PEMBANGUNAN SISTEM INFORMASI BIMBINGAN SKRIPSI PTIIK MENGGUNAKAN FITUR PENYIMPANAN BERKAS BERBASIS *CLOUD*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
mencapai gelar Sarjana Komputer

Disusun Oleh :
SA'DIYAH NURUL FAUZIYAH
NIM : 115060800111071

PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2015

Lembar Persetujuan

PEMBANGUNAN SISTEM INFORMASI BIMBINGAN SKRIPSI PTIIK MENGGUNAKAN FITUR PENYIMPANAN BERKAS BERBASIS *CLOUD*

SKRIPSI

KONSENTRASI REKAYASA PERANGKAT LUNAK

Diajukan untuk memenuhi sebagian persyaratan

mencapai gelar Sarjana Komputer

Disusun oleh :

Sa'diyah Nurul Fauziyah

NIM. 115060800111071

Skripsi ini telah diperiksa dan disetujui oleh dosen pembimbing pada

Tanggal 28 Juni 2015

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

Lembar Pengesahan

PEMBANGUNAN SISTEM INFORMASI BIMBINGAN SKRIPSI PTIIK MENGGUNAKAN FITUR PENYIMPANAN BERKAS BERBASIS *CLOUD*

SKRIPSI

KONSENTRASI REKAYASA PERANGKAT LUNAK

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun oleh :

Sa'diyah Nurul Fauziyah

NIM. 115060800111071

Skripsi ini telah diuji dan dinyatakan lulus pada tanggal 7 Agustus 2015

Penguji I	**Penguji II**
Fajar Pradana, S.ST., M.Eng.	**Aryo Pinandito, S.T, M.MT**
NIK. 871121 16 1 1 0371	**NIP. 198305192014041001**

Penguji III

Mahendra Data, S.Kom
NIK. 201503 861117 1 0001

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 disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam 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, 10 Juli 2015

Mahasiswa

Sa'diyah Nurul Fauziyah

NIM. 115060800111071

KATA PENGANTAR

Puji dan syukur Penulis panjatkan kehadirat Allah SWT, karena hanya dengan rahmat dan bimbingannya Penulis dapat menyelesaikan tugas akhir dengan judul "Pembangunan Sistem Informasi Bimbingan Skripsi PTIIK Menggunakan Fitur Penyimpanan Berkas Berbasis *Cloud*" dengan baik. Tanpa rahmat dan bimbingan dari Tuhan Yang Maha Esa, maka niscaya Penulis tidak akan dapat menyelesaikan tugas akhir ini dengan baik dan tepat waktu.

Penulisan dan penyusunan laporan skripsi ini dapat terlaksana dengan baik karena adanya bantuan secara langsung maupun tidak langsung dari pihak tertentu diantaranya:

1. Bapak Denny Sagita Rusdianto, S.Kom., M.Kom., dan Bapak Satrio Agung Wicaksono, S.Kom., M.Kom. selaku dosen pembimbing I dan dosen pembimbing II yang telah banyak memberikan ilmu dan saran untuk laporan skripsi ini.
2. Bapak Drs. Marji, M.T. selaku Ketua Program Studi Informatika / Ilmu Komputer Program Teknologi Informasi dan Ilmu Komputer (PTIIK) Universitas Brawijaya.
3. Bapak dan Ibu dosen serta seluruh staff Program Teknologi Informasi dan Ilmu Komputer (PTIIK) Universitas Brawijaya yang telah memberikan ilmunya serta arahan selama masa perkuliahan.
4. Kedua orang tua saya yang telah memberi motivasi, kasih sayang serta dukungan moril dan materil. Dan keluarga besar di Lamongan yang telah memberikan semangat dari awal sampai akhir pengerjaan skripsi.
5. Sahabat-sahabat kuliahku di Malang, Dhemmy, Kiki, Olin, Ester, Nani yang telah banyak meberikan dukungan dan doa demi kelancaran skripsi ini.
6. Sahabat-sahabatku di pesantren luhur tercinta Tyas, Arifa, Arina, Nana, Happy, Mbak Elok, dan yang lainnya.

7. Teman – teman saya TIF H 2011 dan IF 2011 yang tidak bisa disebutkan satu persatu yang selalu mendukung saya.
8. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya tugas akhir ini.

Semoga jasa dan amal baik mendapatkan balasan dari Allah SWT. Dengan segala kerendahan hati, penulis menyadari sepenuhnya bahwa skripsi ini masih jauh dari sempurna karena keterbatasan materi dan pengetahuan yang dimiliki penulis. Saran dan kritikan yang bersifat membangun dapat disampaikan melalui email penulis diyahnf@gmail.com Akhirnya semoga skripsi ini dapat bermanfaat dan berguna bagi pembaca terutama mahasiswa Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.

Malang, 09 Juli 2015

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRACT

Sa'diyah Nurul Fauziyah. 2015 : ***Development of Thesis Guidance Information System With Cloud Based Storage System. Information Technology and Computer Science Program, Brawijaya University, Malang.***

***Advisory Lecturer* : Denny Sagita Rusdianto, S.Kom., M.Kom *and* Satrio Agung Wicaksono, S.Kom., M.Kom.**

Thesis is a final project that must be taken by the student to obtain a bachelor's degree. Implementation of the thesis will be guided by one or two lecturer to help student during the process of thesis. Until now, the implementation guidance in PTIIK thesis is still done through direct face to face with the lecturer. Based on the results of a questionnaire given to 40 respondents of PTIIK students, it can be concluded that the implementation of the guidance in the conventional thesis has several problems including, process guidance can be neglected due to the difference in distance and schedules between faculty and students, in addition to the guidance can only be carried out on the day and hour thus causing a particular queue. This paper describes the development of systems to support the thesis that the guidance is supported by the presence of features to make online file storage. Testing is done through three stages, black box testing, white box, and the rest-api testing. Results of white box testing indicates that the program has a good structure and the code complexity to make improvements is low. Black box testing provides valid status indicate that a program has been run in fit with the design that had been made. And the rest-api testing, each tested method returns a status code of 200 that indicates successful http request.

***Keywords** : cloud storage, cloud computing, thesis guidance*

ABSTRAK

Sa'diyah Nurul Fauziyah. 2015: Pembangunan Sistem Informasi Bimbingan Skripsi PTIIK Menggunakan Fitur Penyimpanan Berkas Berbasis *Cloud*. Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang.

Dosen Pembimbing : Denny Sagita Rusdianto, S.Kom., M.Kom *and* Satrio Agung Wicaksono, S.Kom., M.Kom.

Skripsi merupakan sebuah tugas akhir yang wajib ditempuh oleh mahasiswa untuk memperoleh gelar sarjana. Dalam pelaksanaannya, mahasiswa akan dibimbing oleh satu sampai dua dosen pembimbing yang bertujuan untuk membantu kelancaran mahasiswa dalam proses pengerjaan skripsi. Sampai saat ini pelaksanaan bimbingan skripsi di PTIIK masih dilakukan secara konvensional melalui tatap muka secara langsung dengan dosen pembimbing. Berdasarkan hasil dari kuisioner yang diberikan kepada 40 responden mahasiswa PTIIK, dapat disimpulkan bahwa pelaksanaan bimbingan skripsi secara konvensional memiliki beberapa masalah diantaranya, proses bimbingan dapat terbengkalai akibat adanya perbedaan jarak dan jadwal antara dosen dan mahasiswa, selain itu bimbingan hanya dapat dilaksanakan pada hari dan jam tertentu sehingga menyebabkan terjadinya antrian. Skripsi ini akan menjelaskan tentang pengembangan sistem untuk mendukung pelaksanaan bimbingan skripsi yang didukung oleh adanya fitur untuk melakukan penyimpanan berkas secara *online*. Pengujian dilakukan melalui tiga tahap yaitu pengujian *black box*, *white box*, dan pengujian *rest-api*.

Kata kunci : *cloud computing*, *cloud storage*, bimbingan skripsi.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Skripsi merupakan karya ilmiah mahasiswa program sarjana (S1) yang merupakan wujud dari kajian pengetahuan dan/atau penerapan teknologi berdasarkan kaidah ilmiah dalam bidang studi yang telah dipelajarinya. Setiap mahasiswa wajib mengambil mata kuliah tersebut, karena skripsi digunakan sebagai salah satu prasyarat bagi mahasiswa untuk memperoleh gelar sarjana. Penulisan skripsi mencakup beberapa tahapan yaitu pengajuan proposal, pelaksanaan bimbingan, revisi, seminar, pameran, dan ujian skripsi.

Tahap awal dalam skripsi adalah pengajuan proposal. Mahasiswa harus mengajukan proposal skripsi terlebih dahulu, kemudian setelah proposal yang diajukan mendapat persetujuan maka mahasiswa dapat mulai menulis skripsi. Dalam hal ini, mahasiswa akan dibimbing oleh satu sampai dua dosen pembimbing yang bertujuan untuk membantu kelancaran mahasiswa dalam proses pengerjaan skripsi.

Program Teknologi Informasi dan Ilmu Komputer (PTIIK) merupakan salah satu fakultas yang ada di Universitas Brawijaya. Pelaksanaan bimbingan skripsi di PTIIK saat ini masih dilaksanakan secara konvensional. Secara umum prosedur tersebut dilaksanakan dengan cara mahasiswa terlebih dahulu membuat janji atau melihat jadwal bimbingan dari masing-masing dosen pembimbing, datang ke ruangan dosen pembimbing sesuai dengan jam dan hari yang telah terjadwal, dan mengantri untuk melaksanakan bimbingan skripsi.

Berdasarkan hasil dari kuisioner yang diberikan kepada 40 *stakeholder* mahasiswa PTIIK, dapat disimpulkan bahwa pelaksanaan bimbingan skripsi secara konvensional memiliki beberapa masalah, yaitu: prosedur tidak dapat dilaksanakan apabila pihak-pihak terkait sedang tidak berada di tempat, proses bimbingan skripsi hanya dapat dilaksanakan pada hari dan jam tertentu sehingga menyebabkan terjadinya antrian, pemborosan biaya yang dikeluarkan untuk mencetak dokumen skripsi secara berulang-ulang akibat adanya revisi, proses bimbingan skripsi dapat

terbengkalai akibat adanya perbedaan jarak dan jadwal antara dosen dan mahasiswa, serta dokumen skripsi mahasiswa kurang terorganisir dengan baik.

Dengan berkembangnya teknologi informasi yang semakin pesat, manual prosedur pelaksanaan bimbingan skripsi di PTIIK dapat diadopsi dalam suatu sistem yang dapat memberi kemudahan dan pelayanan secara lebih optimal. Sistem yang dapat diimplementasikan adalah sistem informasi bimbingan skripsi yang dapat diakses secara *online* melalui media internet, sehingga dapat mengurangi keterbatasan waktu, tempat, dan meminimalisir biaya yang dibutuhkan dalam proses pelaksanaan bimbingan skripsi.

Dalam melaksanakan penulisan skripsi kebutuhan akan data merupakan hal yang tak bisa terhindarkan lagi. Semua hasil kerja mahasiswa berupa data, dimana data tersebut perlu diorganisir dengan baik dalam suatu ruang penyimpanan. Selain itu, pada kondisi tertentu dosen pembimbing perlu memeriksa data hasil kerja dari mahasiswa untuk mengetahui sejauh mana progres skripsi yang telah dikerjakan oleh mahasiswa yang dibimbingnya.

Dalam hal ini, fitur yang dapat digunakan untuk memenuhi kebutuhan akan penyimpanan data adalah fitur penyimpanan berbasis *cloud*. Dengan fitur tersebut mahasiswa dapat meyimpan dan mengakses data serta dokumen skripsi miliknya secara *online*. Selain itu mahasiswa juga dapat menggunakan layanan untuk berbagi dokumen dengan dosen pembimbing maupun dengan mahasiswa lain.

Cloud computing dapat didefinisiskan sebagai akses layanan *on-demand* ke sekumpulan sumber daya komputasi seperti jaringan, *server*, penyimpanan, aplikasi dan layanan [MEL-11]. *Cloud storage* merupakan salah satu aspek dari *cloud computing*, dimana penyimpanan dilakukan secara *online* dan *user* dapat terhubung ke media penyimpanan tersebut melalui komputer miliknya. *User* menyimpan data tanpa mengetahui dimana lokasi penyimpanan data, sehingga *user* dapat mengakses data tersebut kapanpun dan dimanapun selama mereka terhubung ke jaringan *internet* [ADE-14]. Akses yang dapat dilakukan oleh *user* antara lain *upload*, *delete*, *rename*, *share*, serta *dowload* dokumen miliknya.

Secara umum arsitektur *cloud storage* terdiri dari *front end* yang memiliki API untuk mengakses penyimpanan, *storage logic*, dan *back-end storage* [JON-10:2]. *Front end* dapat menggunakan *web service*, *file based*, dan beberapa tradisional *front end* lain. Dengan demikian sistem tidak dapat diakses secara langsung oleh pengguna melainkan dibutuhkan sistem lain yang mengakses layanan *cloud* melalui API yang telah disediakan. Sehingga sistem informasi bimbingan skripsi dan sistem penyimpanan *cloud* akan dikembangkan secara terpisah.

Dengan sistem informasi bimbingan skripsi ini diharapkan dapat mengatasi masalah dalam proses pelaksanaan bimbingan skripsi. Melalui sistem ini, mahasiswa dapat melaksanakan bimbingan skripsi secara *online* tanpa adanya batasan ruang dan waktu. Hal ini diharapkan dapat memberikan kemudahan dan pelayanan bagi mahasiswa secara lebih optimal. Dan dengan adanya fitur penyimpanan berkas berbasis *cloud*, diharapkan dokumen skripsi mahasiswa akan lebih terorganisir dengan baik karena dokumen yang telah di-*upload* dan direvisi akan tersimpan secara sistematis di dalam sistem.

1.2 Rumusan Masalah

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

1. Bagaimanana menganalisis kebutuhan sistem informasi bimbingan skripsi sesuai dengan standar prosedur yang ada di PTIIK?
2. Bagaimana merancang dan mengimplementasikan sistem informasi bimbingan skripsi yang dapat mendukung pelaksanaan bimbingan skripsi secara *online* di PTIIK?
3. Bagaimana merancang dan mengimplementasikan fitur penyimpanan berkas berbasis *cloud* pada sistem informasi bimbingan skripsi ?
4. Bagaimana validitas sistem informasi bimbingan skripsi yang mengimplementasikan fitur penyimpanan berkas berbasis *cloud*?

1.3 Batasan Masalah

Agar pembahasan tidak menyimpang dari tujuan maka dilakukan pembatasan masalah sebagai berikut:

1. Implementasi sistem penyimpanan berkas berbasis *cloud* difokuskan pada sisi pengembangan perangkat lunak dan fungsionalitas sistem.
2. Sistem penyimpanan berbasis *cloud* terintegrasi dengan sistem informasi bimbingan skripsi sebagai sebuah fitur yang digunakan untuk menyimpan dokumen.
3. Pengembangan perangkat lunak hanya sampai pada tahap pengujian dan tidak mencakup tahap *maintenance* (pemeliharaan).
4. *Web service* yang digunakan tidak mencakup masalah keamanan.

1.4 Tujuan Penelitian

1. Menganalisis, merancang, mengimplementasikan, dan menguji sistem informasi bimbingan skripsi di PTIIK.
2. Merancang fitur penyimpanan berkas berbasis *cloud* untuk diimplementasikan pada sistem informasi bimbingan skripsi .

1.5 Manfaat Penelitian

Manfaat yang nantinya dapat diambil dari pengembangan sistem ini adalah dapat memberi kemudahan bagi mahasiswa dalam melaksanakan bimbingan skripsi. Memberi kemudahan bagi dosen dalam memantau perkembangan skripsi mahasiswa yang dibimbingnya. Serta dapat memberi kemudahan bagi mahasiswa dalam menyimpan dokumen skripsi secara *online*.

1.6 Sistematika Penulisan

Untuk mencapai tujuan yang diharapkan, maka sistematika penulisan yang disusun dalam tugas akhir ini adalah sebagai berikut:

BAB I Pendahuluan

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

BAB II Tinjauan Pustaka

Tinjauan pustaka menjelaskan tentang kajian pustaka dan dasar teori yang mendasari pengembangan sistem informasi bimbingan skripsi menggunakan fitur penyimpanan berkas berbasis *cloud*.

BAB III Metodologi Penelitian

Membahas tentang metode yang digunakan dalam penulisan yang terdiri dari studi literatur, analisa kebutuhan sistem, perancangan sistem, implementasi sistem, pengujian dan analisis, serta pengambilan kesimpulan.

BAB V Analisis dan Perancangan

Membahas tentang analisis kebutuhan dari sistem informasi bimbingan skripsi menggunakan fitur penyimpanan berkas berbasis *cloud* kemudian merancang hal-hal yang berhubungan dengan analisa tersebut.

BAB V Implementasi

Membahas tentang implementasi sistem informasi bimbingan skripsi menggunakan fitur penyimpanan berkas berbasis *cloud*.

BAB VI Pengujian dan Analisis

Memuat tentang hasil pengujian dan analisis terhadap sistem yang telah direalisasikan, yaitu sistem informasi bimbingan skripsi menggunakan fitur penyimpanan berkas berbasis *cloud*.

Bab VII Penutup

Memuat kesimpulan yang diperoleh dari pembuatan dan pengujian terhadap sistem informasi bimbingan skripsi menggunakan fitur penyimpanan berkas berbasis *cloud*, serta saran–saran untuk pengembangan lebih lanjut.

BAB II
DASAR TEORI

Pada bab ini berisi tinjauan pustaka dan dasar teori yang diperlukan untuk penelitian. Kajian pustaka membahas penelitian yang dilakukan sebelumnya. Sedangkan dasar teori membahas teori yang diperlukan untuk menyusun penelitian yang diusulkan.

2.1 Kajian Pustaka

Beberapa penelitian yang berkaitan dengan sistem untuk melakukan bimbingan skripsi secara dan teknologi *cloud storage* telah dilakukan oleh Aprillita Dwiyani dan Eletta Adeola O.

Bimbingan merupakan petunjuk (penjelasan) cara mengerjakan sesuatu [KBB-15]. Sedangkan skripsi adalah istilah yang digunakan di Indonesia untuk mengilustrasikan suatu karya tulis ilmiah berupa paparan tulisan hasil penelitian sarjana S1 yang membahas suatu permasalahan/fenomena dalam bidang ilmu tertentu dengan menggunakan kaidah-kaidah yang berlaku [WIK-15]. Sehingga bimbingan skripsi dapat diartikan sebagai petunjuk atau penjelasan yang diberikan oleh dosen pembimbing untuk membantu mahasiswa dalam proses pengerjaan skripsi.

Penelitian pertama dilakukan oleh Aprillita Dwiyani dengan judul Perancangan Sistem Pendukung Bimbingan *Online* Tugas Akhir Mahasiswa Program Studi Teknik Informatika. Penelitian tersebut membahas tentang perancangan sistem pendukung bimbingan skripsi *online*, dengan menerapkan teknologi informasi sehingga dapat bekerja layaknya bimbingan secara tatap muka dengan perantara sistem. Tujuan penelitian tersebut adalah untuk mengatasi kendala dari pelaksanaan bimbingan skripsi yang masih dilakukan dengan cara tatap muka sesuai dengan jadwal bimbingan dari dosen pembimbing. Implementasi sistem berbasis *web*, menggunakan bahasa pemrogramana PHP dan basis data MySQL. Penelitian ini membuktikan bahwa sistem dapat membantu proses pelaksanaan bimbingan tugas akhir antara dosen dan mahasiswa yang dibimbingnya. [APR-15]

Penelitian selanjutnya dilakukan oleh Eletta Adeola O dengan judul *Cloud Based Document Storage System for a Higher Institution*. Penelitian tersebut membahas tentang perancangan sistem penyimpanan dokumen berbasis *cloud* untuk diimplementasikan pada suatu institusi. Tujuan penelitian adalah untuk membangun sistem yang dapat digunakan untuk menyimpan *file* dan dokumen secara *online* dan mudah diakses kapanpun. Implementasi sistem berbasis *web*, menggunakan bahasa pemrograman PHP, basis data MySQL, dan konsep MVC (*Model-View-Controller*). Penelitian ini membuktikan bahwa sistem dapat menyediakan suatu ruang penyimpanan dokumen yang dapat diakses secara *online* melalui semua *web browser* dari berbagai *device* baik *personal computer* maupun *mobile*.[ADE-14]

Sistem yang akan dibangun oleh penulis nantinya merupakan sistem informasi bimbingan skripsi yang dilengkapi dengan fitur penyimpanan berkas berbasis *cloud*. Tujuan utama dari sistem tersebut adalah untuk membantu mahasiswa untuk melaksanakan bimbingan skripsi secara *online*. Dan dengan adanya fitur penyimpanan berkas berbasis *cloud*, diharapkan dokumen skripsi mahasiswa akan lebih terorganisir dengan baik karena dokumen yang telah di-*upload* dan direvisi akan tersimpan secara sistematis di dalam sistem. Sistem yang dibangun berbasis *web*, menggunakan bahasa pemrograman PHP dengan *framework* CodeIgniter dan basis data MySQL.

2.2 Sistem Informasi

Sistem informasi adalah suatu sistem di dalam suatu oranisasi yang mempertemukan kebutuhan pengelolaan transaksi harian, mendukung operasi, bersifat manajerial, dan kegiatan strategi dari organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang dibutuhkan [HUT-14]. Dalam arti yang sangat luas, istilah sistem informasi yang sering digunakan merujuk kepada interaksi antara orang, proses algoritmik, data, dan teknologi. Dalam pengertian ini, istilah ini digunakan untuk merujuk tidak hanya pada penggunaan organisasi teknologi informasi dan komunikasi (TIK), tetapi juga untuk cara dimana orang berinteraksi dengan teknologi ini dalam mendukung proses bisnis. [SIS-15]

2.3 Perangkat Lunak

Perangkat lunak dapat diartikan sebagai [PRE-01:6]:

1. Kumpulan instruksi (program komputer) yang jika dieksekusi akan menyediakan fungsi dan layanan yang diinginkan.
2. Kumpulan struktur data yang memungkinkan program untuk memanipulasi informasi secukupnya.
3. Kumpulan dokumen yang menggambarkan operasi dan penggunaan program.

2.4 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak adalah sebuah disiplin ilmu yang membahas semua aspek produksi perangkat lunak dari tahap awal spesifikasi sistem sampai pemeliharaan sistem setelah memasuki tahap penggunaan. Pada definisi ini, ada dua istilah kunci [SOM-03:7]:

1. ‘disiplin rekayasa’ Perekayasaan membuat suatu alat bekerja. Mereka menerapkan teori, metode, dan alat bantu yang sesuai, selain itu mereka menggunakannya dengan selektif dan selalu mencoba mencari solusi terhadap permasalahn, walaupun tidak ada teori dan metode yang mendukung. Perekayasa juga menyadari bahwa mereka harus bekerja dalam batasan organisasi dan keuangan, sehingga mereka berusaha mencari solusi dalam batasan-batasan ini.
2. ‘semua aspek produksi perangkat lunak’ Rekayasa perangkat lunak tidak hanya berhubungan dengan proses teknis dari pengembangan perangkat lunak tetapi juga dengan kegiatan seperti manajemen proyek perangkat lunak dan pengembangan alat bantu, metode, dan teori untuk mendukung produksi perangkat lunak.

Secara umum, perekayasa perangkat lunak memakai pendekatan yang sistematis dan terorganisir terhadap pekerjaan mereka karena cara ini seringkali paling efektif untuk menghasilkan perangkat lunak berkualitas tinggi. Namun demikian, rekayasa ini sebenarnya mencakup masalah pemilihan metode yang paling

sesuai untuk satu set keadaan dan pendekatan yang lebih kreatif, informal terhadap pengembangan yang mungkin efektif pada beberapa keadaan.

Pendekata sistematis yang digunakan dalam rekayasa perangkat lunak seringkali disebut dengan istilah proses perangkat lunak. Proses perangkat lunak adalah urutan kegiatan untuk menghasilkan suatu produk perangkat lunak. Ada empat kegiatan dasar yang umum untuk semua proses perangkat lunak. Kegiatan ini adalah [SOM-03:8]:

1. Spesifikasi perangkat lunak, fungsionalitas perangkat lunak dan batasan kemampuan operasinya harus didefinisikan.
2. Pengembangan perangkat lunak, perangkat lunak yang memenuhi spesifikasi tersebut harus diproduksi.
3. Validasi perangkat lunak, perangkat lunak harus divalidasi untuk menjamin bahwa perangkat lunak melakukan apa yang diinginkan oleh pelanggan.
4. Evolusi perangkat lunak, perangkat lunak harus berkembang untuk memenuhi kebutuhan pelanggan yang berubah-ubah.

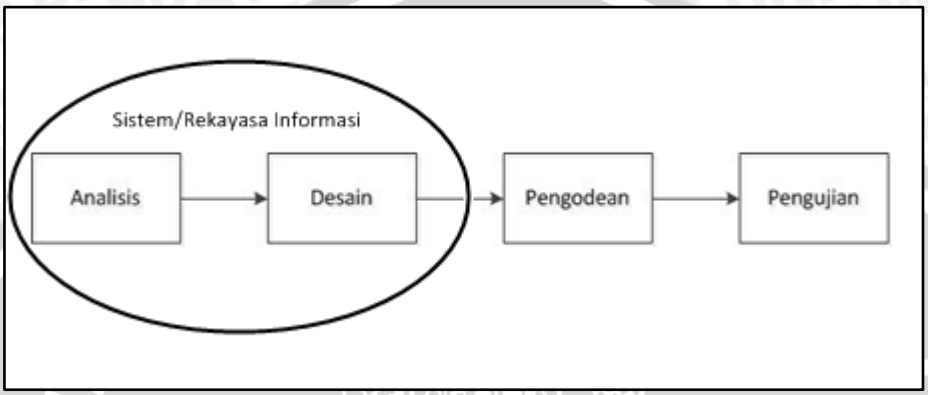
2.5 Model SDLC (*System development life cycle*)

SDLC (*Systems Development Life Cycle*) atau siklus hidup pengembangan sistem adalah proses atau tahapan yang dilakukan oleh suatu organisasi dalam mengembangkan sebuah proyek perangkat lunak. Tahapan-tahapan tersebut terdiri dari rencana pengembangan secara rinci, pemeliharaan, dan perbaikan atau peningkatan perangkat lunak. Siklus hidup mendefinisikan metodologi untuk meningkatkan kualitas perangkat lunak dan proses pembangunan secara keseluruhan [SDL-15:01]. SDLC memiliki beberapa model dalam penerapan tahapan prosesnya, salah satunya adalah model *waterfall*.

2.6 Model *Waterfall*

Model *waterfall* merupakan model proses dalam rekayasa perangkat lunak pertama yang diperkenalkan. Hal ini juga disebut sebagai *sequential* model siklus hidup *linear*. *Waterfall* merupakan model yang sangat sederhana untuk dipahami dan

digunakan. Dalam model *waterfall*, setiap fase harus diselesaikan sebelum tahap berikutnya dapat dimulai dan tidak ada tumpang tindih dalam fase [SDL-15:4]. Gambar 2.1 adalah ilustrasi model *waterfall*.



Gambar 2.1 Ilustrasi Model *Waterfall*

Sumber: [ROS-14:29]

2.6.1 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. [ROS-14:29]

Pengumpulan data pada proses analisis kebutuhan dapat dilakukan melalui teknik wawancara, observasi, dan kuisioner [ROS-14:17]. Selanjutnya kebutuhan yang telah dikumpulkan dikelompokkan menjadi kebutuhan fungsional dan kebutuhan non-fungsional. Kebutuhan fungsional adalah jenis kebutuhan yang berisi proses-proses apa saja yang nantinya dilakukan oleh sistem. sedangkan kebutuhan non fungsional adalah tipe kebutuhan yang berisi properti perilaku yang dimiliki oleh sistem. [ALF-07:63]

2.6.2 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. [ROS-14:29]

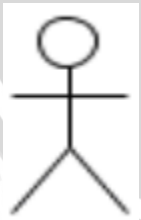
Desain dapat dilakukan dengan menggunakan pemodelan UML (*Unified Modelling Language)*. UML adalah sekumpulan simbol dan diagram untuk memodelkan *software*. Dengan menggunakan UML, desain *software* dapat diwujudkan dalam bentuk simbol dan diagram, kemudian diterjemahkan dalam kode program. Implementasi kode program dari diagram UML dapat menggunakan berbagai bahasa pemrograman dengan syarat bahasa pemrograman tersebut mendukung pemrograman berorientasi objek (OOP). [AZI-08:116]

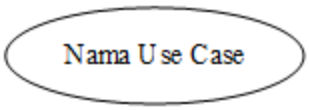
2.6.2.1 Diagram Use Case Diagram

Diagram *Use case* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem dengan menekankan tentang "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* digunakan untuk memodelkan bisnis proses berdasarkan prespektif pengguna sistem. *Use case* diagram terdiri atas diagram untuk *use case* dan aktor [AZI-08:118].

Sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu [DHA-03:4]. Keterangan simbol - simbol *use case diagram* ditunjukkan pada tabel 2.1.

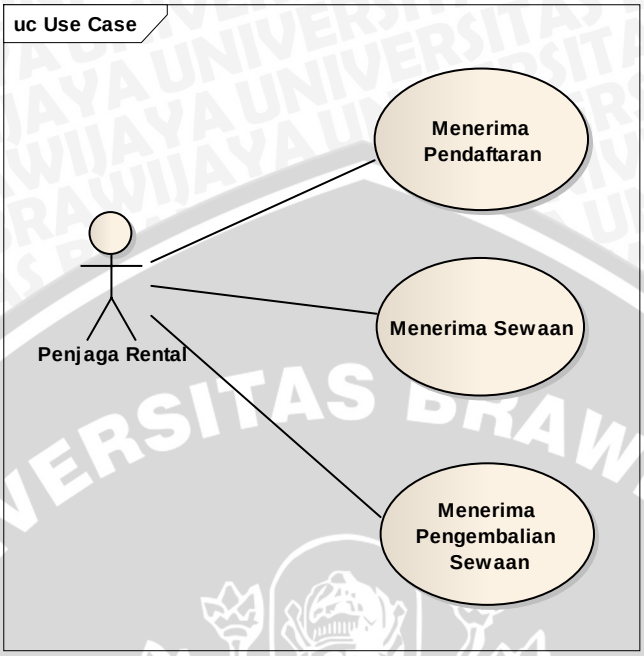
Tabel 2.1 Keterangan simbol – simbol Diagram *Use Case*

No	Simbol	Deskripsi
1	Aktor 	Orang, proses, atau sistem lain yang berinteraksi dengan si stem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
2	*Use Case*	Fungsionalitas yang disediakan sistem

		sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*.
3	Asosiasi/*Assosiation* ———————	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interksi dengan aktor.
4	Ekstensi/*Extend* - - - -<<extend>>- - - ->	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
5	Generalisasi ——————▷	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
6	*Include* — —<<include>>— —>	*Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan.

Sumber : [ROS-14 :130]

Contoh lengkap dari diagram use case ditunjukkan oleh gambar 2.2



Gambar 2.2 Contoh Lengkap dari Diagram *Use Case*

Sumber: [HER-04:40]

Setiap *use case* dilengkapi dengan skenario. Skenario *use case* adalah alur jalannya proses *use case* dari sisi aktor dan sistem. Tabel 2.2 berikut adalah format tabel skenario *use case*:

Tabel 2.2 Skenario *Use Case Diagram*

Aksi Aktor	Reaksi Sistem
Skenario Normal	
Skenario Alternatif	

Sumber : [ROS-14:161]

Skenario *use case* dibuat per *use case* terkecil, misalkan untuk generalisasi maka skenario yang dibuat adalah use case yang lebih khusus. Skenario normal adalah skenario bila sistem berjalan normal tanpa mengalami kesalahan atau *error*. Sedangkan skenario alternatif adalah skenario bila sistem tidak berjalan normal atau

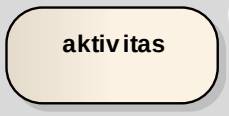
mengalami *error*. Skenario normal dan skenario alternatif dapat lebih dari satu. [ROS-14 :161]

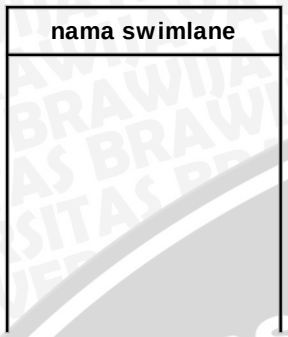
2.6.2.2 Diagram *Activity*

Diagram *Activity* adalah sebuah diagram alur kerja yang menjelaskan berbagai kegiatan pengguna (sistem), orang yang melakukan masing masing aktivitas, dan aliran sekuensial dari aktivitas-aktivitas tersebut [TRI-12:37].

Simbol-simbol yang digunakan dalam *activity* diagram ditunjukkan oleh tabel 2.3

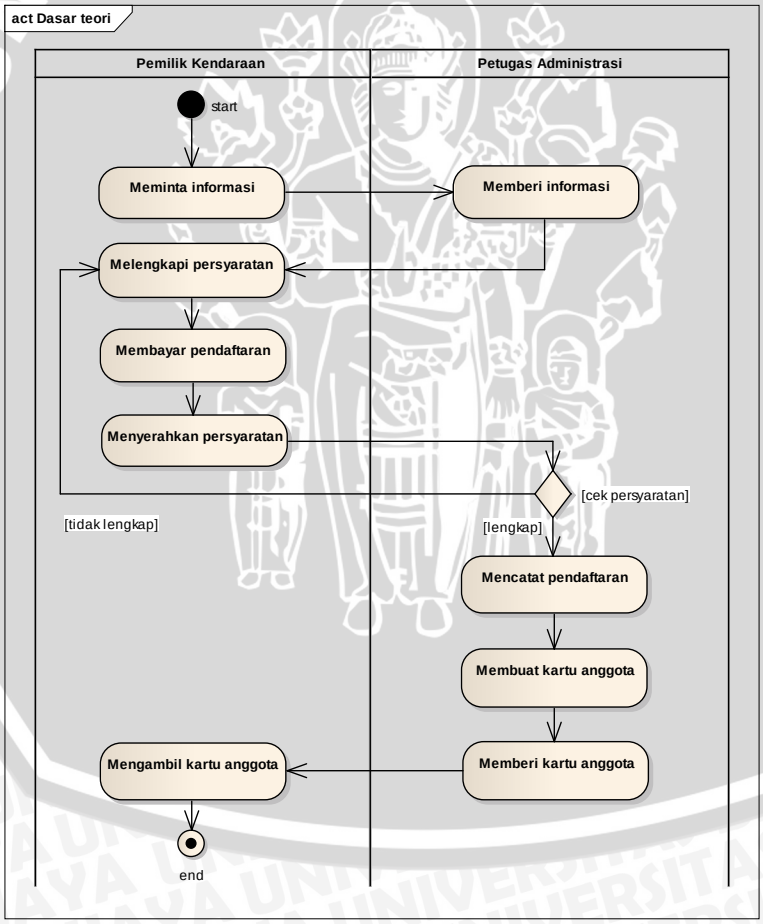
Tabel 2.3 Keterangan simbol - simbol diagram *activity*

No	Simbol	Deskripsi
1	Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2	Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3	Percabangan/decision 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4	Penggabungan/join	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5	Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

6	Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Sumber : [ROS-14 :162-163]

Contoh diagram activity ditunjukkan oleh gambar 2.3.



Gambar 2.3 Contoh Diagram *Activity*

Sumber: [SUM-15:4]

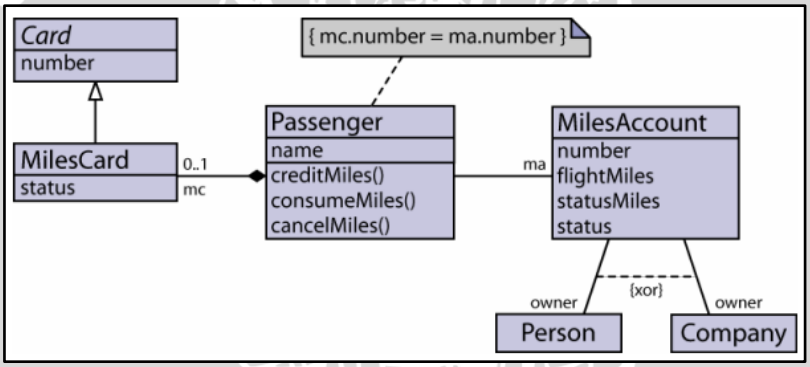
2.6.2.3 Diagram *Class*

Diagram *Class* menggambarkan *class* dan hubungan antar *class* di dalam sistem. Diagram *Class* dibangun berdasarkan diagram *use case* dan diagram *sequence* yang telah dibuat sebelumnya.

