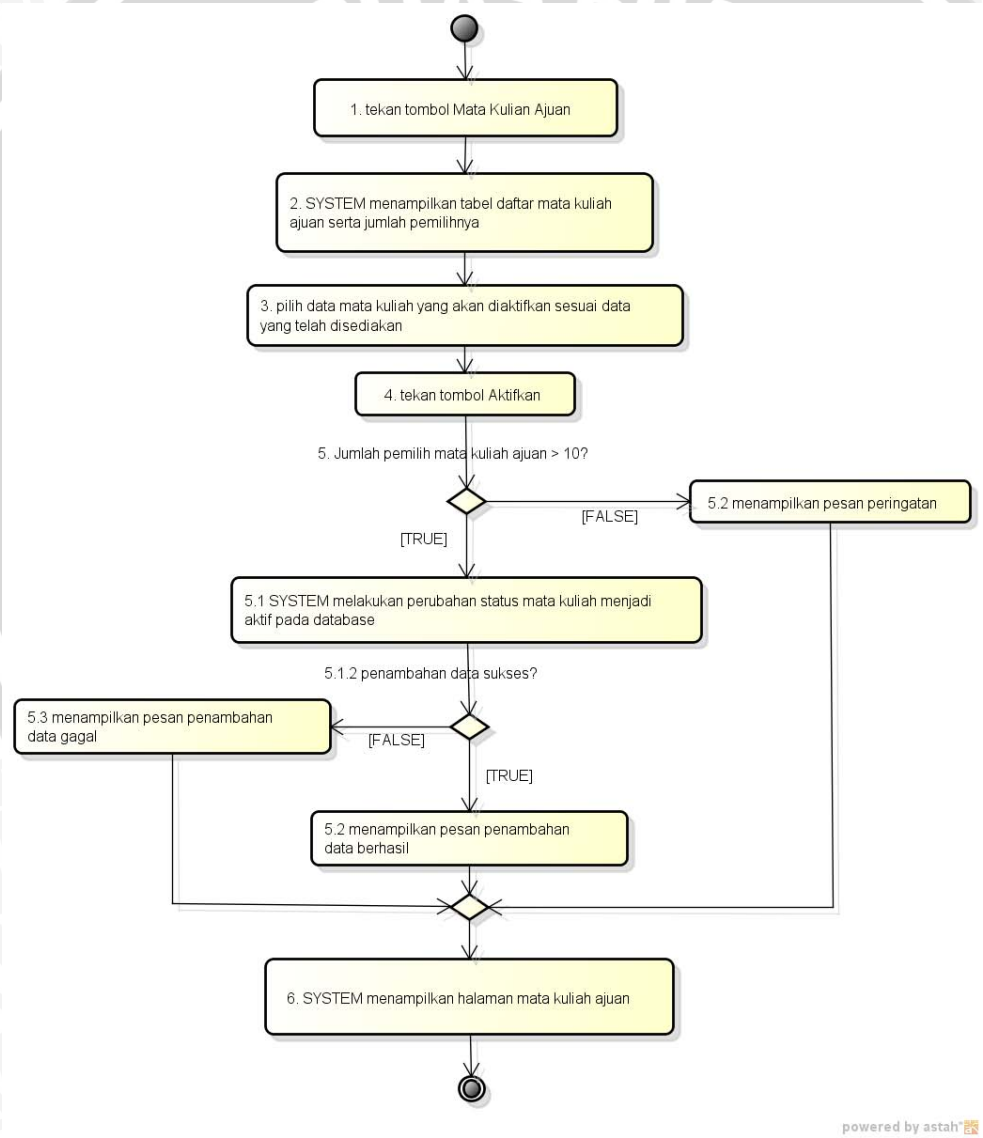
Gambar 4.11 *Activity Diagram* Batalkan Mata Kuliah Semester Pendek

2. *Activity Diagram* Sistem *Administrator* (akademik)

Activity diagram sistem *admin* akademik menggambarkan proses interaksi *admin* dengan sistem informasi semester pendek. Berikut akan dijelaskan beberapa contoh *activity diagram* utama yang terdapat pada sistem *admin* akademik.

a. *Activity Diagram* Tambah Mata Kuliah Aktif Semester Pendek

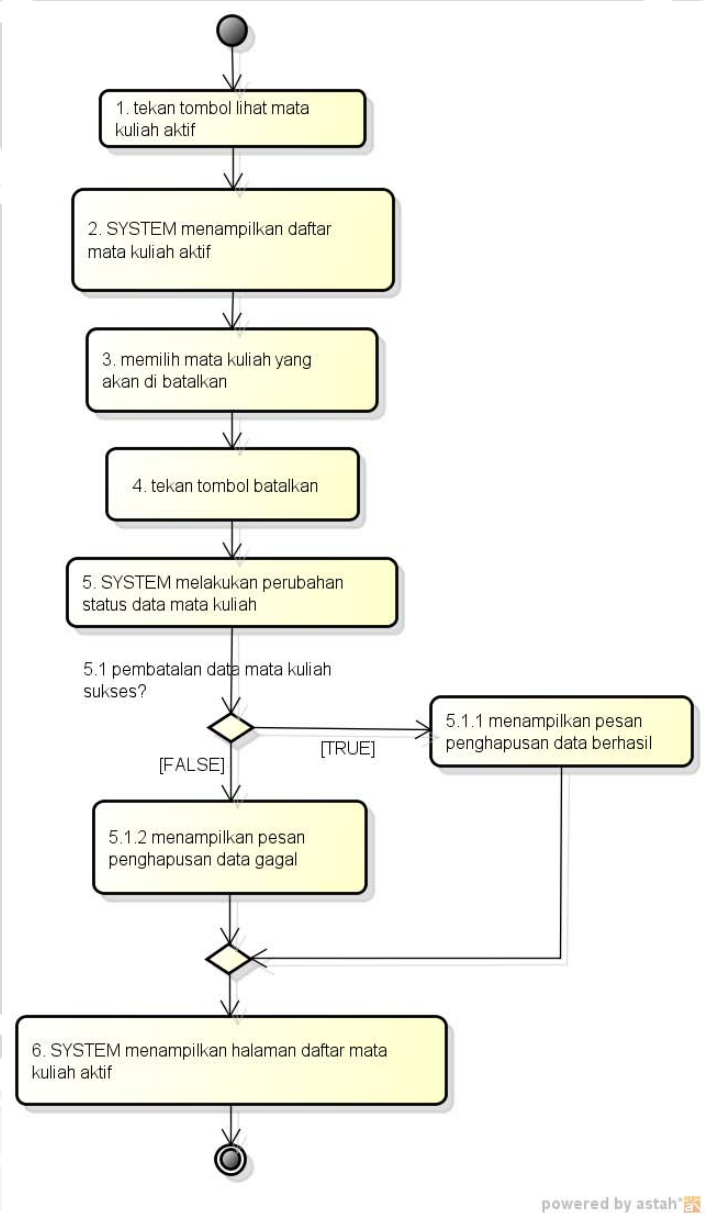
Activity diagram tambah mata kuliah aktif adalah *event* ketika *admin* ingin membuka dan mengubah status mata kuliah ajuan menjadi aktif sehingga dapat dipilih oleh mahasiswa. Mata kuliah yang telah diubah menjadi aktif akan disimpan dalam database dan dibuka selama semester pendek berlangsung. Perancangan *activity diagram* tambah mata kuliah aktif semester pendek dapat dilihat pada Gambar 4.12.



Gambar 4.12 *Activity Diagram* Tambah Mata Kuliah Aktif

b. *Activity Diagram* Batal Mata Kuliah Aktif Semester Pendek

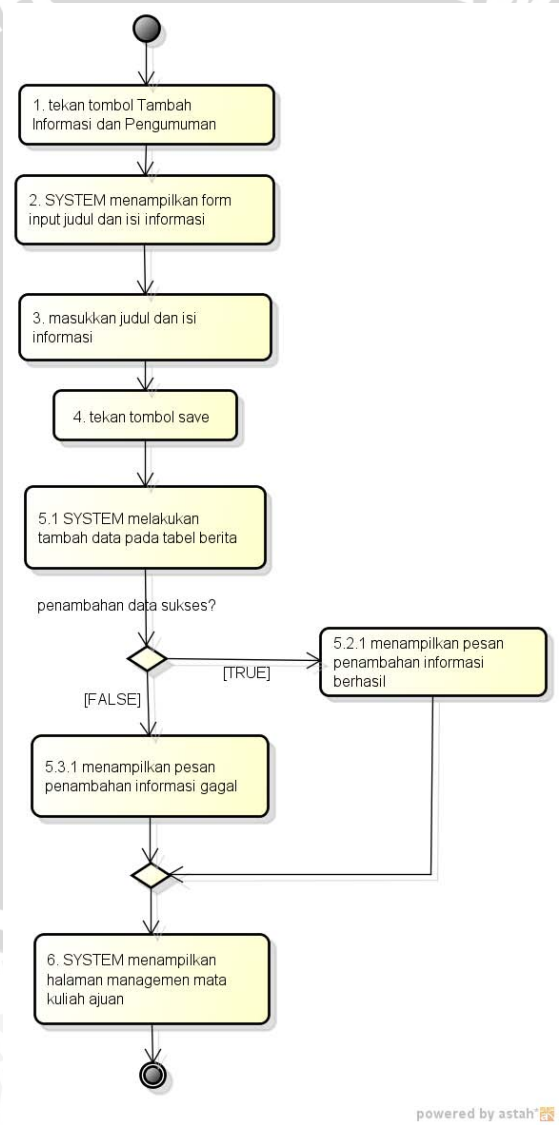
Activity diagram batal mata kuliah aktif adalah *event* ketika *admin* ingin membatalkan daftar mata kuliah yang telah dibuka atau berstatus aktif pada sistem informasi semester pendek. Perancangan *activity diagram* batal mata kuliah aktif semester pendek dapat dilihat pada Gambar 4.13.



Gambar 4.13 *Activity Diagram* Batal Mata Kuliah Aktif

c. *Activity Diagram* Tambah Informasi atau Pengumuman

Activity diagram tambah informasi adalah *event* ketika *admin* ingin menambah berbagai berita, informasi atau pengumuman yang berkaitan dengan semester pendek sehingga mahasiswa memperoleh mengenai pengadaan dan alur proses semester pendek. Isi informasi yang ditambahkan akan disimpan dalam database dan dapat dilihat oleh *user* (mahasiswa) selama semester pendek berlangsung. Perancangan *activity diagram* tambah informasi atau pengumuman semester pendek dapat dilihat pada Gambar 4.14.



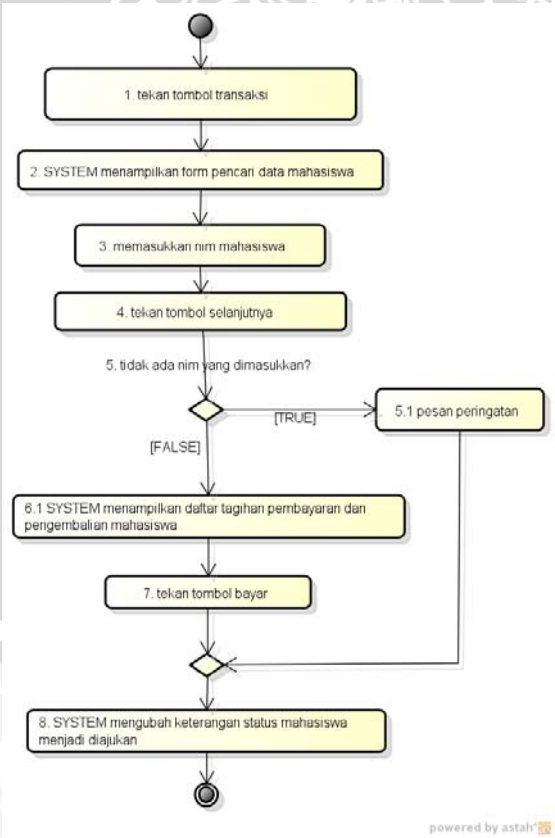
Gambar 4.14 *Activity Diagram* Tambah Informasi

3. *Activity Diagram* Sistem *Administrator* (keuangan)

Activity diagram sistem *admin* keuangan menggambarkan proses interaksi *admin* keuangan dengan sistem informasi semester pendek. Berikut akan dijelaskan beberapa contoh *activity diagram* utama yang terdapat pada sistem *admin* keuangan akademik.

a. *Activity Diagram* Tambah Transaksi Semester Pendek

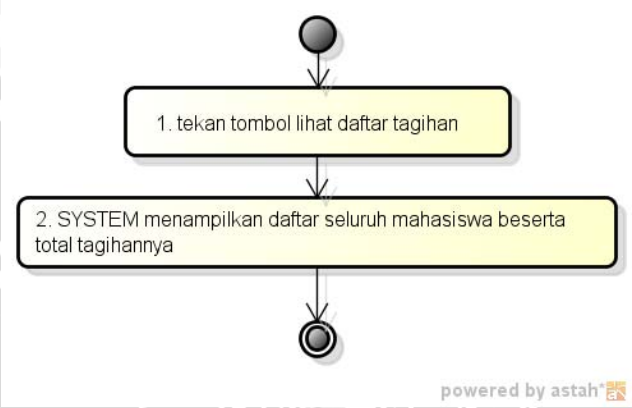
Activity diagram tambah transaksi adalah *event* ketika *admin* ingin menambah transaksi yang berkaitan dengan semester pendek baik transaksi pembayaran maupun transaksi poengembalian. Transaksi yang ditambahkan akan disimpan dalam database dan dapat dilihat oleh *user* (mahasiswa) selama semester pendek berlangsung. Transaksi pembayaran juga digunakan untuk membuat status mata kuliah yang dipilih mahasiswa berstatus diajukan. Perancangan *activity diagram* tambah transaksi pembayaran semester pendek dapat dilihat pada Gambar 4.15.



Gambar 4.15 *Activity Diagram* Tambah Transaksi

b. *Activity Diagram* Lihat Tagihan Semester Pendek

Activity diagram lihat tagihan adalah *event* ketika *admin* ingin melihat tagihan seluruh mahasiswa yang telah mendaftar mata kuliah semester pendek. Perancangan *activity diagram* lihat tagihan semester pendek dapat dilihat pada Gambar 4.16.