Class memiliki tiga area pokok [DHA-03:5]:

1. Nama (dan *stereotype*)
2. Atribut
3. Metoda

Class digambarkan dengan sebuah kotak dibagi menjadi tiga bagian. Bagian paling atas diisikan nama *class*, bagian tengah diisikan variabel yang dimiliki *class*, dan bagian bawah diisikan *method-method class* [AZI-08:123]. Contoh diagram class ditunjukkan oleh gambar 2.4.

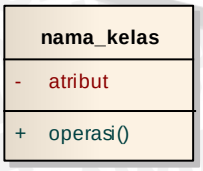


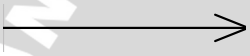
Gambar 2.4 Contoh Diagram *Class*

Sumber [KNA-17]

Berikut adalah simbol-simbol yang ada pada diagram class:

Tabel 2.4 Keterangan simbol – simbol Diagram *Class*

No	Simbol	Deskripsi
1	Kelas 	Kelas pada struktur sistem

2	Asosiasi/association	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disebut dengan *multiplicity*
3	Asosiasi berarah /directed association 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
4	Generalisasi 	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum khusus)
5	Kebergantungan/ dependency 	Relasi antarkelas dengan makna kebergantungan antarkelas
6	Agregasi/agregation 	Relasi antarkelas dengan makna semua-bagian (*whole-part*)

Sumber : [ROS-14 :146-147]

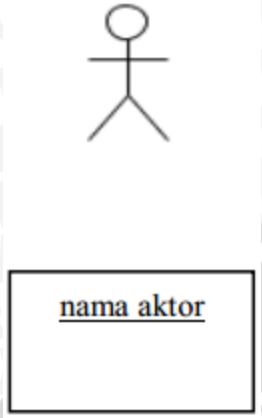
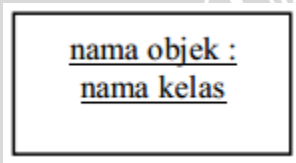
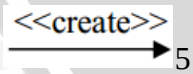
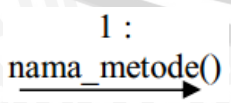
2.6.2.4 Diagram *Sequence*

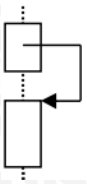
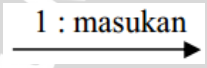
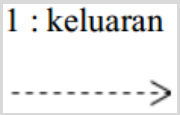
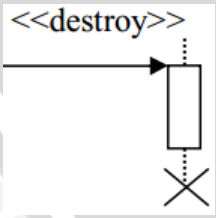
Diagram *Sequence* menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait) [DHA-03:7].

Diagram *Sequence* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai *respons* dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan. Keterangan simbol - simbol *squence diagram* ditunjukkan pada tabel 2.5.

Tabel 2.5 Keterangan simbol - simbol *Sequence Diagram*

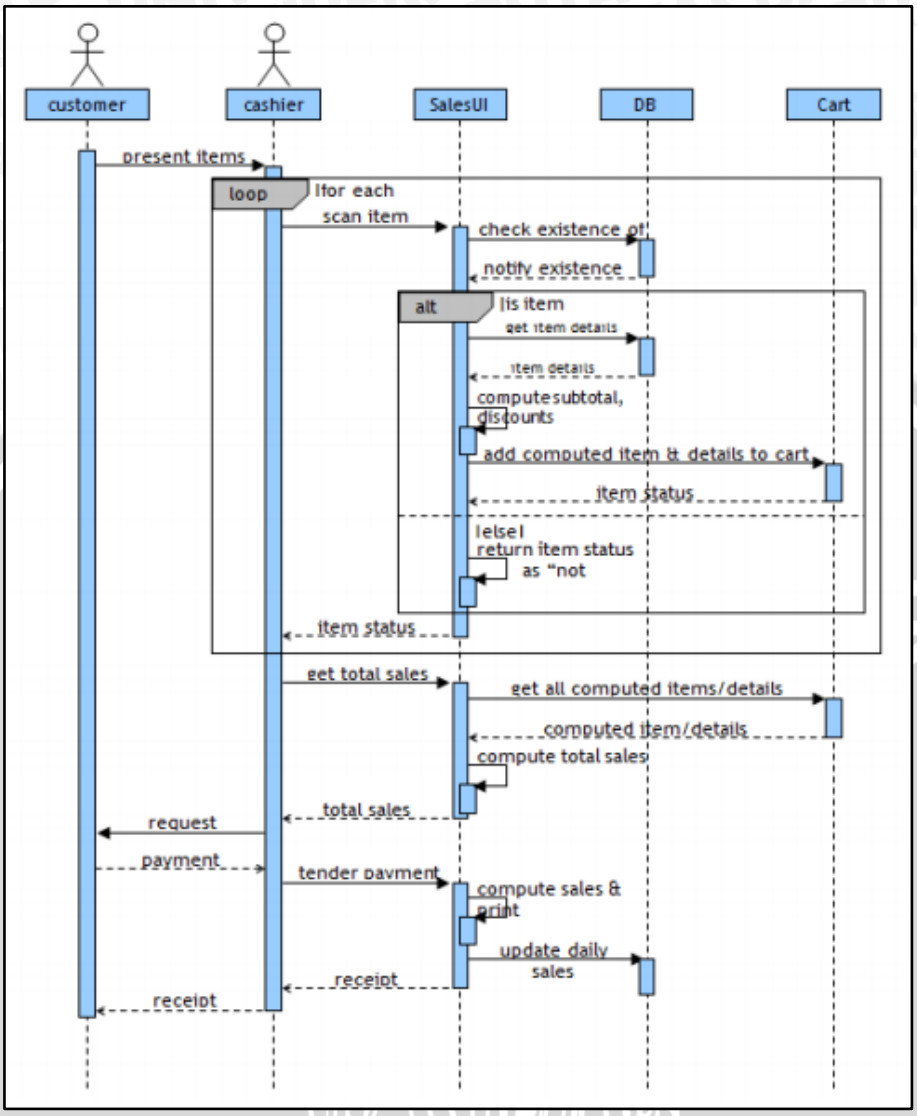
No	Simbol	Deskripsi
1	Aktor/*Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang

		akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama *actor*.
2	Garis hidup/*Lifeline* 	Menyatakan kehidupan suatu objek.
3	Objek/*Object* 	Menyatakan objek yang berinteraksi pesan.
4	Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.
5	Pesan tipe *create*  5	Menyatakan suatu objek membuat objek lain, arah panah mengarah pada objek yang dibuat.
6	Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri.

		 1 : nama_metode() Arah panah mengarah pada objek yang memiliki operasi atau metode, karena ini memanggil operasi atau metode maka operasi atau metode yang dipanggil harus ada pada diagram kelas sesuai dengan objek yang berinteraksi.
7	Pesan tipe *send*  1 : masukan	Menyatakan bahwa suatu objek mengirimkan data masukan atau informasi ke objek lainnya, arah panah mengarah pada objek yang akan dikirim.
8	Pesan tipe *return*  1 : keluaran	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9	Pesan tipe *destroy*  <<destroy>>	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy*.

Sumber : [ROS-14 :138]

Contoh diagram sequence ditunjukkan oleh gambar 2.5.



Gambar 2.5 Contoh diagram *sequence*

Sumber [LEE-12:161]

2.6.3 Pengodean

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain. [ROS-14:29]

2.6.4 Pengujian

Pengujian adalah proses pemeriksaan atau evaluasi sistem atau komponen sistem secara manual atau otomatis untuk memverifikasi apakah sistem memenuhi

kebutuhan-kebutuhan yang dispesifikasikan atau mengidentifikasi perbedaan-perbedaan antara hasil yang diharapkan dengan hasil yang terjadi.

Sebelum perangkat lunak diserahkan ke pemilik, perangkat lunak harus dievaluasi untuk menjamin telah memenuhi kebutuhan-kebutuhan fungsional dan non fungsional. Dokumen spesifikasi kebutuhan yang ditulis bagus menyatakan kebutuhan-kebutuhan itu secara eksplisit. Pengujian seharusnya meliputi tiga konsep berikut:

1. Demonstrasi validitas perangkat lunak pada masing-masing tahap di siklus pengembangan sistem
2. Penentuan validitas sistem akhir dikaitkan dengan kebutuhan pemakai
3. Pemeriksaan perilaku sistem dengan mengeksekusi sistem pada data *sample* pengujian.

Awalnya, pengujian diartikan sebagai aktivitas yang dapat/hanya dilakukan setelah pengkodean (kode program selesai). Namun pengujian seharusnya dilakukan dalam skala lebih luas. Pengujian dapat dilakukan begitu spesifikasi kebutuhan telah dapat didefinisikan. Evaluasi terhadap sepsifikasi dan perancangan juga merupakan teknik di pengujian. [HAR-04:569]

2.6.4.1 *Black Box Testing*

Konsep kotak hitam digunakan untuk merepresentasikan sistem yang cara kerja di dalamnya tidak tersedia untuk diinspeksi. Di dalam kotak hitam item-item yang diuji dianggap “gelap” karena logikanya tidak diketahui, yang diketahui hanya apa yang masuk dan apa yang keluar dari kotak hitam.

Pada pengujian kotak hitam, kasus-kasus pengujian berdasarkan pada spesifikasi sistem. Rencana pengujian dapat dimulai sedini mungkin di proses pengembangan perangkat lunak. Pada pengujian *black box*, kita mencoba beragam masukan dan memeriksa keluaran yang dihasilkan. Kita dapat mempelajari apa yang dilakukan kotak, tapi tidak mengetahui sama sekali mengenai cara konversi dilakukan.

Teknik pengujian *black box* juga dapat digunakan untuk pengujian berbasis skenario, dimana isi dalam sistem mungkin tidak tersedia untuk diinspeksi tapi masukan dan keluaran yang didefinisikan dengan *use case* dan informasi analisis yang lain. [HAR-04:577]

2.6.4.2 *White Box Testing*

White box testing secara umum merupakan jenis *testing* yang lebih berkosentrasi terhadap "isi" dari perangkat lunak itu sendiri. Jenis ini lebih banyak berkosentrasi kepada *source code* dari perangkat lunak yang dibuat sehingga membutuhkan proses testing yang jauh lebih lama dan "mahal" dikarenakan membutuhkan ketelitian dari para *tester* serta kemampuan teknis pemrograman bagi para *tester*-nya. [RIZ-11:261]

Pengujian *white box* mengasumsikan bahwa logik spesifik adalah penting dan harus diuji untuk menjamin sistem melakukan fungsi dengan benar. Penggunaan utama *white box* adalah pengujian berbasis kesalahan ketika kita siap menguji semua objek di aplikasi dan semua metode eksternal. Pada pengujian *white box*, kita mencari bug yang mempunyai peluang eksekusi yang rendah, atau yang diimplementasikan secara sembrono. [HAR-04:578]

2.6.4.3 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. [YUS-15]

2.7 Konsep Basis data

2.7.1 Definisi Basis Data

Database/basis data terdiri dari dua suku kata, yaitu data dan *base*/basis. Data dapat diartikan sebagai representasi fakta dunia nyata yang mewakili suatu objek, misalnya manusia, hewan, barang, peristiwa, konsep yang direkam dalam bentuk huruf, teks, simbol, angka, suara, gambar, dan lain sebagainya. Sedangkan basis/*base* dapat diartikan sebagai tempat berkumpul, sarang atau gudang untuk menyimpan

sesuatu. Dengan demikian, basis data dapat diartikan sebagai tempat berkumpul/menyimpan data-data suatu benda atau kejadian yang saling berhubungan.

Database merupakan kumpulan *file*, tabel, arsip yang saling berhubungan yang disimpan dalam media elektronis. Data yang disimpan tersebut tidak dibiarkan begitu saja, namun dikelola dan diorganisir oleh suatu perangkat lunak yang dikenal dengan *Database Management System* (DBMS). Dengan demikian data yang tersimpan di dalam *database* dapat disusun dengan rapi dan terstruktur sehingga data tersebut dapat dimanipulasi, diambil, dan dicari secara cepat.

Database sangat bermanfaat untuk mengatasi berbagai masalah yang sering terjadi dalam penyusunan data. Masalah dalam penyusunan data dapat berupa redudansi dan inkonsistensi data, kesulitan pengaksesan data, dan lain lain. [WAH-10:140]

Secara prinsip, dalam suatu database tercakup dua komponen penting yaitu data dan informasi. Jadi tujuan akhir dari suatu database adalah bagaimana mengelola data sehingga mampu menjadi informasi yang diinginkan. Selain itu proses pengambilan, penghapusan, dan pengeditan terhadap data dapat dilakukan secara mudah dan cepat. [YUH-08:2]

2.7.2 Normalisasi

Secara umum sasaran perancangan basis data adalah untuk menghasilkan himpunan skema relasi yang mengizinkan pengguna untuk menyimpan informasi-informasi tanpa redudansi yang tidak dikehendaki (meminimalkan redudansi dan meningkatkan derajat konsistensi data dengan pemeliharaan integritas data) serta mengizinkan pengguna untuk mencari informasi yang dikehendaki dengan cara yang mudah. Salah satu pendekatan yang dapat digunakan adalah perancangan relasi-relasi menjadi bentuk normal (normal *form*). [NGH-11: 188-189]

Normalisasi dapat dipahami sebagai tahapan-tahapan yang masing-masing berhubungan dengan bentuk normal. Bentuk normal adalah kedaan relasi yang dihasilkan dengan menerapkan aturan sederhana berkaitan dengan berkaitan dengan

konsep kebergantungan fungsional pada relasi yang bersangkutan. Garis besar normalisasi adalah berikut [NGH-11: 201]:

1. Bentuk normal pertama (1NF/*First Normal Form*)

   Bentuk normal pertama adalah suatu bentuk relasi dimana atribut bernilai banyak (*multivalues atribute*) telah dihilangkan sehingga kita akan menjumpai nilai tunggal (mungkin saja nilai *null*) pada pada setiap perpotongan garis dan kolom pada tabel.

2. Bentuk normal kedua (2NF/*First Normal Form*)

   Semua kebergantungan fungsional yang bersifat sebagian (*partial functional dependency*) telah dihilangkan.

3. Bentuk normal ketiga (3NF/*First Normal Form*)

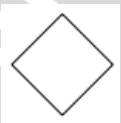
   Semua kebergantungan transitif (*transitive dependency*) telah dihilangkan.

2.8 BPMN (*Business Process Modeling Notation*)

Business Process Management Initiative (BPMI) telah mengembangkan BPMN. Tujuan utama dari BPMN adalah menyediakan notasi yang mudah dimengerti oleh semua pengguna bisnis, mulai dari analis bisnis yang menciptakan konsep awal proses, para pengembang teknis yang bertanggung jawab untuk melaksanakan teknologi pada proses-proses tersebut, sampai pada orang-orang bisnis yang akan mengelola dan memantau proses. Dengan demikian, BPMN menciptakan jembatan standar untuk kesenjangan antara desain proses bisnis dan proses implementasi. Notasi-notasi yang digunakan pada BPMN adalah sebagai berikut:

Tabel 2.6 Keterangan simbol – simbol BPMN

No	Notasi	Deskripsi
1	*Start* 	Sesuai dengan namanya, *start event* mengindikasikan bahwa proses akan dimulai.
2	*Intermediate*	*Intermediate event* terdapat diantara *start event* dan *end event*, yang akan

		mempengaruhi aliran proses tetapi tidak memulai atau (secara langsung) mengakhiri proses.
3	*End* 	Sesuai dengan namanya, *end event* mengindikasikan bahwa proses akan berhenti.
4	Activity 	*Activity* adalah istilah umum untuk suatu pekerjaan yang diselenggarakan oleh perusahaan.
5	Gateway 	Sebuah *Gateway* digunakan dapat mengontrol perbedaan dan konvergensi urutan arus. Dengan demikian, akan menentukan bercabang, *forking*, penggabungan, dan bergabung jalur.
6	Sequence Flow 	Digunakan untuk menunjukkan urutan kegiatan yang akan dilakukan di sebuah proses.
7	Association 	Sebuah Asosiasi digunakan untuk menghubungkan informasi dengan *flow object*.
8	Pool 	*Pool* mewakili partisipan dalam sebuah proses
9	Data Object 	Data *object* dipertimbangkan sebagai *artifact* karena mereka tidak memiliki pengaruh secara langsung pada *sequence flow* ataupun *message flow* pada proses,

		akan tetapi menyediakan informasi tentang aktivitas apa yang butuh untuk dilakukan dan/atau apa yang mereka hasilkan.

[Sumber: OBJ-11:18-20]

2.9 Definisi *Cloud Computing*

Cloud computing merupakan akses layanan *on-demand* ke sekumpulan sumber daya komputasi seperti jaringan, *server*, penyimpanan, aplikasi dan layanan [MEL-11]. Komputasi awan (*cloud computing*) adalah gabungan antara pemanfaatan teknologi komputer dengan pengembangan berbasis internet. Awan (*cloud*) adalah metafora dari internet, sebagaimana awan yang sering digambarkan pada diagram jaringan komputer, awan (*cloud*) dalam *cloud computing* juga merupakan abstraksi dari infrastruktur kompleks yang disembunyikan. [WAL-12:1]

Cloud computing menerapkan suatu metode komputasi, yaitu kapabilitas yang terkait teknologi informasi yang disajikan sebagai suatu layanan atau *service* sehingga pengguna dapat mengaksesnya lewat internet, tanpa mengetahui apa yang ada di dalamnya, ahli dengannya, atau memiliki kendali terhadap infrastruktur teknologi yang membantunya. [WAL-12:1]

2.10 *Cloud Storage*

2.10.1 Definisi *Cloud Storage*

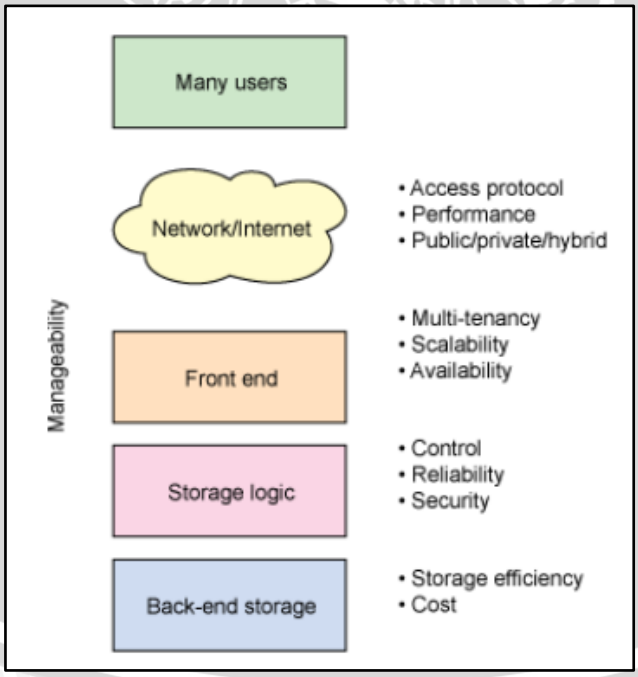
Cloud storage adalah layanan penyimpanan berbasis *web*. Saat ini *cloud storage* mulai populer digunakan sebagai penyimpanan *backup*. Salah satu keuntungan menggunakan penyimpanan *cloud* adalah kita dapat mengakses file dari komputer maupun *smartphone*. *Cloud storage* dapat diakses dimanapun menggunakan koneksi internet [CAN-15].

Cloud Storage merupakan media penyimpanan data yang dapat diakses oleh para penggunanya lewat jaringan internet. Untuk dapat mengakses data, para pengguna akan dihubungkan dengan *server* melalui halaman *web*. [ELC-12:11]

2.10.2 Arsitektur *Cloud Storage*

Arsitektur *cloud storage* terutama tentang bagaimana menyediakan media penyimpanan secara *on demand* dengan skalabilitas tinggi dan bersifat *multi-tenant*. Secara umum arsitektur *cloud storage* terdiri dari *front end* yang memiliki API untuk mengakses penyimpanan. Pada sistem penyimpanan tradisional, API dapat berupa protokol SCSI, tapi pada sistem *cloud* protokol ini berkembang. Dalam hal ini dapat menggunakan *web service*, *file based*, dan beberapa tradisional *front end* (seperti SCSI atau iSCSI). Di belakang *front end* adalah layer *middleware* yang disebut *storage logic*. Layer ini menerapkan berbagai fitur, seperti replikasi dan pengurangan data. Dan yang terakhir adalah *back end* yang menerapkan penyimpanan fisik untuk data. Ini dapat berupa protokol internal yang menerapkan fitur tertentu atau tradisional *back end* ke *physical disk*. [JON-10:2]

Arsitektur *cloud storage* dapat dilihat pada gambar 2.6.



Gambar 2.6 Arsitektur *cloud storage*

Sumber: [JON-10:2]

Dari gambar 2.6 dapat dilihat beberapa karakteristik untuk arsitektur *cloud storage* yang didefinisikan pada tabel 2.7

Table 2.7 Karakteristik *cloud storage*

Karakteristik	Deskripsi
Manageability	Kemampuan untuk mengatur sistem dengan *resource* minimal
Access method	Perbedaan utama antara *cloud storage* dengan tadisional *storage* terlatak pada metode akses. Sebagian besar *provider* menerapkan lebih dari satu metode akses, tetapi yang paling umum adalah *web service* API. Sebagian besar API diimplementasikan berdasarkan prinsip REST, yang dibangun diatas protokol HTTP.
Performance	Performance dapat diukur melalui *bandwith* dan *latency*.
Multi-tenancy	Mendukung *multiple user* (atau *tenants*)
Scalability	Kemampuan sistem untuk dikembangkan
Data availability	Menyediakan data kepada *user* kapanpun *request* dilakukan
Control	Kemampuan untuk melakukan kontrol terhadap sistem, untuk melakukan konfigurasi harga, *performance*, atau karakteristik lain.
Storage eficiency	Mengukur efisiensi penggunaan *raw storage*. Solusi yang umum adalah melakukan *reduce* terhadap *source*

	data sehingga membutuhkan lebih sedikit ruang penyimpanan.
Cost	Megukur besar biaya (harga) untuk ruang penyimpanan yang digunakan.

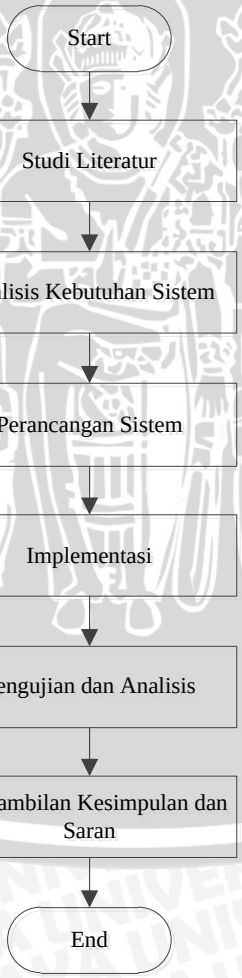
Sumber: [JON-10:2-3]

BAB III
METODOLOGI PENELITIAN

Bab ini menjelaskan metode penelitian yang digunakan pada sistem informasi bimbingan skripsi *online* di PTIIK. Perangkat lunak dikembangkan dengan model pengembangan perangkat lunak *waterfall*.

3.1 Metode Penelitian

Metode penelitian berisi langkah-langkah yang digunakan dalam pembuatan sistem yang terdiri dari studi literatur, analisa kebutuhan, perancangan sistem, implementasi, pengujian dan analisis, serta pengambilan kesimpulan dan saran. Diagram alir dari metode penelitian ditunjukkan oleh gambar 3.1.



Gambar 3.1 Diagram Alir Metode Penelitian

Model proses perangkat lunak yang diimplementasikan adalah model *waterfall* yang merupakan paradigma rekayasa perangkat lunak yang paling tua dan paling banyak digunakan. *Waterfall* model menggunakan pendekatan yang bersifat sistematis dan berurutan dalam membangun *software*. Pengembangan perangkat lunak dengan menggunakan model *waterfall* dibagi atas empat tahap yaitu *requirements analysis and definition, sistem and software design*, *implementation*, dan *testing*.

3.1.1 Studi Literatur

Studi literatur menjelaskan dasar teori yang digunakan sebagai sumber acuan dalam penulisan skripsi. Teori-teori tersebut meliputi:

1. Sistem Informasi
2. Perangkat Lunak
3. Rekayasa Perangkat Lunak
4. Model SDLC
5. Model *Waterfall*
6. Konsep Basis Data
    a. Definisi basis data
    b. Normalisasi
7. BPMN (*Business Process Modeling Notation*)
8. *Cloud Computing*
9. *Cloud Storage*

3.1.2 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem merupakan tahap pertama sebelum penelitian dilaksanakan. Hal tersebut dilakukan melalui kuisioner dan observasi. Proses analisis kebutuhan tersebut adalah sebagai berikut:

1. Kuisioner

   Kuisioner diberikan kepada 40 responden mahasiswa PTIIK untuk menganalisis permasalahan apa saja yang ada pada proses pelaksanaan bimbingan skripsi.

2. Observasi

   Observasi dilakukan dengan mencari informasi mengenai prosedur pelaksanaan skripsi melalui buku panduan manual prosedur skripsi di PTIIK. Pada observasi juga dilakukan pengamatan untuk mendapatkan informasi mengenai proses pelaksanaan bimbingan skripsi dan apa saja yang dibutuhkan dalam pelaksanaan bimbingan skripsi.

Setelah itu, hasil dari kuisioner dan observasi dijadikan sebagai acuan untuk melakukan analisis kebutuhan.

Proses analisis kebutuhan ini diawali dengan memahami domain permasalahan dari manual prosedur pelaksanaan bimbingan skripsi di PTIIK untuk mengetahui apa dan mengapa permasalahan tersebut terjadi. Selanjutnya dilakukan pengumpulan kebutuhan sesuai dengan permasalahan yang telah dianalisis.

Setelah daftar kebutuhan dikumpulkan tahap selanjutnya adalah mengurutkan kebutuhan sesuai dengan prioritas dan memodelkannya ke dalam diagram *use case*. Analisis kebutuhan ini bertujuan untuk menggambarkan fitur-fitur yang harus disediakan oleh sistem agar dapat memenuhi kebutuhan pengguna.

3.1.3 Perancangan Sistem

Perancangan sistem berdasarkan *object oriented design* yaitu menggunakan pemodelan UML (*Unified Modeling Language*). Perancangan sistem terdiri dari perancangan arsitektural, perancangan basis data, perancangan diagram *class*, diagram *activity*, diagram *sequence*, dan perancangan antarmuka.

Basis data yang dimiliki oleh sistem ada dua, basis data pertama yang terhubung dengan sistem informasi bimbingan skripsi dan basis data kedua yang terhubung dengan *server cloud*. *Server cloud* sendiri menyediakan API (*Application Programming Interface*) berupa REST *web service* yang nantinya akan digunakan oleh sistem informasi untuk mengakses data-data yang ada di *server cloud*.

3.1.4 Implementasi

Sistem yang dikembangkan berbasis *web* menggunakan bahasa pemrograman PHP dan *framework* CodeIgniter. Implementasi sistem informasi dilakukan pada

lingkungan sistem operasi *Windows* 7, dengan menggunakan basis data MySQL dan software XAMPP. Twitter bootstrap juga diimplementasikan untuk membuat sistem *responsive* terhadap semua *device*.

Implementasi *server cloud* dilakukan pada lingkungan sistem operasi *Windows 7*, dengan menggunakan basis data MySQL dan *server* XAMPP. Sistem informasi berinteraksi dengan *server cloud* melaui REST API menggunakan format pertukaran data JSON.

3.1.5 Pengujian dan Analisis

Metode pengujian yang dilakukan adalah *white box testing* dan *black box testing*. Pengujian *white box* dilakukan dengan menggunakan teknik *basis path testing*. Teknik pengujian ini dilakukan dengan menggunakan rancangan atau kode program sebagai dasar untuk membuat *flow graph*. Setelah itu, berdasarkan *flow graph* yang telah dibuat akan ditentukan kompleksitas siklomatik serta himpunan basis dari jalur-jalur independen secara linier.

Skenario pada pengujian *black box* dilakukan dengan menjalankan sistem sesuai dengan fungsi-fungsi internal yang akan dijabarkan pada bab analisis dan perancangan sub bab diagram *use case*. Setelah itu dilakukan pencatatan dalam bentuk tabel apakah hasil kebutuhan fungsional berhasil terpenuhi oleh sistem (valid) atau tidak (tidak valid).

Pada sistem *cloud* akan dilakukan pengujian untuk mengetahui kinerja API sistem dalam melayani *request* dan mengirim respon ke *user*. Dalam skenarionya, akan dilakukan perhitungan waktu akses dan pencatatan status kode dari HTTP *request* yang dilakukan oleh *user*.

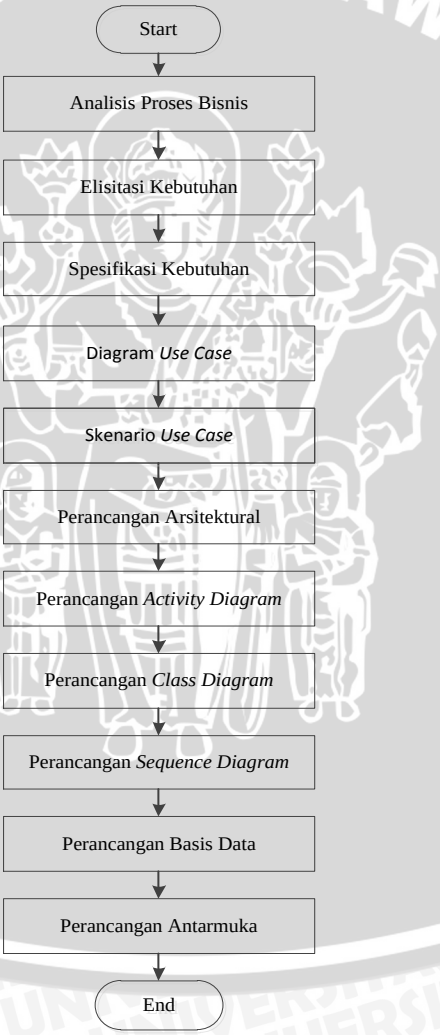
Analisis dilakukan untuk mengetahui hasil dan melakukan evaluasi dari pengujian perangkat lunak yang telah dilakukan sehingga didapatkan suatu kesimpulan dari pengembangan perangkat lunak yang dilakukan. Apabila terjadi kesalahan pada sistem maka proses kembali pada tahap implementasi.

3.1.6 Pengambilan Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan, implementasi dan pengujian sistem aplikasi telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisis terhadap sistem yang dibangun. Tahap terakhir dari penulisan adalah saran yang dimaksudkan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan perangkat lunak lebih lanjut.

BAB IV
ANALISIS DAN PERANCANGAN

Bab ini membahas mengenai analisis dan perancangan. Analisis terdiri dari analisis proses bisnis, elisitasi kebutuhan, spesifikasi kebutuhan, diagram *use case*, dan skenario *use case*. Sedangkan perancangan terdiri dari perancangan arsitektural, perancangan diagram *activity*, perancangan diagram *class*, perancangan daiagram *sequence*, perancangan basis data, dan perancangan antarmuka. Diagram alir analisis dan perancangan ditunjukkan oleh gambar 4.1.



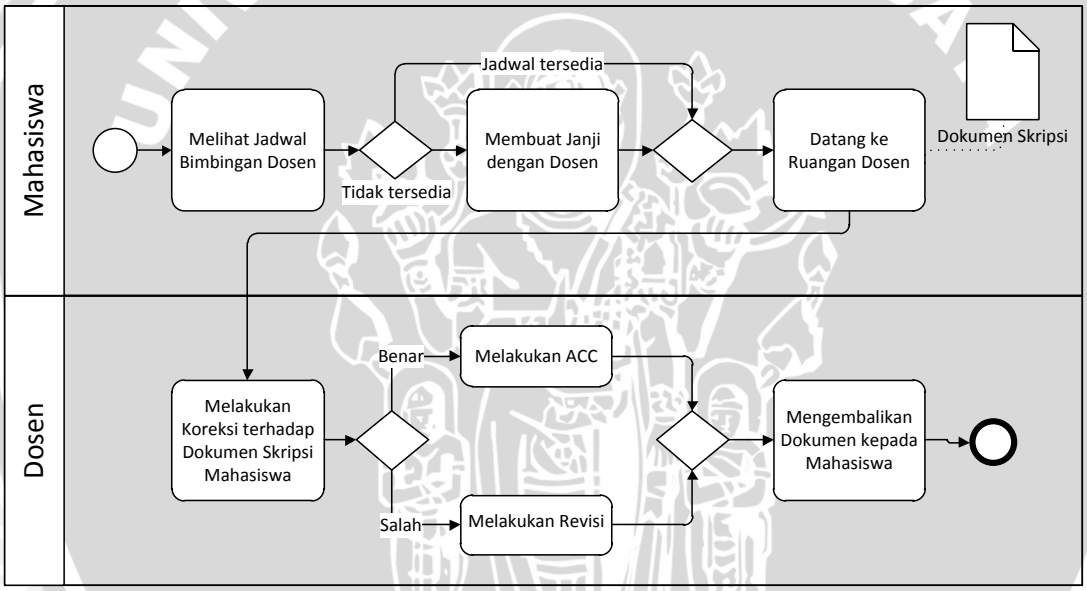
Gambar 4.1 Diagram Alir Analisis dan Perancangan

4.1 Analisis Proses Bisnis

Pada bagian ini akan di bahas mengenai proses-proses bisnis yang saat ini berjalan di PTIIK sebelum menggunakan sistem (as-is) dan proses bisnis yang berjalan setelah menggunakan sistem (to-be).

4.4.1 Proses Bisnis As-Is Bimbingan Skripsi

Analisis dilakukan melalui observasi dan studi literatur melalui buku panduan manual prosedur skripsi di PTIIK. Selanjutnya, proses bisnis tersebut digambarkan dengan menggunakan diagram *activity*. Proses bisnis as-is pada pelaksanaan bimbingan skripsi di PTIIK dapat dilihat pada gambar 4.2.



Gambar 4.2 Proses Bisnis as-is Bimbingan Skripsi

Skenarion proses bisnis as-is pada pelaksanaan bimbingan skripsi di PTIIK dapat dilihat pada tabel 4.1.