Gambar 4.16 *Activity Diagram* Lihat Tagihan

4.1.2.5 Perancangan *Sequence Diagram*

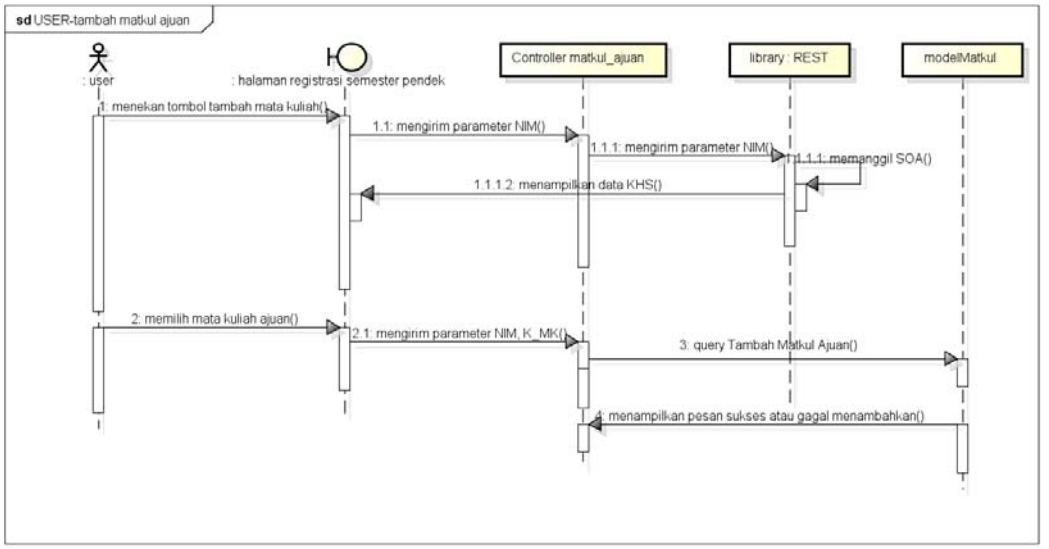
Diagram *sequence* menunjukkan pemodelan aliran jalannya proses interaksi antar objek atau *class* yang disusun berdasarkan urutan waktu. *Sequence diagram* digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai *respons* dari sebuah *event* untuk menghasilkan *output* tertentu.

1. *Sequence Diagram* Sistem *User*

Sequence Diagram aplikasi *user* menggambarkan interaksi *class* dan objek yang terjadi ketika *user* melakukan suatu *event* terhadap sistem informasi semester pendek.

a. *Sequence Diagram* Tambah Mata Kuliah Ajuan

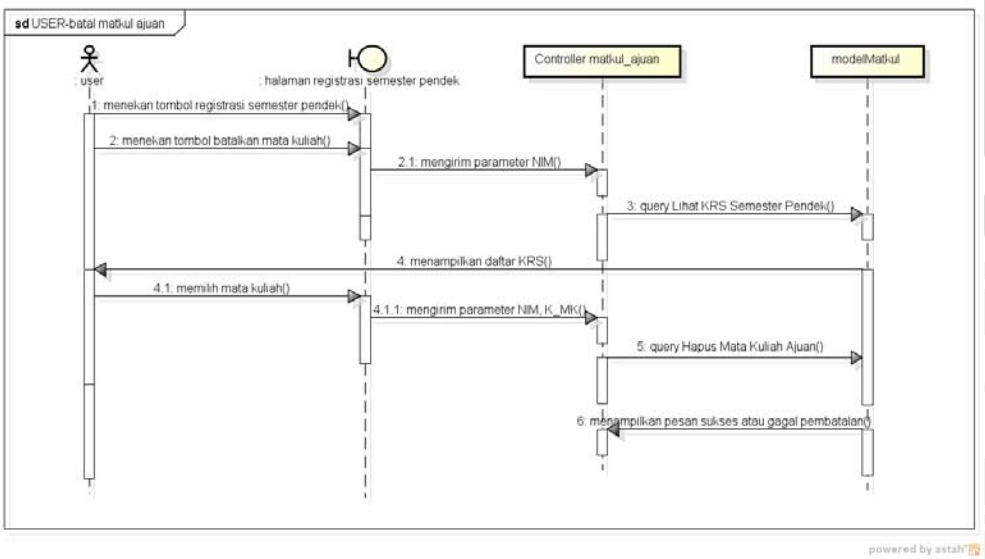
Sequence diagram tambah mata kuliah ajuan adalah event ketika *user* atau mahasiswa ingin melakukan pendaftaran semester pendek, langkah awal yang harus dilakukan adalah melihat daftar seluruh KHS melalui *web service* dan memilih mata kuliah yang akan diajukan. Perancangan *sequence diagram* tambah mata kuliah ajuan dapat dilihat pada Gambar 4.17.



Gambar 4.17 Sequence Diagram Tambah Mata Kuliah Ajuan

b. *Sequence Diagram* Batalkan Mata Kuliah Ajuan

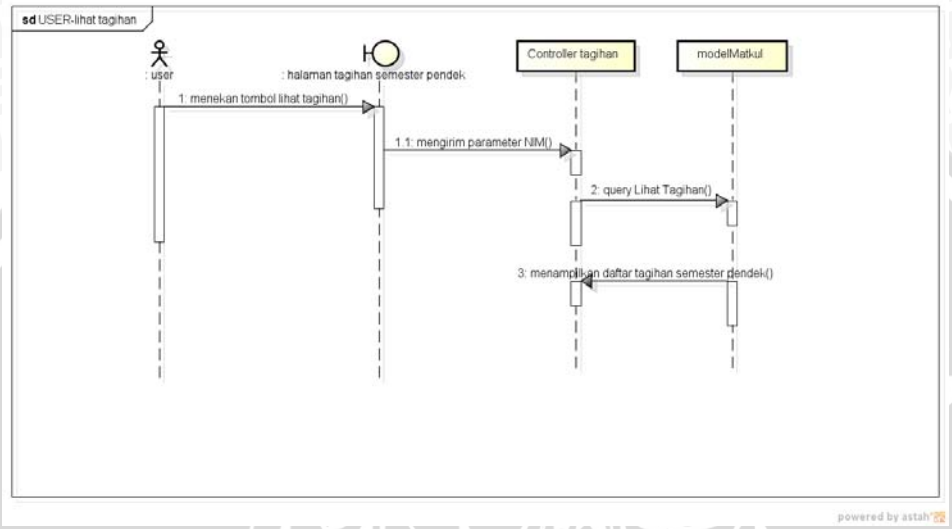
Sequence diagram batalkan mata kuliah ajuan adalah event ketika *user* atau mahasiswa ingin melakukan pembatalan mata kuliah yang diajukan pada semester pendek, langkah awal yang harus dilakukan adalah melihat daftar seluruh KRS semester pendek dan memilih mata kuliah yang akan dibatalkan. Perancangan *sequence diagram* batalkan mata kuliah ajuan semester pendek dapat dilihat pada Gambar 4.18.



Gambar 4.18 *Sequence Diagram* Batalkan Mata Kuliah Ajuan

c. *Sequence Diagram* Lihat Tagihan

Sequence diagram lihat tagihan adalah event ketika *user* atau mahasiswa ingin menampilkan seluruh tagihan pembayaran pada semester pendek sesuai dengan ketentuan yang diberikan sebelumnya. Perancangan *sequence diagram* lihat tagihan semester pendek dapat dilihat pada Gambar 4.19.



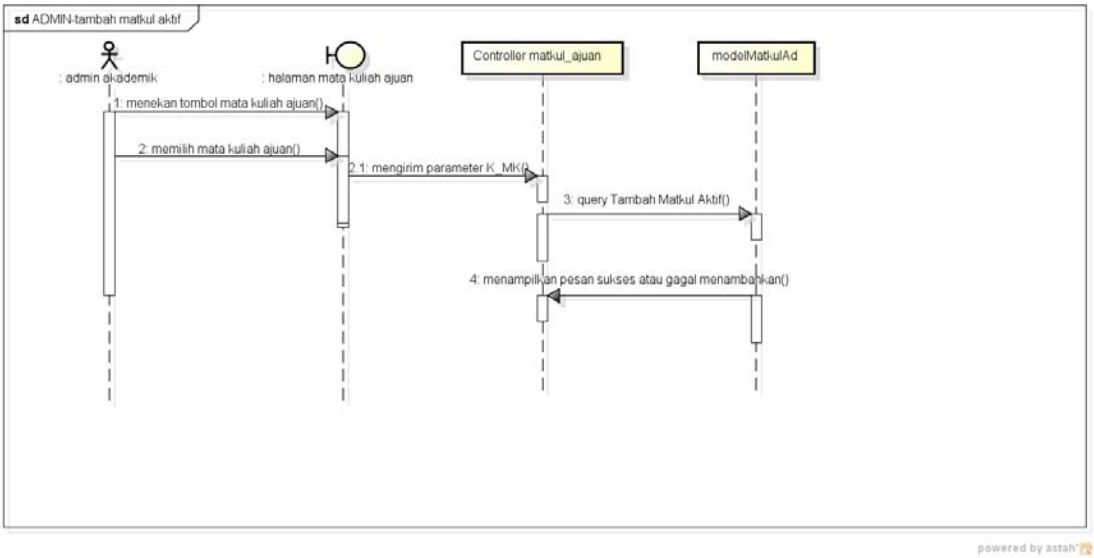
Gambar 4.19 *Sequence Diagram* Lihat Tagihan

2. *Sequence Diagram* Sistem *Administrator* Akademik

Sequence diagram sistem administrator akademik menggambarkan interaksi *class* dan objek yang terjadi ketika administrator melakukan suatu *event* terhadap sistem administrator. Berikut ini akan dijelaskan beberapa contoh *sequence diagram* utama yang terdapat pada sistem administrator.

a. *Sequence Diagram* Tambah Mata Kuliah Aktif

Sequence diagram tambah mata kuliah aktif adalah event ketika *administrator* atau akademik ingin menampilkan seluruh daftar mata kuliah ajuan beserta jumlah peminatnya dan akan menambah daftar mata kuliah ajuan menjadi status aktif atau dibuka pada semester pendek. Perancangan *sequence diagram* tambah mata kuliah aktif dapat dilihat pada Gambar 4.20.



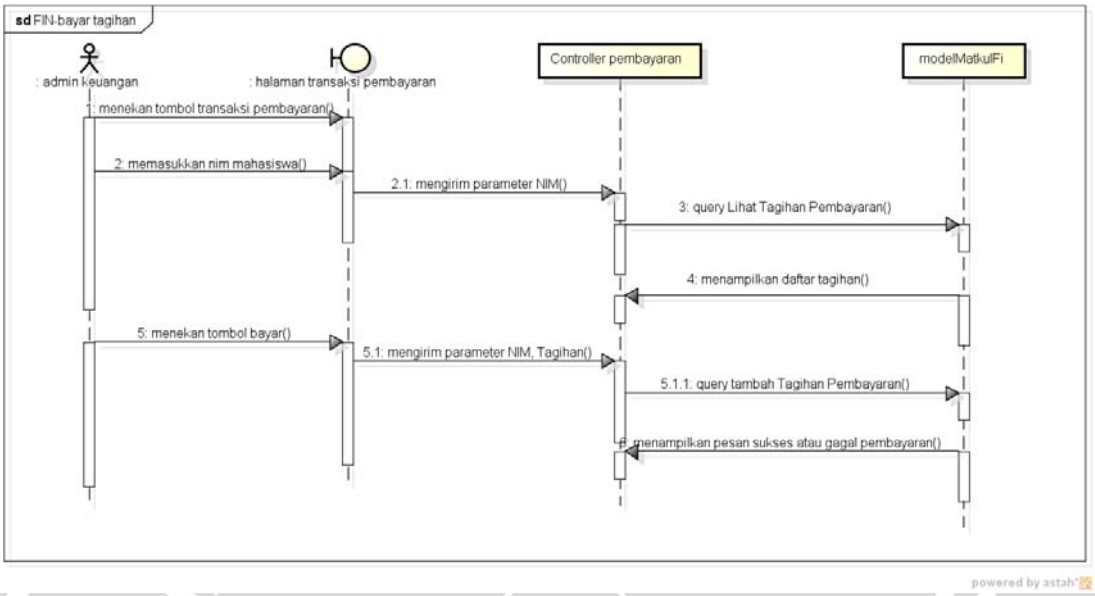
Gambar 4.20 *Sequence Diagram* Tambah Matkul Aktif

3. *Sequence Diagram* Sistem *Administrator* Keuangan

Sequence diagram sistem administrator keuangan menggambarkan interaksi *class* dan objek yang terjadi ketika administrator melakukan suatu *event* terhadap sistem keuangan. Berikut ini akan dijelaskan beberapa contoh *sequence diagram* utama yang terdapat pada sistem administrator keuangan.

a. *Sequence Diagram* Bayar Tagihan Pembayaran

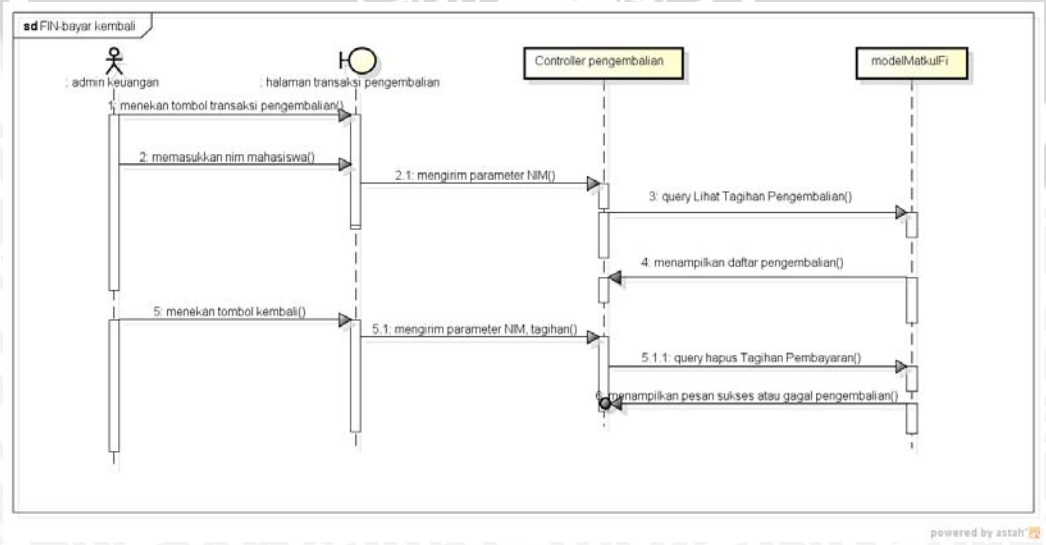
Sequence diagram bayar tagihan pembayaran adalah *event* ketika *administrator* atau keuangan ingin menampilkan seluruh daftar tagihan salah satu mahasiswa dan mengubah status tagihannya menjadi lunas. Perancangan *sequence diagram* bayar tagihan pembayaran dapat dilihat pada Gambar 4.21.



Gambar 4.21 *Sequence Diagram* Bayar Tagihan Pembayaran

b. *Sequence Diagram* Bayar Tagihan Pengembalian

Sequence diagram bayar tagihan pengembalian adalah *event* ketika *administrator* atau keuangan ingin menampilkan seluruh daftar tagihan pengembalian salah satu mahasiswa dan mengubah status pengembaliannya menjadi lunas. Perancangan *sequence diagram* bayar tagihan pengembalian dapat dilihat pada Gambar 4.22.



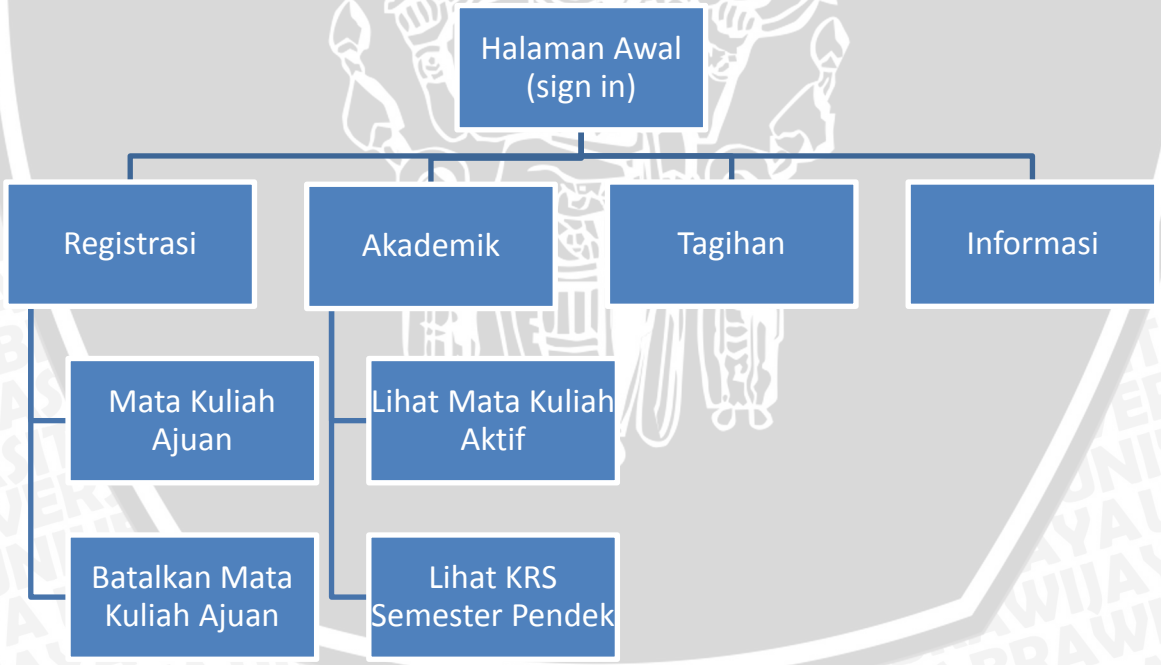
Gambar 4.22 *Sequence Diagram* Bayar Tagihan Pengembalian

4.1.2.6 Perancangan Antarmuka

Pada bagian ini akan dijelaskan tentang perancangan antarmuka sistem informasi semester pendek PTIIK. Antarmuka ini akan digunakan oleh pengguna untuk berinteraksi dengan sistem informasi semester pendek PTIIK. Antarmuka sistem ini dibagi menjadi tiga, yaitu antarmuka untuk sistem *user*, antarmuka untuk sistem *administrator* akademik dan antarmuka untuk sistem *administrator* keuangan.

1. **Perancangan Antarmuka Sistem Informasi *User* (Mahasiswa)**

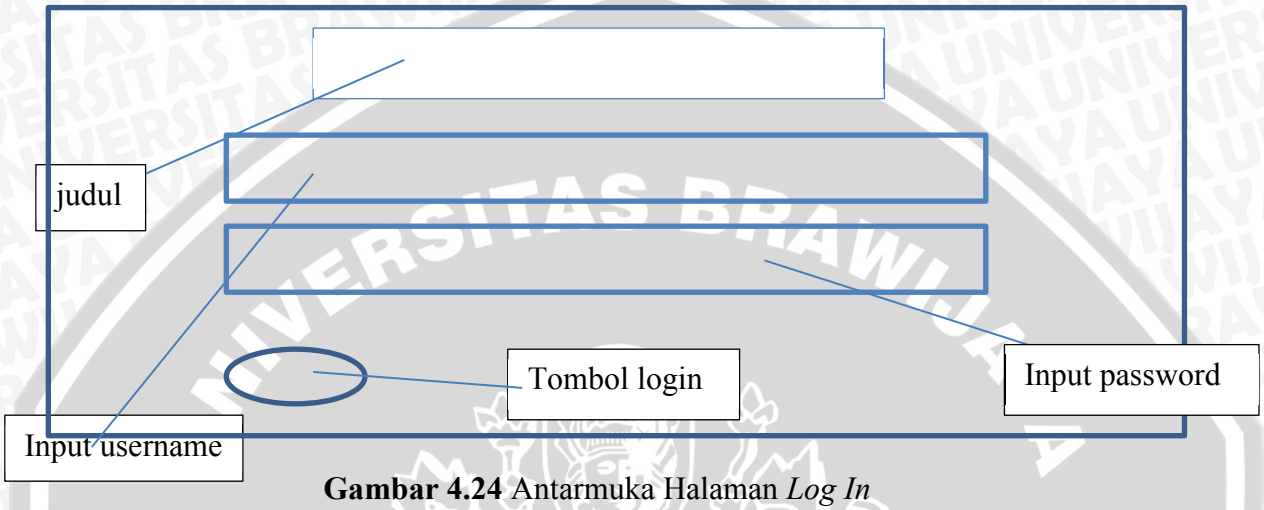
Antarmuka pengguna untuk sistem *user* berupa halaman web yang menggunakan *framework* PHP CodeIgniter, sehingga akan lebih optimal dalam menampilkan content yang ada berdasarkan keefektifannya dalam mengakses proses. Sitemap sistem *user* dari sistem informasi semester pendek PTIIK ditunjukkan oleh gambar 4.23.



Gambar 4.23 *Sitemap* Antarmuka Sistem *User*

a. Halaman Sign In

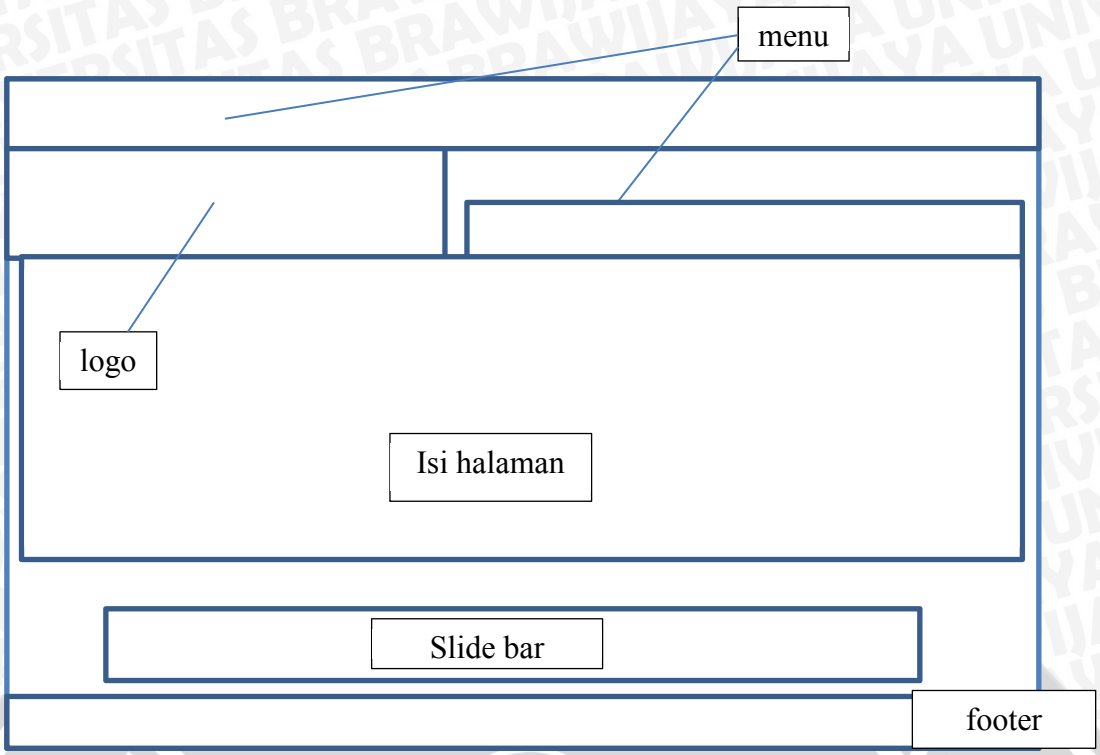
Halaman *log in* merupakan salah satu antarmuka pengguna untuk sistem user yang berfungsi untuk mempermudah user dalam melakukan *log in*. Perancangan halaman *log in* dapat dilihat pada Gambar 4.24.



Gambar 4.24 Antarmuka Halaman *Log In*

b. Halaman Registrasi

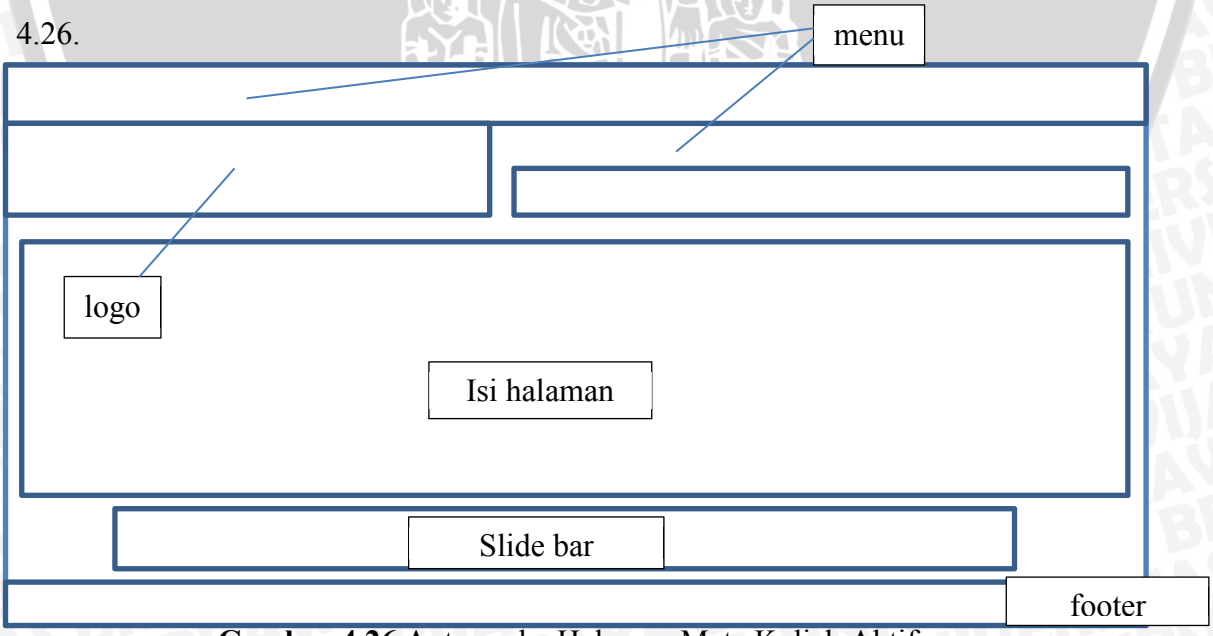
Halaman ini merupakan salah satu antarmuka pengguna untuk sistem user yang berfungsi untuk mempermudah user dalam melakukan registrasi mata kuliah menjadi ajuan untuk didaftarkan pada semester pendek. Selain itu pada halaman ini juga akan dibuat tab untuk proses pembatalan mata kuliah yang telah diajukan sebelumnya. Perancangan halaman registrasi dapat dilihat pada Gambar 4.25.