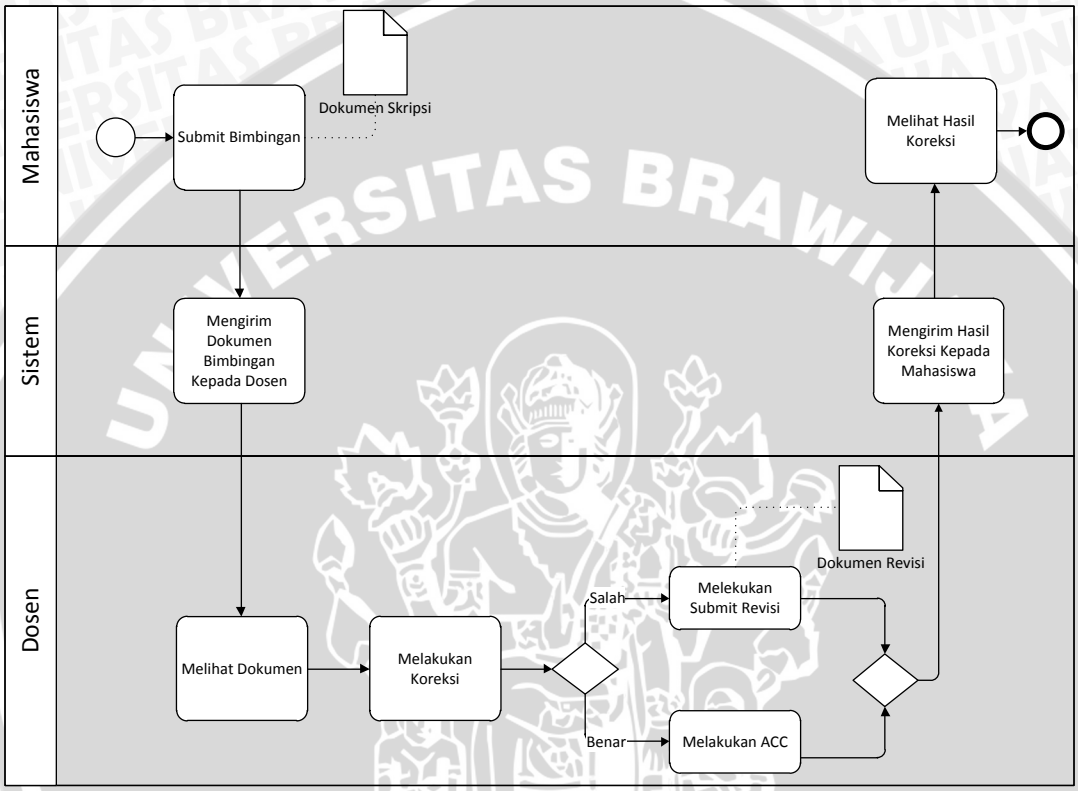
Tabel 4.1 Skenario Model Bisnis As-Is

No	Alur bisnis proses bimbingan skripsi
1.	Melihat jadwal bimbingan dosen Pada proses ini, mahasiswa melihat jadwal bimbingan dari dosen pembimbing, pada umumnya jadwal akan tertempel di depan

	ruangan dosen.
2.	Membuat janji dengan dosen Beberapa dosen tidak memiliki jadwal bimbingan khusus sehingga mahasiswa harus membuat janji terlebih dahulu ketika akan melaksanakan bimbingan.
3.	Datang ke ruangan dosen Setelah dan jam hari ditentukan, maka mahasiswa harus datang ke ruangan dosen pembimbingan untuk melaksanakan bimbingan skripsi, dengan membawa dokumen skripsi yang akan dikonsultasikan.
4.	Melakukan koreksi terhadap dokumen skripsi mahasiswa Dokumen yang telah diserahkan selanjutnya akan dikoreksi oleh dosen pembimbing untuk dicari kesalahan-kesalahan yang ada.
5.	Melakukan revisi Apabila masih terdapat kesalahan maka dilakukan revisi untuk kemudian kesalahan tersebut dibenarkan oleh mahasiswa.
6.	Melakukan acc Sedangkan apabila tidak terdapat kesalahan maka dosen pembimbing akan melakukan acc terhadap dokumen skripsi mahasiswa
7.	Mengembalikan dokumen kepada mahasiswa Dokumen skripsi yang telah dikoreksi oleh dosen pembimbing selanjutnya akan dikembalikan lagi kepada mahasiswa.

4.4.2 Proses Bisnis To-Be Bimbingan Skripsi

Proses bisnis to-be menjelaskan bagaimana proses bisnis usulan yang akan diimplementasikan pada proses bimbingan skripsi di PTIIK. Proses bisnis to-be dapat dilihat pada gambar 4.3.



Gambar 4.3 Proses Bisnis to-be

Skenarion proses bisnis to-be pada pelaksanaan bimbingan skripsi di PTIIK dapat dilihat pada tabel 4.2.

Tabel 4.2 Skenario Model Bisnis To-Be

No	Alur bisnis proses bimbingan skripsi
1.	Submit bimbingan Pada proses ini, mahasiswa melakukan *upload* dokumen skripsi yang digunakan untuk bimbingan.
2.	Mengirim dokumen bimbingan kepada dosen

	Pada proses ini, sistem akan mengirim dokumen yang telah di-*upload* oleh mahasiswa kepada dosen pembimbing.
3.	Melihat dokumen Pada proses ini, dosen akan melihat dokumen skripsi yang telah dikirim kepadanya.
4.	Melakukan koreksi Selanjutya dosen akan melakukan koreksi terhadap dokumen skripsi mahasiswa untuk mencari kesalahan yang ada.
5.	Melakukan *submit* revisi Apabila masih terdapat kesalahan maka dosen akan melakukan *submit*.
6.	Melakukan acc Sedangkan apabila tidak terdapat kesalahan maka dosen pembimbing akan melakukan acc terhadap dokumen skripsi mahasiswa
7.	Mengirim hasil koreksi kepada mahasiswa Sistem akan mengirim koreksi terhadap dokumen skripsi yang telah dilakukan oleh dosen kepada mahasiswa.
8.	Melihat hasil koreksi Mahasiswa akan melihat hasil koreksi yang telah dilakukan oleh dosen pembimbing.

4.2 Analisis Kebutuhan Sistem

Proses analisis kebutuhan dilakukan pada tahap pertama proses perancangan sistem informasi. Tahap analisis kebutuhan terdiri atas empat langkah yaitu melakukan identifikasi, membuat daftar spesifikasi kebutuhan pengguna, pemodelan kebutuhan dalam diagram *use case*, dan analisis skenario pada *use case*.

4.2.1 Elisitasi Kebutuhan

Elisitasi kebutuhan untuk sistem informasi bimbingan skripsi dilakukan dengan mendeskripsikan permasalahan sesuai dengan hasil dari kuisioner yang diberikan kepada responden. Selanjutnya dilakukan identifikasi kebutuhan sebagai solusi atas permasalahan yang ada. Elisitasi kebutuhan ditunjukkan oleh tabel 4.3.

Tabel 4.3 Elisitasi kebutuhan

No.	Identifikasi Permasalahan	Identifikasi Kebutuhan
1	Jarak dan jadwal yang berbeda antara dosen dan mahasiswa kadangkala menciptakan diskomunikasi sehingga proses bimbingan dapat terbengkalai.	Sistem harus mampu menyediakan fitur untuk melaksanakan bimbingan yang dapat diakses kapanpun dan dimanapun.
2	Pemborosan biaya yang dikeluarkan untuk mencetak dokumen skripsi secara berulang-ulang akibat adanya revisi.	Sistem harus mampu menyediakan fitur untuk bimbingan skripsi dengan cara melakukan submit *softcopy* dokumen skripsi.
3	Pencatatan laporan bimbingan kurang terorganisasi dengan baik.	Sistem harus mampu menyediakan fitur untuk melakukan pencatatan laporan bimbingan skripsi secara otomatis.
4	Dokumen skripsi mahasiswa kurang terorganisasi dengan baik.	Sistem harus mampu menyediakan fitur penyimpanan berkas secara *online*.

4.2.2 Spesifikasi Kebutuhan

Spesifikasi kebutuhan untuk sistem informasi bimbingan skripsi dilakukan dengan mendeskripsikan kebutuhan sesuai dengan hasil dari solusi permasalahan pada tahap elisitasi kebutuhan, selanjutnya daftar spesifikasi kebutuhan tersebut akan diurutkan sesuai dengan prioritas. Ketika kebutuhan tersebut menjawab permasalahan utama yang telah dideskripsikan sebelumnya, maka kebutuhan tersebut akan memiliki prioritas tinggi. Apabila kebutuhan tersebut mendukung kebutuhan utama, maka akan memiliki prioritas medium. Sedangkan prioritas rendah, ketika kebutuhan tersebut hanya mendukung sistem dan tidak berkaitan dengan kebutuhan utama.

Daftar spesifikasi kebutuhan untuk sistem informasi bimbingan skripsi diurutkan berdasarkan prioritas ditunjukkan oleh tabel 4.4.

Tabel 4.4 Spesifikasi kebutuhan fungsional sistem informasi bimbingan skripsi

Nomor SRS	Kebutuhan	*Use Case*	Aktor	Prioritas
SRS_1_01	Perangkat lunak harus mampu menyediakan fasilitas untuk *upload* dan *submit* dokumen skripsi dalam *format* .pdf.	*Submit* Dokumen Skripsi	Mahasiswa	*High*
SRS_1_02	Perangkat lunak harus mampu menyediakan fasilitas untuk melihat dokumen skripsi yang telah di-*upload* oleh mahasiswa. Kemudian menampilkan dua pilihan, untuk revisi dan untuk acc.	Melihat Dokumen Skripsi	Dosen	*High*

SRS_1_03	Perangkat lunak harus mampu menyediakan fasilitas untuk melakukan ACC atas dokumen skripsi mahasiswa	Melakukan ACC	Dosen	*High*
SRS_1_04	Perangkat lunak harus mampu menyediakan fasilitas untuk *submit* revisi atas dokumen skripsi yang telah di-*upload* oleh mahasiswa.	Submit Revisi	Dosen	*High*
SRS_1_05	Perangkat lunak harus mampu menyediakan fasilitas untuk menampilkan revisi.	Melihat Revisi	Mahasiswa	*High*
SRS_1_06	Perangkat lunak harus mampu menyediakan fasilitas untuk melakukan akses ke fitur penyimpanan *cloud*. Akses yang dapat dilakukan adalah *view*, *upload*, *delete*, *rename*, serta *share* dokumen.	Mengakses Fitur *Cloud*	Dosen dan Mahasiswa	*High*
SRS_1_07	Perangkat lunak harus mampu menyediakan	Melihat Laporan	Dosen dan Mahasiswa	*Medium*

	fasilitas untuk melihat detail laporan bimbingan yang telah dilakukan oleh mahasiswa.	Bimbingan		
SRS_1_08	Perangkat lunak harus mampu menyediakan fasilitas untuk melihat daftar mahasiswa yang melaksanakan bimbingan	Melihat Daftar Bimbingan	Dosen	*Medium*
SRS_1_09	Perangkat lunak harus mampu menyediakan fasilitas untuk melihat, edit, serta menghapus data mahasiswa dan dosen.	Kelola *User*	*Staff* Akademik	*Medium*
SRS_1_10	Perangkat lunak harus mampu menyediakan fasilitas untuk mengaktifkan akun mahasiswa. Aktivasi dilakukan dengan memasukkan nama dosen pembimbing satu dan dua, serta nama laboratorium keminatan mahasiswa. Kemudian secara	Mengaktifkan Akun Mahasiswa	*Staff* Akademik	*Medium*

	otomatis sistem akan mengirim *email* pemberitahuan kepada mahasiswa bahwa mahasiswa telah dapat melaksanakan bimbingan kepada dosen yang telah ditetapkan.			
SRS_1_11	Perangkat lunak harus mampu menyediakan fasilitas untuk mengaktifkan akun dosen. Aktivasi dilakukan dengan mengirim *email* pemberitahuan kepada dosen bahwa akun telah diaktivasi.	Mengaktifkan Akun Dosen	*Staff* Akademik	*Medium*
SRS_1_12	Perangkat lunak harus mampu menyediakan fasilitas bagi *user* untuk melakukan registrasi.	Melakukan Registrasi	*User*	*Low*
SRS_1_13	Perangkat lunak harus mampu menyediakan fasilitas bagi *user* untuk *login* ke dalam sistem.	*Login*	*User*	*Low*

SRS_1_14	Perangkat lunak harus mampu menyediakan fasilitas bagi *user* untuk melakukan mengubah *password* akun miliknya.	*Edit Password*	Dosen dan Mahasiswa	*Low*

Daftar spesifikasi kebutuhan untuk sistem penyimpanan berbasis *cloud* diurutkan berdasarkan prioritas ditunjukkan oleh tabel 4.5.

Tabel 4.5 Spesifikasi kebutuhan fungsional sistem penyimpanan *cloud*

Nomor SRS	Kebutuhan	Use Case	Aktor	Prioritas
SRS_2_01	Sistem harus menyediakan layanan untuk melihat dokumen yang tersimpan di dalam sistem.	*View* Dokumen	Sistem informasi bimbingan skripsi	*High*
SRS_2_02	Sistem harus menyediakan layanan untuk *upload* dokumen.	*Upload* Dokumen	Sistem informasi bimbingan skripsi	*High*
SRS_2_03	Sistem harus menyediakan layanan untuk menghapus dokumen.	*Hapus* Dokumen	Sistem informasi bimbingan skripsi	*Medium*
SRS_2_04	Sistem harus menyediakan layanan untuk merubah nama	*Rename* Dokumen	Sistem informasi bimbingan skripsi	*Medium*

	dokumen.			
SRS_2_05	Sistem harus menyediakan layanan untuk *sharing* dokumen dengan pengguna lain.	*Sharing* Dokumen	Sistem informasi bimbingan skripsi	*Medium*
SRS_2_06	Sistem harus menyediakan layanan untuk mengatur kapasitas *storage* untuk *user*.	Mengatur kapasitas storage	*Administrator*	*Medium*
SRS_2_07	Sistem harus menyediakan layanan untuk melihat daftar *user*.	Melihat *user*	*Administrator*	*Medium*
SRS_2_08	Sistem harus menyediakan layanan untuk menghapus *user*.	Menghapus *user*	*Administrator*	*Medium*
SRS_2_09	Sistem harus menyediakan layanan untuk *login* ke dalam sistem.	*Login*	*User*	*Low*

Tabel 4.6 Spesifikasi kebutuhan non fungsional

Nomor SRS	Parameter	Kebutuhan
SRS_3_01	Compatibility	Sistem harus dapat dijalankan di berbagai *browser* PC atau

		laptop (*Google Chrome, Internet Explorer, Mozilla Firefox, Opera, Safari*)
SRS_3_02	Usability	Sistem harus memiliki kemudahan antarmuka/*interface* bagi *user*

4.2.3 Diagram *Use Case*

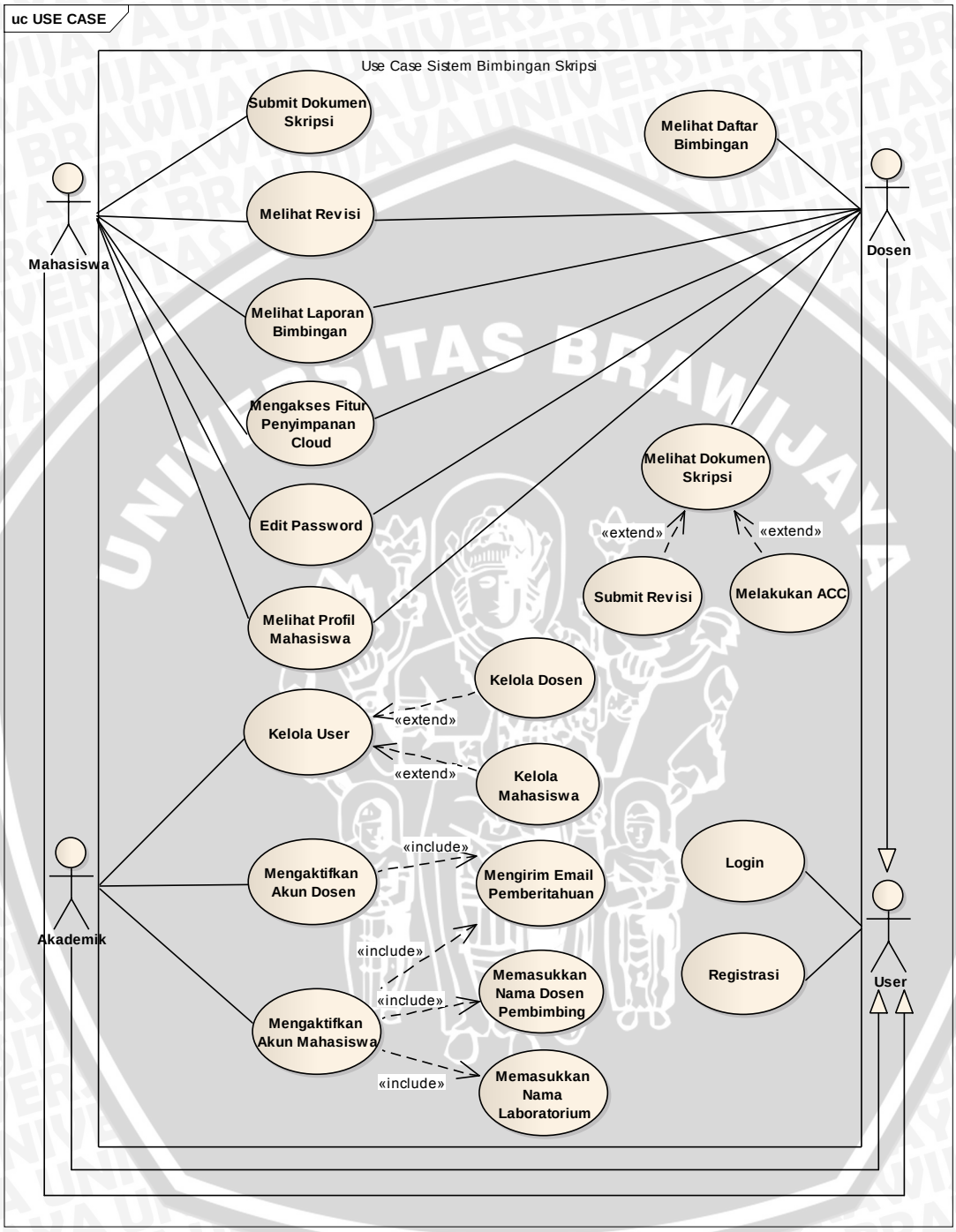
Tabel 4.7 memperlihatkan aktor-aktor yang terlibat dalam sistem informasi bimbingan skripsi beserta penjelasannya.

Tabel 4.7 Identifikasi Aktor SI Bimbingan Skripsi

Aktor	Deskripsi
User	*User* adalah pengguna sistem informasi yang memiliki hak untuk melakukan registrasi dan *login* ke sistem.
Mahasiswa	Mahasiswa adalah pengguna sistem informasi yang memiliki hak untuk *upload* dokumen skripsi, melihat revisi, melihat laporan bimbingan, dan mengakses fitur penyimpanan *cloud*.
Dosen	Dosen adalah pengguna sistem informasi yang memiliki hak untuk melihat dokumen skripsi mahasiswa, mengakses fitur penyimpanan *cloud*, melakukan acc, submit revisi, serta melihat laporan bimbingan.
Akademik	*Staff* akademik adalah pengguna sistem informasi yang memiliki hak untuk mengelola data dosen dan mahasiswa

	serta mengaktifkan akun mahasiswa dan dosen.

Gambar 4.4 memperlihatkan diagram *use case* sistem informasi bimbingan skripsi.



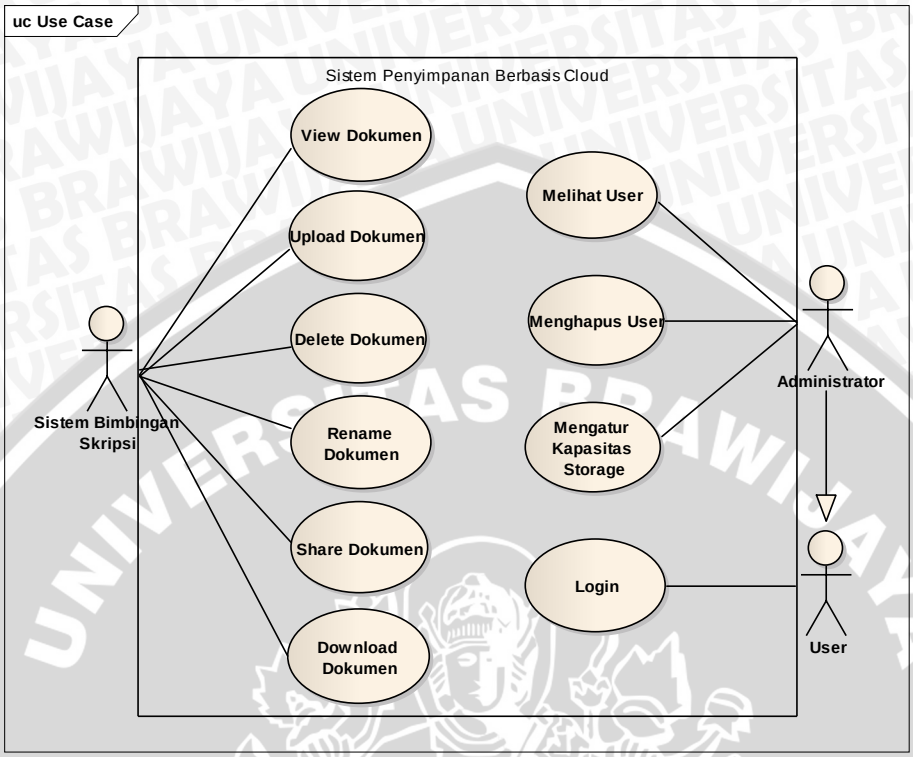
Gambar 4.4. Diagram *use case* sistem informasi bimbingan skripsi

Tabel 4.8 memperlihatkan aktor-aktor yang terlibat dalam sistem penyimpanan berbasis *cloud* beserta penjelasannya.

Tabel 4.8 Identifikasi Aktor Sistem *Cloud*

Aktor	Deskripsi
Sistem Informasi Bimbingan Skripsi	Sistem Informasi Bimbingan Skripsi merupakan sistem yang berinteraksi dengan sistem *cloud* untuk melakukan *request* layanan melalui *web service*. Layanan tersebut adalah *view*, *upload*, *delete*, *rename*, *share*, serta *download* dokumen.
Administrator	Administrator adalah pengguna sistem informasi yang memiliki hak untuk melihat daftar *user*, mengahapus *user*, dan mengatur kapasitas *storage* untuk *user*.
User	*User* adalah pengguna sistem informasi yang memiliki hak untuk melakukan *login* ke sistem.

Sedangkan diagram *use case* yang menggambarkan interaksi aktor dengan sistem penyimpanan berbasis *cloud* ditunjukkan oleh gambar 4.5. Pada *use case* ini terdapat aktor sistem informasi bimbingan skipsi yang mengakses layananan pada sistem *cloud* melaui API yang telah disediakan. Layanan yang dapat diakses adalah layanan untuk *view*, *upload*, *delete*, *rename*, dan *sharing* dokumen. Sedangkan administrator merupakan aktor yang dapat melihat, menghapus, dan mengubah kapasitas *strorage* untuk masing-masing *user* pada sistem *cloud*.



Gambar 4.5 Diagram *use case* sistem penyimpanan *cloud*

4.2.4 Skenario *Use Case*

Dan berikut akan digambarkan beberapa skenario utama dari *use case* yang telah dirancang pada gambar 4.4 dan 4.5.

Tabel 4.9 *Use case* submit dokumen skripsi

Skenario Kasus Pada Sistem	
Nomor Use Case	SRS_1_01
Nama	*Submit* dokumen skripsi
Tujuan	Agar mahasiswa dapat melakukan *submit* dokumen skripsi untuk melaksanakan bimbingan skripsi secara *online*
Deskripsi	*Use Case* ini menjelaskan bagaimana proses *upload* dan *submit* dokumen

	skripsi
Aktor	Mahasiswa
Skenario Utama	
Kondisi Awal	Aktor telah melakukan login dan telah masuk ke halaman bimbingan skripsi
Aksi Aktor	Reaksi Sistem
1. Aktor melakukan *upload* dokumen skripsi	5. Sistem melakukan validasi terhadap data yang telah di-*submit* oleh aktor
2. Aktor mengisi kolom bab	
3. Aktor mengisi kolom pesan	6. Sistem menampilkan pemberitahuan bahwa dokumen telah berhasil di-*submit*.
4. Aktor menekan tombol submit	
Skenario Alternatif 1: Jika format dokumen yang di-*submit* tidak dalam format .pdf	
	7. Sistem menampilkan pemberitahuan bahwa dokumen harus dalam *format* .pdf
Kondisi Akhir	Aktor telah berhasil melakukan submit dokumen bimbingan skripsi kepada dosen pembimbing

Tabel 4.10 *Use case* melihat dokumen skripsi

Skenario Kasus Pada Sistem	
Nomor Use Case	SRS_1_02
Nama	Melihat dokumen skripsi
Tujuan	Agar dosen dapat melihat dokumen skripsi yang di-*upload* oleh mahasiswa
Deskripsi	*Use case* ini menjelaskan bagaimana dosen dapat melakukan proses untuk

	melihat dokumen skripsi yang telah di-*upload* mahasiswa untuk selanjutnya melakukan ACC atau revisi.
Aktor	Dosen
Skenario Utama	
Kondisi Awal	Aktor telah melakukan login dan telah masuk ke halaman daftar bimbingan
Aksi Aktor	Reaksi Sistem
1. Aktor memilih mahasiswa	3. Sistem menampilkan dokumen skripsi beserta rincian bimbingan
2. Aktor menekan tombol "lihat dokumen"	
Kondisi Akhir	Aktor telah berhasil melihat dokumen skripsi mahasiswa

Tabel 4.11 *Use case* melakukan ACC

Skenario Kasus Pada Sistem	
Nomor Use Case	SRS_1_03
Nama	Melakukan ACC
Tujuan	Agar dosen pembimbing dapat melakukan ACC atas dokumen skripsi mahasiswa
Deskripsi	*Use case* ini menjelaskan bagaimana dosen dapat melakukan ACC atas dokumen skripsi yang di-*upload* oleh mahasiswa
Aktor	Dosen
Skenario Utama	
Kondisi Awal	Aktor telah melakukan login dan telah

	masuk ke halaman lihat dokumen skripsi
Aksi Aktor	Reaksi Sistem
1. Aktor menekan tombol "ACC skripsi"	2. Sistem menampilkan halaman konfirmasi
3. Aktor melakukan konfirmasi	4. Sistem menampilkan pemberitahuan bahwa dokumen skripsi telah berhasil di ACC
Skenario Alternatif 1: jika dosen tidak melakukan konfirmasi	
	5. Sistem tidak melakukan ACC atas dokumen skripsi mahasiswa
Kondisi Akhir	Aktor telah berhasil melakukan ACC terhadap dokumen skripsi mahasiswa

Tabel 4.12 *Use case* submit revisi

Skenario Kasus Pada Sistem	
Nomor Use Case	SRS_1_04
Nama	Submit revisi
Tujuan	Agar dosen dapat melakukan submit revisi
Deskripsi	*Use case* ini menjelaskan bagaimana dosen dapat melakukan proses untuk submit revisi atas dokumen skripsi mahasiswa
Aktor	Dosen
Skenario Utama	
Kondisi Awal	Aktor telah melakukan *login* dan telah berada pada halaman lihat dokumen skripsi

Aksi Aktor	Reaksi Sistem
1. Dosen menekan tombol “revisi”	2. Sistem menampilkan *form* untuk submit revisi
3. Dosen memilih dokumen revisi dari komputer miliknya atau mengetik teks komentar pada *text area* yang telah disediakan	5. Sistem melakukan validasi terhadap data yang di-*submit* oleh aktor
4. Dosen menekan tombol “submit”	6. Sistem menampilkan pemberitahuan bahwa *submit* berhasil dilakukan
Skenario Alternatif 1: jika dokumen tidak dalam format .pdf	
	7. Sistem menampilkan pemberitahuan bahwa dokumen harus dalam format .pdf
Kondisi Akhir	Aktor telah berhasil melakukan submit dokumen revisi kepada mahasiswa

Tabel 4.13 *Use case* mengakses fitur penyimpanan *cloud*

Skenario Kasus Pada Sistem	
Nomor Use Case	SRS_1_05
Nama	Mengakses fitur penyimpanan *cloud*
Tujuan	Agar aktor dapat melakukan akses ke fitur penyimpanan *cloud*
Deskripsi	*Use Case* ini menjelaskan bagaimana aktor melakukan proses akses ke fitur *cloud*
Aktor	Mahasiswa, Dosen
Skenario 1 : Melihat Dokumen	
Kondisi Awal	Aktor telah melakukan login dan telah

	berada pada halaman penyimpanan *cloud*
Aksi Aktor	Reaksi Sistem
2. Aktor memilih dokumen yang ingin dilihat dengan cara melakukan "klik"	1. Sistem menampilkan daftar dokumen milik aktor yang tersimpan di dalam sistem
	3. Sistem menampilkan dokumen yang dipilih oleh aktor
Kondisi Akhir	Aktor berhasil melihat dokumen yang telah dipilih
Skenario 2: *Upload* dokumen	
Kondisi Awal	Aktor telah melakukan *login* dan telah berada pada halaman penyimpanan *cloud*
Aksi Aktor	Reaksi Sistem
1. Aktor memilih menu "*upload* dokumen"	
2. Aktor memilih dokumen yang ingin di upload dari komputer miliknya	
3. Aktor menekan tombol "upload"	
Kondisi Akhir	Aktor berhasil melakukan *upload* dokumen ke sistem *cloud*

Tabel 4.14 *Use case* melihat laporan bimbingan

Skenario Kasus Pada Sistem	
Nomor Use Case	SRS_1_06
Nama	Melihat laporan bimbingan
Tujuan	Agar pengguna dapat melihat laporan bimbingan yang telah dilakukan oleh mahasiswa
Deskripsi	*Use case* ini menjelaskan bagaimana pengguna dapat melakukan proses untuk melihat *list* yang berisi laporan bimbingan skripsi yang telah dilakukan oleh mahasiswa
Aktor	Mahasiswa, Dosen
Skenario Utama	
Kondisi Awal	Aktor telah melakukan login dan telah masuk pada halaman aktor
Aksi Aktor	Reaksi Sistem
1. Aktor memilih menu untuk menampilkan laporan bimbingan skripsi	2. Sistem menampilkan detail laporan bimbingan skripsi mahasiswa

Tabel 4.15 *View* Dokumen

Skenario Kasus Pada Sistem	
Nomor Use Case	SRS_2_01
Nama	*View* Dokumen
Tujuan	Agar sistem informasi bimbingan skripsi dapat menggunakan layanan pada *server cloud* untuk *view* dokumen
Deskripsi	*Use case* ini menjelaskan bagaimana sistem informasi bimbingan skripsi melakukan request layanan untuk melihat dokumen yang tersimpan di dalam sistem penyimpanan *cloud.*
Aktor	Sistem informasi bimbingan skripsi
Skenario Utama	
Kondisi Awal	Sistem informasi bimbingan skripsi telah berjalan
Aksi Aktor	Reaksi Sistem
1. Aktor mengirim parameter berbentuk *url*	2. Sistem membaca parameter yang dikirim oleh aktor
	3. Sistem mengambil data dari *database*
	4. Sistem mengirim data kepada aktor dalam format JSON
Kondisi Akhir	Aktor berhasil membaca data JSON yang dikirim oleh *server cloud*

Tabel 4.16 Mengatur Kapasitas *Storage*

Skenario Kasus Pada Sistem	
Nomor Use Case	SRS_2_02
Nama	Mengatur Kapasitas *Storage*
Tujuan	Agar *administrator* dapat menggunakan layanan untuk mengatur kapasitas penyimpanan *user*
Deskripsi	*Use case* ini menjelaskan bagaimana administrator melakukan proses untuk mengatur kapasitas penyimpanan untuk masing-masing *user*
Aktor	*Administrator*
Skenario Utama	
Kondisi Awal	Sistem penyimpana *cloud* telah berjalan dan menampilkan daftar *user*
Aksi Aktor	Reaksi Sistem
1. Aktor memilih user yang akan diubah	5. Melakukan refresh halaman
2. Aktor memilih menu "ubah kapasitas"	
3. Memilih jumlah kapasitas	
Kondisi Akhir	Kapasitas penyimpanan *user* berhasil diubah

4.3 Perancangan Aplikasi

Perancangan aplikasi dilakukan dalam enam tahap, yaitu perancangan arsitektural, perancangan diagram *activity*, perancangan diagram *class*, perancangan diagram *sequence*, perancangan basis data, dan perancangan antarmuka. Perancangan

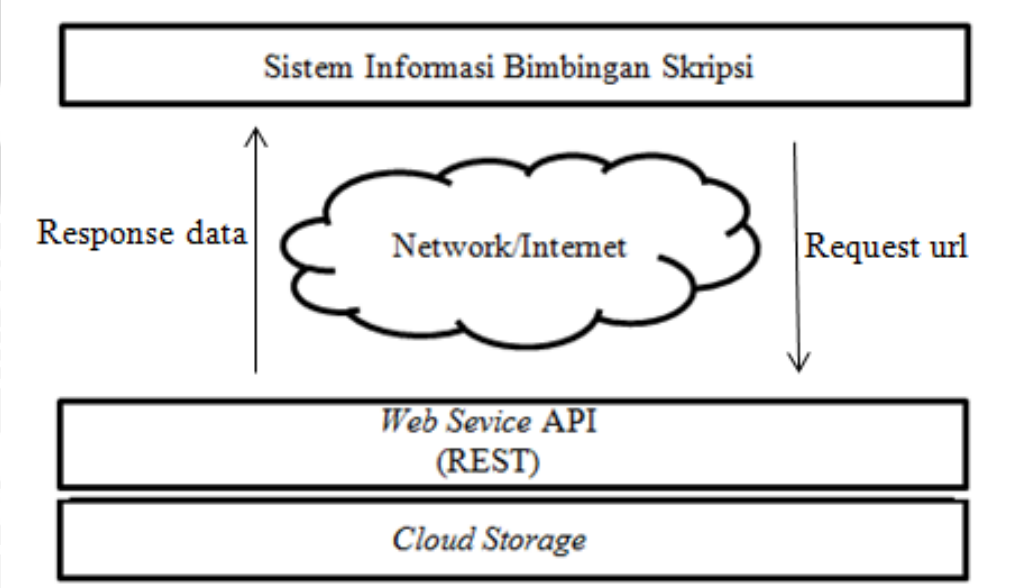
aplikasi pada skripsi ini menggunakan pendekatan desain berorientasi objek yang direpresentasikan dengan menggunakan UML.

4.3.1 Perancangan Arsitektural

Sistem penyimpanan berbasis *cloud* (*cloud storage*) dibangun menggunakan *framework* CodeIgniter didampingi mySQL sebagai penyimpanan dokumen skripsi. Setiap dokumen yang diunggah ke sistem penyimpanan *cloud* akan tersimpan disini. Sistem ini menyediakan teknologi REST *web service* sebagai API, dimana API tersebut dapat digunakan oleh sistem lain (dalam hal ini sistem informasi bimbingan skripsi) untuk mengakses layanan yang telah disediakan oleh sistem *cloud*.

Sistem informasi bimbingan skripsi dibangun menggunakan *framework* CodeIgniter dan didampingi mySQL sebagai penyimpanan data untuk proses bimbingan skripsi. Sistem ini berinteraksi dengan sistem *cloud* melalui *web service* API dalam sebuah jaringan. Sistem informasi bimbingan skripsi akan melakukan *request* layanan dan mengirim parameter melalui url sesuai dengan format yang telah disediakan oleh sistem *cloud*.

Perancangan arsitektur sistem ditunjukkan oleh gambar 4.6.



Gambar 4.6 Perancangan Arsitektural

4.3.2 Perancangan Diagram *Activity*

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

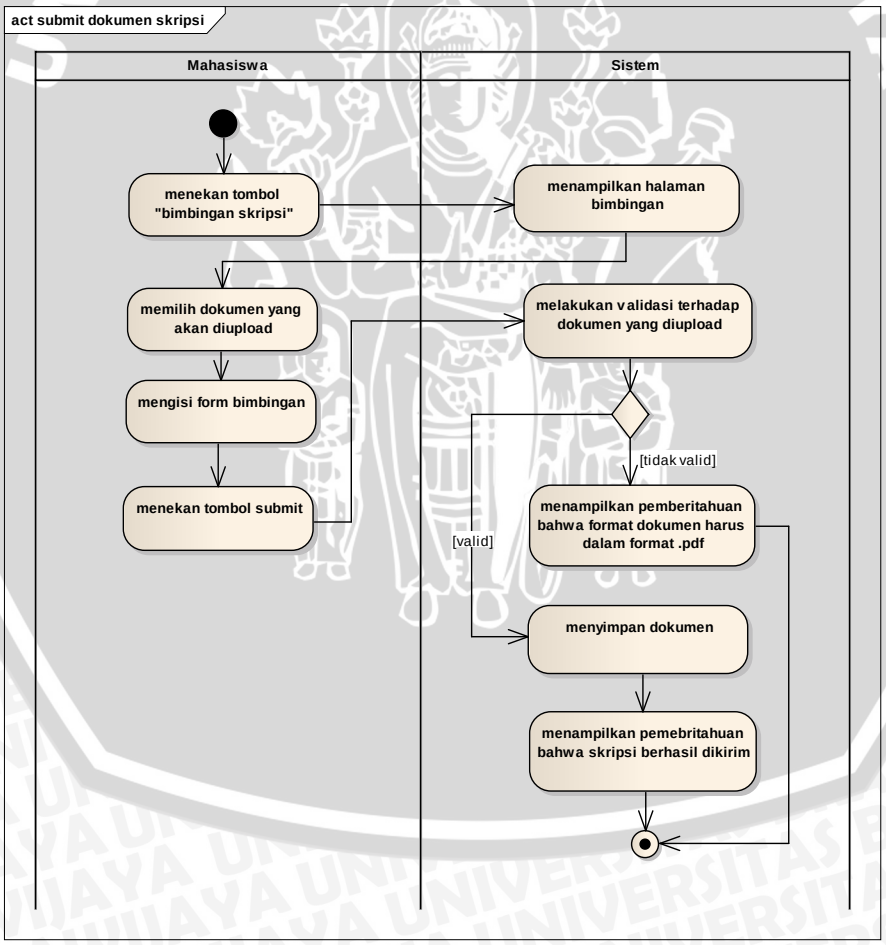
4.3.2.1 Diagram *Activity* Sistem informasi bimbingan skripsi

Diagram *activity* ini menggambarkan proses interaksi *aktor* dengan sistem informasi bimbingan skripsi . *Aktor* yang dimaksud adalah mahasiswa, dosen, *staff* akademik, dan *user*.