Gambar 4.25 Antarmuka Halaman Registrasi

c. Halaman Mata Kuliah Aktif

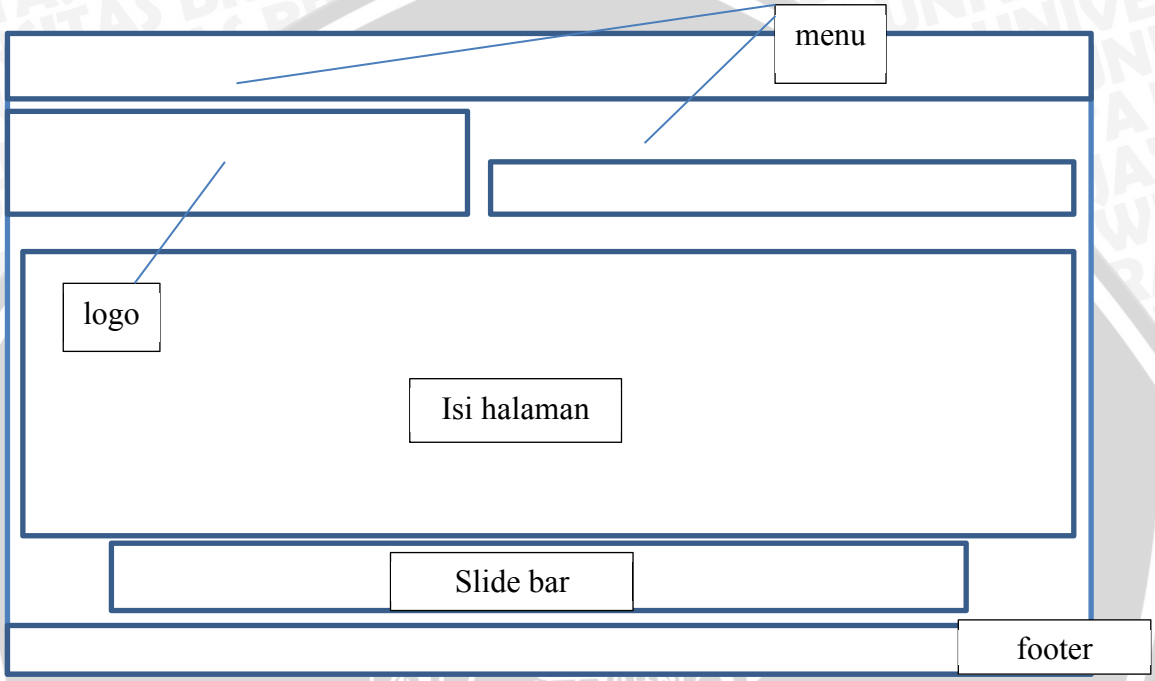
Halaman Mata Kuliah Aktif semester pendek merupakan salah satu antarmuka pengguna untuk sistem user yang berfungsi untuk mempermudah user dalam mengetahui mata kuliah apa saja yang telah aktif (telah dibuka) oleh administrator. Perancangan halaman mata kuliah aktif dapat dilihat pada Gambar 4.26.



Gambar 4.26 Antarmuka Halaman Mata Kuliah Aktif

d. Halaman Lihat KRS Semester Pendek

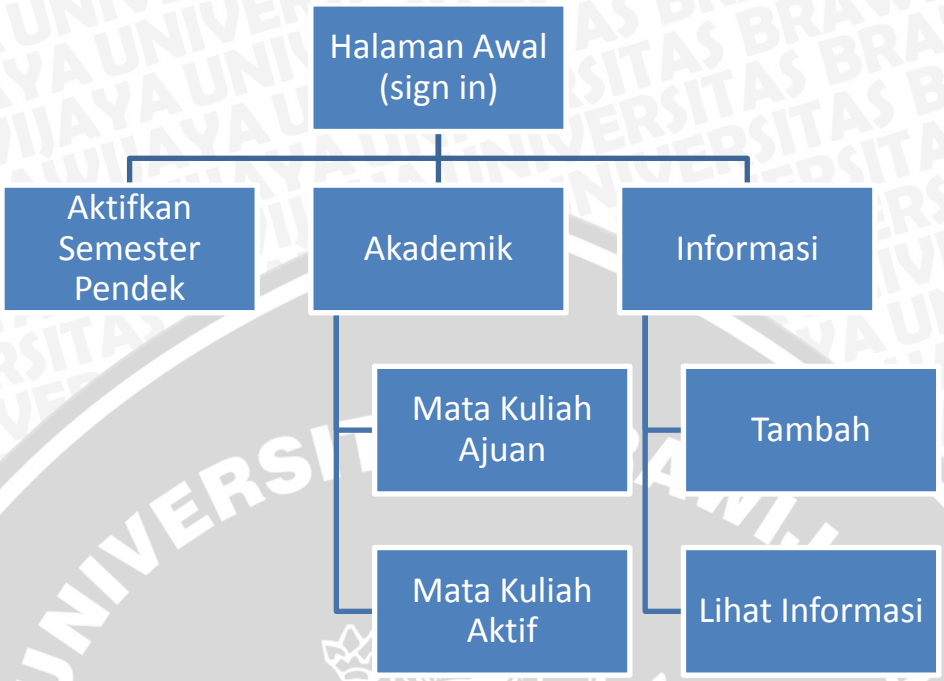
Halaman Lihat KRS Semester Pendek merupakan salah satu antarmuka pengguna untuk sistem user yang berfungsi untuk mempermudah user dalam mengetahui mata kuliah apa saja yang didaftarkan di semester pendek. Perancangan halaman lihat KRS semester pendek dapat dilihat pada Gambar 4.27.



Gambar 4.27 Antarmuka Halaman Lihat KRS

2. Perancangan Antarmuka Sistem Informasi *Administrator* Akademik

Antarmuka pengguna untuk sistem *administrator* berupa halaman web yang menggunakan *framework* PHP CodeIgniter serta desain menggunakan *template* bootstrap, sehingga akan lebih optimal dalam menampilkan content yang ada berdasarkan keefektifannya dalam mengakses proses. Sitemap sistem *administrator* dari sistem informasi semester pendek PTIIK ditunjukkan oleh Gambar 4.28.



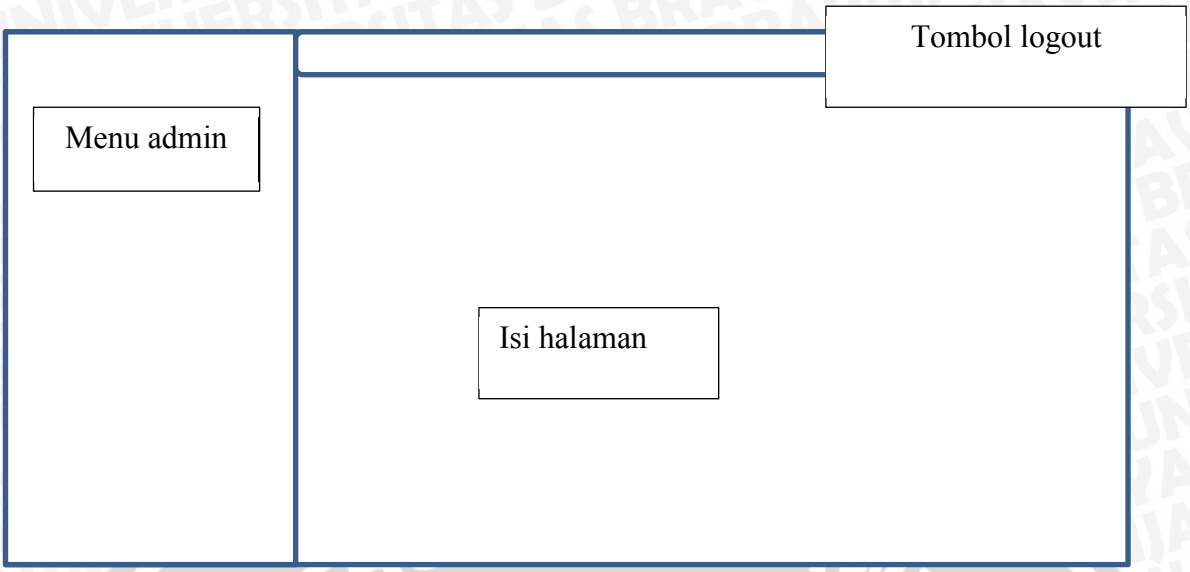
Gambar 4.28 *Sitemap* Antarmuka Pengguna Sistem Administrator

a. Halaman Mata Kuliah Ajuan

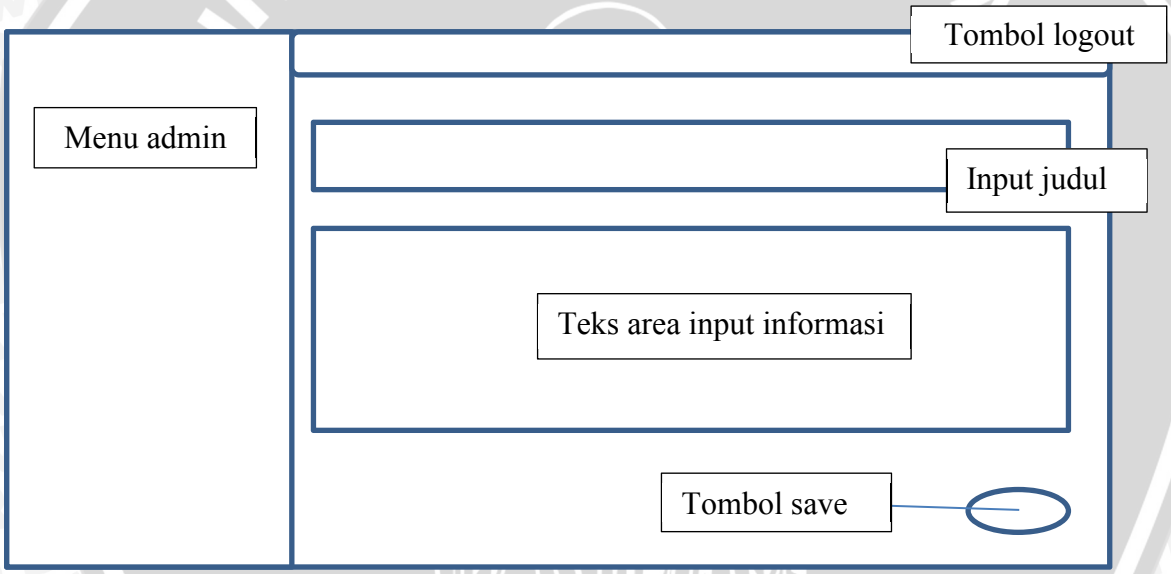
Halaman Mata Kuliah Ajuan semester pendek merupakan salah satu antarmuka pengguna untuk sistem *admin* yang berfungsi untuk mempermudah *admin* dalam melihat daftar mata kuliah ajuan yang telah dipilih oleh beserta jumlah pemilihnya. Pada halaman ini *admin* dapat melakukan perngubahan status suatu mata kuliah agar berstatus aktif, yang berati mata kuliha tersebut dibuka pada semester pendek. Perancangan halaman mata kuliah ajuan dapat dilihat pada Gambar 4.29.

b. Halaman Tambah Informasi

Halaman Tambah Informasi merupakan salah satu antarmuka pengguna untuk sistem *admin* yang berfungsi untuk mempermudah *admin* dalam memberikan informasi dan pengumuman mengenai semester pendek kepada mahasiswa. Perancangan halaman tambah informasi dapat dilihat pada Gambar 4.30.



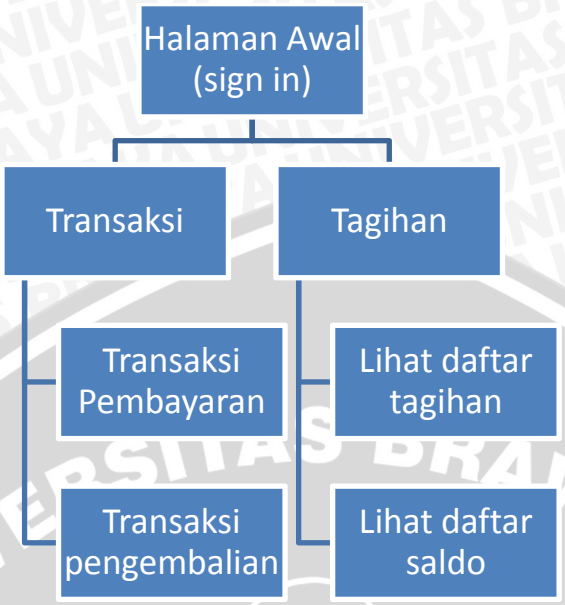
Gambar 4.29 Antarmuka Halaman Mata Kuliah Ajuan *Administrator*



Gambar 4.30 Antarmuka Halaman Tambah Informasi

3. Perancangan Antarmuka Sistem Informasi *Administrator* Keuangan

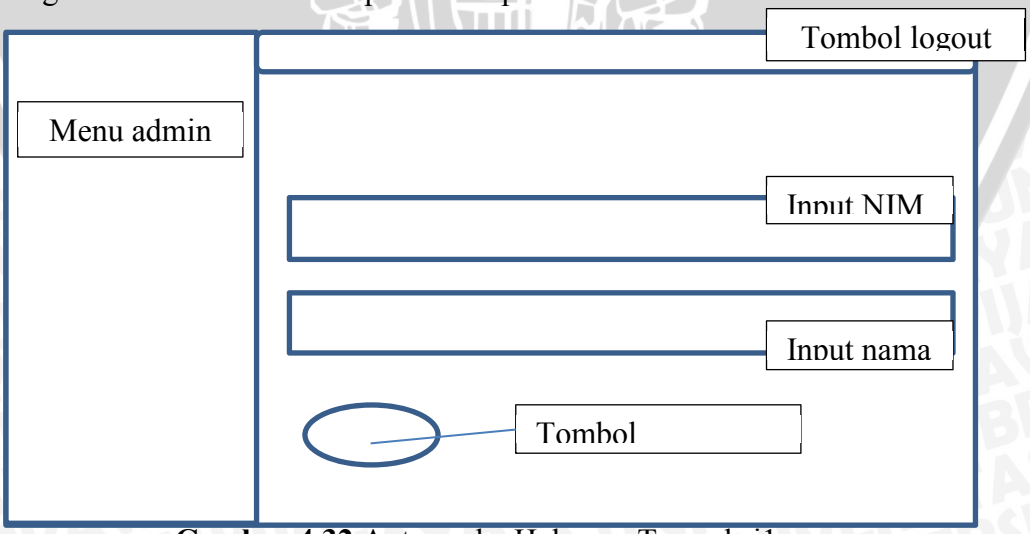
Antarmuka pengguna untuk sistem *administrator* berupa halaman web yang menggunakan *framework* PHP CodeIgniter serta desain menggunakan *template* bootstrap, sehingga akan lebih optimal dalam menampilkan content yang ada berdasarkan keefektifannya dalam mengakses proses. Sitemap sistem *administrator* dari sistem informasi semester pendek PTIIK ditunjukkan oleh Gambar 4.31.



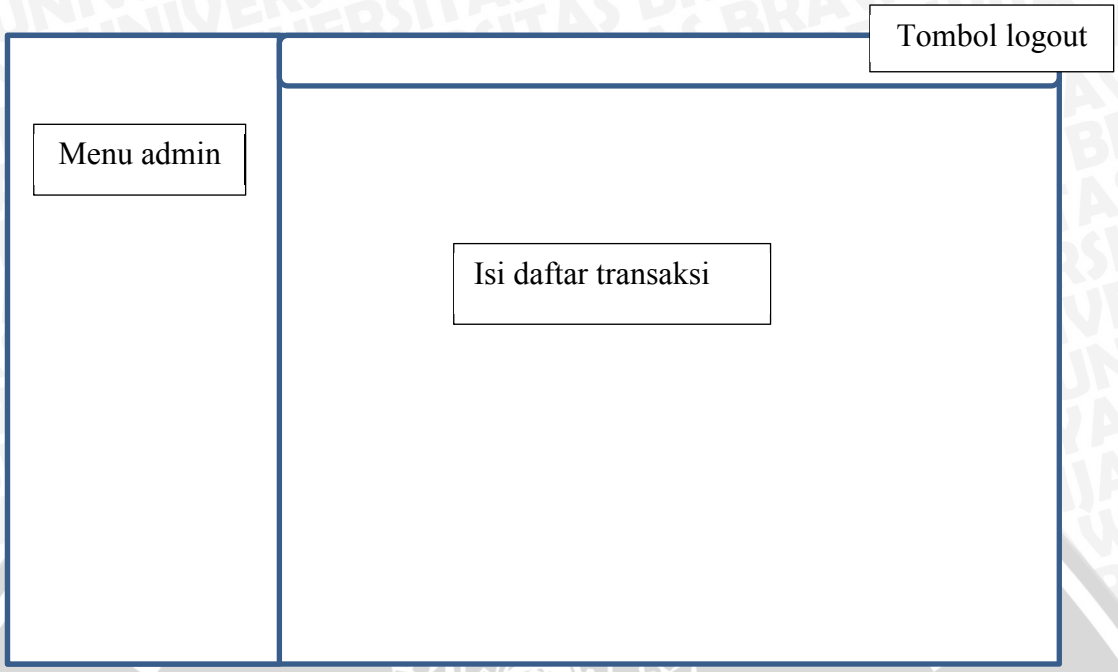
Gambar 4.31 *Sitemap* Antarmuka Halaman *Administrator* Keuangan

a. Halaman Transaksi

Halaman Transaksi Mata Kuliah semester pendek merupakan salah satu antarmuka pengguna untuk sistem *admin* yang berfungsi untuk mempermudah *admin* dalam melihat dan mencari daftar tagihan mahasiswa yang telah dipilih. Pada halaman ini, *admin* dapat melakukan pencarian mahasiswa terlebih dahulu. Perancangan halaman transaksi dapat dilihat pada Gambar 4.32 dan 4.33.



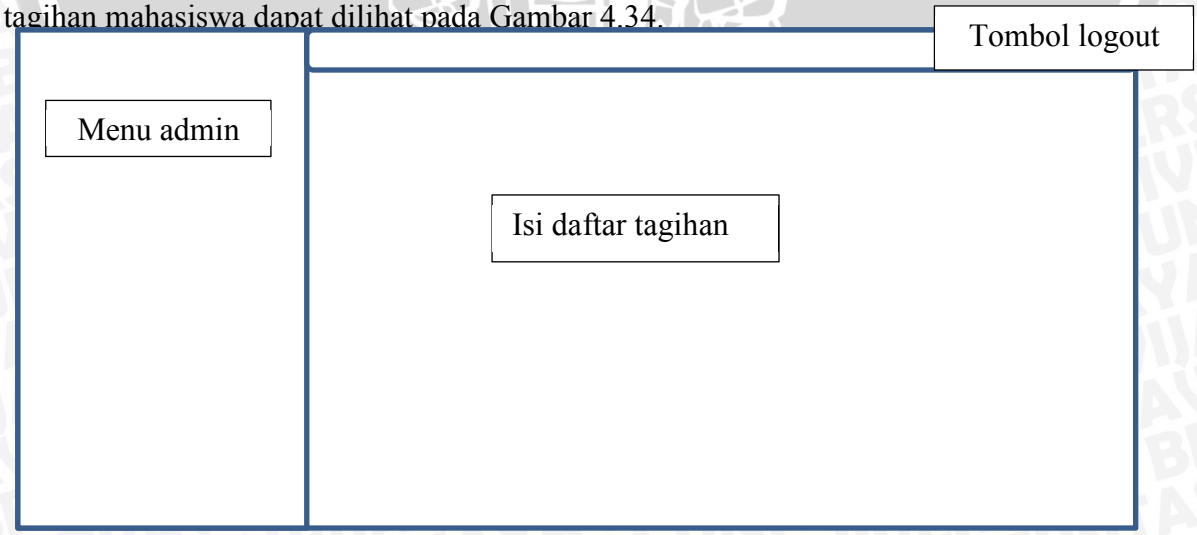
Gambar 4.32 Antarmuka Halaman Transaksi1



Gambar 4.33 Antarmuka Halaman Transaksi2

b. Halaman Tagihan Mahasiswa

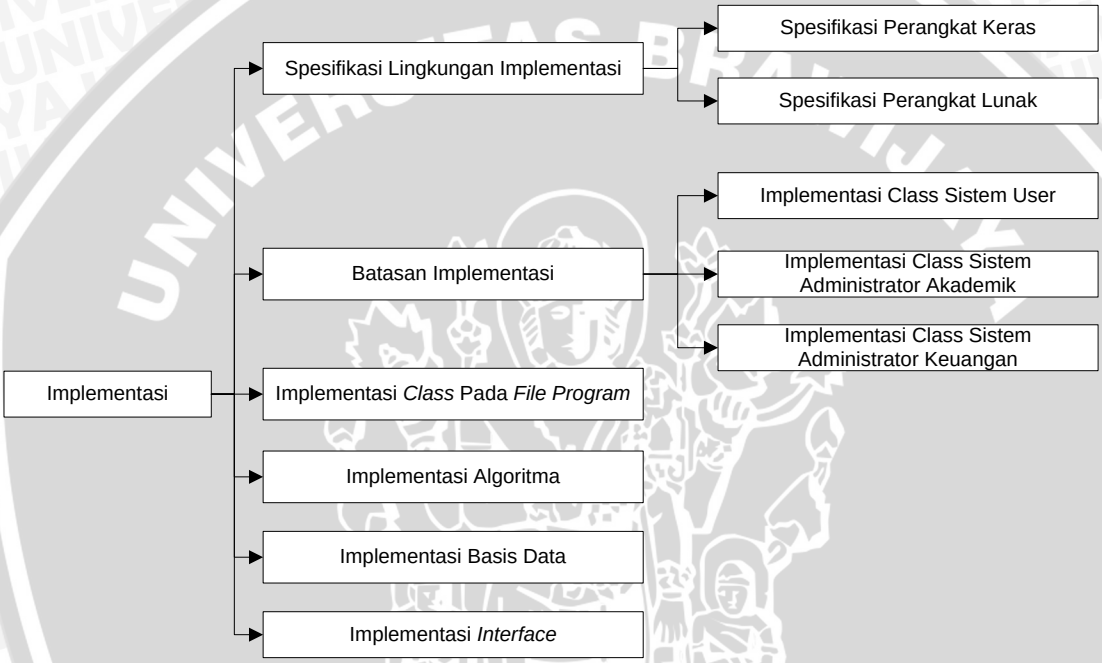
Halaman Tagihan Mahasiswa semester pendek merupakan salah satu antarmuka pengguna untuk sistem *admin* yang berfungsi untuk mempermudah *admin* dalam melihat daftar tagihan mahasiswa yang telah mendaftar semester pendek dan belum melakukan pelunasan pembayaran secara keseluruhan. Perancangan halaman tagihan mahasiswa dapat dilihat pada Gambar 4.34.



Gambar 4.34 Antarmuka Halaman Tagihan Mahasiswa

4.2 Implementasi

Bagian ini membahas mengenai implementasi sistem berdasarkan hasil yang telah didapatkan dari analisis kebutuhan dan proses perancangan aplikasi. Pembahasan terdiri atas penjelasan tentang spesifikasi lingkungan implementasi, batasan – batasan dalam implementasi, implementasi tiap *class* pada *file* program, dan implementasi algoritma. Struktur sub bab implementasi dapat ditunjukkan pada Gambar 4.35.



Gambar 4.35 Struktur Bab Implementasi

4.2.1 Spesifikasi Lingkungan Implementasi

Sistem informasi semester pendek PTIIK dikembangkan dalam lingkungan implementasi yang terdiri dari perangkat keras dan perangkat lunak yang menggunakan bahasa pemrograman tertentu.

4.2.1.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang dipakai dalam proses pengembangan dijelaskan pada tabel 4.29 :

Tabel 4.29 Spesifikasi perangkat keras komputer

PC- DELL inspiron 1440	
Processor	Pentium ® Dual-Core CPU T4500 @2.30GHz
Memory (RAM)	2048MB RAM
Harddisk	2.3GHz
Monitor	Generic PnP Monitor

4.2.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang dipakai dalam proses pengembangan sistem informasi semester pendek PTIIK dijelaskan pada Tabel 4.30.

Tabel 4.30 Spesifikasi perangkat keras komputer

PC Dell-Inspiron N1440	
Operating System	Microsoft Windows 7 Ultimate N 32-bit
Programming Language	PHP, HTML, CSS 3, java script
Database Management System & Web Server	XAMPP 1.7.3

4.2.2 Batasan Implementasi

Beberapa batasan dalam mengimplementasikan sistem informasi semester pendek ini adalah sebagai berikut :

1. Penggunaan SOA berasal dari SIAKAD Universitas Brawijaya.
2. Sistem yang dibangun menangani pengadaan semester pendek PTIIK khususnya di bagian pendaftaran dan pembayaran tagihan semeester pendek.
3. Sistem informasi semester pendek menggunakan syarat dan ketentuan yang berlaku pada pengelolaan semester pendek PTIIK.
4. *Database Management System* yang digunakan adalah MySQL yang terletak di *localhost*.
5. Penerapan *database* belum memakai metode keamanan data yang sempurna, seperti penggunaan enkripsi untuk *password*.

4.2.3 Implementasi *Class* Pada *File* Program

Setiap *class* yang telah dirancang pada proses perancangan direalisasikan pada sebuah *file* program dengan ekstensi *.php, baik menggunakan bahasa PHP maupun HTML. Penjelasan Mengenai pasangan antara *class* dengan *file* program yang digunakan ditunjukkan tabel di bawah ini.

4.2.3.1 Implementasi *Class* Sistem *User*

Tabel 4.31 Implementasi *class* aplikasi *user*

No.	*Package*	Nama *Class*	Nama *File* Program
1.	*Controllers*	Main	main.php
2.	*Controllers*	User	user.php
3.	*Controllers*	Matkul_ajuan	matkul_ajuan.php
4.	*Controllers*	Matkul_aktif	matkul_aktif.php
5.	*Controllers*	Tagihan	tagihan.php
6.	*Controllers*	Berita	berita.php
7.	*Controllers*	Logout	logout.php
8.	*views/content*	HTML *document*	beranda.php
9.	*views/content*	HTML *document*	registrationuser.php
10.	*views/content*	HTML *document*	registrationuser2.php
11.	*views/content*	HTML *document*	academicuser.php
12.	*views/content*	HTML *document*	academicuserajuan.php
13.	*views/content*	HTML *document*	academicusersempen.php
14.	*views/content*	HTML *document*	tagihanuser.php
15.	*views/content*	HTML *document*	beritauser.php
16.	*models*	ModelMatkul	modelMatkul.php

4.2.3.2 Implementasi *Class* Sistem Administrator Akademik

Tabel 4.32 Implementasi *class* aplikasi akademik

No.	*Package*	Nama *Class*	Nama *File* Program
1.	*Controllers*	Main	main.php

2.	*controllers*	Admin	admin.php
3.	*Controllers*	Matkul_ajuan	matkul_ajuan.php
4.	*Controllers*	Matkul_aktif	matkul_aktif.php
5.	*Controllers*	Berita	berita.php
6.	*Controllers*	Logout	logout.php
7.	*models*	ModelMatkulAd	modelmatkulAd.php
8.	*views/content*	HTML *document*	index.php
9.	*views/content*	HTML *document*	matkulajuan.php
10.	*views/content*	HTML *document*	matkulaktif.php
11.	*views/content*	HTML *document*	tambahberita.php
12.	*views/content*	HTML *document*	atursempen.php
13.	*views/content*	HTML *document*	lihatberita.php
14.	*views/content*	HTML *document*	tambahberita.php

4.2.3.3 Implementasi Class Sistem Administrator Keuangan

Tabel 4.33 Implementasi *class* administrator keuangan

No.	***Package***	Nama *Class*	Nama *File* Program
1.	*controllers*	Finance	finance.php
2.	*controllers*	Tagihan	tagihan.php
3.	*controllers*	Pembayaran	pembayaran.php
4.	*controllers*	Pengembalian	pengembalian.php
5.	*controllers*	Main	main.php
6.	*controllers*	Logout	logout.php
7.	*models*	ModelMatkulFi	modelmatkulFi.php
8.	*views/content*	HTML *document*	index.php
9.	*views/content*	HTML *document*	transaksi1.php
10.	*views/content*	HTML *document*	transaksi2.php
11.	*views/content*	HTML *document*	kembali1.php
12.	*views/content*	HTML *document*	kembali2.php

13.	*views/content*	HTML *document*	tagihanMhs.php
14.	*views/content*	HTML *document*	saldoMhs.php

4.2.4 Implementasi Algoritma

Sistem informasi semester pendek PTIIK menggunakan integrasi web service SIAKAD Universitas Brawijaya ini memiliki beberapa *method* atau proses. Dalam pengimplementasian ini, penulis menggunakan SOA yang telah ada sebelumnya dari SIAKAD Universitas Brawijaya. Beberapa *method* yang dicantumkan dalam skripsi ini hanya algoritma dari beberapa proses (operasi) sehingga tidak semua *method* akan dicantumkan.