Berikut akan dijelaskan diagram *activity* utama yang terdapat pada sistem.

a. Diagram *Activity Submit* Dokumen Skripsi

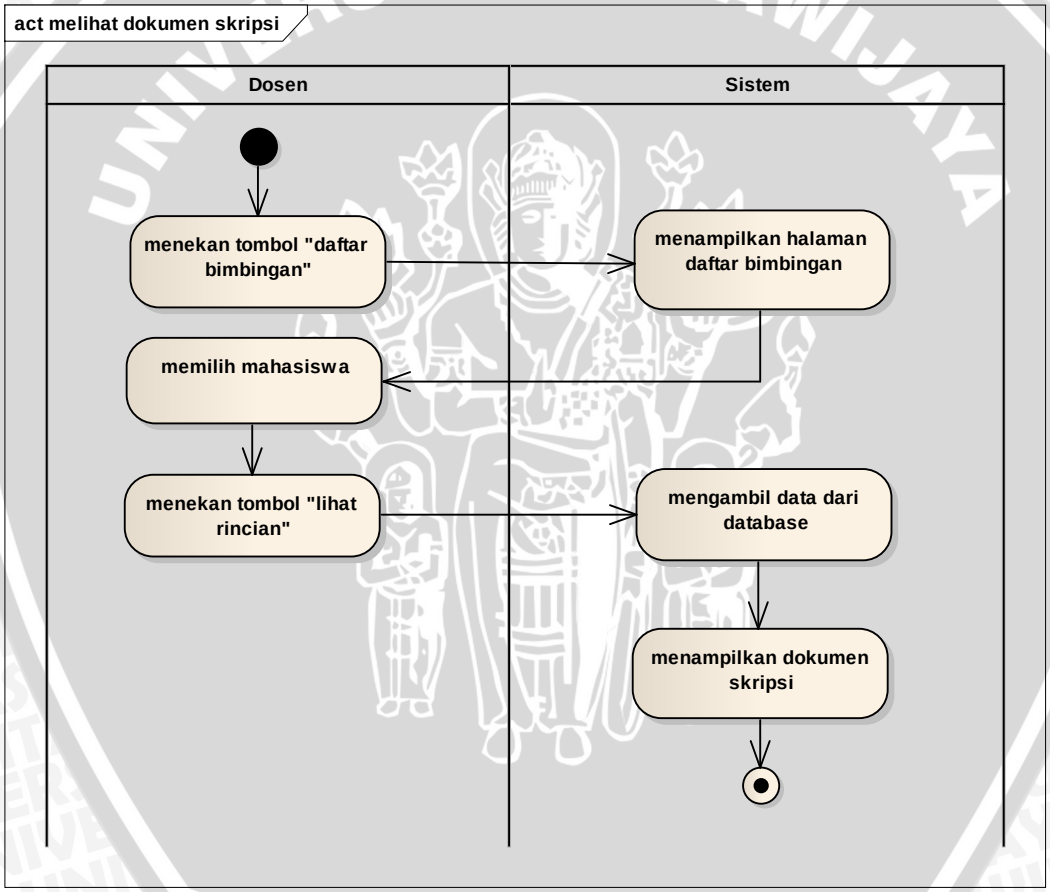
Perancangan diagram *activity submit* dokumen skripsi dapat dilihat pada gambar 4.7.



Gambar 4.7 Diagram Activity Submit Dokumen Skripsi

b. Diagram *Activity* Melihat Dokumen Skripsi

Melihat dokumen skripsi merupakan *event* ketika dosen ingin melihat dokumen skripsi mahasiswa yang melaksanakan bimbingan. Pada aktivitas ini, sistem menampilkan daftar mahasiswa yang melaksanakan bimbingan, kemudian dosen dapat memilih mahasiswa yang ingin dilihat. Setelah dosen memilih mahasiswa, maka sistem akan menampilkan dokumenskripsi dari mahasiswa tersebut. Perancangan diagram *activity* melihat dokumen skripsi dapat dilihat pada gambar 4.8.

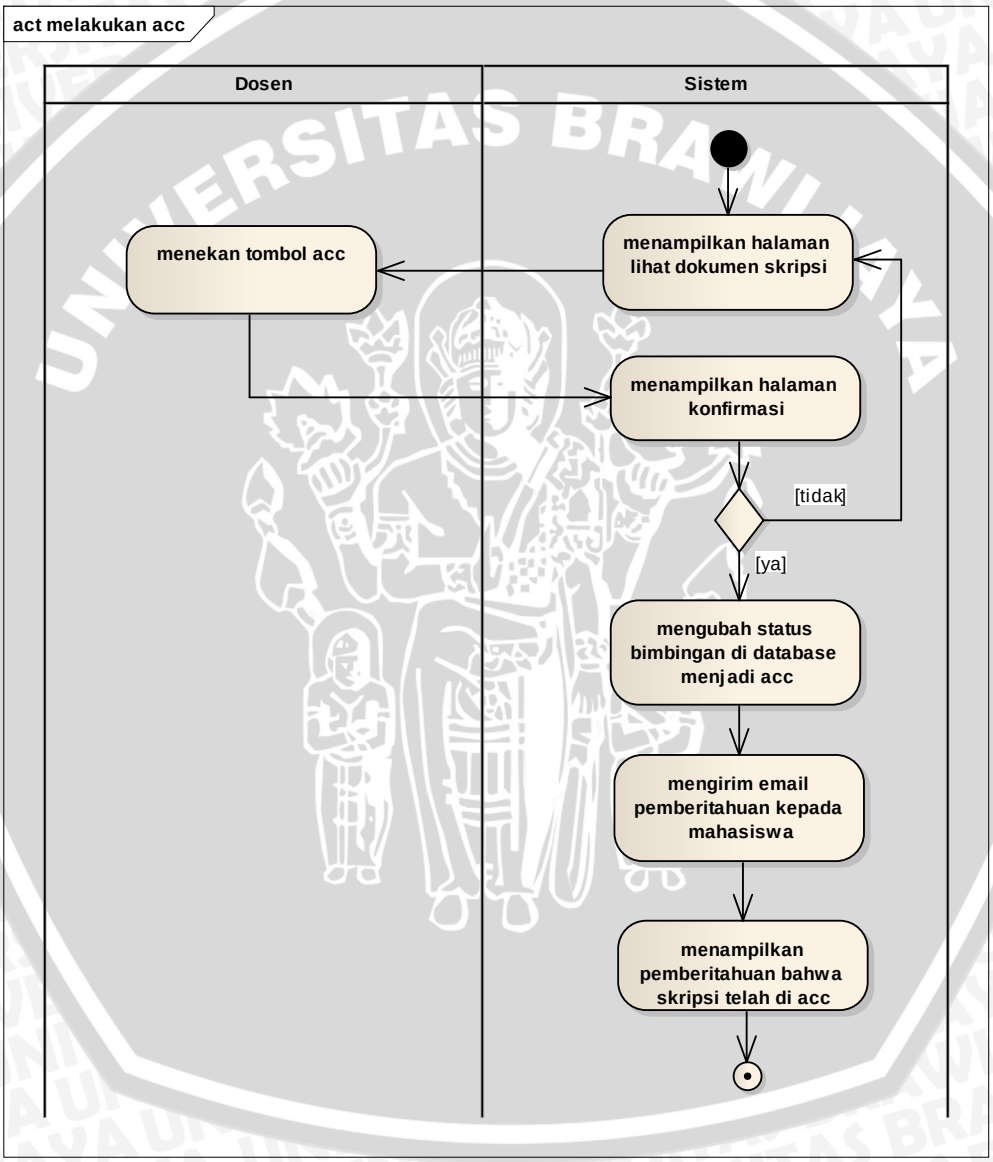


Gambar 4.8 Diagram *Activiy* Melihat Dokumen Skripsi

c. Diagram *Activity* Melakukan ACC

Melakukan ACC merupakan *event* ketika dokumen skripsi mahasiswa sudah dianggap benar dan tidak ada yang perlu direvisi sehingga dosen melakukan ACC

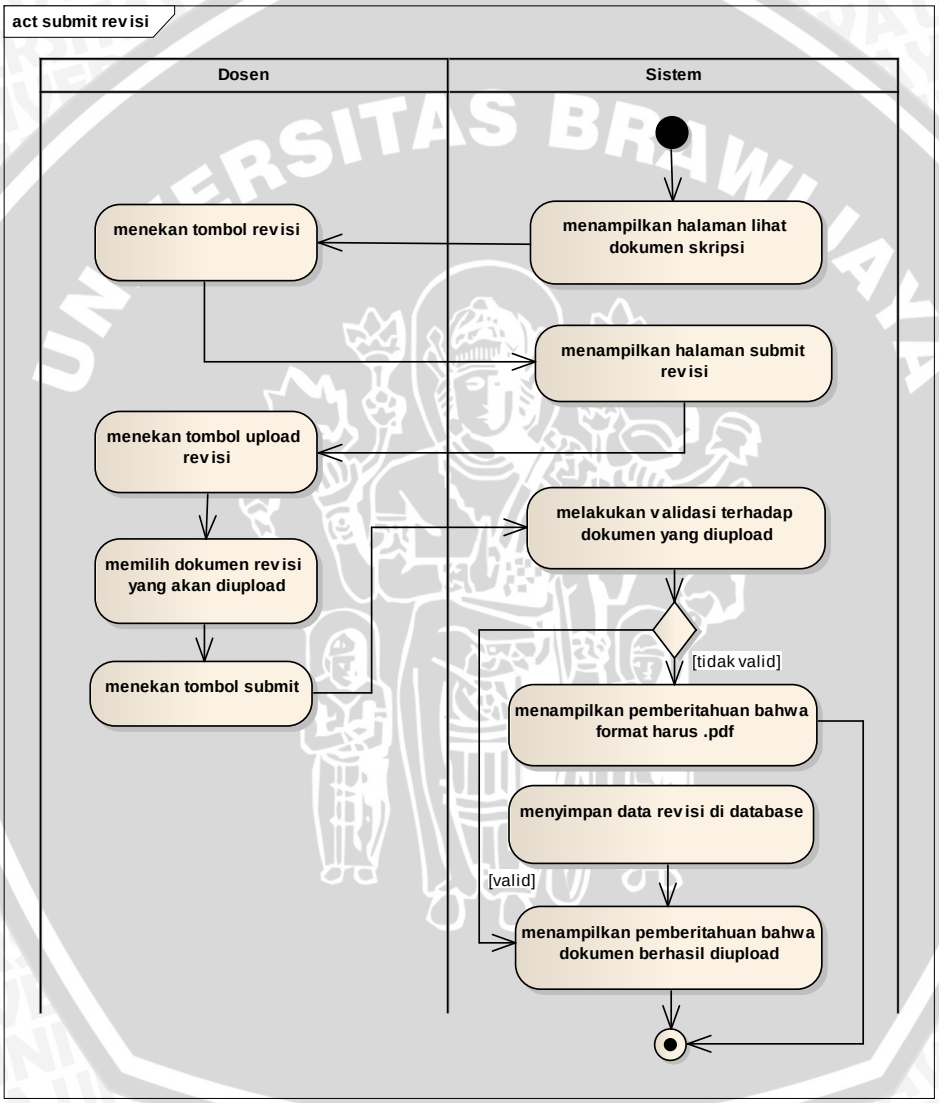
terhadap dokumen tersebut. Pada aktivitas ini, dosen menekan tombol ACC dan melakukan konfirmasi ACC. Selanjutnya sistem akan mengubah status bimbingan mahasiswa menjadi ACC dan menampilkan pemberitahuan bahwa dokumen skripsi berhasil di ACC. Perancangan diagram *activity* melakukan ACC dapat dilihat pada gambar 4.9.



Gambar 4.9 Diagram *Activiy* Melakukan ACC

d. Diagram *Activity* Submit Revisi

Submit revisi merupakan *event* ketika dokumen skripsi mahasiswa dianggap belum benar dan ada yang perlu direvisi sehingga dosen mengunggah dokumen revisi atas skripsi mahasiswa yang bersangkutan. Perancangan diagram *activity* submit revisi dapat dilihat pada gambar 4.10.



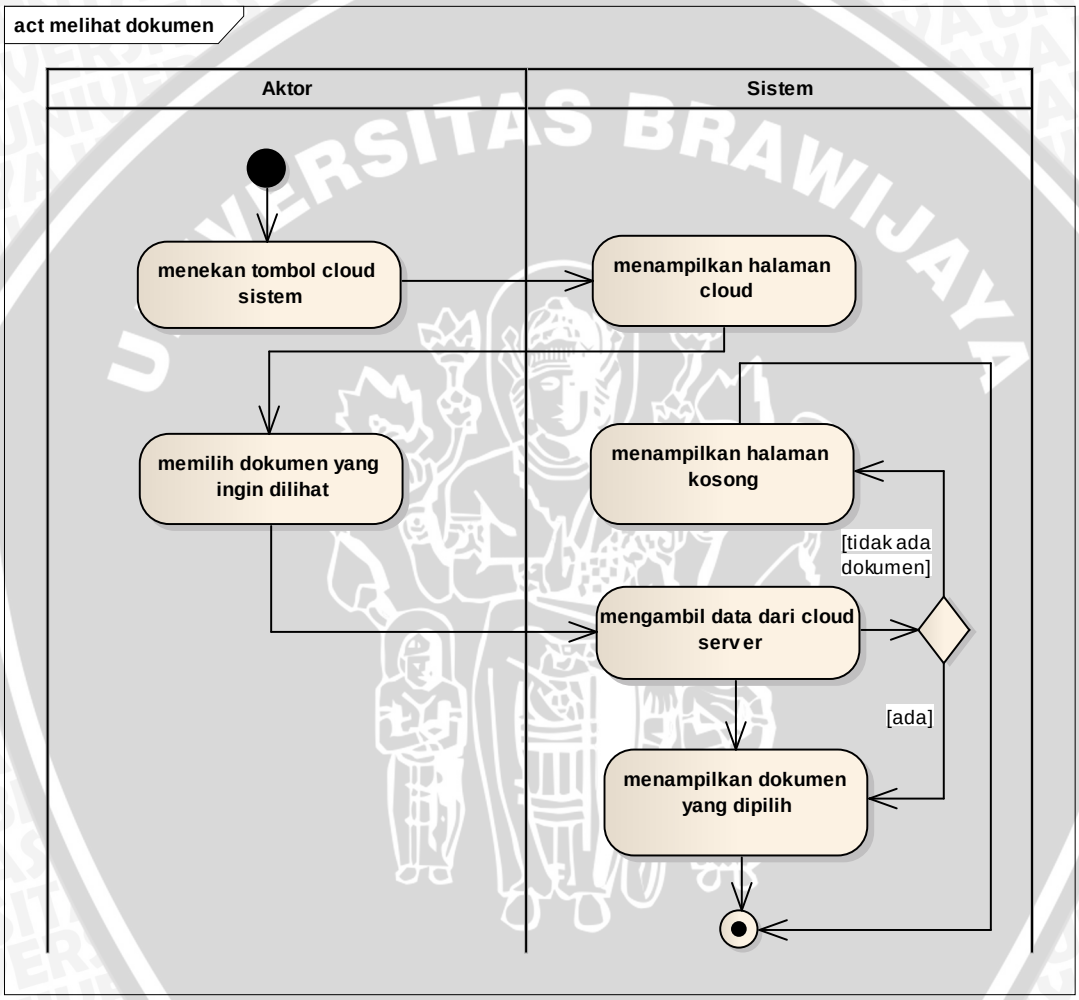
Gambar 4.10 Diagram *Activiy Submit* Revisi

e. Mengakses Fitur Penyimpanan *Cloud*

Diagram *activity* mengakses fitur penyimpanan *cloud* adalah *event* ketika aktor ingin melakukan akses terhadap fitur penyimpanan berbasis *cloud* yang telah

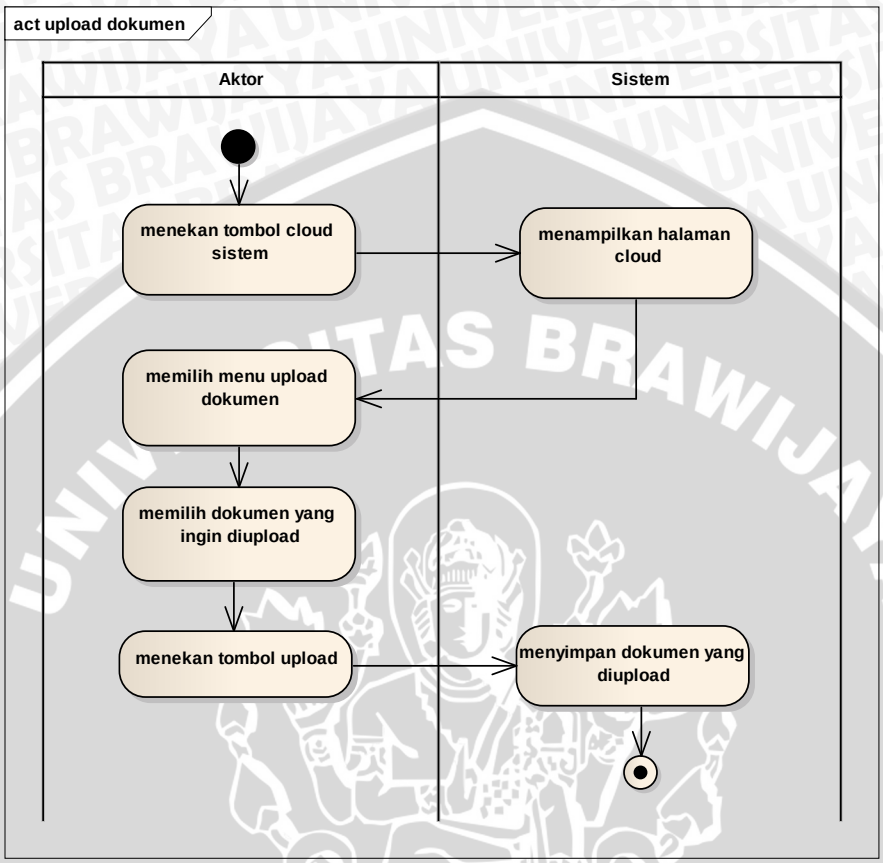
disediakan oleh sistem. Akses disini terdiri dari melihat dokumen, *upload* dokumen, menghapus dokumen, *rename* dokumen, serta *sharing* dokumen.

Melihat dokumen merupakan *event* ketika aktor ingin melihat dokumen miliknya yang telah disimpan di dalam sistem penyimpanan *cloud*. Perancangan diagram *activity* melihat dokumen dapat dilihat pada gambar 4.11.



Gambar 4.11 Diagram *Activiy* Melihat Dokumen

Upload dokumen merupakan *event* ketika aktor ingin mengunggah dokumen miliknya untuk disimpan di dalam sistem penyimpanan *cloud*. Pada aktivitas ini aktor dapat memilih dokumen dari komputer miliknya untuk di-*upload* di sistem penyimpanan cloud. Perancangan diagram *activity upload* dokumen dapat dilihat pada gambar 4.12.



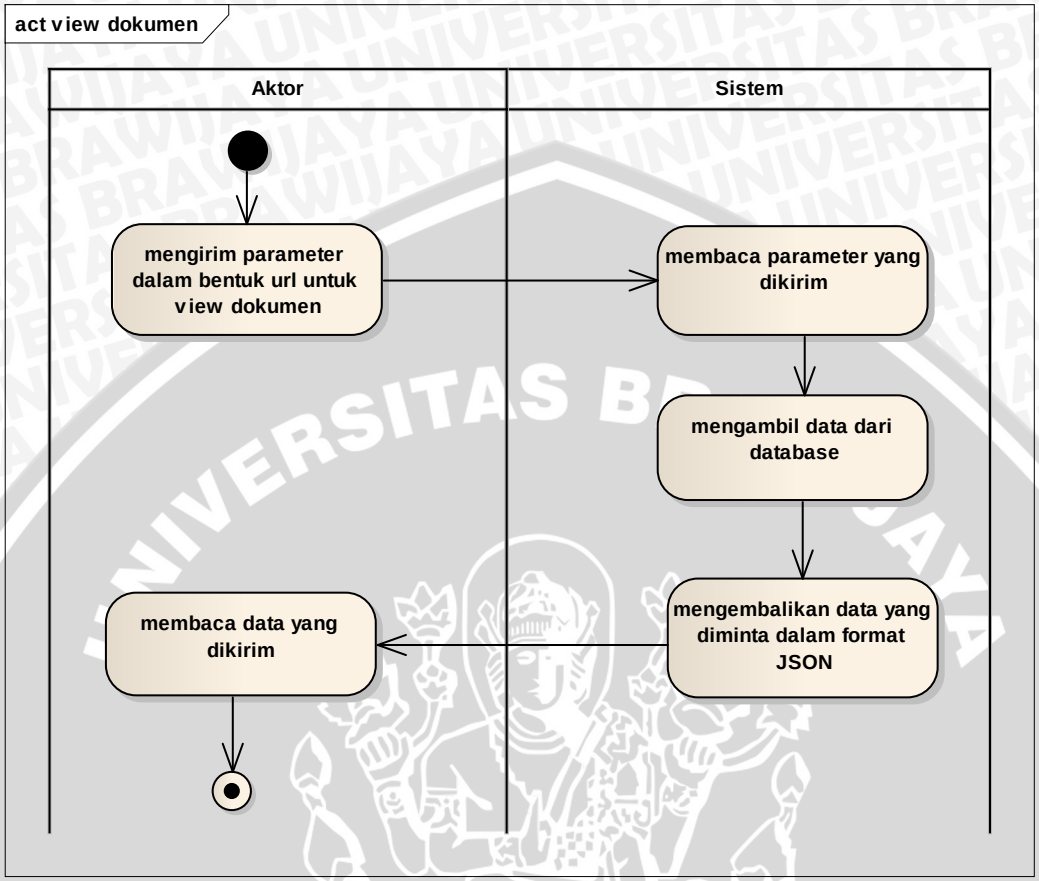
Gambar 4.12 Diagram *Activiy Upload* Dokumen

4.3.2.2 Diagram *Activity* Sistem Penyimpanan Berbasis *Cloud*

Diagram *activity* ini menggambarkan proses interaksi aktor dengan sistem penyimpanan berkas berbasis *cloud*. Aktor yang dimaksud adalah sistem informasi bimbingan skripsi dan *administrator*.

a. Diagram *Activity View* Dokumen

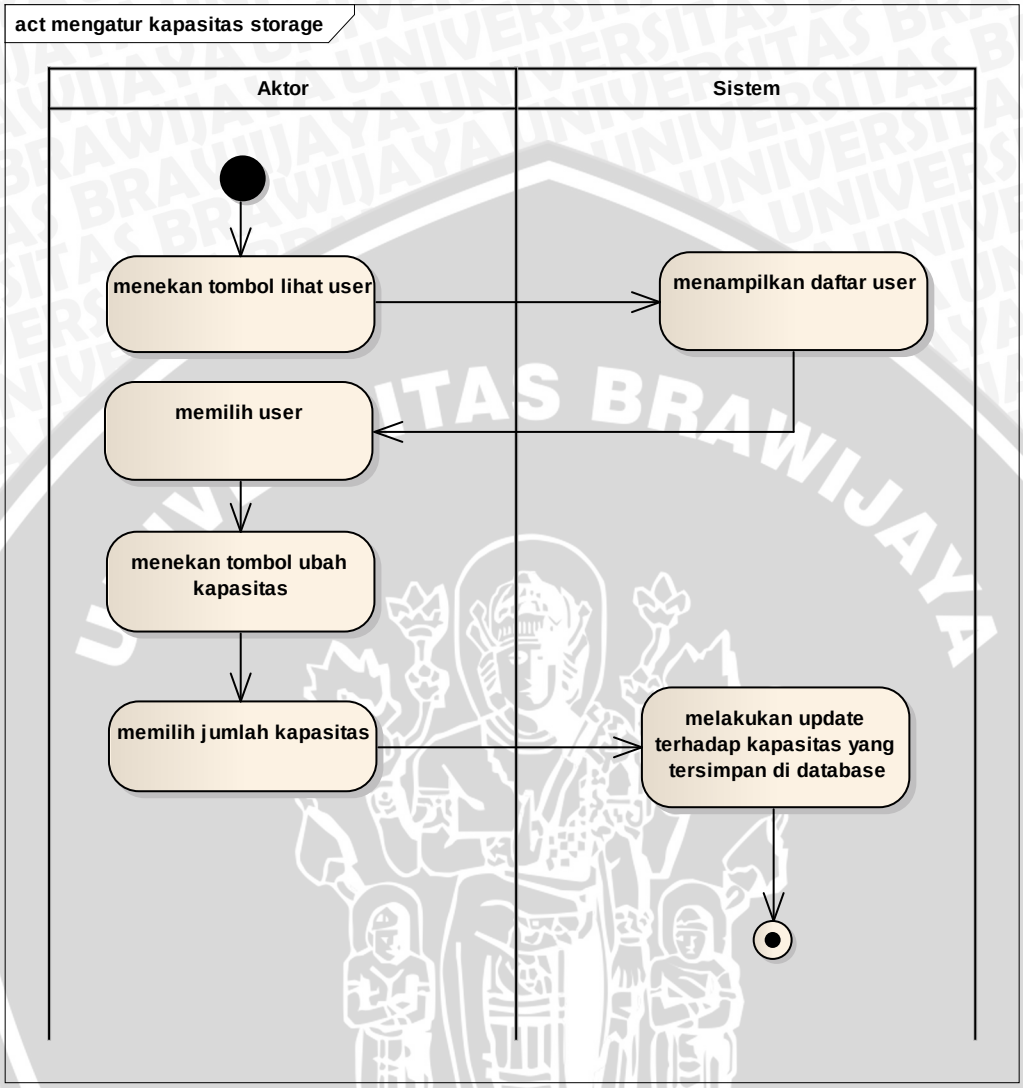
View dokumen merupakan *event* ketika sistem informasi bimbingan skripsi melakukan *request* layanan untuk *view* dokumen dengan mengirim parameter *url* melaui REST API yang telah disediakan oleh *server* penyimpanan *cloud*. Diagram *Activity view* dokumen dapat dilihat pada gambar 4.13.



Gambar 4.13 Diagram *Activiy View* Dokumen

b. Diagram *Activity* Mengatur Kapasitas *Storage*

Mengatur kapasitas *storage* merupakan *event* ketika *administrator* ingin mengubah kapasitas storage untuk masing-masing *user* pada sistem penyimpanan berbasis *cloud*. Pada aktivitas ini, administrator memilih user terlebih dahulu kemudian memilih jumlah kapasitas untuk user tersebut. Setelah ubah kapasitas berhasil, maka sistem akan menampilkan pemberitahuan bahwa kapasitas berhasil diubah. Diagram *Activity* mengatur kapasitas *storage* dapat dilihat pada gambar 4.14.



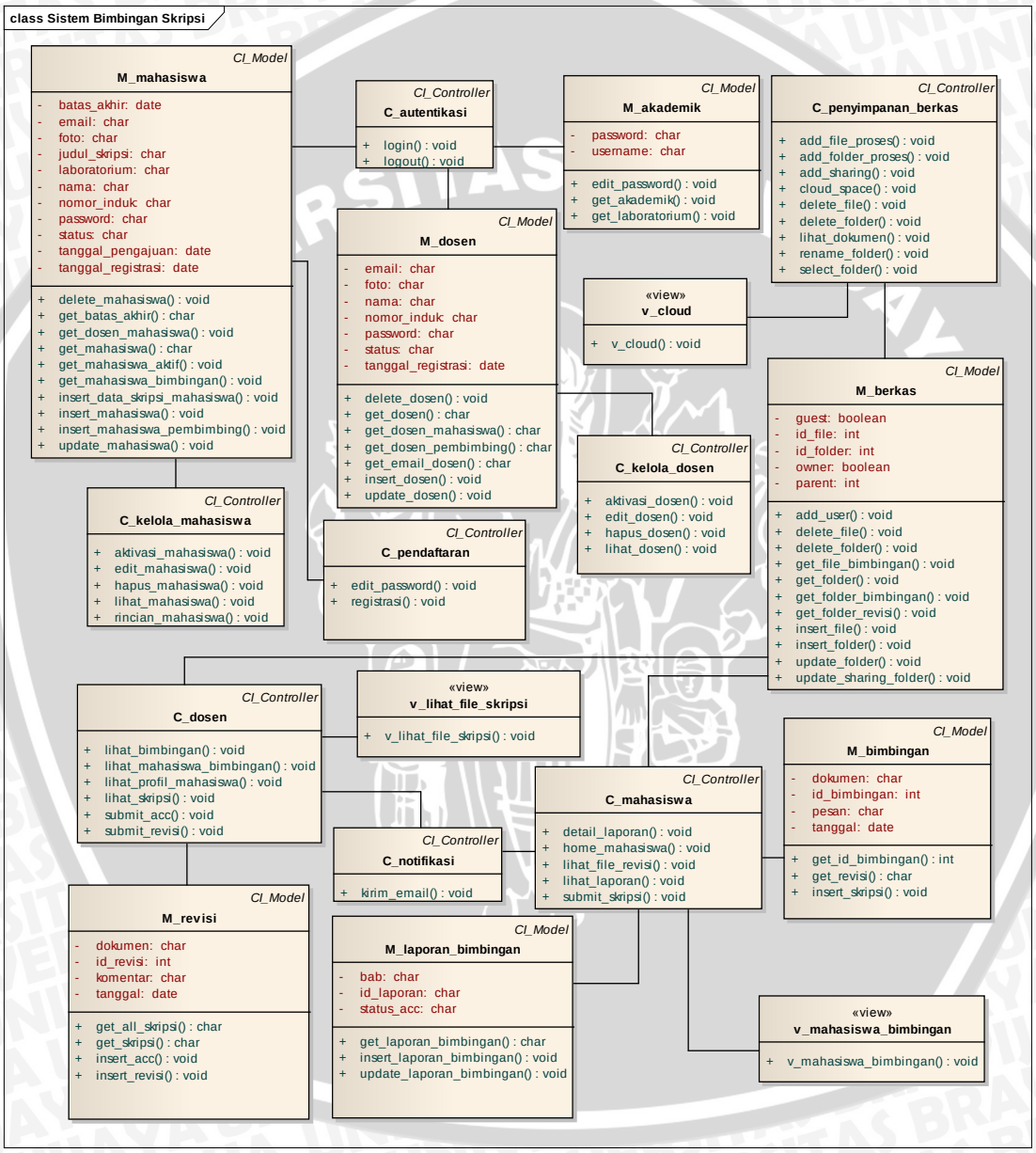
Gambar 4.14 Diagram *Activiy* Mengatur Kapasitas *Storage*

4.3.3 Perancangan Diagram *Class*

Diagram *class* memberikan gambaran rancangan kelas-kelas serta atribut yang nantinya akan digunakan dalam penyimpanan data. Selain itu juga akan disebutkan operasi-operasi dalam tabel yang menjelaskan operasi apa yang dapat dilakukan terhadap tabel tersebut. Pada perancangan ini, diagram *class* dirancang berdasarkan struktur *codeigniter* yaitu terdapat *class controller* yang saling menggunakan *class model*.

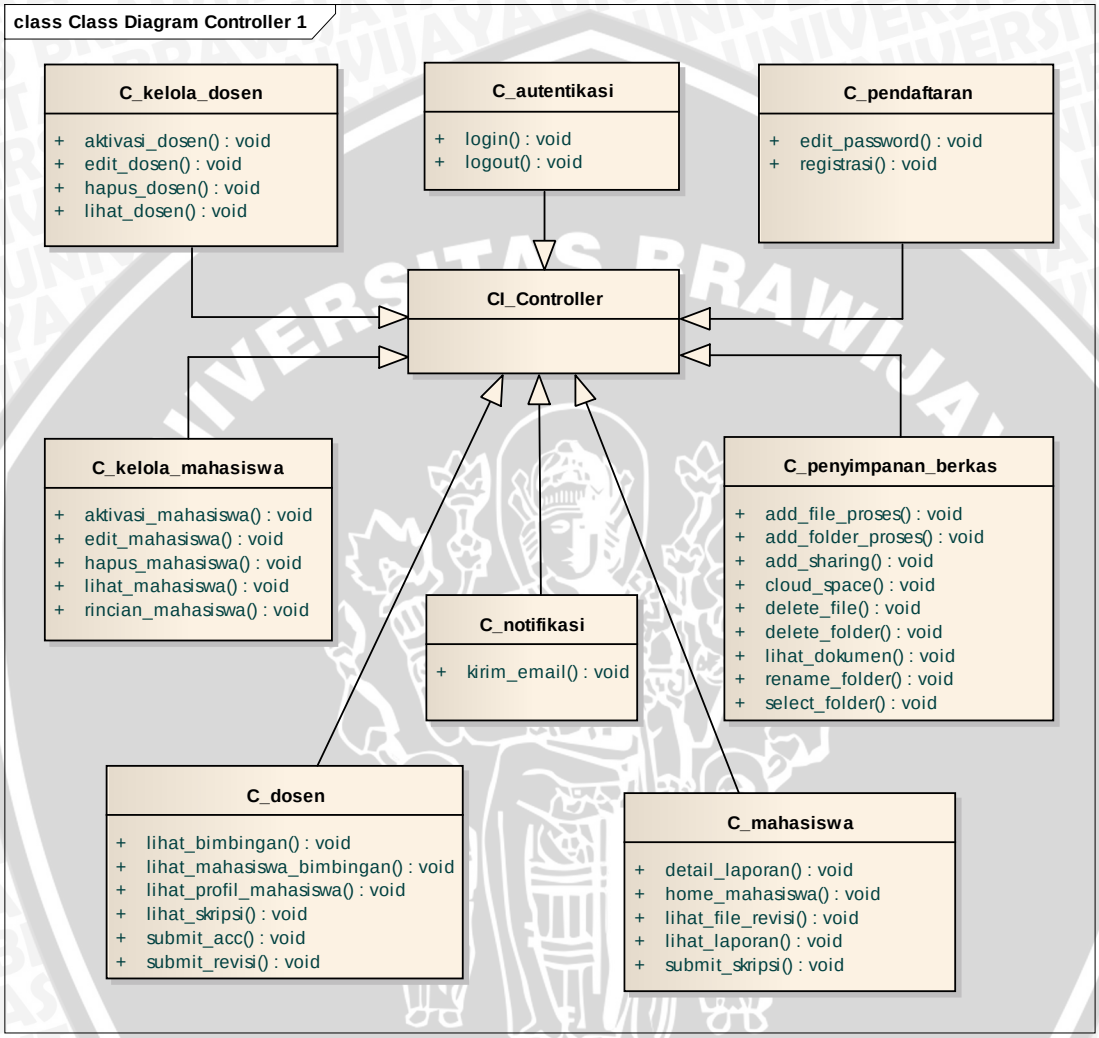
4.3.3.1 Diagram *Class* Sistem informasi bimbingan skripsi

Diagram *calss* ini menggambarkan pemodelan elemen-elemen *class* yang membentuk sistem informasi bimbingan skripsi . Diagram *class* sistem informasi bimbingan skripsi dapat dilihat pada gambar 4.15.



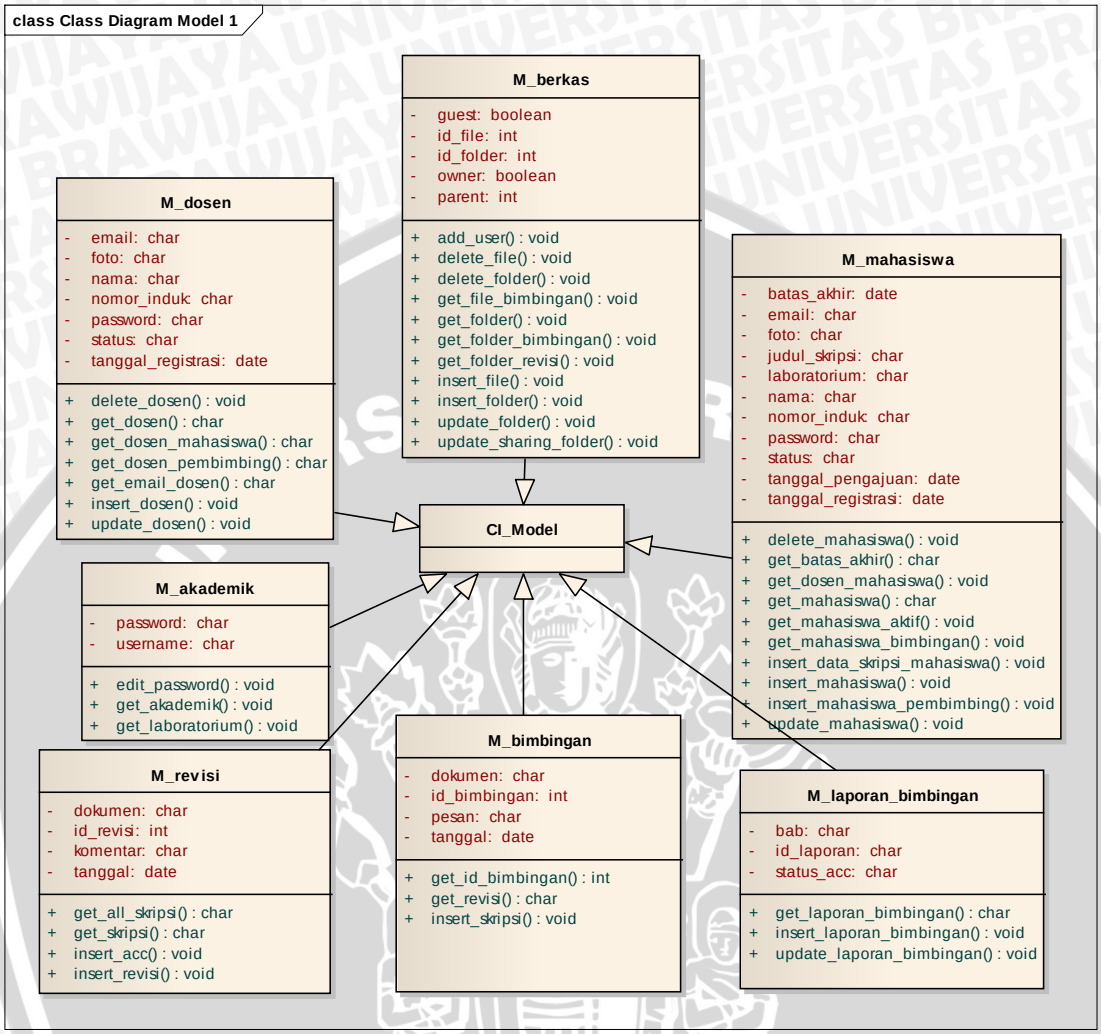
Gambar 4.15 Diagram *Class* SI Bimbingan Skripsi

Pada Gambar 4.16 akan dijelaskan rancangan kelas-kelas *controller* pada sistem yang nantinya akan diimplementasikan pada saat proses pembuatan aplikasi.



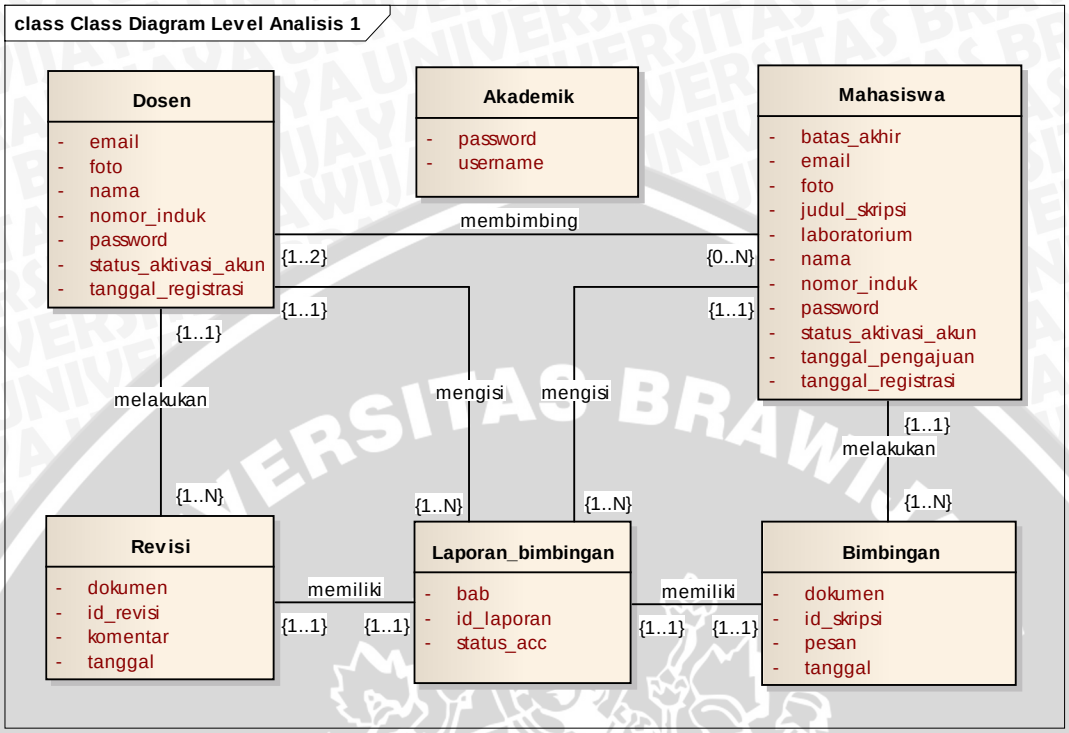
Gambar 4.16 Diagram *Class Controller* SI Bimbingan Skripsi

Pada Gambar 4.17 akan dijelaskan rancangan kelas-kelas *model* pada sistem informasi bimbingan skripsi yang nantinya akan diimplementasikan pada saat proses pembuatan aplikasi. Terdapat tujuh kelas model yaitu M_berkas, M_dosen, M_mahasiswa, M_akademik, M_revisi, M_bimbingan, dan M_laporan_bimbingan.



Gambar 4.17 Diagram *Class Model* SI Bimbingan Skripsi

Pada Gambar 4.18 akan dijelaskan rancangan diagram *class* level analisis, dimana diagram tersebut meggambarkan atribut-atribut yang nantinya akan digunakan untuk memodelkan *database*.

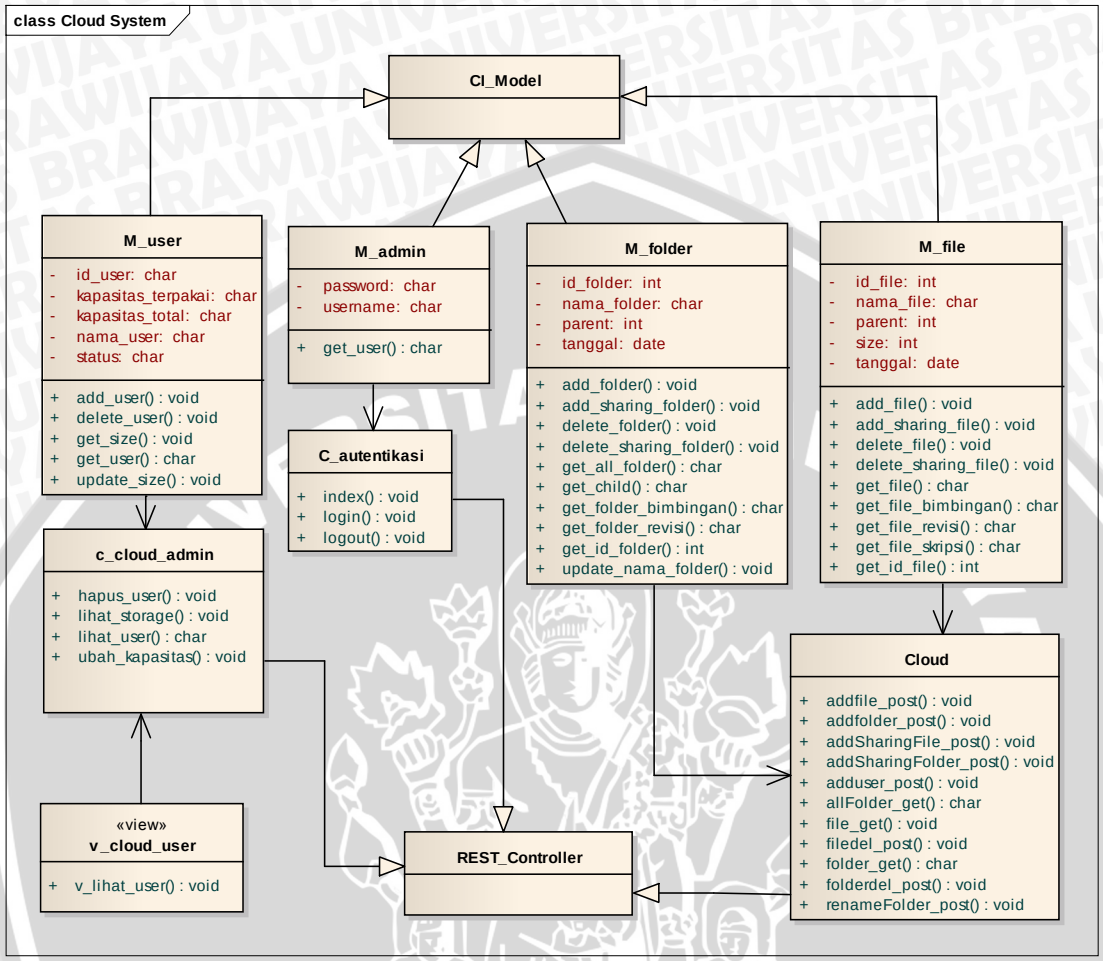


Gambar 4.18 Diagram *Class* Analisis SI Bimbingan Skripsi

4.3.3.2 Diagram *Class* Sistem Penyimpanan Berbasis Cloud

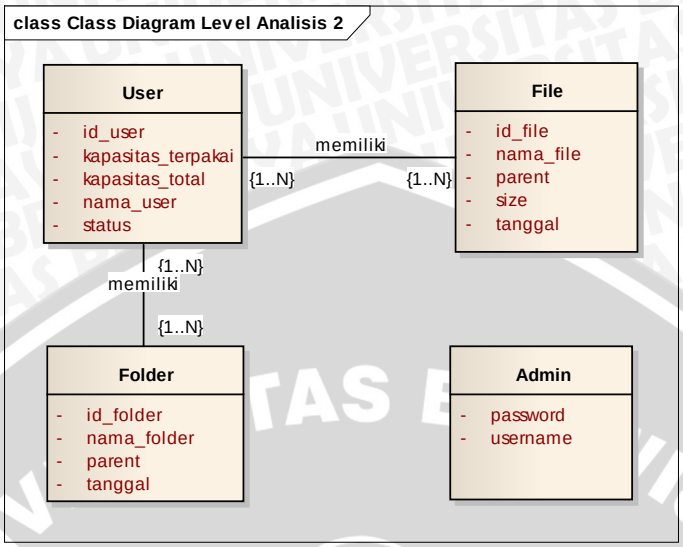
Diagram *class* ini menggambarkan pemodelan elemen-elemen *class* yang membentuk sistem penyimpanan berbasis *cloud*. Terdapat tiga *class controller* yaitu C_cloud_admin, C_cloud_login, dan C_cloud. Sedangkan *class model*-nya ada empat yaitu M_user, M_admin, M_folder, dan M_file. Terdapat juag *view* yang digunakan oleh sistem yaitu v_lihat_user.

Diagram *class* pada sistem penyimpanan berbasis *cloud* ini dirancang berdasarkan struktur *codeigniter* yaitu terdapat *class controller* yang saling menggunakan *class model* dan *view*. Diagram *class* sistem penyimpanan berbasis *cloud* dapat dilihat pada gambar 4.19.



Gambar 4.19 Diagram *Class* Sistem Penyimpanan *Cloud*

Pada Gambar 4.20 akan dijelaskan rancangan diagram *class* level analisis, dimana diagram tersebut meggambarkan atribut-atribut yang nantinya akan digunakan untuk memodelkan *database*. *Diagram class* level analisis merupakan analisis atribut dari tiap *model* pada *class* diagram sistem *cloud*. Sehingga nama *class* juga akan mengikuti nama model. Selanjutya masing-masing *class* pada class diagram level analisis tersebut akan direlasikan.



Gambar 4.20 Diagram *Class* Analisis Sistem *Cloud*

4.3.4 Perancangan Diagram *Sequence*

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

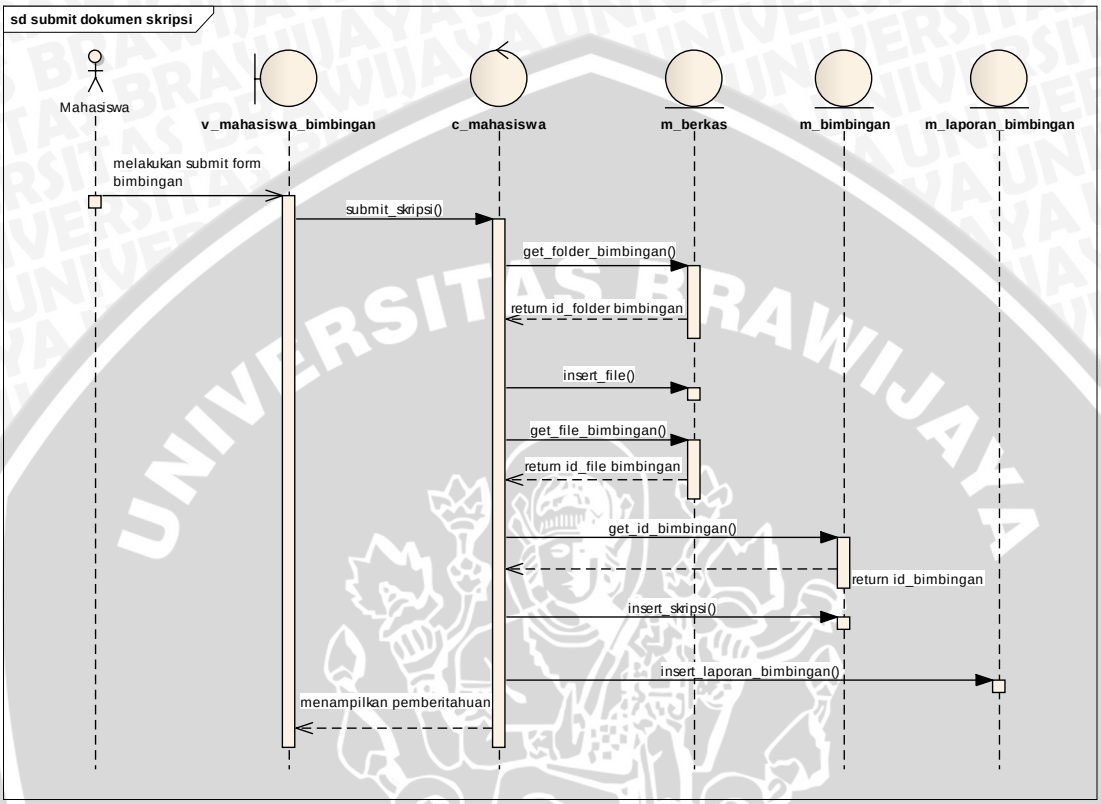
4.3.4.1 Diagram *Sequence* Sistem informasi bimbingan skripsi

Diagram *sequence* ini menggambarkan interaksi yang terjadi pada *class* dan objek ketika aktor melakukan suatu *event* terhadap sistem informasi bimbingan skripsi . Aktor yang dimaksud adalah *user*, mahasiswa, dosen, dan *staff* akademik.

a. Diagram *Sequence* Submit Dokumen Skripsi

Perancangan diagram *sequence* untuk sistem melibatkan satu *view* v_mahasiswa_bimbingan, satu *class controller* c_mahasiswa, *class model* m_bimbingan, m_laporan_bimbingan dan *class model* m_berkas. *Aktor* mahasiswa melakukan *submit form* bimbingan, kemudian v_mahasiswa_bimbingan memanggil *function* submit_skripsi() pada c_mahasiswa. *Controller* tersebut selanjutnya akan memanggil beberapa *function* pada m_bimbingan, m_laporan_bimbingan dan m_berkas untuk melakukan *query* ke *database*. Setelah data berhasil disimpan maka

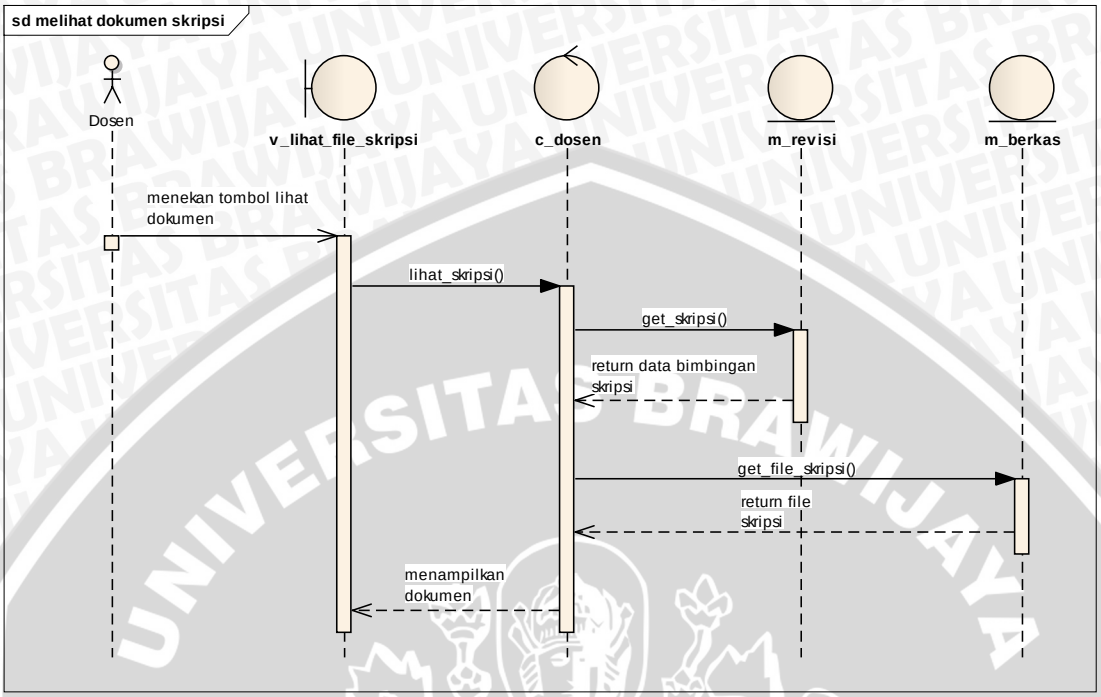
controller akan kembali menampilkan pemberitahuan bahwa skripsi berhasil dikirim. Diagram *sequence submit* dokumen skripsi dapat dilihat pada gambar 4.21.



Gambar 4.21 Diagram *Sequence* Submit Dokumen Skripsi

b. Diagram *Sequence* Melihat Dokumen Skripsi

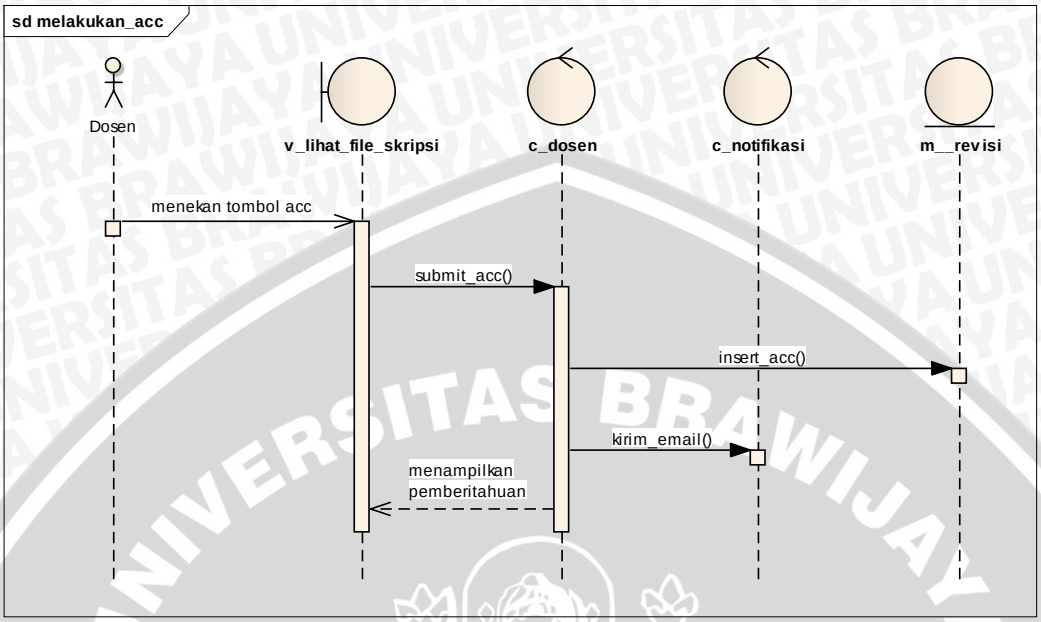
Dosen menekan tombol lihat dokumen. *Function* lihat_skripsi() pada c_dosen akan memanggil *function* get_skripsi() pada m_revisi untuk melakukan *query select* terhadap data mahasiswa terpilih yang tersimpan di *database*. Kemudian memanggil *function* get_file_skripsi() pada m_berkas untuk mengambil dokumen skripsi. Setelah data dokumen skripsi dari m_berkas di *return* atau dikembalikan, maka c*ontroller* akan memanggil v_lihat_file_skripsi untuk menampilkan dokumen skripsi mahasiswa yang dipilih. Diagram *sequence* melihat dokumen skripsi dapat dilihat pada gambar 4.22.



Gambar 4.22 Diagram *Sequence* Melihat Dokumen Skripsi

c. Diagram *Sequence* Melakukan ACC

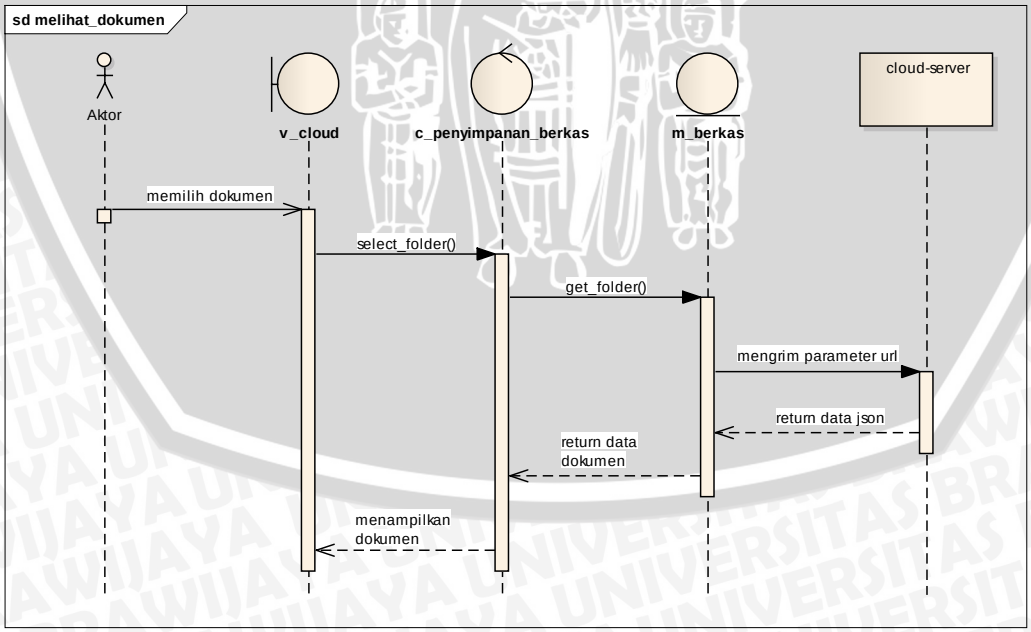
Perancangan diagram *sequence* melibatkan satu *view* v_lihat_file_skripsi, *class controller* c_dosen dan c_notifikasi, seta satu *class model* m_revisi. Setelah *aktor* dosen menekan tombol acc dan melakukan konfirmasi, maka *function* submit_acc() pada c_dosen akan memanggil *function* insert_acc() pada m_revisi untuk melakukan *query update* terhadap status laporan bimbingan mahasiswa yang telah tersimpan di *database* menjadi acc, kemudian memanggil *function* kirim_email() untuk mengirim email. Setelah *update* berhasil dilakukan maka c*ontroller* akan memanggil v_lihat_file_skripsi dan menampilkan pemberitahuan bahwa acc berhasil. Diagram *sequence* melakukan acc dapat dilihat pada gambar 4.23.



Gambar 4.23 Diagram *Sequence* Melakukan ACC

d. Diagram *Sequence* Mengakses Fitur *Cloud*

Perancangan diagram *sequence* melihat dokumen melibatkan satu *view* v_cloud, satu *class* c_penyimpanan_berkas, m_berkas dan *cloud-server api*. Diagram *sequence* melihat dokumen dapat dilihat pada gambar 4.24.



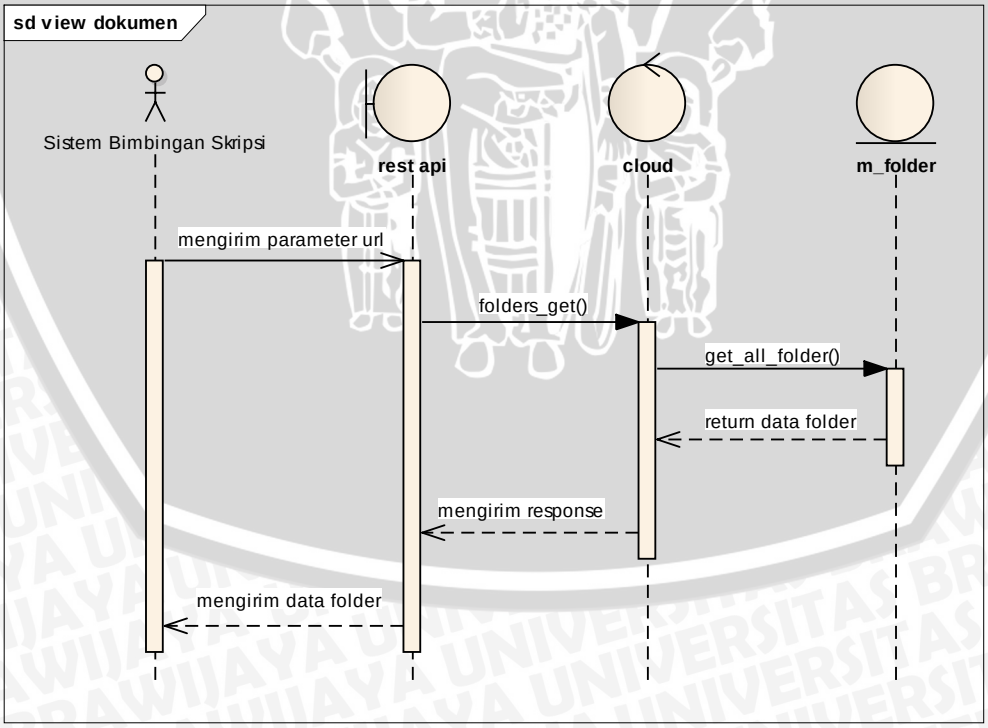
Gambar 4.24 Diagram *Sequence* Melihat Dokumen

4.3.4.2 Diagram *Sequence* Sistem Penyimpanan Berbasis *Cloud*

Diagram *sequence* ini menggambarkan interaksi yang terjadi pada *class* dan objek ketika aktor melakukan suatu *event* terhadap sistem penyimpanan berbasis *cloud*. Aktor yang dimaksud adalah *administrator* dan sistem informasi bimbingan skripsi .

a. Diagram *Sequence View* Dokumen

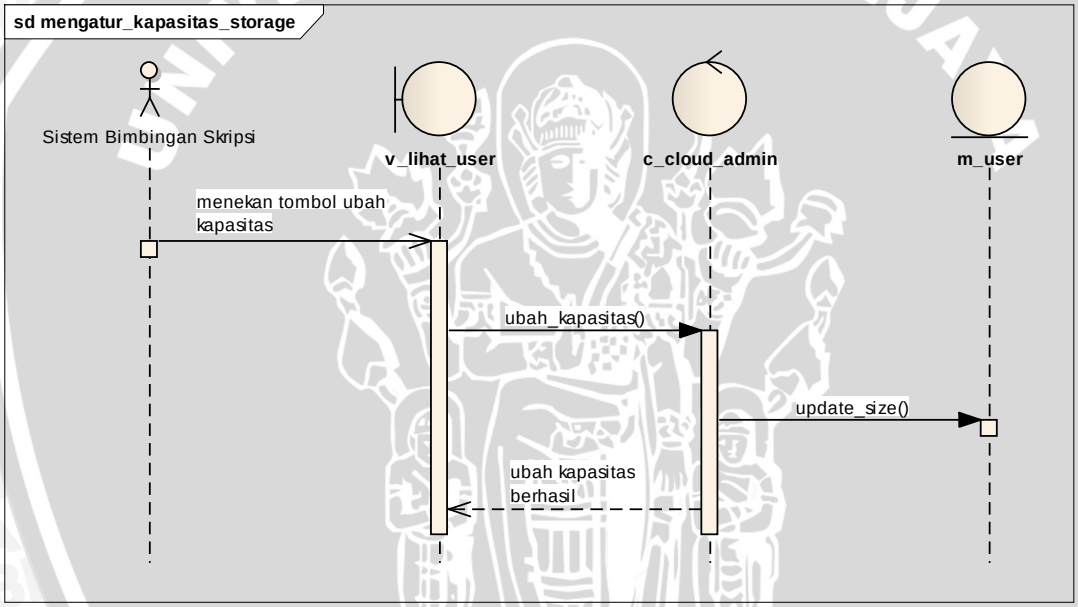
Perancangan diagram *sequence* melibatkan *interface rest api*, *class controller cloud*, dan *class model* m_folder. Sistem informasi bimbingan skripsi melakukan *request* layanan dengan mengirim parameter melalui *url* ke *rest server api*. Selanjutnya *rest api* akan memanggil *function* folders_get() pada *controller* cloud untuk membaca parameter yang dikirim dan memanggil *function* get_all_folder() pada m_folder untuk melakukan *query select* terhadap dokumen yang ada di database *cloud server*. Setelah data di *return*, maka *controller* mengirimnya ke sistem informasi bimbingan skripsi . Diagram *sequence view* dokumen dapat dilihat pada gambar 4.25.



Gambar 4.25 Diagram *Sequence View* Dokumen

b. Mengatur Kapasitas Srorage

Perancangan diagram *sequence* melibatkan satu *view* yaitu v_lihat_user, satu *class controller* c_cloud_admin, dan satu *class model* m_user. Setelah *aktor administrator* menekan tombol ubah kapasitas, maka *function* ubah_kapasitas() akan memanggil *function* updat_size() pada m_user untuk melakukan *query update* terhadap kapasitas penyimpanan *user* yang tersimpan di *database*. Setelah *update* berhasil maka c*ontroller* akan me-*load* v_lihat_user. Diagram *sequence* mengatur kapasitas *storage* dapat dilihat pada gambar 4.26.



Gambar 4.26 Mengatur Kapasitas *Storage*

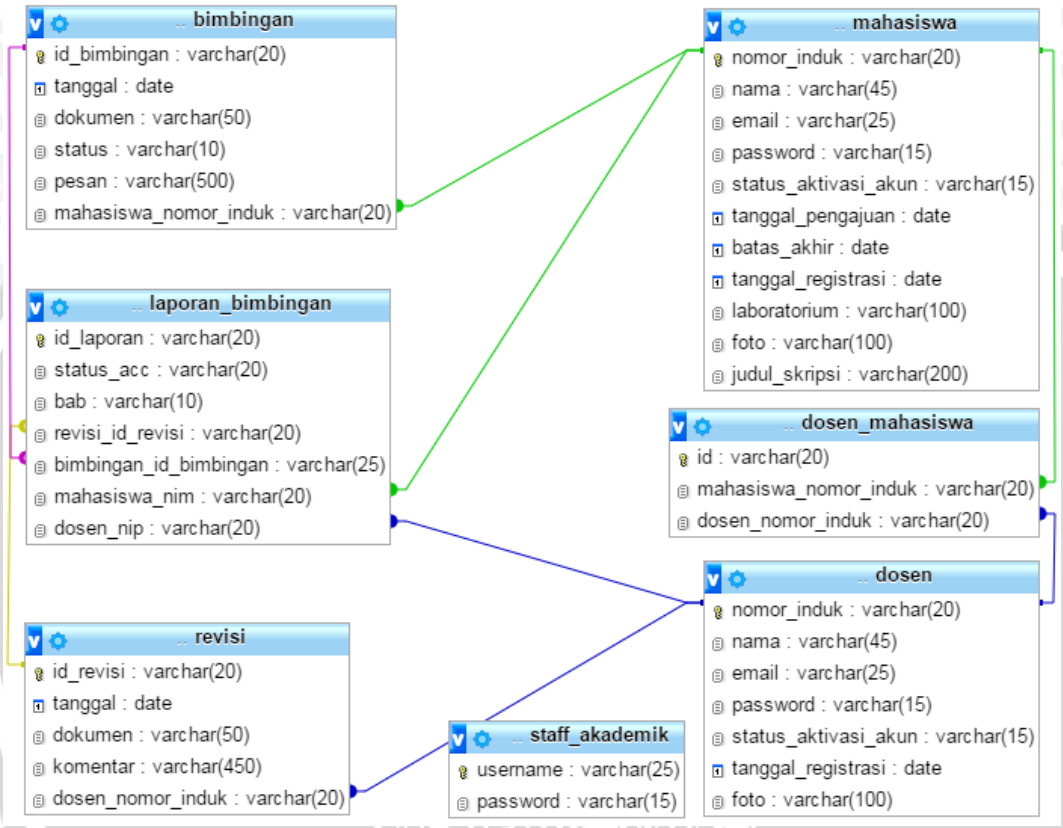
4.3.5 Perancangan Basis Data

Perancangan basis data digunakan untuk merancang basis data yang akan dibuat agar masukan dan keluaran sistem sesuai dengan apa yang diharapkan. Perancangan basis data terdiri dari dua bagian yaitu *relational mapping* dan *normalisasi*. *Normalisasi* sendiri terdiri dari tiga tahap yaitu normalisasi 1NF, normalisasi 2NF, dan normalisasi 3NF.

1. Perancangan Basis Data Sistem Informasi Bimbingan Skripsi

a. Mapping Relational

Dari diagram *class* level analisis yang telah dibuat sebelumnya, maka akan dilakukan *mapping* yang ditunjukkan oleh gambar 4.27.