4.2.4.1 Algoritma *Method* registrationuser2 *Class User* Sistem User

Method registrationuser2 merupakan salah satu *method* yang terdapat pada sistem *user* yang berfungsi untuk mengirimkan *request* untuk mendapatkan *resource* daftar mata kuliah yang telah diambil mahasiswa. Method ini juga berfungsi untuk menampilkan daftar serta tombol sebagai inputan keterangan tambah atau batalkan mata kuliah ajuan.

```
1   public function registrationuser2($nim) /*ini masuk
ke mata       kuliah ajuan mahasiswa*/
2     {
3
$dataajuan=file_get_contents("http://localhost/ws-
semester-
pendek/test.nilai.khs.php?nim=$nim&semester=2&is_pendek=
0&is_current=0");
4         $dataajuanarr=json_decode($dataajuan);
6         $this->load->model('modelMatkul');
7          $datalihat        =        $this->modelMatkul-
>view_matkulsp_user($nim);
8           if($dataajuanarr > 0){
9               $data['matkul_user'] = $datalihat;
10              $data['matkul_ajuan'] = $dataajuanarr;
11           $this->load->view('user/registrationuser2',
$data);
```

Baris untuk *request* data dari SIAKAD UB menggunakan *web service*

Mengubah data array dgn JSON

Kondisi jika data yang diambil ada. Mengirim nilai ke halaman view

```
12          }
13          else{
14                $data['matkul_user'] = $datalihat;
15                $data['matkul_ajuan'] = NULL;
16                $this->session-
>set_flashdata('msg_errorajuan2','<div
class="isa_error">Data  mata  kuliah  ajuan  anda  belum
tersedia.</div>');
17                $this->session-
>set_flashdata('msg_errorajuan','<div
class="isa_error">Data  mata  kuliah  aktif  anda  belum
tersedia.</div>');
18                $this->load-
>view('user/registrationuser2', $data);
19          }
20    }
```

... code lanjutan sebelumnya

... code lanjutan sebelumnya

Kondisi jika data web service yang diambil tidak ada. Sistem menjalankan pesan error

4.2.4.2 Algoritma *Method* tambahajuanadmin *Class* matkul_ajuan Sistem Administrator

Method ini berfungsi untuk menambahkan daftar mata kuliah aktif yang dilakukan oleh administrator. Hal pertama yang dilakukan adalah mengecek daftar mata kuliah yang telah aktif, selanjutnya sistem akan mengecek jumlah peminat mata kuliah, apabila jumlah mata kuliah yang dipilih lebih dari 10 mahasiswa, maka sistem melanjutkan proses selanjutnya yaitu menambahkan daftar mata kuliah aktif.

```
0   public   function   tambahajuanadmin($nama,   $k_mk,
$thn_mk)
1     {
2     $this->load->model('modelMatkulAd');
3
4     $datacek          =          $this->modelMatkulAd-
>cek_matkul_aktif(str_replace("%20","  ",$nama),  $k_mk,
$thn_mk);
5     if($datacek > 0){
6                 $this->session-
>set_flashdata('msg_danger','<div  class="alert  alert-
```

Memanggil modelMatkulAd fungsi cek_matkul_aktif dgn mengirim parameter

```
danger alert-dismissable">
7                   <button type="button" class="close"
data-dismiss="alert" aria-hidden="true">&times;</button>
8                    Mata kuliah <a class="alert-link"
href="#">'.str_replace("%20","      ",$nama).'      yang
ditambahkan GAGAL</a> karena telah tersimpan dalam
database.
9                </div>');
10                  redirect
('matkul_ajuan/matkulajuanadmin');
11              }
12              else{
13                  $datajml    =    $this->modelMatkulAd-
>cek_jumlah_aktif($k_mk);
14                  if($datajml > 0){
15                  $datamasuk   =   $this->modelMatkulAd-
>ins_matkul_aktif(str_replace("%20","  ",$nama),  $k_mk,
$thn_mk);
16                      if($datamasuk > 0 ){
17                          $this->session-
>set_flashdata('msg_succ','<div   class="alert   alert-
success alert-dismissable">
18                              <button     type="button"
class="close"         data-dismiss="alert"         aria-
hidden="true">&times;</button>
19                              Penambahan    data    mata
kuliah              <a              class="alert-link"
href="#">'.str_replace("%20","  ",$nama).'  BERHASIL</a>
untuk dibuka pada semester pendek.
20                              </div>');
21                              redirect
('matkul_ajuan/matkulajuanadmin');
22                      }
23                      else{
24                          $this->session-
>set_flashdata('msg_danger','<div   class="alert   alert-
```

Mengecek apakah matkul yang dipilih telah ditambahkan sebelumnya

Memanggil modelMatkulAd utk proses cek jml matkul aktif

Memanggil fungsi ins_matkul_aktif dgn mengirim parameter utk proses insert

Kondisi ketika proses penambahan sukses, muncul pesan berhasil

```
danger alert-dismissable">
25                          <button      type="button"
class="close"          data-dismiss="alert"          aria-
hidden="true">&times;</button>
26                          Penambahan    data    mata
kuliah              <a                class="alert-link"
href="#">'.str_replace("%20","    ",$nama).'    GAGAL</a>
untuk dibuka pada semester pendek.
27                          </div>');
28                          redirect
('matkul_ajuan/matkulajuanadmin');
29                    }
30
31               }
32               else{
33                    $this->session-
>set_flashdata('msg_danger','<div  class="alert  alert-
danger alert-dismissable">
34                      <button          type="button"
class="close"          data-dismiss="alert"          aria-
hidden="true">&times;</button>
35                      Penambahan data mata kuliah <a
class="alert-link"       href="#">'.str_replace("%20","
",$nama).' GAGAL</a> untuk dibuka pada semester pendek,
jumlah peserta < 10.
36                    </div>');
37                    redirect
('matkul_ajuan/matkulajuanadmin');
38               }
39          }
40    }
```

…code lanjutan sebelumnya

kondisi ketika penambahan matkul gagal, menampilkan pesan error

Kondisi ketika penambahan matkul gagal karena jml peminat < 10 orang

4.2.4.3 Algoritma *Method* bayartagihan *Class* tagihan Sistem Administrator Keuangan

```
0     public function bayartagihan($nim, $nama, $k_mk,
$sks, $tagihan) {
            $this->load->model('modelMatkulFi');
2           $datamasuk = $this->modelMatkulFi-
            >reg_matkul_mhs($nim, str_replace("%20","
            ",$nama), $k_mk, $sks, $tagihan);
3           if($datamasuk > 0){
4                 $this->session-
                  >set_flashdata('msg_succ','<div
                  class="alert alert-success alert-
                  dismissable">
5                 button type="button" class="close"
                  data-dismiss="alert" aria-
                  hidden="true">&times;</button>
6                 Pembayaran tagihan mata kuliah <a
                  class="alert-link" href="#"><?php echo
                  "$nama"; php> BERHASIL</a>, data
                  ditambahkan ke daftar mata kuliah
                  ajuan</div>');
7
      redirect('Pembayaran/transaksi3/'.$nim);
8           }
9           else{
11                $this->session-
                  >set_flashdata('msg_danger','<div
                  class="isa_error">Pembayaran mahasiswa
                  gagal, kesalahan pada database
                  </div>');
12                redirect('Pembayaran/transaksi2');
13          }
14    }
```

Menjalankan fungsi reg_matkul_mhs

Menjalankan kondisi ketika proses pembayaran berhasil

Menjalankan kondisi ketika proses pembayaran gagal

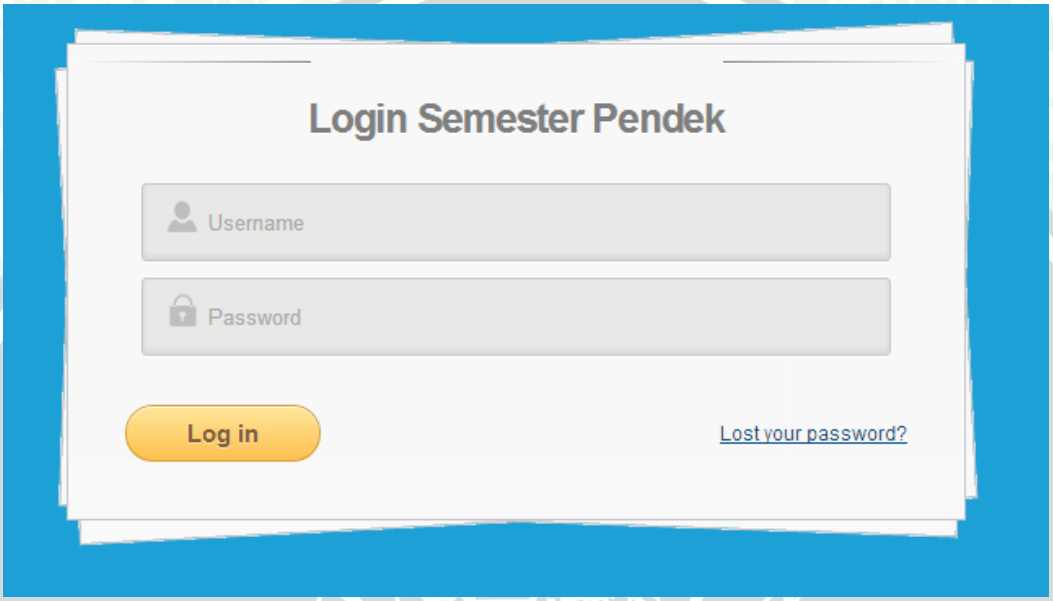
4.2.5 Implementasi Basis Data

Implementasi Database dilakukan berdasarkan perancangan data pada subab sebelumnya pada sub bab 4.1.2.2 menggunakan MySQL. Struktur tabel yang digunakan pada implementasi sistem ini sama dengan perancangan yang telah dibuat sebelumnya. Struktur tabel dapat dilihat pada Gambar 4.7 mengenai *entity relationship* sistem.

4.2.6 Implementasi *Interface*

a. Implementasi *Interface Login*

Gambar 4.36 merupakan halaman login termasuk halaman yang pertama kali diakses ketika sistem informasi semester pendek dijalankan. Didalam sistem login terdapat form input *username, password, button login*.

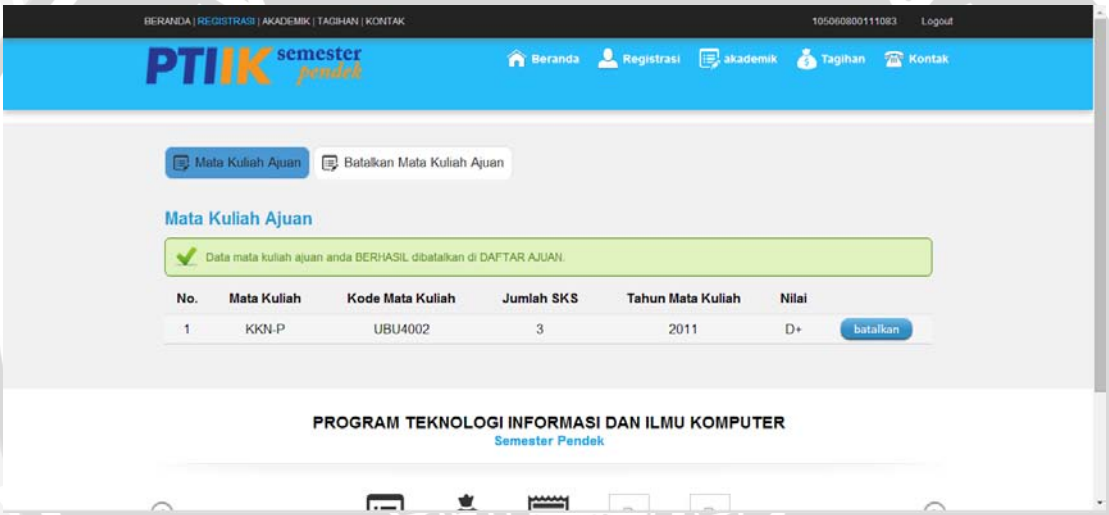


Gambar 4.36 Implementasi *Interface Login*

b. Implementasi *Interface* Registrasi Semester Pendek Sistem *User*

Halaman pendaftaran semester pendek (*registrationuser*2) ini menampilkan daftar mata kuliah yang telah diambil dan ada pada tabel KHS SIAKAD Universitas Brawijaya. Selain itu terdapat tab yang menghubungkan dengan proses pembatalan mata kuliah ajuan. Halaman ini juga menampilkan menu-menu yang berhubungan dengan semester pendek. Berikut ini adalah implementasi *interface* sistem untuk proses registrasi *user* dapat dilihat Gambar 4.37 dan Gambar 4.38.

Gambar 4.37 Implementasi *Interface Registrationuser*2 Tab Mata Kuliah Ajuan



Gambar 4.38 Implementasi *Interface Registrationuser*2 Tab Batalkan Mata Kuliah Ajuan

c. Implementasi *Interface* Tampilan KRS Semester Pendek Sistem *User*

Halaman tampilan KRS semester pendek (*academicusersempen*) ini menampilkan daftar mata kuliah yang telah diambil pada semester pendek. Halaman ini juga menampilkan menu-menu yang berhubungan dengan semester pendek. Gambar 4.39 menunjukkan implementasi *interface* untuk halaman tampilan KRS semester pendek *user*.

Gambar 4.39 Implementasi *Interface Academicusersempen* Sistem *User*

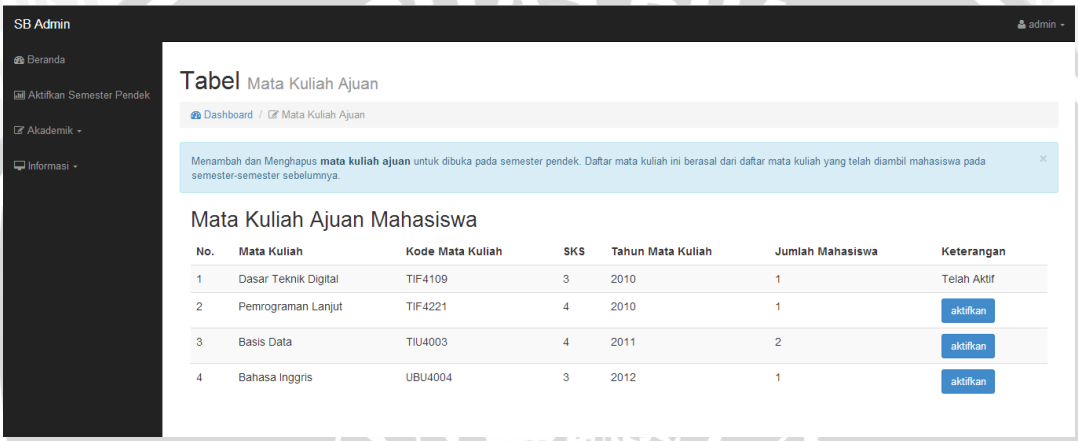
d. Implementasi *Interface* Tampilan Tagihan Semester Pendek Sistem *User*

Gambar 4.39 adalah halaman tampilan tagihan semester pendek (tagihanuser) ini menampilkan daftar tagihan semester pendek yang telah dihitung sesuai sistem semester pendek yang digunakan. Halaman ini juga menampilkan keterangan tagihan, jumlah yang telah dibayarkan serta sisa saldo *user*.