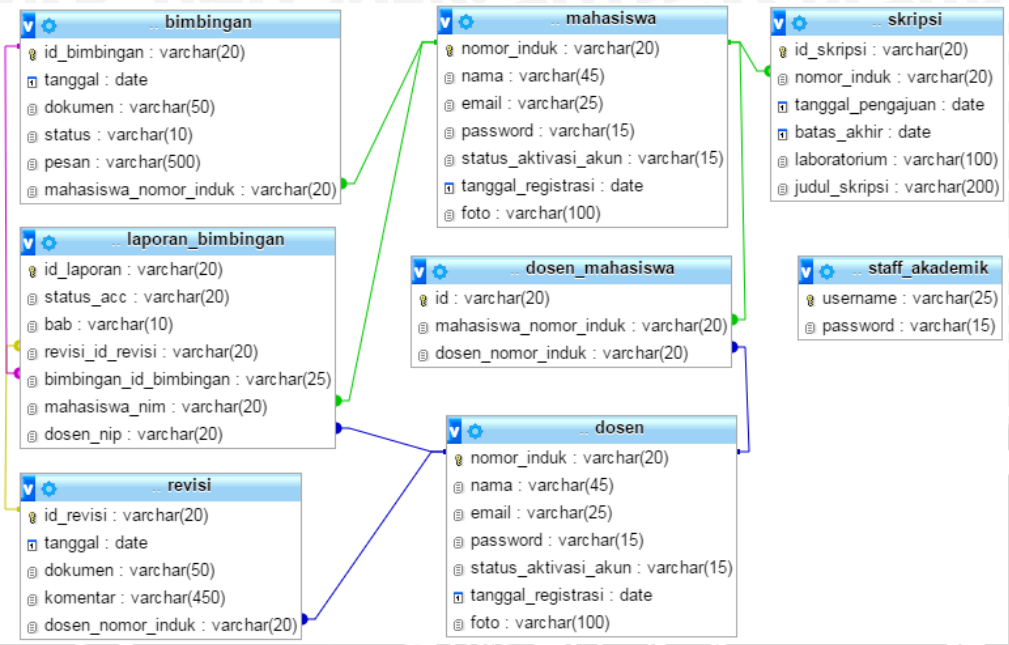
Gambar 4.27 *Mapping* Sistem informasi bimbingan skripsi

b. Normalisasi 1NF

Pada gambar 4.27 bentuk normal pertama sudah tercapai, dimana tidak terdapat atribut bernilai banyak (*multivalues atribute*).

c. Normalisasi 2NF

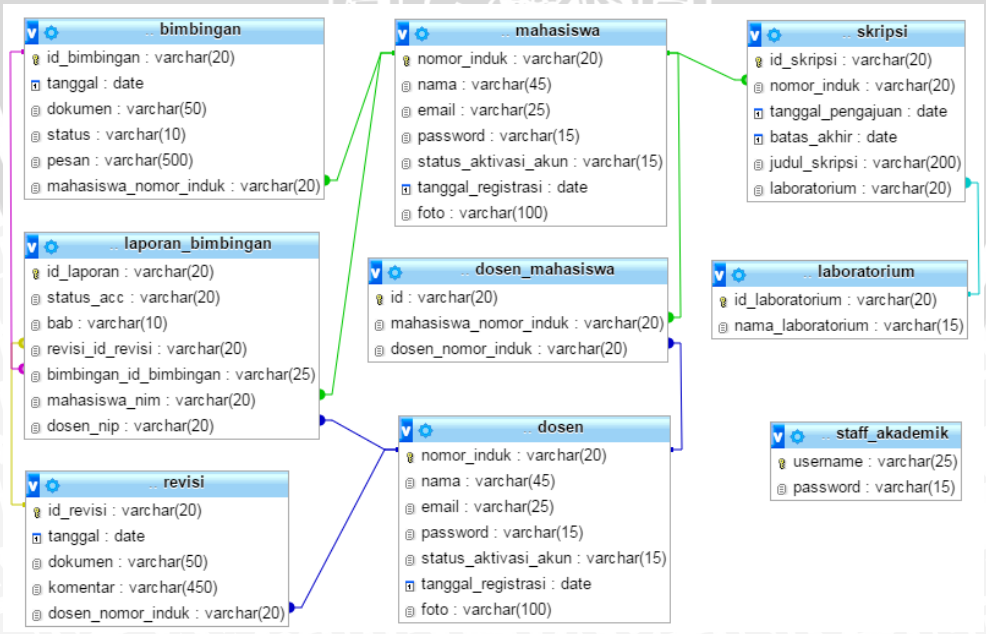
Pada gambar 4.28 bentuk normal kedua tercapai dengan memisahkan atribut judul_skripsi, laboratorium, batas_akhir, dan tanggal pengajuan menjadi tabel baru bernama skripsi.



Gambar 4.28 Bentuk Normal Kedua Sistem informasi bimbingan skripsi

d. Normalisasi 3NF

Pada gambar 4.28 bentuk normal ketiga sudah tercapai, akan tetapi terdapat atribut yang dapat menyebabkan terjadinya redudansi yaitu laboratorium sehingga dilakukan pemisahan menjadi tabel baru.

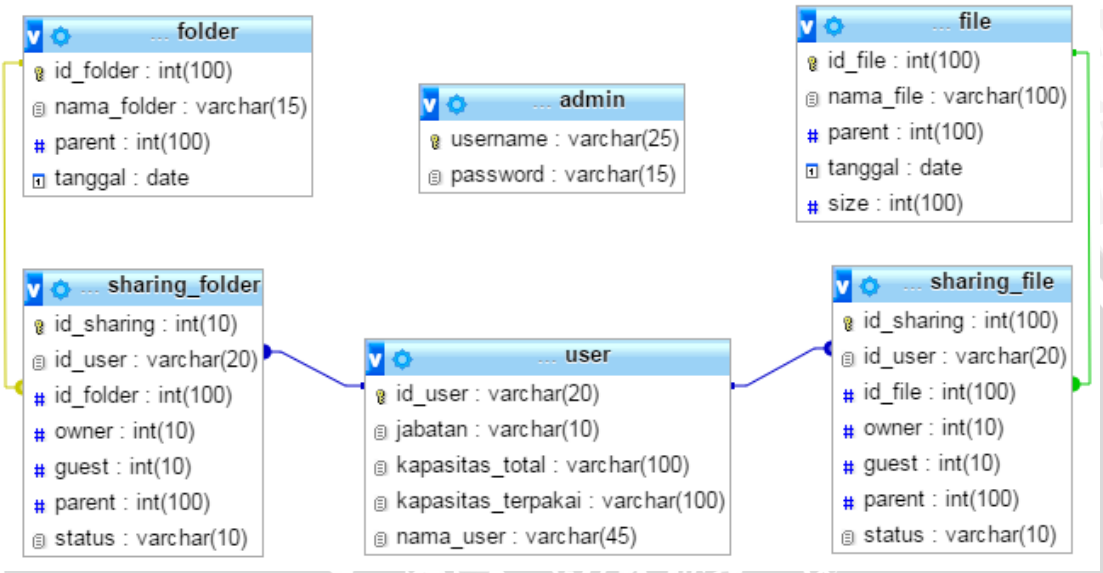


Gambar 4.29 Normalisasi 3NF Sistem informasi bimbingan skripsi

2. Perancangan Basis Data Sistem Penyimpanan Berbasis *Cloud*

a. Mapping Relational

Dari diagram *class* level analisis yang telah dibuat sebelumnya, maka akan dilakukan *mapping* yang ditunjukkan oleh gambar 4.28.



Gambar 4.30 *Mapping Relational* Sistem *Cloud*

b. Normalisasi 1NF

Pada gambar 4.30 bentuk normal pertama sudah tercapai, dimana tidak terdapat atribut bernilai banyak (*multivalues atribute*).

c. Normalisasi 2NF

Pada gambar 4.30 bentuk normal kedua sudah tercapai, dimana tidak terdapat atribut yang memiliki kebergantungan fungsional sebagian.

d. Normalisasi 3NF

Pada gambar 4.30 bentuk normal ketiga sudah tercapai, dimana tidak terdapat atribut yang memiliki kebergantungan transitif.

4.3.6 Perancangan Antarmuka

Pada bagian ini akan dijelaskan tentang perancangan antarmuka sistem. Antarmuka ini digunakan oleh pengguna untuk berinteraksi dengan sistem. Perancangan antarmuka terdiri dari antarmuka sistem informasi bimbingan skripsi dan antarmuka sistem penyimpanan *cloud*.

4.3.6.1 Perancangan Antarmuka Sistem informasi bimbingan skripsi

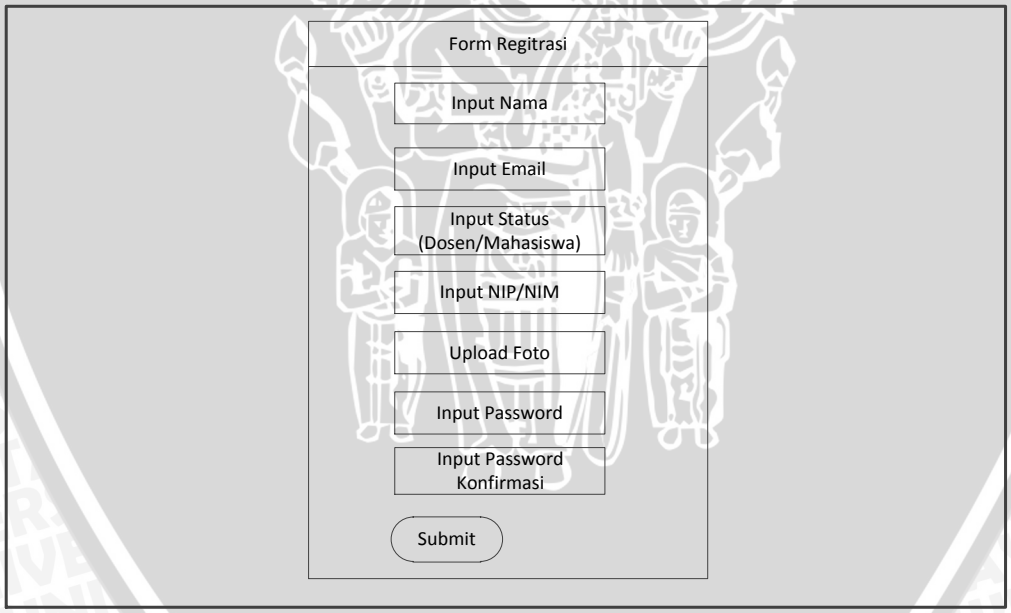
Perancangan antarmuka sistem informasi bimbingan skripsi dibagi menjadi empat, yaitu antarmuka untuk sistem *user*, antarmuka untuk sistem mahasiswa, antarmuka untuk sistem dosen, dan antarmuka untuk sistem akademik.

1. Perancangan Antarmuka Sistem User

Antarmuka pengguna untuk sistem *user* berupa halaman *web* yang menggunakan *framework* PHP *CodeIgniter*.

a. Halaman Registrasi

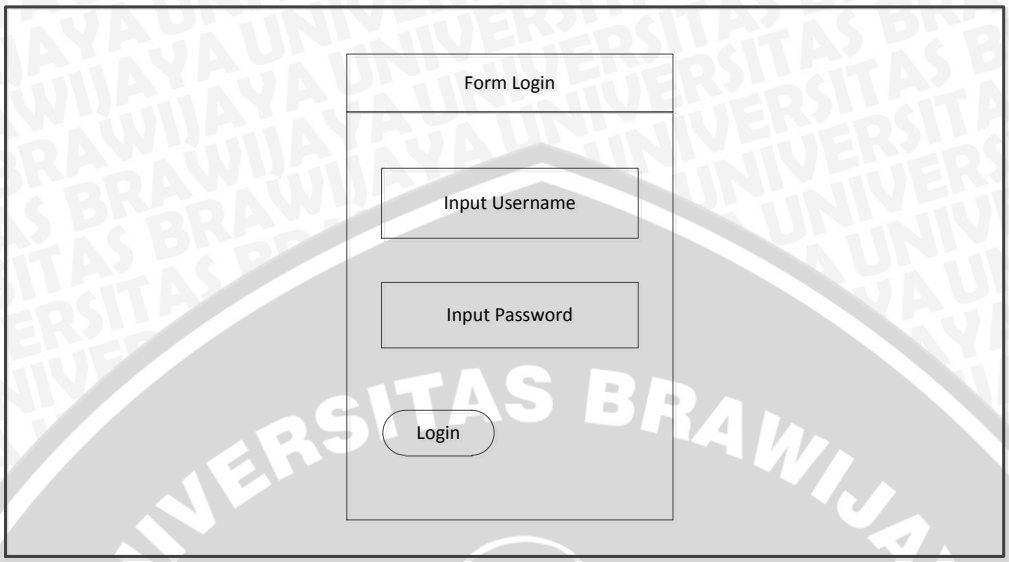
Halaman registrasi merupakan antarmuka bagi *user* untuk melakukan registrasi dengan mengisi *form* yang telah disediakan. Kolom-kolom yang terdapat pada halaman ini antara lain nama, *email*, status (dosen atau mahaiswa), nomor induk, foto, *password*, dan konfirmasi *password*. Antarmuka halaman registrasi ditunjukkan oleh gambar 4.31.



Gambar 4.31 Antarmuka halaman registrasi

b. Halaman Login

Halaman *login* merupakan antarmuka bagi *user* untuk melakukan *login* dengan mengisi *field username* dan *password* pada *form* yang telah disediakan. Antarmuka halaman *login* ditunjukkan oleh gambar 4.32.

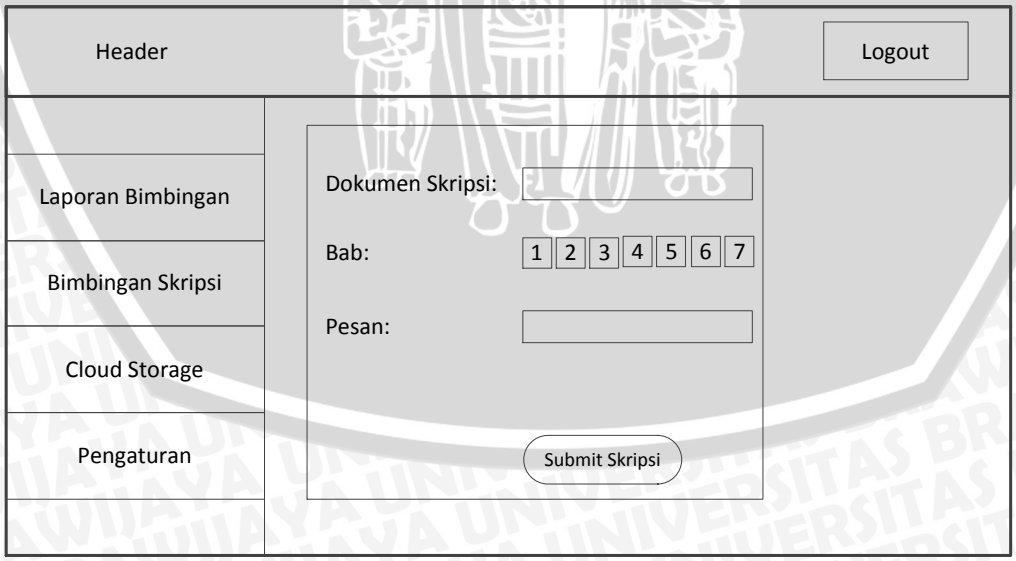


Gambar 4.32 Antarmuka halaman *login*

2. Perancangan Antarmuka Sistem Mahasiswa

a. Submit Dokumen Skripsi

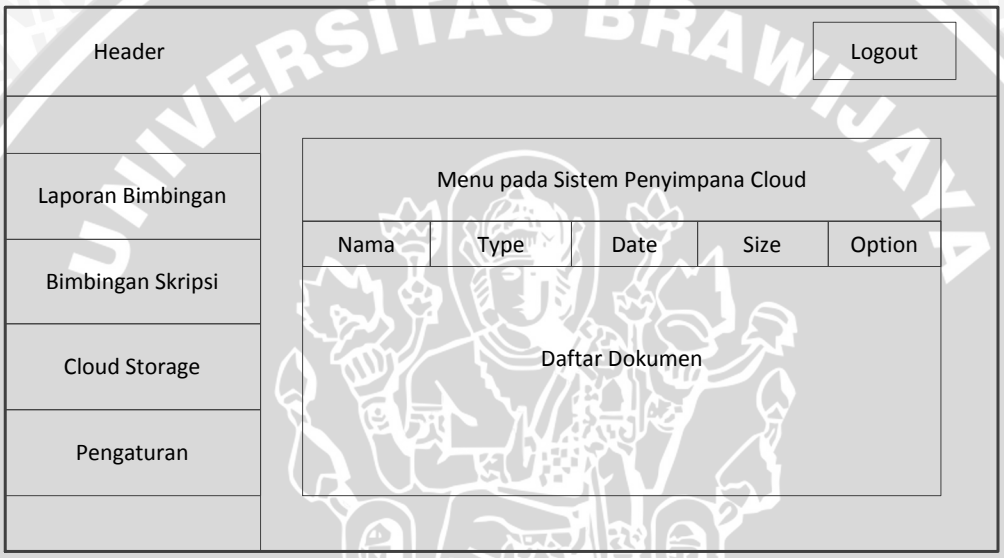
Submit dokumen skripsi merupakan antarmuka bagi mahasiswa untuk melakukan *submit* dokumen skripsi. Pada *field* dokumen skripsi mahasiswa dapat meng-*upload* dokumen skripsi, *field* bab. Antarmuka halaman *submit* dokumen skripsi ditunjukkan oleh gambar 4.33.



Gambar 4.33 Antarmuka halaman *submit* dokumen skripsi

b. Fitur *Cloud*

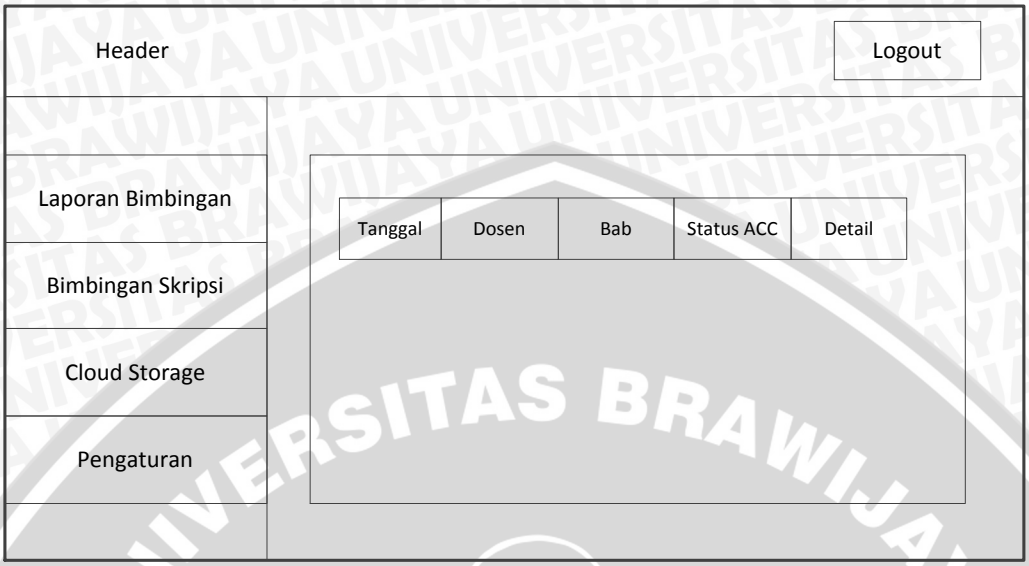
Pada antarmuka fitur *cloud* akan ditampilkan daftar *file* dan *folder* yang telah disimpan beserta rinciannya, antara lain nama dokumen, tipe dokumen, tanggal pembuatan, ukuran, serta *option* untuk menu *delete* untuk menghapus dokumen, menu *share* untuk melakukan sharing dokumen, dan menu *rename* untuk mengubah nama dokumen. Menu-menu tersebut akan ditampilkan dalam bentuk *dropdown*. Antarmuka halaman *cloud* ditunjukkan oleh gambar 4.34.



Gambar 4.34 Antarmuka halaman penyimpanan *cloud*

c. Melihat Laporan Bimbingan

Pada antarmuka fitur *cloud* akan ditampilkan daftar bimbingan laporan bimbingan beserta rinciannya yaitu tanggal untuk menampilkan tanggal bimbingan mahasiswa, dosen untuk menampilkan nama dosen pembimbing, bab untuk menampilkan bab bimbingan mahasiswa, status ACC untuk menunjukkan apakah dokumen bimbingan sudah mendapatkan ACC atau belum, dan menu rincian untuk melihat laporan secara lebih rinci. Antarmuka halaman laporan bimbingan ditunjukkan oleh gambar 4.35.

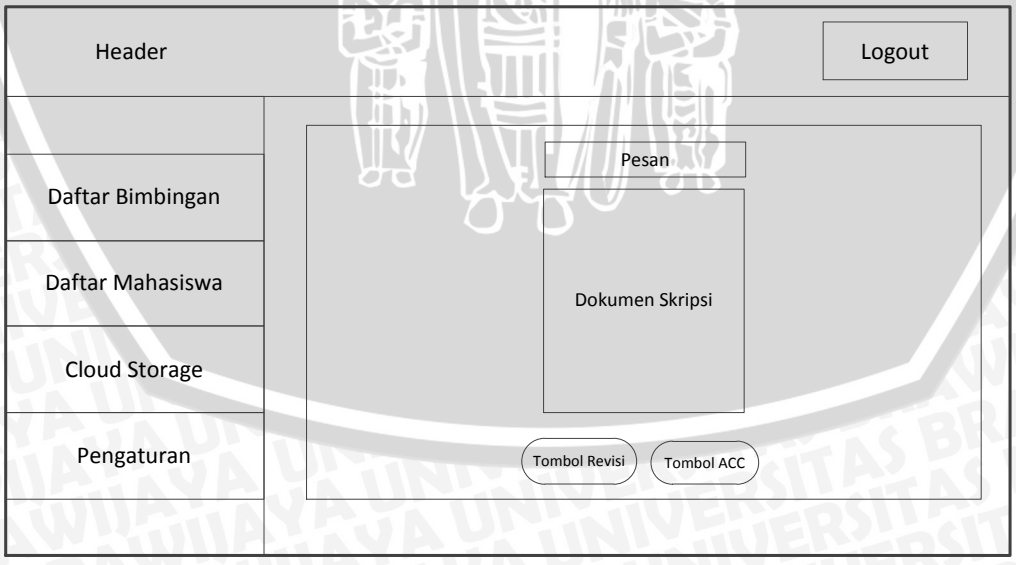


Gambar 4.35 Antarmuka halaman laporan bimbingan

3. Perancangan Antarmuka Sistem Dosen

a. Melihat Dokumen Skripsi

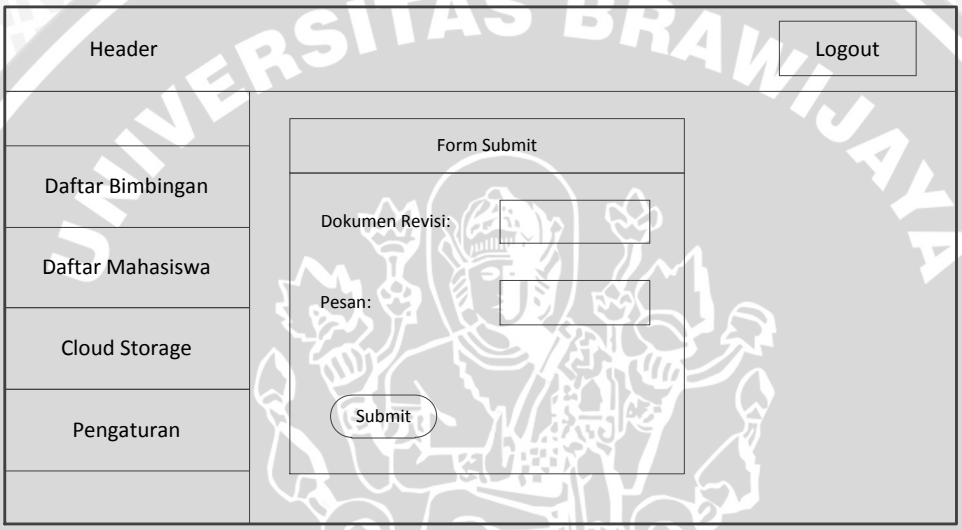
Antarmuka bagi dosen pembimbing untuk melihat dokumen skripsi mahasiswa yang melakukan bimbingan. Pada halaman ini ditampilkan pesan dari mahasiswa, dan dokumen skripsi dalam format pdf. Antarmuka halaman lihat dokumen skripsi ditunjukkan oleh gambar 4.36.



Gambar 4.36 Antarmuka halaman lihat dokumen skripsi

b. Submit Revisi

Submit revisi merupakan antarmuka bagi dosen untuk melakukan *submit* revisi terhadap mahasiswa yang telah melakukan bimbingan skripsi. Pada halaman ini, dosen mengisi pesan yang ingin disampaikan pada *field* pesan dan melakukan *upload* dokumen pada *field* dokumen revisi dengan mengunggah dokumen revisi dari komputer miliknya. Antarmuka halaman lihat dokumen skripsi ditunjukkan oleh gambar 4.37.



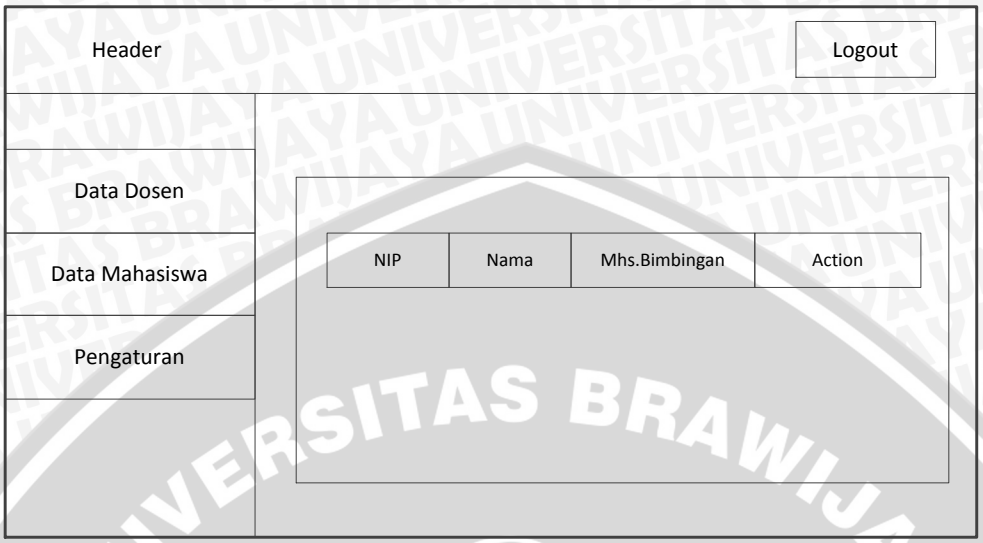
Gambar 4.37 Antarmuka halaman submit revisi

4. Perancangan Antarmuka Sistem Akademik

Perancangan ini merupakan perancangan antarmuka yang digunakan pada sistem untuk *staff* akademik.

a. Melihat Dosen Aktif

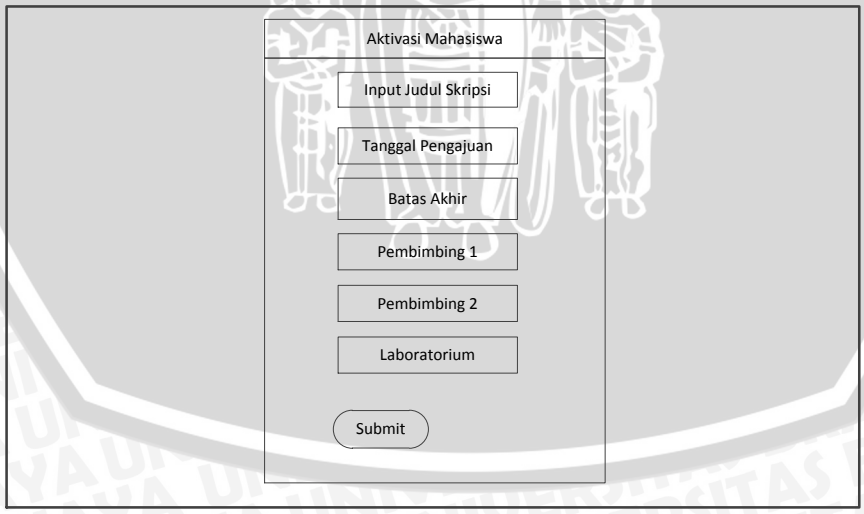
Melihat dosen aktif merupakan antarmuka bagi staff akademik untuk melihat data dosen yang berstatus aktif. Pada halaman ini, akan ditampilkan nama dosen, nip dosen, daftar mahasiswa bimbingan dari dosen yang bersangkutan, serta pilihan untuk hapus dan edit dosen. Antarmuka halaman melihat dosen aktif ditunjukkan oleh gambar 4.38.



Gambar 4.38 Antarmuka halaman lihat dosen aktif

b. Mengaktifkan Akun Mahasiswa

Halaman ini merupakan antarmuka bagi *staff* akademik untuk melakukan aktivasi akun mahasiswa dengan mengisi *form* yang telah disediakan. Form tersebut berisi kolom judul skripsi, tanggal pengajuan skripsi, pembimbing 1, pembimbing 2, dan nama laboratorium. Antarmuka halaman mengaktifkan akun mahasiswa ditunjukkan oleh gambar 4.39.



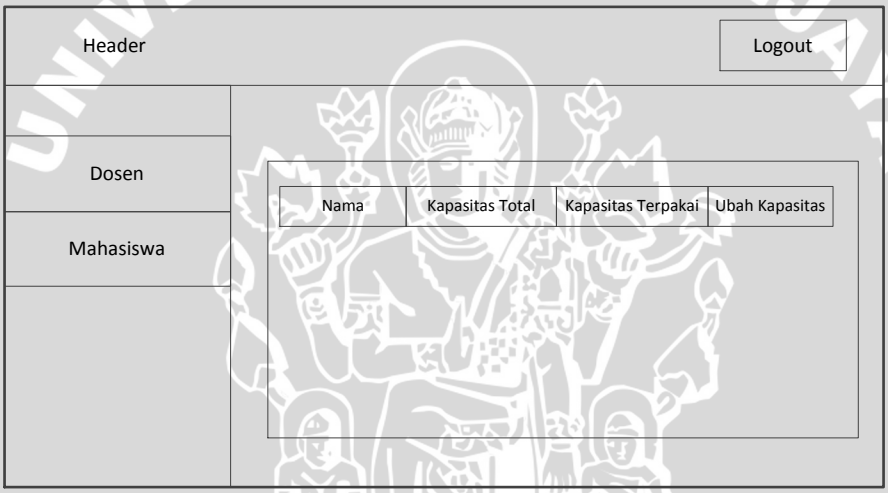
Gambar 4.39 Antarmuka halaman mengaktifkan akun mahasiswa

4.3.6.2 Perancangan Antarmuka Sistem Penyimpanan Berbasis *Cloud*

Perancangan antarmuka untuk administrator pada sistem penyimpanan *cloud* adalah sebagai berikut.

a. Melihat *User*

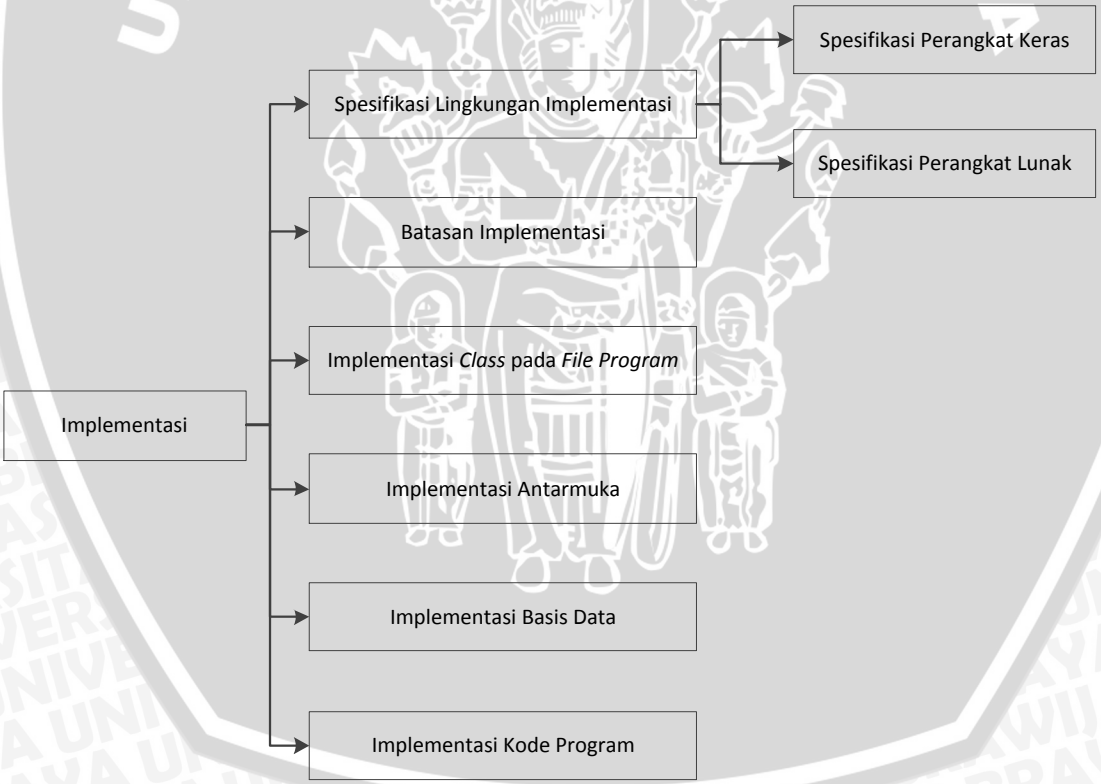
Halaman ini merupakan antarmuka bagi *administrator* untuk melihat *user*. Pada halaman ini akan ditampilkan daftar *user* beserta rinciannya antara lain nama, kapasitas total, dan kapasitas terpakai. Pada halaman ini juga terdapat menu untuk mengubah kapasitas penyimpanan *user*. Antarmuka halaman melihat user ditunjukkan oleh gambar 4.40.



Gambar 4.40 Antarmuka halaman melihat *user*

BAB V
IMPLEMENTASI

Bab ini membahas mengenai implemetasi sistem informasi bimbingan skripsi menggunakan fitur penyimpanan berkas berbasis *cloud*. Implementasi didasarkan pada hasil yang didapat dari proses analisis kebutuhan dan perancangan aplikasi. Pembahasan terdiri atas penjelasan tentang spesifikasi lingkungan implementasi, batasan-batasan dalam implementasi, implementasi tiap *class* pada *file program*, implementasi kode *program*, implementasi basis data, dan implementasi antarmuka sistem. Struktur sub bab implementasi ditunjukkan oleh gambar 5.1.



Gambar 5.1 Struktur Bab Implementasi

5.1 Spesifikasi Lingkungan Implementasi

Sistem informasi bimbingan skripsi dikembangkan dalam lingkungan implementasi yang terdiri dari perangkat keras dan perangkat lunak dengan menggunakan bahasa pemrograman PHP.

5.1.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang dipakai dalam proses pengembangan dijelaskan pada tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras Sistem

Perangkat Keras	Spesifikasi
Type	ASUS A46CM
Processor	Intel Core i3 @1.80 GHz
Memory (RAM)	4.00 GB

5.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang dipakai dalam proses pengembangan dijelaskan pada tabel 5.2.

Tabel 5.2 Spesifikasi Perangkat Lunak

Perangkat Lunak	Jenis
OS	Windows 7 Ultimate 32 bit
Aplikasi	XAMPP 3.2.1, PhpMyAdmin, Notepad++

5.2 Batasan Impelementasi

Beberapa batasan dalam mengimplementasikan sistem informasi bimbingan skripsi PTIIK menggunakan sistem penyimpanan berkas berbasis *cloud* adalah sebagai berikut :

1. *Database Management System* yang digunakan dalam sistem ini adalah MySQL PhpMyAdmin.
2. Sistem menggunakan dua basis data, basis data pertama untuk sistem informasi bimbingan skripsi dan basis data kedua untuk sistem penyimpanan berkas berbasis *cloud*.
3. Sistem menggunakan dua *web server*, *server* pertama untuk sistem informasi bimbingan skripsi dan *server* kedua untuk sistem penyimpanan berkas berbasis *cloud*.

5.3 Implementasi *Class* pada *File Program*

Setiap class yang dirancang pada proses perancangan direalisasikan pada sebuah file prgram dengan ekstensi *.php, baik menggunakan bahas pemrograman PHP maupun HTML. Penjelasan mengenai pasangan antara class dengan file program yang digunakan pada sistem informasi bimbingan skripsi ditunjukkan oleh tabel 5.3. Sedangkan pasangan antara class dengan file program yang digunakan pada sistem penyimpanan *cloud* ditunjukkan oleh tabel 5.4.

Tabel 5.3 Implementasi *class* pada sistem informasi bimbingan skripsi

No.	*Package*	Nama *Class*	Nama *File* Program
1	*Controllers*	C_autentikasi	c_autentikasi.php
2	*Controllers*	C_dosen	c_dosen.php
3	*Controllers*	C_kelola_dosen	c_ kelola_dosen.php
4	*Controllers*	C_kelola_mahasiswa	c_kelola_mahasiswa.php
5	*Controllers*	C_mahasiswa	c_mahasiswa.php
6	*Controllers*	C_pendaftaran	c_ pendaftaran.php
7	*Controllers*	C_penyimpanan_berkas	c_ penyimpanan_berkas.php
8	*Models*	M_berkas	m_berkas.php
9	*Models*	M_dosen	m_dosen.php
10	*Models*	M_laporan_bimbingan	m_laporan_bimbingan.php

11	*Models*	M_mahasiswa	m_mahasiswa.php
12	*Models*	M_revisi	m_revisi.php
13	*Models*	M_bimbingan	m_bimbingan.php

Tabel 5.4 Implementasi *class* pada sistem penyimpanan *cloud*

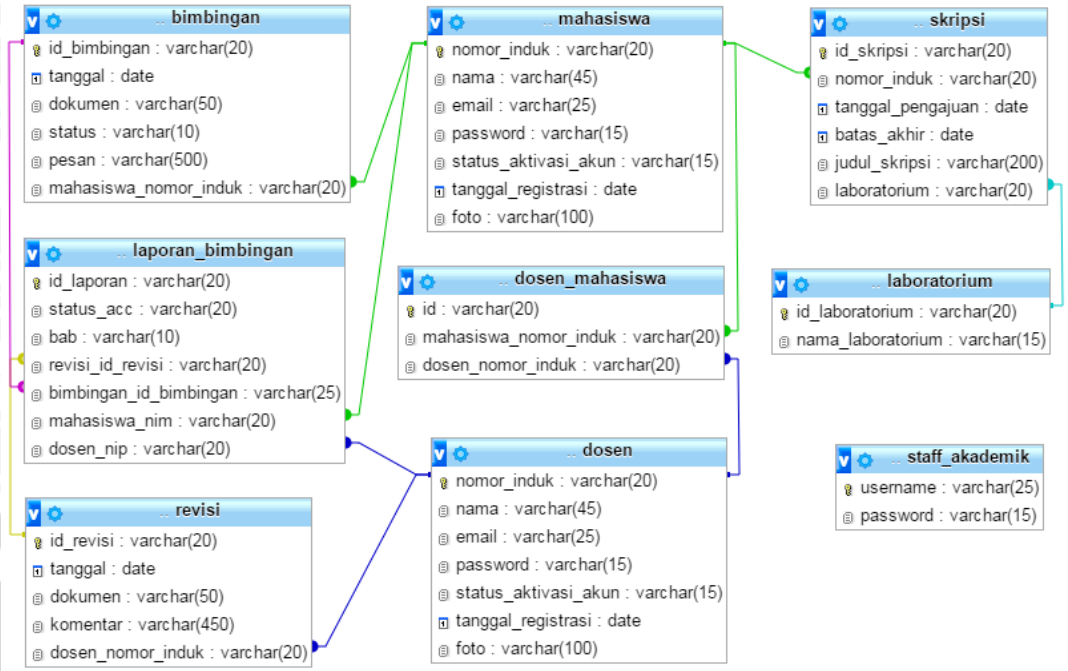
No.	***Package***	**Nama *Class***	**Nama *File* Program**
1	*Controllers*	Cloud	cloud.php
2	*Controllers*	C_cloud_admin	c_cloud_admin.php
3	*Controllers*	C_cloud_login	c_cloud_login.php
4	*Models*	M_user	m_user.php
5	*Models*	M_admin	m_admin.php
6	*Models*	M_folder	m_folder.php
7	*Models*	M_file	m_file.php

5.4 Implementasi Basis Data

Implementasi basis data dilakukan berdasarkan physical data model pada sub bab 4.2.2 menggunakan MySQL. Struktur tabel yang digunakan pada implementasis sistem ini sama dengan perancangan yang telah dibuat sebelumnya.

5.4.1 Implementasi Basis Data Sistem informasi bimbingan skripsi

Basis data ini merupakan basis data yang digunakan untuk menyimpan data bimbingan skripsi. Sesuai dengan perancangan yang telah dijabarkan pada bab analisis dan perancangan, basis data ini memiliki sembilan tabel yaitu tabel dosen, mahasiswa, dosen_mahasiswa, bimbingan, revisi, laporan_bimbingan, skripsi, laboratorium, dan akademik. Struktur tabel beserta atribut pada sistem informasi bimbingan skripsi dapat dilihat pada gambar 5.2.



Gambar 5.2 Implementasi basis data sistem informasi bimbingan skripsi

a. Tabel dosen

Tabel 5.5 Implementasi Tabel Dosen

```
CREATE TABLE IF NOT EXISTS `dosen` (
  `nomor_induk` varchar(20) NOT NULL,
  `nama` varchar(45) DEFAULT NULL,
  `email` varchar(25) DEFAULT NULL,
  `password` varchar(15) DEFAULT NULL,
  `status_aktivasi_akun` varchar(15) DEFAULT NULL,
  `tanggal_registrasi` date DEFAULT NULL,
  `foto` varchar(100) DEFAULT NULL,
   primary key (nomor_induk)
)
```

Pada gambar 5.3 dijelaskan bahwa tabel dosen akan digunakan untuk menyimpan data dosen. Pada tabel ini terdapat beberapa atribut, diantaranya nomor_induk sebagai *primary key*, nama untuk menyimpan nama dosen, *email* untuk menyimpan *email* dosen, *password* untuk menyimpan *password* dosen, status_aktivasi untuk menyimpan status aktivasi terhadap akun dosen,

tanggal_registrasi untuk menyimpan tanggal registrasi dosen ke sistem, dan foto untuk menyimpan data foto dosen.

b. Tabel mahasiswa

Tabel 5.6 Implementasi Tabel Mahasiswa

```
CREATE TABLE IF NOT EXISTS `mahasiswa` (
  `nomor_induk` varchar(20) NOT NULL,
  `nama` varchar(45) DEFAULT NULL,
  `email` varchar(25) DEFAULT NULL,
  `password` varchar(15) DEFAULT NULL,
  `status_aktivasi_akun` varchar(15) DEFAULT NULL,
  `tanggal_registrasi` date DEFAULT NULL,
  `foto` varchar(100) DEFAULT NULL,
  primary key (nomor_induk)
)
```

Pada tabel 5.6 dijelaskan bahwa tabel mahasiswa akan digunakan untuk menyimpan data mahasiswa. Pada tabel ini terdapat beberapa atribut, diantaranya nomor_induk sebagai *primary key*, nama untuk menyimpan nama mahasiswa, *email* untuk menyimpan *email* mahasiswa, *password* untuk menyimpan *password* mahasiswa, status_aktivasi untuk menyimpan status aktivasi terhadap akun mahasiswa, tanggal_registrasi untuk menyimpan tanggal registrasi mahasiswa ke sistem, dan foto untuk menyimpan data foto mahasiswa.

c. Tabel bimbingan

Tabel 5.7 Implementasi Tabel Bimbingan

```
CREATE TABLE IF NOT EXISTS `bimbingan` (
  `id_bimbingan` varchar(20) NOT NULL,
  `tanggal` date DEFAULT NULL,
  `dokumen` varchar(50) DEFAULT NULL,
  `status` varchar(10) DEFAULT NULL,
  `pesan` varchar(500) DEFAULT NULL,
  `mahasiswa_nomor_induk` varchar(20) DEFAULT NULL,
  primary key (id_bimbingan`),
  foreign key `mahasiswa_nomor_induk` references
mahasiswa(`nomor_induk`)
)
```

Pada tabel 5.7 dijelaskan bahwa tabel bimbingan akan digunakan untuk menyimpan data bimbingan mahasiswa. Pada tabel ini terdapat beberapa atribut, diantaranya id_bimbingan sebagai *primary key*, tanggal untuk menyimpan tanggal pada saat mahasiswa melakukan *upload* dokumen bimbingan, dokumen untuk menyimpan data nama dokumen bimbingan yang di-*upload* oleh mahasiswa, status untuk menyimpan data status bimbingan mahasiswa yaitu terkirim atau dibaca, pesan digunakan untuk menyimpan pesan yang ditulis mahasiswa pada saat bimbingan, serta mahasiswa_nomor_induk sebagai *foreign key* yang me-*reference* ke tabel mahasiswa.

d. Tabel Revisi

Tabel 5.8 Implementasi Tabel Revisi

```
CREATE TABLE IF NOT EXISTS `revisi` (
  `id_revisi` varchar(20) NOT NULL,
  `tanggal` date DEFAULT NULL,
  `dokumen` varchar(50) DEFAULT NULL,
  `komentar` varchar(450) DEFAULT NULL,
  `dosen_nomor_induk` varchar(20) DEFAULT NULL,
   primary key (id_revisi``),
   foreign key `dosen_nomor_induk` references
dosen(`nomor_induk`)
)
```

Pada tabel 5.8 dijelaskan bahwa tabel revisi akan digunakan untuk menyimpan data revisi dari dosen pembimbing. Pada tabel ini terdapat beberapa atribut, diantaranya id_revisi sebagai *primary key*, tanggal untuk menyimpan tanggal pada saat dosen melakukan *upload* dokumen revisi, dokumen untuk menyimpan data nama dokumen revisi yang di-*upload* oleh dosen, komentar digunakan untuk menyimpan pesan atau komentar yang ditulis dosen pada saat melakukan revisi terhadap dokumen skripsi mahasiswa, serta dosen_nomor_induk sebagai *foreign key* yang me-*reference* ke tabel dosen.

e. Tabel Laporan_bimbingan

Tabel 5.9 Implementasi Tabel Laporan_bimbingan

```
CREATE TABLE IF NOT EXISTS `laporan_bimbingan` (
  `id_laporan` varchar(20) NOT NULL,
  `status_acc` varchar(20) DEFAULT NULL,
  `bab` varchar(10) DEFAULT NULL,
  `revisi_id_revisi` varchar(20) DEFAULT NULL,
  `bimbingan_id_bimbingan` varchar(25) DEFAULT NULL,
  `mahasiswa_nim` varchar(20) DEFAULT NULL,
  `dosen_nip` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`id_laporan`),
  FOREIGN KEY (`dosen_nip`) REFERENCES `dosen`
(`nomor_induk`),
  FOREIGN KEY (`mahasiswa_nim`) REFERENCES
`mahasiswa` (`nomor_induk`),
  FOREIGN KEY (`revisi_id_revisi`) REFERENCES
`revisi` (`id_revisi`),
  FOREIGN KEY (`bimbingan_id_bimbingan`) REFERENCES
`bimbingan` (`id_bimbingan`)
)
```

Pada tabel 5.9 dijelaskan bahwa tabel laporan_bimbingan akan digunakan untuk menyimpan data laporan bimbingan skripsi mahasiswa. Pada tabel ini terdapat beberapa atribut, diantaranya id_laporan sebagai *primary key*, status_acc digunakan untuk menyimpan status bimbingan skripsi mahasiswa apakah sudah acc atau masih revisi, bab digunakan untuk menyimpan bab dari skripsi yang diupload oleh mahasiswa, revisi_id_revisi me-*reference* id_revisi pada tabel revisi, bimbingan_id_bimbingan me-*reference* id_bimbingan pada tabel bimbingan, mahasiswa_nim me-*reference* nomor_induk pada tabel mahasiswa, dosen_nip me-*reference* nomor_induk pada tabel dosen.

f. Tabel Staff_akademik

Tabel 5.10 Implementasi Tabel Staff_akademik

```
CREATE TABLE IF NOT EXISTS `staff_akademik` (
  `username` varchar(25) NOT NULL,
  `password` varchar(15) DEFAULT NULL,
  PRIMARY KEY (`username`)
)
```

Pada tabel 5.10 dijelaskan bahwa tabel staff_akademik akan digunakan untuk menyimpan data staff akademik. Pada tabel ini terdapat beberapa atribut, diantaranya *username* sebagai *primary key*, dan *password* digunakan untuk menyimpan *password staff* akademik.

g. Tabel Dosen_mahasiswa

Tabel 5.11 Implementasi Tabel Dosen_mahasiswa

```
CREATE TABLE IF NOT EXISTS `dosen_mahasiswa` (
  `id` varchar(20) NOT NULL,
  `mahasiswa_nomor_induk` varchar(20) DEFAULT NULL,
  `dosen_nomor_induk` varchar(20) DEFAULT NULL,
  FOREIGN KEY (`dosen_nomor_induk`) REFERENCES `dosen` (`nomor_induk`),
  FOREIGN KEY (`mahasiswa_nomor_induk`) REFERENCES `mahasiswa` (`nomor_induk`)
)
```

Pada tabel 5.11 dijelaskan bahwa tabel dosen_mahasiswa akan digunakan untuk menyimpan data mahasiswa dan dosen yang membimbingnya. Pada tabel ini terdapat beberapa atribut, diantaranya id sebagai *primary key*, mahasiswa_nomor_induk untuk menyimpan nomor_induk mahasiswa dan dosen_nomor_induk untuk menyimpan nomor_induk dosen yang membimbing mahasiswa tersebut. mahasiswa_nomor_induk me-*reference* nomor_induk pada tabel mahasiswa, dosen_nomor_induk me-*reference* nomor_induk pada tabel dosen.

h. Tabel Laboratorium

Tabel 5.12 Implementasi Tabel Laboratorium

```
CREATE TABLE IF NOT EXISTS `laboratorium` (
  `id_laboratorium` varchar(20) NOT NULL,
  `nama_laboratorium` varchar(15) DEFAULT NULL,
  primary key (`id_laboratorium`)
)
```

Pada tabel 5.12 dijelaskan bahwa tabel laboratorium akan digunakan untuk menyimpan data laboratorium. Pada tabel ini terdapat beberapa atribut, diantaranya id_laboratorium sebagai *primary key*, dan nama_laboratorium untuk menyimpan data nama laboratorium.

i. Tabel Skripsi

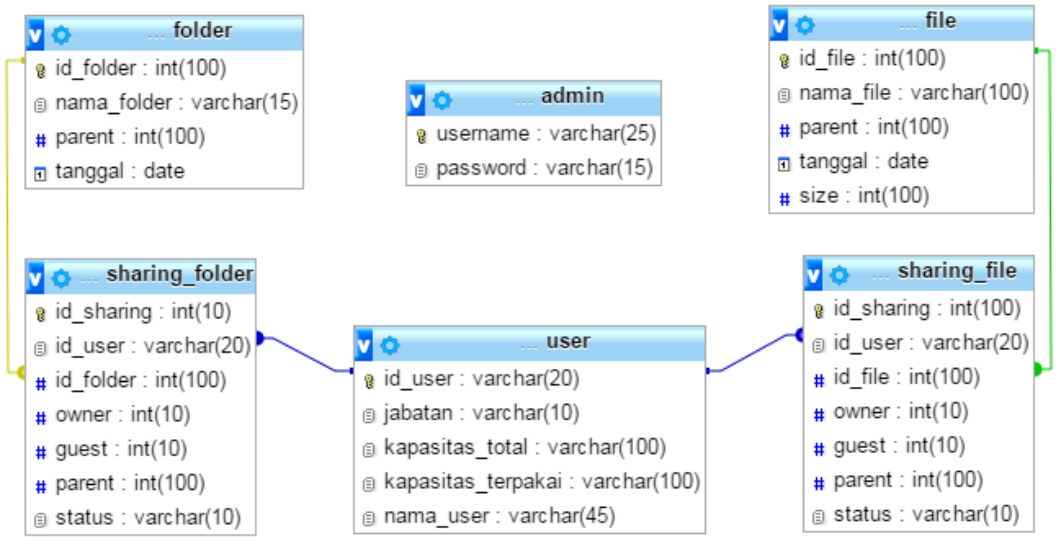
Tabel 5.13 Implementasi Tabel Skripsi

```
CREATE TABLE IF NOT EXISTS `skripsi` (
  `id_skripsi` varchar(20) NOT NULL,
  `nomor_induk` varchar(20) NOT NULL,
  `tanggal_pengajuan` date DEFAULT NULL,
  `batas_akhir` date DEFAULT NULL,
  `judul_skripsi` varchar(200) DEFAULT NULL,
  `laboratorium` varchar(20) DEFAULT NULL,
  FOREIGN KEY (`nomor_induk`) REFERENCES `mahasiswa` (`nomor_induk`),
  FOREIGN KEY (`laboratorium`) REFERENCES `laboratorium` (`id_laboratorium`)
)
```

Pada tabel 5.13 dijelaskan bahwa tabel laboratorium akan digunakan untuk menyimpan data laboratorium. Pada tabel ini terdapat beberapa atribut, diantaranya id_skripsi sebagai *primary key*, nomor_induk untuk menyimpana data NIM mahasiswa, tanggal_pengajuan untuk menyimpan tanggal pengajuan skripsi mahasiswa, batas akhir untuk menyimpan tanggal batas akhir dari skripsi mahasiswa, judul_skripsi untuk menyimpan judul skripsi mahasiswa, laboratorium untuk menyimpan laboratorium skripsi mahasiswa. Atribut nomor_induk dan laboratorium merupakan *foreign key* yang masing-masing me-*reference* tabel mahasiswa dan tabel laboratorium.

5.4.2 Implementasi Basis Data Sistem Penyimpanan *Cloud*

Basis data ini merupakan basis data yang digunakan untuk menyimpan data berkas yang diunggah ke sistem penyimpanan *cloud*. Sesuai dengan perancangan yang telah dijabarkan pada bab analisis dan perancangan, basis data ini memiliki enam tabel yaitu tabel folder, tabel sharing_folder, tabel file, tabel sharing_file, tabel user, dan tabel admin . Struktur tabel pada sistem penyimpanan berbasis *cloud* dapat dilihat pada gambar 5.9.



Gambar 5.3 Implementasi basis data sistem penyimpanan berbasis *cloud*

a. Tabel user

Tabel 5.14 Implementasi Tabel User

```
CREATE TABLE IF NOT EXISTS `user` (
  `id_user` varchar(20) NOT NULL,
  `jabatan` varchar(10) DEFAULT NULL,
  `kapasitas_total` varchar(100) DEFAULT NULL,
  `kapasitas_terpakai` varchar(100) DEFAULT NULL,
  `nama_user` varchar(45) DEFAULT NULL,
  primary key (`id_user`)
)
```

Pada tabel 5.14 dijelaskan bahwa tabel *user* akan digunakan untuk menyimpan data *user* baik itu dosen maupun mahasiswa. Pada tabel ini terdapat beberapa atribut, diantaranya id_user sebagai *primary key*, status digunakan untuk menyimpan status *user* sebagai dosen ataukah mahasiswa, kapasitas_total digunakan untuk menyimpan kapasitas penyimpanan total yang dimiliki *user*, kapasitas_terpakai digunakan untuk menyimpan kapasitas penyimpanan yang telah dipakai oleh *user*, nama_user digunakan untuk menyimpan nama user.

b. Tabel admin

Tabel 5.15 Implementasi Tabel Admin

```
CREATE TABLE IF NOT EXISTS `admin` (
  `username` varchar(25) NOT NULL,
  `password` varchar(15) DEFAULT NULL,
  primary key (`username`)
)
```

Pada tabel 5.15 dijelaskan bahwa tabel admin akan digunakan untuk menyimpan data admin pada sistem penyimpanan *cloud*. Pada tabel ini terdapat beberapa atribut, diantaranya *username* sebagai *primary key*, dan *password* yang digunakan untuk menyimpan *password* admin.

c. Tabel file

Tabel 5.16 Implementasi Tabel File

```
CREATE TABLE IF NOT EXISTS `file` (
  `id_file` int(100) NOT NULL,
  `nama_file` varchar(100) DEFAULT NULL,
  `parent` int(100) DEFAULT NULL,
  `tanggal` date DEFAULT NULL,
  `size` int(100) DEFAULT NULL,
  primary key (`id_file`)
)
```

Pada tabel 5.16 dijelaskan bahwa tabel *file* akan digunakan untuk menyimpan data *file* yang di-*upload* oleh *user*. Pada tabel ini terdapat beberapa atribut, diantaranya id_file sebagai *primary key*, nama_file digunakan untuk menyimpan nama file, *parent* digunakan untuk menyimpan id_folder yang merupakan *parent* dari file yang bersangkutan, tanggal.digunakan untuk menyimpan tanggal *file* dibuat, *size* digunakan untuk menyimpan ukuran file.

d. Tabel sharing_file

Tabel 5.17 Implementasi Tabel Sharing_file

```
CREATE TABLE IF NOT EXISTS `sharing_file` (
  `id_sharing` int(100) NOT NULL,
  `id_user` varchar(20) DEFAULT NULL,
```

```
  `id_file` int(100) DEFAULT NULL,
  `owner` int(10) DEFAULT NULL,
  `guest` int(10) DEFAULT NULL,
  `parent` int(100) DEFAULT NULL,
  `status` varchar(10) DEFAULT NULL,
  primary key (id_sharing`),
  FOREIGN KEY (`id_file`) REFERENCES `file` (`id_file`),
  FOREIGN KEY (`id_user`) REFERENCES `user` (`id_user`)
)
```

Pada tabel 5.17 dijelaskan bahwa tabel sharing_file akan digunakan untuk menyimpan data *file* yang di-*share* oleh *user*. Pada tabel ini terdapat beberapa atribut, diantaranya id_sharing sebagai *primary key*, id_user digunakan untuk menyimpan data *user* yang memiliki hak akses atas *file* yang bersangkutan, owner menunjukkan pemilik *file* yang bersangkutan, guest menunjukkan guest dari *file* yang bersangkutan, status digunakan untuk menyimpan tipe dokumen. Terdapat dua *foreign key* yaitu id_file dan id_user.

e. Tabel folder

Tabel 5.18 Implementasi Tabel Folder

```
CREATE TABLE IF NOT EXISTS `folder` (
  `id_folder` int(100) NOT NULL,
  `nama_folder` varchar(15) DEFAULT NULL,
  `parent` int(100) DEFAULT NULL,
  `tanggal` date DEFAULT NULL,
  primary key (`id_folder`)
)
```

Pada tabel 5.18 dijelaskan bahwa tabel *folder* akan digunakan untuk menyimpan data *folder* yang di-*upload* oleh *user*. Pada tabel ini terdapat beberapa atribut, diantaranya id_folder sebagai *primary key*, nama_folder digunakan untuk menyimpan nama *folder*, *parent* digunakan untuk menyimpan id_folder yang merupakan *parent* dari *folder* yang bersangkutan, tanggal.digunakan untuk menyimpan tanggal *folder* dibuat.

f. Table sharing_folder

Tabel 5.19 Implementasi Tabel Sharing_folder

```
CREATE TABLE IF NOT EXISTS `sharing_folder` (
  `id_sharing` int(10) NOT NULL,
  `id_user` varchar(20) DEFAULT NULL,
  `id_folder` int(100) DEFAULT NULL,
  `owner` int(10) DEFAULT NULL,
  `guest` int(10) DEFAULT NULL,
  `parent` int(100) DEFAULT NULL,
  `status` varchar(10) DEFAULT NULL,
  primary key (`id_sharing`),
  FOREIGN KEY (`id_folder`) REFERENCES `folder`
(`id_folder`),
  FOREIGN KEY (`id_user`) REFERENCES `user` (`id_user`)
)
```

Pada tabel 5.19 dijelaskan bahwa tabel sharing_folder akan digunakan untuk menyimpan data *folder* yang di-*share* oleh *user*. Pada tabel ini terdapat beberapa atribut, diantaranya id_sharing sebagai *primary key*, id_user digunakan untuk menyimpan data *user* yang memiliki hak akses atas *folder* yang bersangkutan, owner menunjukkan pemilik *folder* yang bersangkutan, guest menunjukkan guest dari *folder* yang bersangkutan, status digunakan untuk menyimpan tipe dokumen.

5.5 Implementasi Kode Program

Implementasi kode program yang dicantumkan dalam laporan skripsi ini merupakan lima *method* yang memiliki prioritas tinggi.

5.5.1 Kode Program Aktivasi Mahasiswa

Method aktivasi_mahasiswa ini merupakan *method* dari kelas c_kelola_mahasiswa pada *package controllers*. *Method* ini berfungsi untuk melakukan aktivasi terhadap akun mahasiswa yang telah melaksanakan registrasi. Tabel 5.20 merupakan kode program untuk aktivasi mahasiswa.

Tabel 5.20 Kode Program Aktivasi Mahasiswa

```
1  public function aktivasi_mahasiswa($id)
2  {
3       if($id==0){
4            $tanggal = $this->m_cloud->get_tanggal();
5            foreach($tanggal as $row){
6                 $tgl_sekarang= $row->tanggal;
7            }
8            $id=$this->input->post('nomor_induk');
9            $email=$this->input->post('email');
10           $subjek = "Sistem informasi bimbingan
skripsi ";
11           $pesan = "Akun anda telah diaktivasi,
sehingga anda dapat login dan mulai melaksanakan
bimbingan";
12
13           $data["dosen_nip_1"]=$this->input-
>post('dosen_nip_1');
14           $data["dosen_nip_2"]=$this->input-
>post('dosen_nip_2');
15           $data["tanggal_pengajuan"]=$this->input-
>post('tanggal_pengajuan');
16           $data["batas_akhir"]=$this->input-
>post('batas_akhir');
17           $data["laboratorium"]=$this->input-
>post('laboratorium');
18           $data["judul_skripsi"]=$this->input-
>post('judul');
19           $data["status"]="aktif";
20
21           // create username on cloud
22           session_start();
23           $data2["nama"] = $this->input-
>post('nama_mahasiswa');;
24           $data2["id_user"] = $id;
25           $data2["status"] = "Mahasiswa";
26           $data2["kapasitas_total"] = "2147483648";
27           $data2["kapasitas_terpakai"] = 0;
28
29           $this->m_cloud->add_user($data2);
30
31           // create folder bimbingan
```

```
32              $data3["nama_folder"] = "Bimbingan";
33              $data3["parent"] = 0 ;
34              $data3["id_user"] = $id;
35              $data3["tanggal"] = $tgl_sekarang;
36              $this->m_cloud->insert_folder($data3);
37              // create folder revisi
38              $data4["nama_folder"] = "Revisi";
39              $data4["parent"] = 0 ;
40              $data4["id_user"] = $id;
41              $data4["tanggal"] = $tgl_sekarang;
42              $this->m_cloud->insert_folder($data4);
43              $this->m_user->update_mahasiswa($id,
$data);
44              // send message via email
45              $this->c_akun->kirim_email($email, $subjek,
$pesan);
46              redirect
('c_mahasiswa/lihat_mahasiswa_belum_aktif',
'refresh');
47      }else{
48              $data["nomor_induk"]= $id;
49              $list = $this->m_user->get_mahasiswa();
50              foreach($list as $row){
51              if($data["nomor_induk"] == $row-
>nomor_induk){
52                      $data["nama_mahasiswa"] = $row->nama;
53                      $data["email"] = $row->email;
54              }
55      }
56      $data['list'] = $this->m_user->get_dosen();
57      $this->load-
>view('akademik/v_aktivasi_mahasiswa', $data);
58      }
59      }
```

Penjelasan kode program pada tabel 5.5 adalah sebagai berikut:

Baris 2: apabila id sama dengan nol maka baris ke empat sampai dengan baris ke 47 akan dieksekusi.

Baris 4-7: mengambil data tanggal sekarang dari *database*.

Baris 8-19: mengisi nilai pada variabel.

Baris 22-29: menambahkan user baru pada sistem penyimpanan cloud

Baris 31-36: membuat folder baru dengan nama bimbingan.

Baris 38-42: membuat folder baru dengan nama revisi.

Baris 43: melakukan *update* terhadap data mahasiswa dan mengubah status aktivasi akun di *database* menjadi aktif

Baris 45: mengirim email pemberitahuan ke akun email mahasiswa bahwa akun telah aktif.

Baris 47: apabila id tidak sama dengan nol maka baris ke 48 sampai dengan baris ke 57 akan dieksekusi.

Baris 49-53: mengambil data mahasiswa dari *database*.

Baris 56: mengambil data dosen dari *database*.

Baris 57: menampilkan halaman aktivasi mahasiswa.

5.5.2 Kode Program *Submit* Revisi

Method submit_revisi ini merupakan *method* dari kelas c_revisi pada *package controllers*. *Method* ini berfungsi untuk melakukan revisi terhadap skripsi yang telah di-*submit* oleh mahasiswa, dimana dosen meng-*upload file* untuk revisi kemudian *file* tersebut akan dikirim kepada mahasiswa serta akan tersimpan di *server cloud*. Tabel 5.21 merupakan kode program untuk submit revisi.

Tabel 5.21 Kode Program *Submit* Revisi

```
1  public function submit_revisi(){
2        session_start();
3        $user = $_SESSION["user_id"];
4        $data["nama"] = $_SESSION["nama"];
5        $data["foto"] = $_SESSION["foto"];
6        $data["nim"] = $_SESSION["user_id"];
7        $nim_mhs= $this->input->post('nim_mhs');
8        $id_bimbingan= $this->input-
>post('id_bimbingan');
9        $id = $this->m_cloud-
>get_folder_revisi($nim_mhs);
10       $json_idr = json_decode($id, true);
11       foreach($json_idr as $row ){
12             $id_folder = $row['id_folder'];
```

```
13        }
14        $tanggal = $this->m_cloud->get_tanggal();
15        foreach($tanggal as $row2){
16              $tgl_sekarang= $row2->tanggal;
17        }
18
19        $RealTitleID = $_FILES['nama_file']['name'];
20        $data["size"] = $_FILES['nama_file']['size'];
21        $data['file'] = new
CurlFile($_FILES['nama_file']['tmp_name'],'file/exgp
d',$RealTitleID);
22        $data["parent"]=$id_folder;
23        $data["id_user"]=$nim_mhs;
24        $data["tanggal"]=$tgl_sekarang;
25        $data2["nama_file"]=
$_FILES['nama_file']['name'];
26        $data2["parent"]=$id_folder;
27        $this->m_cloud->insert_file($data);
28        $id2 = $this->m_cloud-
>get_file_revisi($data2["parent"],
$data2["nama_file"]);
29        $json_idr2 = json_decode($id2, true);
30        foreach($json_idr2 as $row6 ){
31              $id_file = $row6['id_file'];
32        }
33        $data3["id_revisi"]= $id_bimbingan;
34        $data3["tanggal"]=$tgl_sekarang;
35        $data3["dokumen"]=$id_file;
36        $data3["komentar"]=$this->input->post('pesan');
37        $data4["revisi_id_revisi"]=
$data3["id_revisi"];
38        $this->m_bimbingan->insert_revisi($data3);
39        $this->m_bimbingan-
>update_laporan($data4["revisi_id_revisi"],
$data["nim"]);
40        $this->load->view('dosen/v_revisi_berhasil',
$data);
41        $email=$this->input->post('email');
42        $subjek = "Sistem informasi bimbingan skripsi
";
43        $pesan = "Skripsi telah mendapat Revisi dari
dosen pembimbing";
```

```
44        $this->c_akun->kirim_email($email, $subjek,
$pesan);
45  }
```

Penjelasan kode program pada tabel 5.6 adalah sebagai berikut:

Baris 2-8: mengisi nilai variabel dengan nilai yang ada pada *session* dan *method post*.

Baris 9: mengambil data folder revisi milik mahasiswa.

Baris 10-13: melakukan parsing json terhadap data folder revisi milik mahasiswa yang bersangkutan.

Baris 14-17: mengambil data tanggal sekarang dari *database*.

Baris 19-26: setting *variabel* untuk *upload file*.

Baris 27: melakukan *insert file* revisi ke *folder* revisi milik mahasiswa pada sistem *cloud*.

Baris 28-31: mengambil data id file revisi.

Baris 38: melakukan *insert* data revisi ke *database*.

Baris 39: melakukan *update* data laporan bimbingan.

Baris 41-44: mengirim *email* pemberitahuan kepada mahasiswa bahwa revisi telah dilakukan.

5.5.3 Kode Program *Upload File*

Method add_file_proses ini merupakan *method* dari kelas c_cloud pada *package controllers*. *Method* ini berfungsi untuk melakukan upload *file* ke *server cloud*. Tabel 5.22 merupakan kode program untuk *upload file*.

Tabel 5.22 Kode Program *Upload File*

```
1   public function add_file_proses(){
2       session_start();
3       $user = $_SESSION["user_id"];
4       $data["nama"] = $_SESSION["nama"];
5       $data["foto"] = $_SESSION["foto"];
6       $data["nim"] = $_SESSION["user_id"];
7       $size = $this->m_cloud->get_space($user );
8       $json_idr = json_decode($size, true);
9       foreach($json_idr as $row ){
10          $data['kapasitas_total'] =
```

```
    $row['kapasitas_total'];
11              $data['kapasitas_terpakai'] =
$row['kapasitas_terpakai'];
12          }
13          $size = $_FILES['nama_file']['size'];
14          $data["kapasitas_upload"]=$size+$data['kapasitas
_terpakai'];
15          if($data["kapasitas_upload"] <=
$data['kapasitas_total']){
16              $tanggal = $this->m_cloud->get_tanggal();
17              foreach($tanggal as $row){
18              $tgl_sekarang= $row->tanggal;
19              }
20              if(isset($_SESSION["parent"]) and
$_SESSION["parent"] !=0){
21              $parent=$_SESSION["parent"];
22              $data = array();
23              $uploadDir = "file/";
24              $RealTitleID =
$_FILES['nama_file']['name'];
25
26              $data["size"] =
$_FILES['nama_file']['size'];
27              $data['file'] = new
CurlFile($_FILES['nama_file']['tmp_name'],'file/exgpd
',$RealTitleID);
28              $data["parent"]=$parent;
29              $data["id_user"]=$_SESSION["user_id"];
30              $data["tanggal"]=$tgl_sekarang;
31              $this->m_cloud->insert_file($data);
32              redirect
('C_cloud/select_folder_direct','refresh');
33              } else{
34              $data = array();
35              $parent =0;
36              $RealTitleID =
$_FILES['nama_file']['name'];
37              $data["size"] =
$_FILES['nama_file']['size'];
38              $data['file'] = new
CurlFile($_FILES['nama_file']['tmp_name'],'file/exgpd
',$RealTitleID);
```

```
39              $data["parent"]=$parent;
40              $data["id_user"]=$_SESSION["user_id"];
41              $data["tanggal"]=$tgl_sekarang;
42              $this->m_cloud->insert_file($data);
43              redirect ('C_cloud/dasboard','refresh');
44              }
45          }else{
46              $this->load->view('cloud/v_upload_gagal',
$data);
47          }
48  }
```

Penjelasan kode program pada tabel 5.7 adalah sebagai berikut:

Baris 7-12: mengambil data kapasitas total dan kapasitas terpakai milik *user*.

Baris 13: mengambil ukuran *file* yang di-*upload* oleh *user*.

Baris 14: menjumlahkan kapasitas yang dipakai dan ukuran *file* yang di-*upload*.

Baris 15: apabila hasil penjumlahan pada baris 14 kurang dari kapasitas total maka baris 16 sampai 32 akan dieksekusi dalam arti perintah untuk *upload file* akan dijalankan.

Baris 45: apabila hasil penjumlahan pada baris 14 lebih dari kapasitas total maka baris 46 akan dieksekusi dalam arti akan ditampilkan pemberitahuan bahwa *upload file* gagal.

5.5.4 Kode program Laporan Bimbingan

Method detail_laporan ini merupakan *method* dari kelas c_bimbingan pada *package controllers*. *Method* ini berfungsi untuk melihat detail laporan bimbingan skripsi yang telah dilakukan oleh mahasiswa. Tabel 5.23 merupakan kode program untuk melihat laporan bimbingan.