Gambar 4.40 Implementasi *Interface* Tagihanuser Sistem *User*

e. Implementasi *Interface* Daftar Mata Kuliah Ajuan Semester Pendek Sistem *Administrator* Akademik

Halaman tampilan daftar mata kuliah ajuan semester pendek (matkulajuan) ini menampilkan daftar mata kuliah ajuan semester pendek yang telah dihitung sesuai sistem untuk melihat jumlah peminatnya per mata kuliah. Halaman ini juga menampilkan tombol untuk menambahkan mata kuliah menjadi mata kuliah berstatus aktif. Gambar 4.41 merupakan tampilan *interface* sistem *administrator* untuk daftar mata kuliah ajuan.



Gambar 4.41 Implementasi *Interface* Daftar Mata Kuliah Ajuan Sistem *Administrator* Akademik

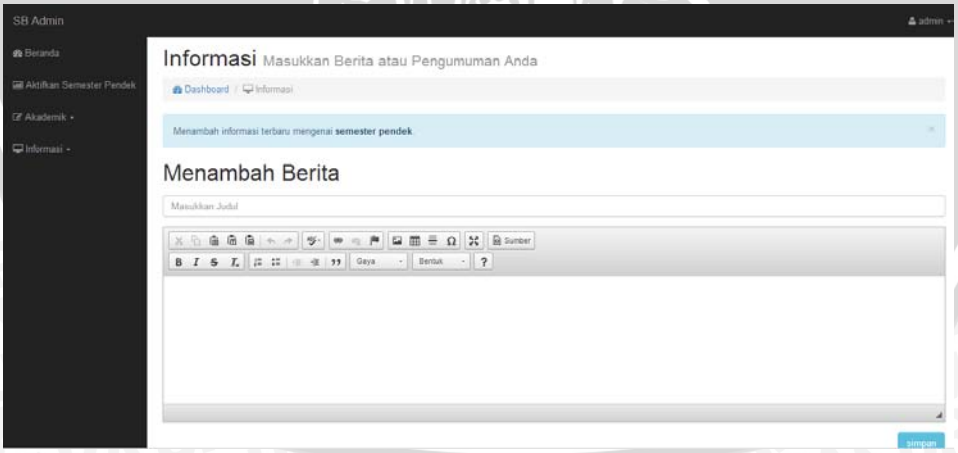
f. Implementasi *Interface* Daftar Mata Kuliah Aktif Semester Pendek Sistem *Administrator* Akademik

Gambar 4.42 adalah halaman tampilan daftar mata kuliah aktif semester pendek (matkulaktif) ini menampilkan daftar mata kuliah aktif semester pendek yang telah dibuka pada semester pendek sesuai ketentuan administrator akademik. Halaman ini juga menampilkan tombol untuk membatalkan mata kuliah yang telah berstatus aktif .

Gambar 4.42 Implementasi *Interface* Daftar Mata Kuliah Aktif Sistem *Administrator* Akademik

g. Implementasi *Interface* Tambah Informasi Semester Pendek Sistem *Administrator* Akademik

Halaman tampilan tambah informasi semester pendek (tambahberita) ini menampilkan *form input* judul dan teks area untuk menambah berita ataupun informasi mengenai semester pendek. Gambar 4.43 menunjukkan halaman tambah informasi semester pendek untuk *administrator*.



Gambar 4.43 Implementasi *Interface* Tambah Informasi Sistem *Administrator* Akademik

Sumber : Perancangan dan Implementasi

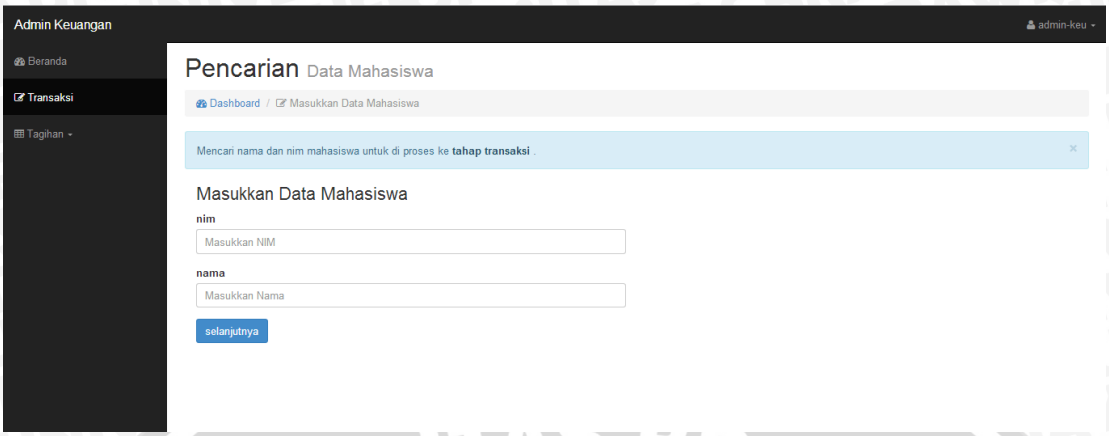
h. Implementasi *Interface* Lihat Informasi Semester Pendek Sistem *Administrator* Akademik

Halaman tampilan lihat informasi semester pendek (lihatberita) ini menampilkan isi berita atau informasi dan pengumuman mengenai semster pendek yang telah dimasukkan oleh administrator sebelumnya. Dibawah ini adalah Gambar 4.44 yang merupakan implementasi *interface* halaman lihat informasi administrator.

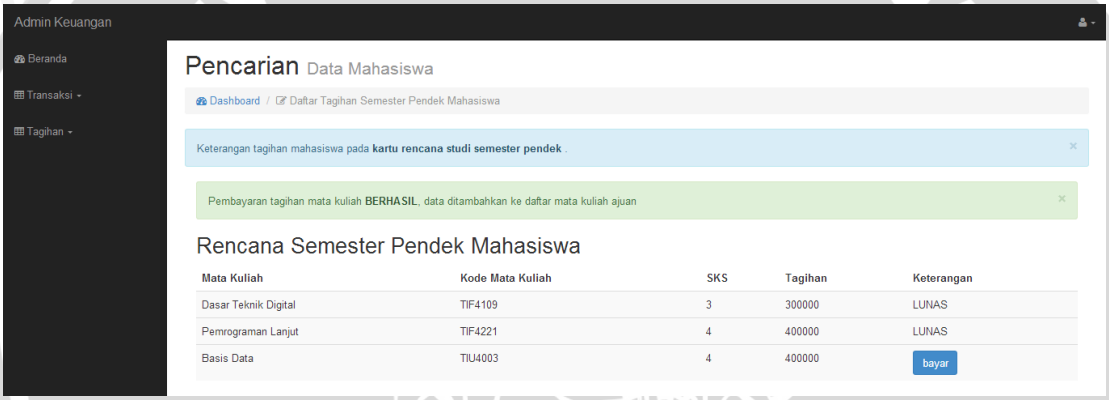
Gambar 4.44 Implementasi *Interface* Lihat Informasi Sistem *Administrator* Akademik

i. Implementasi *Interface* Transaksi Pembayaran Semester Pendek Sistem *Administrator* Keuangan

Halaman tampilan transaksi pembayaran semester pendek (transaksi1) ini menampilkan *form input* NIM dan nama mahasiswa bagi yang ingin melakukan transaksi pembayaran. Selanjutnya, akan ditampilkan halaman yang berisi keterangan mengenai jumlah tagihan pembayaran mahasiswa tersebut (transaksi2). Implementasi *interface* transaksi pembayaran dapat dilihat pada Gambar 4.45 dan Gambar 4.46.



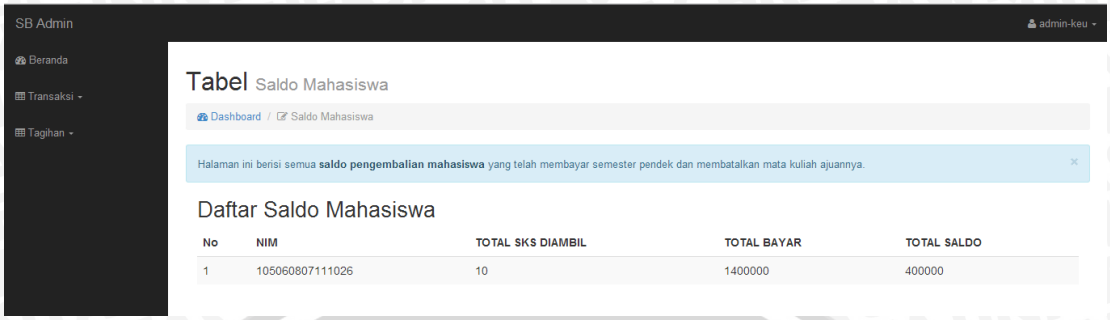
Gambar 4.45 Implementasi *Interface* Transaksi Pembayaran (Transaksi1) *Administrator* Keuangan



Gambar 4.46 Implementasi *Interface* Transaksi Pembayaran (Transaksi2) *Administrator* Keuangan

j. Implementasi *Interface* Daftar Saldo Semester Pendek Sistem *Administrator* Keuangan

Halaman tampilan daftar saldo semester pendek (saldoMhs) ini menampilkan seluruh daftar mahasiswa yang memiliki tagihan pengembalian semester pendek dapat dilihat pengimplementasiannya pada Gambar 4.47.



Gambar 4.47 Implementasi *Interface* Tagihan Pengembalian Mahasiswa (Saldomhs) *Administrator* Keuangan

Halaman tampilan daftar tagihan semester pendek (saldoMhs) ini menampilkan seluruh daftar mahasiswa yang memiliki tagihan pengembalian pada semester pendek.

BAB V
PENGUJIAN DAN ANALISIS

Tahapan terakhir setelah sistem sudah menjadi *prototype* adalah *testing* (pengujian). Pengujian yang dilakukan pada penelitian ini ditinjau dari dua segi yaitu pengujian sistem dan pengujian *web service*. Pengujian sistem dilakukan dengan melakukan pengujian validasi dengan metode *blackbox*. Pengujian metode *blackbox* merupakan metode yang digunakan untuk mengetahui apakah sistem berfungsi dengan baik dan sesuai dengan harapan tanpa memperhatikan struktur logika internal perangkat lunak. Sedangkan pengujian validasi yaitu untuk mengetahui apakah sistem yang sudah dibangun telah benar sesuai dengan kebutuhan. Jadi, tahapan pengujian validasi pada sistem dengan metode *blackbox* karena tidak memerlukan konsentrasi khusus terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan konformitas antara kinerja sistem dengan daftar kebutuhan.

Sedangkan pada pengujian *web service* dilakukan dengan melakukan pengujian REST *web service* pada setiap unit proses *web service*. Pengujian REST *web service* dilakukan menggunakan pengujian validasi sesuai dengan studi literatur yang telah didapatkan.

Proses analisis hasil pengujian mengacu pada dasar teori sesuai dengan hasil pengujian yang didapatkan. Analisis dilakukan terhadap hasil pengujian di setiap tahap pengujian.

5.1 Pengujian

Proses pengujian yang dilakukan melalui tiga tahapan yaitu pengujian sistem secara validasi menggunakan metode *blackbox*, pengujian REST *web service* secara validasi permintaan data ke *web service*. Seluruh proses pengujian ini dilakukan dengan menggunakan spesifikasi perangkat keras dan perangkat lunak sesuai dengan sub bab 4.2.1 yang telah dijelaskan sebelumnya.

5.1.1 Pengujian Sistem

5.1.1.1 Pengujian Validasi Dengan Metode *Blackbox*

Setelah sistem selesai dibangun baru dilakukan pengujian sistem. Penulis menggunakan pengujian *blackbox* untuk mengetahui fungsi-fungsi khusus yang dirancang untuk mengetahui seberapa jauh sistem tersebut berjalan dan seberapa banyak kesalahan yang ada pada sistem tersebut. Pengujian ini berdasar dari analisis kebutuhan fungsional sistem pada sub bab 4.1.1.4 mengenai daftar kebutuhan yang telah dijelaskan sebelumnya dalam Tabel 4.2, 4.3 dan 4.4. Apabila terjadi kesalahan maka sistem tersebut akan segera diperbaiki dan diuji kembali. Hasil dari pengujian validasi ditunjukkan pada Tabel 5.1, Tabel 5.2 dan Tabel 5.3.

Tabel 5.1 Hasil pengujian validasi sistem *user*

No	Fungsi	Prasyarat	Skenario yang Dilakukan	Hasil	Status
1	Registrasi mata kuliah *User*	*Login* sebagai *User*, Memilih nilai < C+, Hanya dibatasi 12 SKS	• *User* melakukan *login* • memilih menu "registrasi" • menekan tombol tambahkan pada mata kuliah yang dipilih pada semester pendek	Dapat menambahkan data mata kuliah ajuan yang dipilih *user* ke dalam *database*	valid
2	Batalkan mata kuliah ajuan semester pendek *User*	*Login* sebagai *User*	• *User* melakukan *login* • memilih menu "registrasi" • memilih tab	Dapat menghapus data mata kuliah ajuan yang dipilih	valid

			"batalkan mata kuliah" • menekan tombol batalkan pada mata kuliah yang dipilih	*user* sebelumnya dari *database*	
3	Lihat mata kuliah aktif	*Login* sebagai *User*	• *User* melakukan *login* • memilih menu "akademik" • memilih menu "lihat mata kuliah aktif"	Dapat menampilkan data mata kuliah berstatus aktif yang telah dibuka oleh *administrator* akademik	valid
4	Lihat KRS Semester Pendek	*Login* sebagai *User*	• *User* melakukan *login* • memilih menu "akademik" • memilih menu "lihat KRS semester pendek"	Dapat menampilkan data mata kuliah di KRS semester pendek	valid
5	Lihat Tagihan Semester Pendek	*Login* sebagai *User*	• *User* melakukan *login* • memilih menu "tagihan"	Dapat menampilkan data tagihan semester pendek *user*	valid

				sesuai dengan syarat yang berlaku di PTIIK	
6	Lihat Informasi Semester Pendek	*Login* sebagai *User*	• *User* melakukan *login* • memilih menu "informasi"	Dapat menampilkan data informasi mengenai semester pendek yang telah ditambahkan oleh *administrator* akademik	valid

Tabel 5.2 Hasil pengujian validasi sistem administrator akademik

No	Fungsi	Prasyarat	Skenario yang Dilakukan	Hasil	Status
1	Lihat daftar mata kuliah ajuan	*Login* sebagai *Admin* akademik	• *Admin* melakukan *login* • memilih menu "akademik" • memilih menu "mata kuliah ajuan"	Dapat menampilkan data mata kuliah ajuan yang telah dipilih *user* serta melihat jumlah peminatnya	Valid
2	Tambah mata kuliah aktif	*Login* sebagai *Admin*	• *Admin* melakukan *login*	Dapat mengubah data	valid

		akademik, jumlah peminat mata kuliah ajuan > 10 orang	• memilih menu "akademik" • memilih menu "mata kuliah ajuan" • menekan tombol tambahkan pada mata kuliah ajuan yang akan dibuka	mata kuliah ajuan menjadi mata kuliah aktif dengan syarat jumlah peminat > 10 orang	
3	Lihat mata kuliah aktif	*Login* sebagai *Admin* akademik	• *Admin* melakukan *login* • memilih menu "akademik" • memilih menu "mata kuliah aktif"	Dapat melihat data mata kuliah berstatus aktif yang telah dibuka	Valid
4	Membatalkan daftar mata kuliah aktif	*Login* sebagai *Admin* akademik	• *Admin* melakukan *login* • memilih menu "akademik" • memilih menu "mata kuliah aktif" • menekan tombol batalkan pada mata kuliah yang dipilih	Dapat menghapus data mata kuliah aktif yang telah dibuka *admin* sebelumnya dari *database*	Valid
5	Menambahkan informasi atau	*Login* sebagai *Admin*	• *Admin* melakukan *login*	Dapat menambah	valid

	berita	akademik	• memilih menu "informasi" • memilih menu "tambah" • mengisi form yang telah disediakan • menekan tombol simpan	informasi atau berita mengenai semester pendek	
6	Lihat informasi atau berita Semester Pendek	*Login* sebagai *Admin* akademik	• *Admin* melakukan *login* • memilih menu "informasi" • memilih menu "lihat informasi"	Dapat menampilkan daftar berita yang telah ditambahkan *admin* ke dalam *database*	Valid

Tabel 5.3 Hasil pengujian validasi sistem *administrator* keuangan

No	Fungsi	Prasyarat	Skenario yang Dilakukan	Hasil yang diharapkan	Hasil
1	Manambah transaksi pembayaran	*Login* sebagai *Admin* keuangan	• *Admin* melakukan *login* • Admin memilih menu "transaksi pembayaran" • Memasukkan NIM dan Nama mahasiswa yang	Dapat memasukkan data pembayaran semester pendek mahasiswa	Valid

			ingin membayar • Menekan tombol bayar		
2	Manambah transaksi pengembalian	*Login* sebagai *Admin* keuangan	• *Admin* melakukan *login* • Admin memilih menu "transaksi pengembalian" • Memasukkan NIM dan Nama mahasiswa yang ingin mmemiliki saldo • Menekan tombol bayar	Dapat memasukkan data pengembalian semester pendek mahasiswa	valid
3	Lihat tagihan mahasiswa	*Login* sebagai *Admin* keuangan	• *Admin* melakukan *login* • Admin memilih menu "tagihan pembayaran"	Dapat melihat data total tagihan pembayaran mahasiswa	valid
4	Lihat pengembalian mahasiswa	*Login* sebagai *Admin* keuangan	• *Admin* melakukan *login* • Admin memilih menu "saldo pengembalian"	Dapat melihat data total saldo pengembalian mahasiswa	valid

5.1.2 Pengujian *Web Service*

5.1.2.1 Pengujian REST *Web Service*

Pengujian REST *web service* dilakukan untuk mengetahui apakah dalam implementasinya baik dari sisi server maupun sisi client (yang me-*request*) sudah berjalan seperti yang diharapkan. Pengujian REST *web service* ini menggunakan strategi pengujian validasi.

1. Kasus Uji GET KHS

Pada tahap ini dilakukan, pengujian REST *web service* dengan menggunakan metode blackbox pada menu registrasi semester pendek di sistem *user*.

Tabel 5.4 Hasil pengujian REST *web service* sistem *user*

Nama Kasus Uji	Kasus Uji *GET KHS* (format JSON)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa REST *web service* yang dibuat dapat memenuhi kebutuhan fungsional untuk menampilkan daftar KHS sesuai dengan format yang ditentukan adalah JSON
Data Masukan	1. NIM mahasiswa 2. Password yang telah ditentukan 3. Format (JSON)
Prosedur Uji	1. Menuliskan *service endpoint* dari *server* di *browser* **(http://devel184.ub.ac.id/siakad_api/api/akademik/CallGetKhsMhs)** 2. Menambahkan parameter *query* berupa NIM mahasiswa, semester mahasiswa (parameter ini tidak berhubungan dengan data yang akan diambil), is_pendek dan is_*current* yang telah ditentukan **(http://localhost/ws-semester-pendek/test.nilai.khs.php?nim=$nim&semester=2&is_pendek=0&is_current=0)**

	3. Menjalankan *browser*
Hasil yang Diharapkan	Browser dapat menampilkan representasi dari KHS mahasiswa dalam format JSON.

5.2 Analisis

Analisis bertujuan untuk menganalisa data hasil pengujian hingga menghasilkan kesimpulan-kesimpulan. Analisis ini dilakukan berdasarkan hasil implementasi hingga pengujian yang telah dilakukan terhadap sistem dan mengacu pada dasar teori. Proses analisis yang dilakukan antara lain adalah analisis hasil pengujian validasi sistem dan analisis hasil pengujian REST *web service*.

Berdasarkan hasil pengujian validasi sistem, implementasi perancangan sistem dapat dikatakan layak dan berhasil untuk setiap sistem penggunanya. Sistem dapat menjalankan fungsi-fungsi yang ada dan memproses sesuai persyaratan yang ditetapkan. Selain itu hasil yang dikeluarkan juga telah sesuai dan tidak ada kendala sama sekali.

Mengacu pada hasil pengujian REST *web service*, sistem telah dapat mengembalikkan nilai sesuai dengan parameter yang diminta. Namun, banyak faktor yang dapat mempengaruhi bahkan memperburuk jalannya proses integrasi ini, seperti kualitas koneksi internet dan padatnya aktivitas pengakses server SIAKAD UB.

Namun dalam proses pengembangan implementasinya penulis memiliki kendala dalam mengintegrasikan *web service* karena faktor *server* localhost yang digunakan. Berdasarkan pada sub bab 4.2.1.2 mengenai spesifikasi perangkat lunak, penulis menggunakan XAMPP version 1.7.3 yang terinstall pada windows 7 ultimate. Hasilnya waktu akses yang dibutuhkan untuk mengakses *web service* cukup lama, untuk mengatasi hal tersebut, pada saat proses pengambilan data melalui *web service* diharuskan untuk me-*restart server* localhost yang digunakan terlebih dahulu sebelum menjalankan proses. Hal ini sangat membantu dan waktu akses *web service* yang diperlukan menjadi sangat cepat.

Secara keseluruhan, sistem dapat dikatakan layak. Hasil pengujian sistem secara keseluruhan menunjukkan sistem bekerja sesuai dengan parameter uji. Ketiga parameter uji bernilai valid. Dimulai dari jalannya sistem dalam proses login, me-*request* REST *web service* SIAKAD UB, melakukan proses penambahan, pembatalan dan melihat tagihan pembayaran bagi *user*, dapat melihat mata kuliah ajuan melihat jumlah peminatnya serta mengaktifkan mata kuliah semester pendek bagi *administrator* akademik dan dapat melakukan proses pembayaran dan pengembalian bagi *administrator* keuangan. Selain itu, sistem akan bekerja lebih optimal apabila didukung dengan kecepatan akses internet serta meminimalisir aktivitas pada server SIAKAD UB.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, pengujian dan analisis yang dilakukan, maka diambil kesimpulan sebagai berikut :

1. Proses analisis dalam membangun sistem informasi semester pendek dilakukan dengan melakukan analisa terkait pengelolaan semester pendek di PTIIK, melakukan studi literatur melalui wawancara mengenai prosedur pengadaan semester pendek serta mendeskripsikan kebutuhan fungsional maupun non-fungsional sistem dengan menggunakan diagram UML (*Unified Modelling Language*) *use case*.
2. Proses perancangan sistem informasi semester pendek dilakukan dengan membuat beberapa perancangan yang sesuai dengan tahap analisa sebelumnya, seperti perancangan arsitektural, perancangan basis data, perancangan *class diagram*, perancangan struktur JSON, perancangan a*ctivity diagram*, perancangan *sequence diagram* dan perancangan antarmuka sistem yang akan dibuat.
3. Proses implementasi dilakukan berdasarkan pada proses perancangan sebelumnya dengan melakukan pembuatan kode program dalam bentuk *class-class* program serta melakukan implementasi basis data berdasarkan perancangan sebelumnya.
4. Perancangan dan implementasi sistem informasi semester pendek PTIIK ini dapat dilakukan dengan menggunakan *web service* SIAKAD UB untuk mengambil data yang diperlukan dan mengintegrasikan data tersebut dengan sistem yang dibangun.
5. Dalam pengimplementasiannya tidak terlepas dengan jaringan SOA SIAKAD UB. Sistem akan me-*request* data apa saja yang dibutuhkan memanfaatkan

web service SIAKAD UB, selanjutnya *web service* SIAKAD UB akan mengembalikkan data yang diminta sesuai parameter yang dimasukkan. Seluruh proses ini telah disediakan oleh SIAKAD UB melalui SOA dan teknologi REST dalam format JSON.

6.2 Saran

Saran yang dapat diberikan setelah menyelesaikan penelitian skripsi ini adalah:

1. Pengadaan semester pendek PTIIK sebaiknya diadakan dan dikelola secara *real time* dan *online.* Sehingga baik peserta maupun pengelola dapat terus terhubung menjalankan tugasnya kapanpun dimanapun tanpa terkendala oleh lambatnya komunikasi yang diterima.
2. Pengimplementasian sistem informasi semester pendek dapat dilakukan dengan menambahkan integrasi *web service* pada kebutuhan fungsi *login.*
3. Pengembangan sistem dapat dilakukan dengan menambahkan fitur penjadwalan menggunakan algoritma genetik
4. Pengembangan sistem dapat dilakukan dengan menambahkan fitur penilaian menggunakan integrasi *web service* ke aplikasi SIADO
5. Pembangunan sistem informasi menggunakan integrasi *web service* SIAKAD UB dapat menggunakan teknologi selain REST dan format pertukaran datanya selain JSON.
6. Pengujian yang dilakukan memiliki beberapa kendala, salah satunya yang telah dijelaskan sebelumnya pada bab pengujian yaitu karena faktor perangkat lunak (*server* localhost XAMPP) yang digunakan. Untuk mengatasi hal ini, dilakukan proses *restart server* localhost sebelum mengakses halaman yang me-*request web service.*

DAFTAR PUSTAKA

[ALL-10] Allamaraju, Subbu. 2010. *RESTful Web Services Cookbook*. Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly.

[DSI-08] DSIPLK (Direktorat Sistem Informasi, Perangkat Lunak & Konten). 2008. *Kerangka Acuan & Pedoman Interoperabilitas Sistem Informasi Instansi Pemerintahan*. Depkominfo RI : Jakarta.

[JSN-14] http://www.json.org/json-id.html (diakses pada : Juni 2014)

[KTU-12] Kepala Tata Usaha (KTU) PTIIK. 2012. *Manual Prosedur Kuliah Semester Pendek (SP) Program Teknologi Informasi Dan Ilmu Komputer Universitas Brawijaya.*

[LCK-08] Lucky. 2008. *XML Web services: Aplikasi Desktop, Internet & Handphone.* Jasakom : Jakarta.

[MUL-02] Mullins, Craig S. 2002. *Database Administration The Complete Guide to Practices and Procedures.*

[NUR-09] Nurseitov, Nurzhan, et al. 2009. *Comparison of JSON and XML Data Interchange Formats: A Case Study*. *Caine*. 9: 157-162.

[PDM-12] Tim Penyusun. 2012. *Pedoman Pendidikan - Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya Tahun Akademik 2012/2013*. Bab II: Sistem Pendidikan, Pasal: 10 Program Semester Pendek.

[PRY-05] Prayitno, Wendhie; S, Kom. 2005. *Desain Model Sistem Perangkat Lunak dengan UML*.

[PTS-08] Pautasso, C., 2008. *REST vs SOAP Making the Right Architectural Decision, SOA Symposium*. Amsterdam.

[PUR-08] Purnamasari, Susan Dian. 2008. *Web Service Sebagai Solusi Integrasi Data Pada Sistem Informasi Akademik Universitas Bina Darma*. Jurnal Imiah MATRIK Vol95. No12,April 2008:1 -20.

[ROS-13] Rosa A. S, M.Salahuddin. 2013. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Informatika : Bandung.

[SKY-09] Sukyadi, D., 2009. *Model Interoperabilitas Sistem Informasi Layanan Publik Studi Kasus: e-Government, Karya Akhir, Prodi Magister Teknologi Informasi, Fasilkom UI*. Jakarta.

[SMT-10] Simarmata, Janner. 2010. *Rekayasa Perangkat Lunak*. C.V Andi OFFSET : Yogyakarta.

[WYN-12] Wiyono, Didiek S. dan Wijayanto, Ardhi, 2012. *Implementasi Rest Web Service Dengan Menggunakan Json Pada Aplikasi Mobile Enterprise Resource Planning*, Perfoma, Vol. 11, No. 2, hal.143 – 152.

[YSR-14] Yusron, Mayang Laily, et al. *Rancang Bangun REST Web Service pada Aplikasi Penentuan Portofolio Saham Menggunakan Model Markowitz dan JQuery Mobile*. (diakses pada : Juni 2014)