Tabel 5.23 Kode Program Laporan Bimbingan

```
1   public function detail_laporan($id){
2       session_start();
3       $user = $_SESSION["user_id"];
4       $data["nama"] = $_SESSION["nama"];
5       $data["foto"] = $_SESSION["foto"];
6       $data["nim"] = $_SESSION["user_id"];
```

```
7       $data['id_bimbingan'] = $this->input->post('id_bimbingan');
8       $data['tanggal_bimbingan'] = $this->input->post('tanggal_bimbingan');
9       $data['nama_dosen'] = $this->input->post('nama_dosen');
10      $data['bab'] = $this->input->post('bab');
11      $data['dokumen1'] = $this->input->post('dokumen_bimbingan');
12      $data['nim'] = $this->input->post('nim');
13      $data['pesan'] = $this->input->post('pesan');
14      $id2 = $this->m_cloud->get_file_skripsi($data['dokumen1']);
15      if($id2 != NULL){
16          $json_idr = json_decode($id2, true);
17          foreach($json_idr as $row ){
18              $data['dokumen_bimbingan'] = $row['nama_file'];
19          }
20      }else{
21          $data['dokumen_bimbingan'] = "-";
22      }
23      if (isset($_POST['nim_mhs'])){
24          $data['nim']= $this->input->post('nim_mhs');
25      }else{
26          $data['nim']=$user;
27      }
28      $list2 = $this->m_bimbingan->get_revisi($data['id_bimbingan']);
29      if($list2 != NULL){
30          foreach($list2 as $row2){
31              $data["dokumen2"] = $row2->dokumen;
32              $data["tanggal_revisi"] = $row2->tanggal;
33              $data["komentar"] = $row2->komentar;
34              $id3 = $this->m_cloud->get_file_skripsi($data['dokumen2']);
35              if($id3 != NULL){
36                  $json_idr2 = json_decode($id3, true);
37                  foreach($json_idr2 as $row3 ){
```

```
38                          $data['dokumen_revisi'] =
$row3['nama_file'];
39                          }
40                      }else{
41                          $data['dokumen_revisi'] = "-";
42                      }
43              }
44              }else{
45                      $data["dokumen_revisi"] = "-";
46                      $data["tanggal_revisi"] = "-";
47                      $data["komentar"] = "-";
48              }
49              if($id==1){
50                      $this->load-
>view('dosen/v_detail_laporan', $data);
51              }else{
52                      $this->load-
>view('bimbingan/v_detail_laporan', $data);
53              }
54      }
```

Penjelasan kode program pada tabel 5.8 adalah sebagai berikut:

Baris 7-13: mengisi variabel dengan data dari method post.

Baris 14-22: mengambil data file bimbingan dari cloud server.

Baris 28-43: mengambil data revisi.

Baris 49-50: jika id sama dengan satu maka akan di load halaman dosen.

Baris 51-52: jika id tidak sama dengan satu maka akan di load halaman mahasiswa.

5.5.5 Kode Program Melihat Dokumen Skripsi

Method liha_skripsi ini merupakan *method* dari kelas c_bimbingan pada *package controllers*. *Method* ini berfungsi untuk melihat dokumen skripsi yang telah disubmit oleh mahasiswa. Tabel 5.24 merupakan kode program untuk melihat dokumen skripsi.

Tabel 5.24 Kode Program Melihat Dokumen Skripsi

```
1   public function lihat_skripsi(){
2       session_start();
3       $user = $_SESSION["user_id"];
```

```
4       $data["nama"] = $_SESSION["nama"];
5       $data["foto"] = $_SESSION["foto"];
6       $data["nim"] = $_SESSION["user_id"];
7       $data["nim_mhs"] = $this->input->post('nim_mhs');
8       $data["email"] = $this->input->post('email');
9       $id = $this->input->post('id_bimbingan');
10      $list1 = $this->m_bimbingan->get_skripsi($id);
11      foreach($list1->result() as $row1)
12      {
13              $data["id_file"] = $row1->dokumen;
14              $data["id_bimbingan"] = $row1->id_bimbingan;
15              $data["pesan"] = $row1->pesan;
16      }
17      $id2 = $this->m_cloud->get_file_skripsi($data["id_file"]);
18      $json_idr = json_decode($id2, true);
19      foreach($json_idr as $row2 ){
20              $data["nama_file"] = $row2['nama_file'];
21      }
22      $this->load->view('dosen/v_lihat_file_skripsi', $data);
23  }
```

Penjelasan kode program pada tabel 5.11 adalah sebagai berikut:

Baris 3-6: mengisi variabel dengan nilai yang ada pada *session*.

Baris 7-9: mengisi variabel dengan nilai yang ada pada *method post*.

Baris 10-16: mengambil data skripsi dari mahasiswa yang melakukan bimbingan.

Baris 17: mengambil *file* skripsi yang ada di *server cloud*.

Baris 18-21: melakukan *parsing* json terhadap data yang diambil pada baris 17.

Baris 22: melakukan *load* terhadap *view* untuk menampilkan file skripsi.

5.6 Implementasi Antarmuka

Pada sub bab ini akan dijelaskan implementasi antarmuka yang terdiri dari antarmuka sistem informasi bimbingan skripsi dan antarmuka sistem penyimpanan *cloud*.

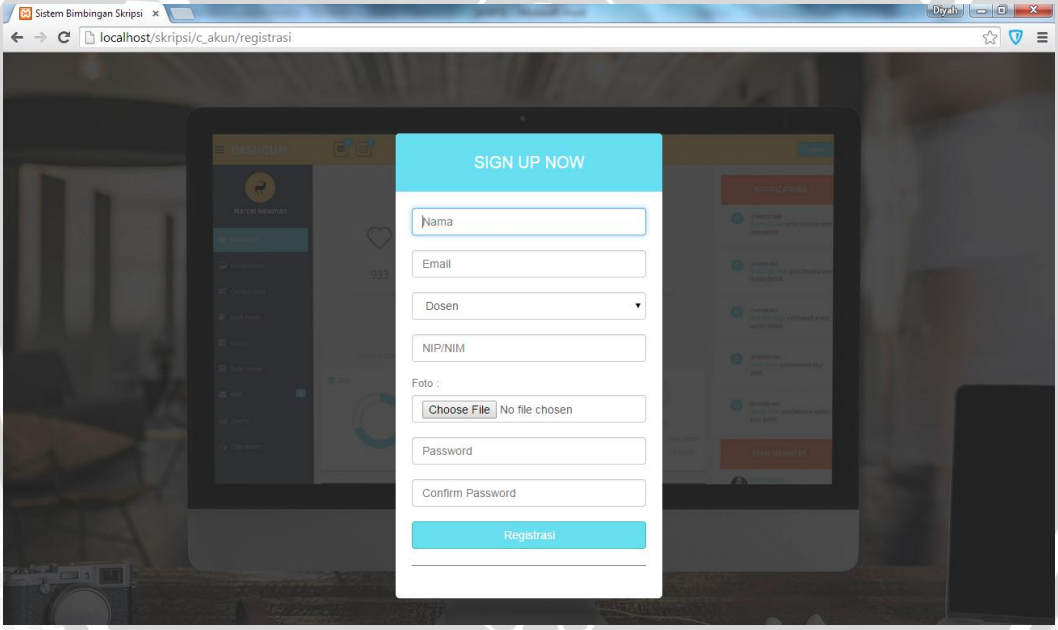
5.6.1 Implementasi Antarmuka Sistem informasi bimbingan skripsi

Implementasi antarmuka sistem informasi bimbingan skripsi terdiri dari dari sistem user, sistem dosen, sistem mahasiswa, dan sistem untuk *staff* akademik. Pada sub bab ini akan dijelaskan antarmuka utama dari sistem informasi bimbingan skripsi

5.4.1.1 Implementasi Antarmuka Sistem *User*

a. Melakukan Registrasi

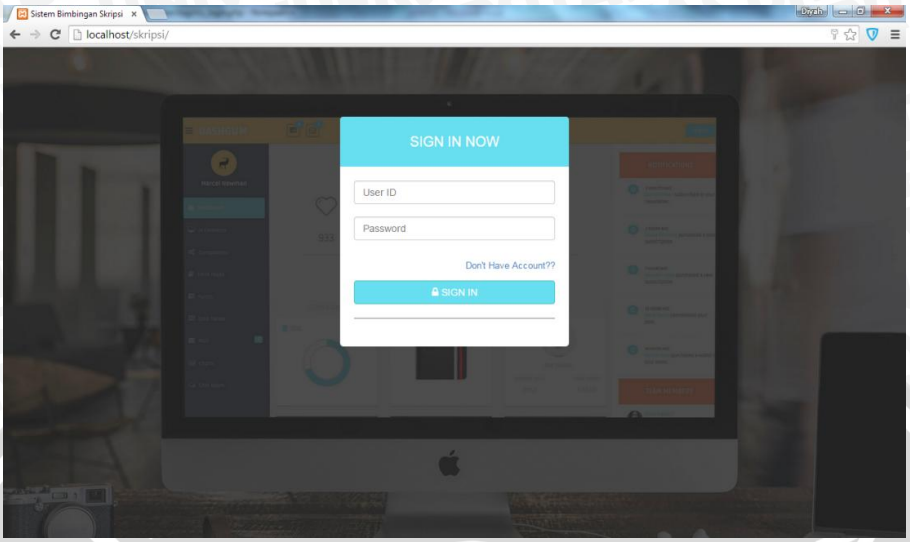
Halaman registrasi merupakan halaman bagi *user* untuk melakukan registrasi. Pada halaman registrasi terdapat *form input* nama, *email*, status dalam hal ini dosen ataukah mahasiswa, NIM/NIP, foto, *password*, *input* untuk konfirmasi password, dan button registrasi. Antarmuka melakukan registrasi ditunjukkan oleh gambar 5.4.



Gambar 5.4 Antarmuka melakukan registrasi

b. Login

Halaman *login* merupakan halaman bagi *user* untuk melakukan *login* ke sistem. Pada halaman login terdapat *form input user id*, *password*, dan button *sign in*. Antarmuka melakukan registrasi ditunjukkan oleh gambar 5.5.

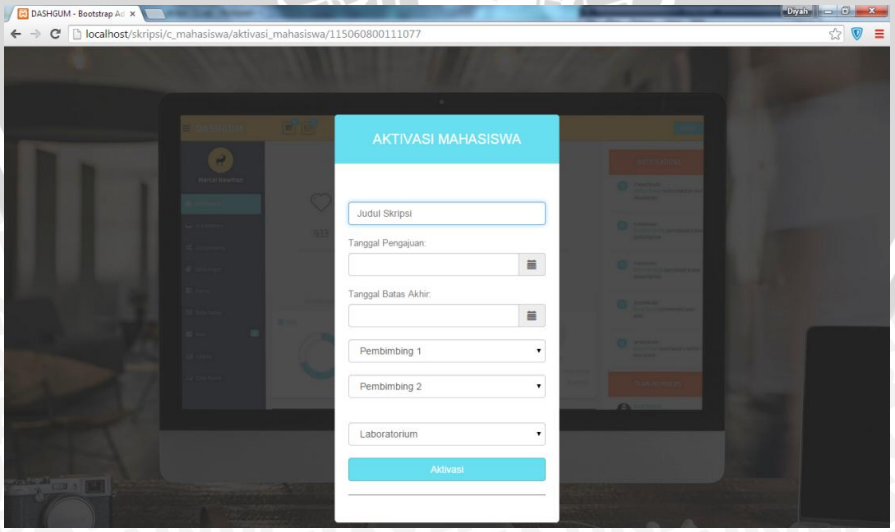


Gambar 5.5 Antarmuka login

5.4.1.2 Implementasi Antarmuka Sistem Akademik

a. Mengaktifkan Akun Mahasiswa

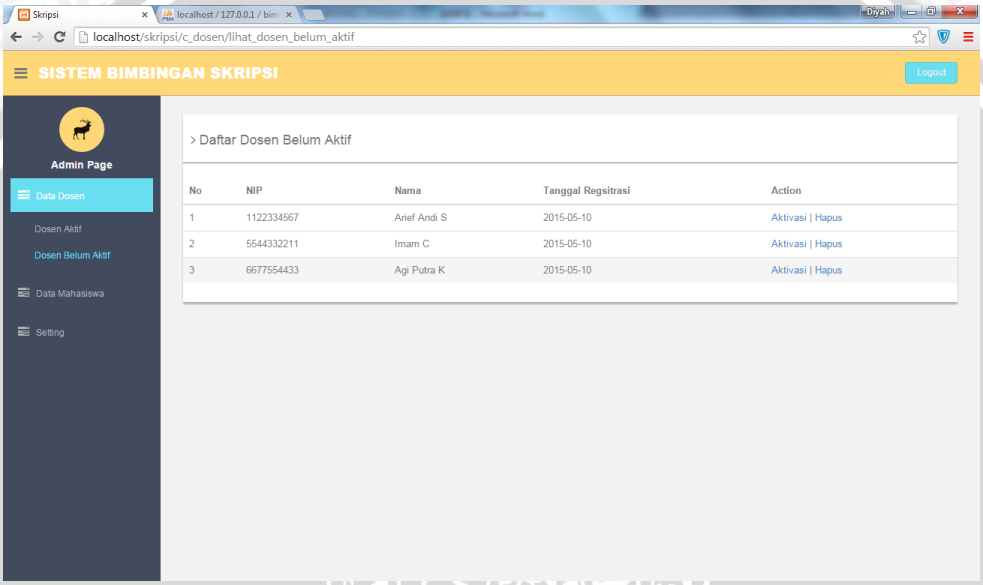
Halaman ini merupakan halaman bagi *staff*_akademik untuk melakukan aktivasi terhadap akun mahasiswa. Pada halaman aktivasi terdapat *form input* judul skripsi, tanggal pengajuan, tanggal batas akhir, pembimbing satu, pembimbing dua, serta laboratorium skripsi mahasiswa. Antarmuka aktivasi mahasiswa ditunjukkan oleh gambar 5.6.



Gambar 5.6 Antarmuka aktivasi mahasiswa

b. Mengaktifkan Akun Dosen

Halaman ini merupakan halaman bagi *staff*_akademik untuk melakukan aktivasi terhadap akun dosen. Pada halaman ini akan ditampilkan daftar dosen yang belum aktif, selanjutnya disebelah kanan terdapat tombol aktivasi dan hapus. Untuk melakukan aktivasi maka staff_akademik harus menekan tombol aktivasi. Antarmuka aktivasi dosen ditunjukkan oleh gambar 5.7.

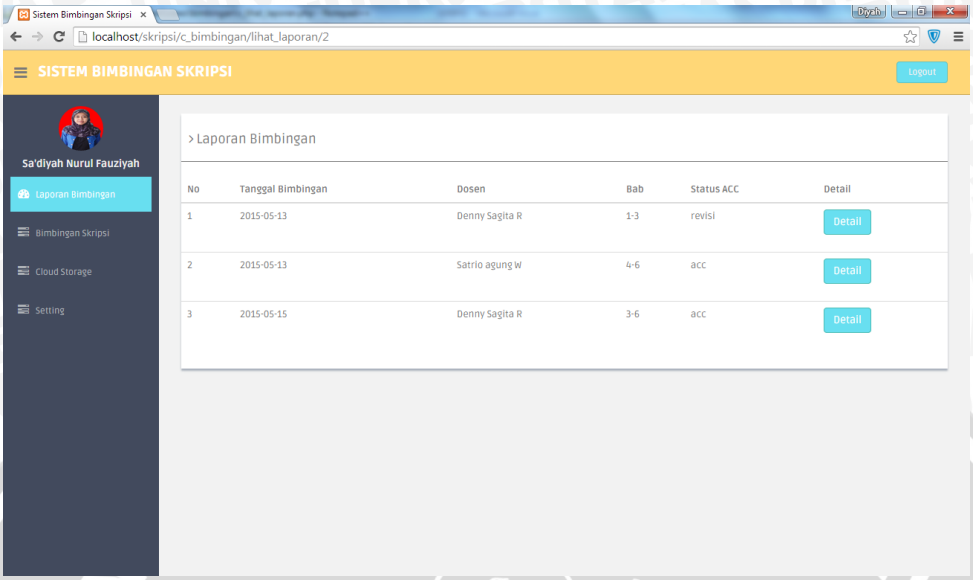


Gambar 5.7 Antarmuka aktivasi dosen

5.4.1.3 Implementasi Antarmuka Sistem Mahasiswa

a. Antamuka Melihat Laporan Bimbingan

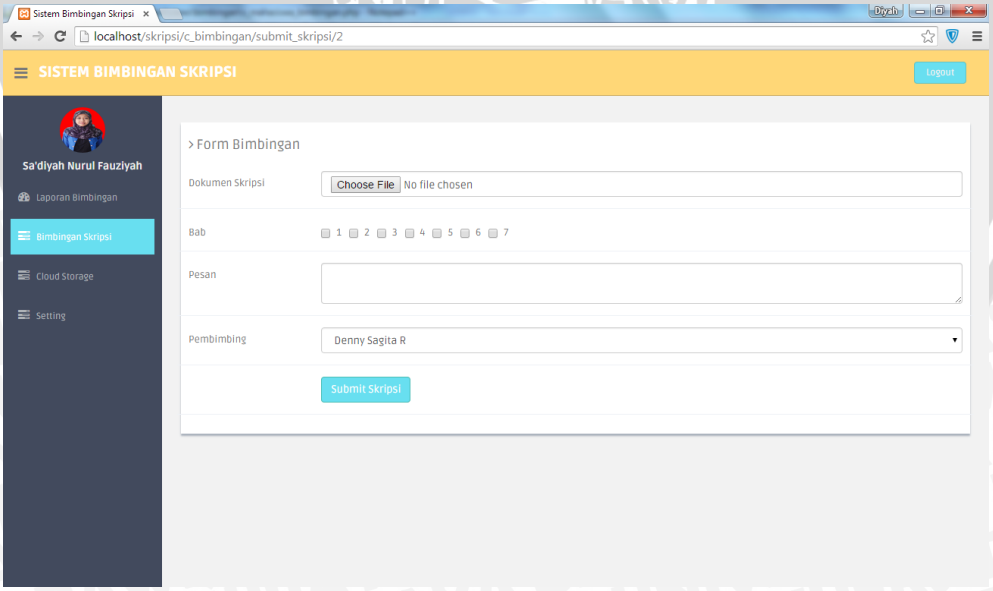
Halaman lihat laporan bimbingan dalam tabel ini menampilkan data laporan bimbingan yang terdiri dari nomor, tanggal bimbingan yang menunjukkan tanggal dimana mahasiswa melaksanakan bimbingan, dosen pembimbing, bab, dan status acc dari dokumen skripsi yang telah diupload oleh mahasiswa, dalam hal ini status tersebut revisi ataukah sudah ACC. Antarmuka melihat laporan bimbingan ditunjukkan oleh gambar 5.8.



Gambar 5.8 Antarmuka melihat laporan bimbingan

b. Antarmuka *Submit* Dokumen Skripsi

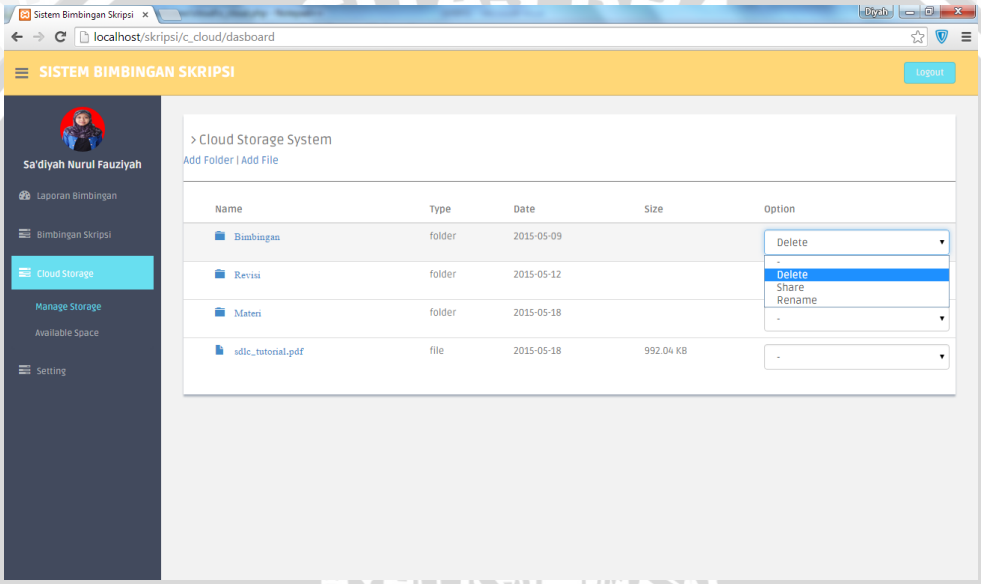
Halaman *submit* dokumen skripsi merupakan halaman bagi mahasiswa untuk melakukan bimbingan dengan cara submit dokumen. Pada halaman ini terdapat *form* bimbingan yang terdiri dari *input* dokumen skripsi, bab, pesan, pembimbing, dan *button submit*. Antarmuka *submit* dokumen skripsi ditunjukkan oleh gambar 5.9.



Gambar 5.9 Antarmuka *submit* dokumen skripsi

c. Mengakses Fitur Penyimpanan *Cloud*

Halaman ini merupakan halaman untuk fitur penyimpan *cloud*. Pada halaman ini ditampilkan *file* dan *folder* yang telah disimpan oleh mahasiswa. Di bagian atas terdapat menu untuk menmbahkan *file* dan *folder*, sedangkan di bagian kanan terdapat menu untuk *delete*, *share*, ataupun mengubah nama dokumen. Rincian dokumen yang ditampilkan antara lain nama, tipe, data, serta size. Antarmuka fitur penyimpanan cloud ditunjukkan oleh gambar 5.10.

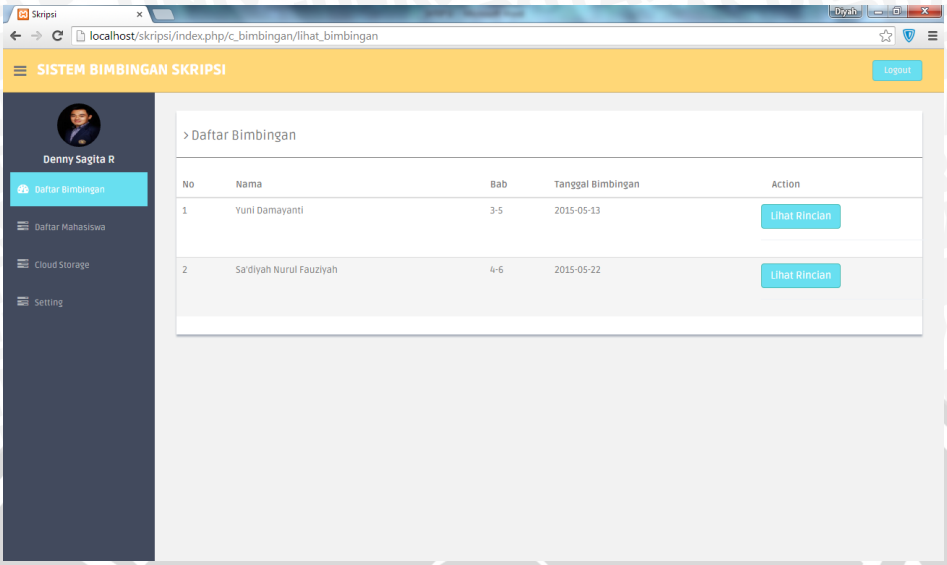


Gambar 5.10 Antarmuka fitur penyimpanan *cloud*

5.4.1.4 Implementasi Antarmuka Sistem Dosen

a. Melihat Daftar Bimbingan

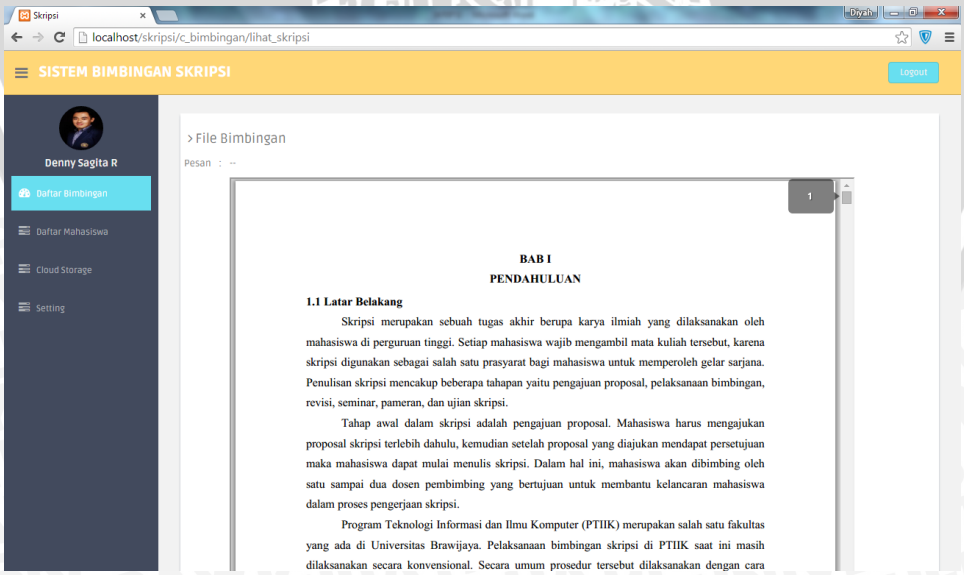
Halaman ini merupakan halaman bagi dosen untuk melihat daftar bimbingan. Pada halaman ini ditampilkan detail mahasiswa yang melakukan bimbingan. Terdiri dari nama, bab, dan tanggal bimbingan. Selain itu pada bagian kanan terdapat menu untuk melihat rincian bimbingan, dimana pada bagian rincian akan ditampilkan dokumen skripsi dari mahasiswa yang bersangkutan. Antarmuka melihat daftar bimbingan ditunjukkan oleh gambar 5.11.



Gambar 5.11 Antarmuka melihat daftar bimbingan

b. Melihat Dokumen Skripsi

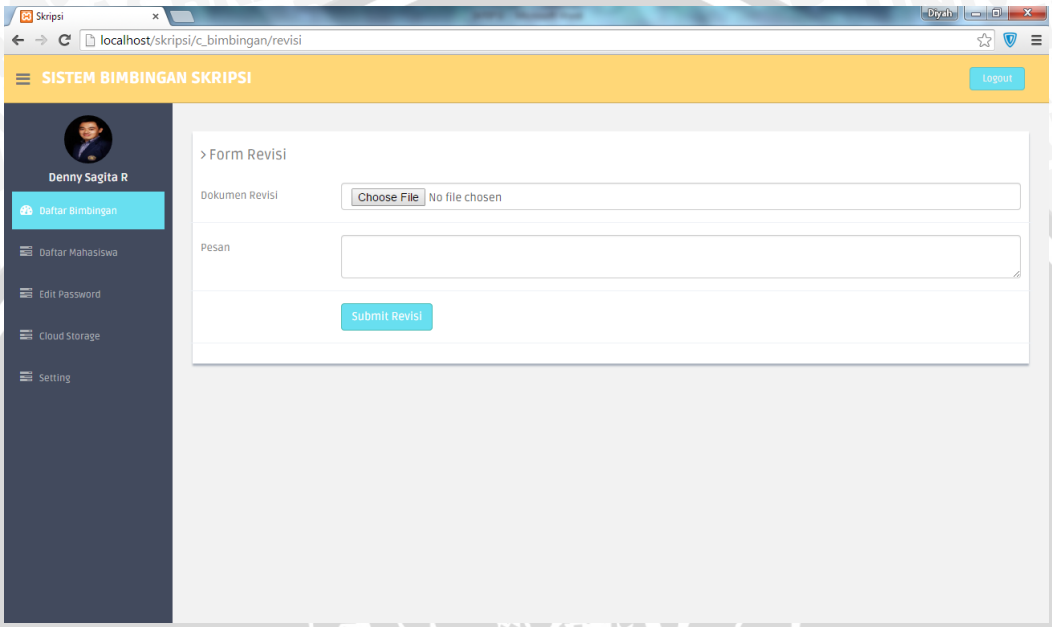
Halaman ini merupakan halaman bagi dosen untuk melihat dokumen skripsi yang di-*upload* oleh mahasiswa. Pada halaman ini ditampilkan pesan dari mahasiswa, serta dokumen skripsi dalam format .pdf. Di bagian bawah terdapat tombol revisi untuk melakukan revisi, dan tombol acc untuk melakukan acc. Antarmuka melihat dokumen skripsi ditunjukkan oleh gambar 5.12.



Gambar 5.12 Antarmuka melihat dokumen skripsi

c. Submit Revisi

Halaman ini merupakan halaman bagi dosen untuk melakukan *submit* revisi. Pada halaman ini ditampilkan *form input* dokumen revisi, pesan, dan *button submit* revisi. Antarmuka *submit* revisi ditunjukkan oleh gambar 5.13.



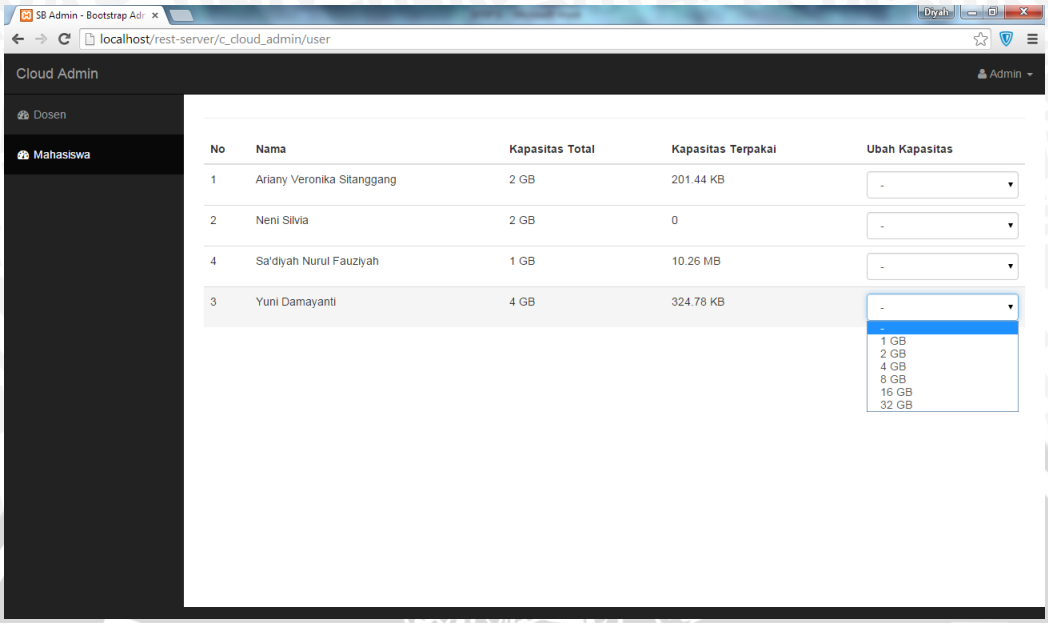
Gambar 5.13 Antarmuka submit revisi

5.6.2 Implementasi Antarmuka Sistem Penyimpanan *Cloud*

Pada sub bab ini akan dijelaskan antarmuka utama dari penyimpanan berbasis *cloud*.

a. Melihat *User*

Halaman ini merupakan halaman bagi adminstrator sistem cloud untuk melihat data user *user*. Pada halaman ini ditampilkan daftar *user* yang terdiri dari dosen dan mahasiswa beserta rinciannya antara lain nama, kapasitas total, dan kapasitas terpakai. Selain itu di bagian kanan terdapat menu ubah kapasitas yang berfungsi untuk mengubah kapasitas penyimpanan *user*. Antarmuka melihat *user* ditunjukkan oleh gambar 5.14.



Gambar 5.14 Antarmuka melihat user

BAB VI
PENGUJIAN DAN ANALISIS

Pada bab ini akan dilakukan tahap pengujian dan analisis dari sistem informasi bimbingan skripsi menggunakan fitur penyimpanan berkas berbasis *cloud*. Pengujian dilakukan melalui tiga tahap yaitu pengujian *black box*, *white box*, dan pengujian *rest-api*. Pengujian *black box* merupakan metode pengujian yang digunakan untuk mengetahui apakah sistem yang dibangun berfungsi dengan baik dan telah sesuai dengan kebutuhan, tanpa memperhatikan struktur logika internal perangkat lunak. Sedangkan pengujian *white box* dilakukan dengan melihat ke dalam modul untuk meneliti kode-kode program dan menganalisis adanya kesalahan atau tidak. Pengujian *rest-api* dilakukan dengan melakukan pengujian pada setiap unit proses pada *web service*. Pengujian REST *web service* dilakukan dengan menggunakan pengujian validasi sesuai dengan studi literatur yang telah didapatkan.

Setelah semua tahap pengujian selesai dilakukan maka akan dilakukan proses analisis terhadap hasil-hasil pengujian. Proses analisis dilakukan terhadap hasil pengujian di setiap tahap pengujian.

6.1 Pengujian White Box

6.1.1 Tujuan

Tujuan dilakukannya pengujian *white box* adalah untuk menguji semua *statement* program pada beberapa algoritma yang memiliki prioritas tinggi telah diimplementasikan sesuai dengan yang diharapkan.

6.1.2 Mekanisme

Pengujian *white box* dilakukan dengan membuat *flowgraph* dilanjutkan dengan menghitung *cyclomatic complexity* dan menentukan *path*. Dari jalur independen (*independen path*) yang telah didefinisikan tersebut maka dapat dibentuk kasus ujinya. Pengujian akan dilakukan pada beberapa path path sebagai *sample*. Pengujian *white box* ini dilakukan pada tiga fitur yaitu aktivasi mahasiswa, melihat detail laporan, dan *upload file*.

6.1.3 Hasil Pengujian

1. Kasus Uji Aktivasi Mahasiswa

Tabel 6.1 memaparkan *method* aktivasi_mahasiswa beserta *node flowgraph.*

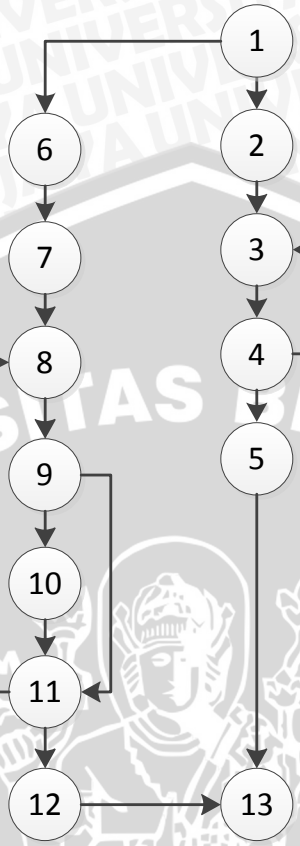
Tabel 6.1 Method Aktivasi Mahasiswa

1	public function aktivasi_mahasiswa($id)
2	{
3	if($id==0){
4	$tanggal = $this->m_cloud->get_tanggal();
5	foreach($tanggal as $row){
6	$tgl_sekarang= $row->tanggal;
7	}
8	$id=$this->input->post('nomor_induk');
9	$email=$this->input->post('email');
10	$subjek = "Sistem informasi bimbingan skripsi ";
11	$pesan = "Akun anda telah diaktivasi, sehingga anda dapat login dan mulai melaksanakan bimbingan";
12	
13	$data["dosen_nip_1"]=$this->input->post('dosen_nip_1');
14	$data["dosen_nip_2"]=$this->input->post('dosen_nip_2');
15	$data["tanggal_pengajuan"]=$this->input->post('tanggal_pengajuan');
16	$data["batas_akhir"]=$this->input->post('batas_akhir');
17	$data["laboratorium"]=$this->input->post('laboratorium');
18	$data["judul_skripsi"]=$this->input->post('judul');
19	$data["status"]="aktif";
20	
21	// create username on cloud
22	session_start();
23	$data2["nama"] = $this->input->post('nama_mahasiswa');;
24	$data2["id_user"] = $id;
25	$data2["status"] = "Mahasiswa";
26	$data2["kapasitas_total"] = "2147483648";
27	$data2["kapasitas_terpakai"] = 0;
28	
29	$this->m_cloud->add_user($data2);
30	
31	// create folder bimbingan
32	$data3["nama_folder"] = "Bimbingan";
33	$data3["parent"] = 0 ;

```
34                  $data3["id_user"] = $id;
35                  $data3["tanggal"] = $tgl_sekarang;
36                  $this->m_cloud->insert_folder($data3);
37                  // create folder revisi
38                  $data4["nama_folder"] = "Revisi";
39                  $data4["parent"] = 0 ;
40                  $data4["id_user"] = $id;
41                  $data4["tanggal"] = $tgl_sekarang;
42                  $this->m_cloud->insert_folder($data4);
43                  $this->m_user->update_mahasiswa($id, $data);
44                  // send message via email
45                  $this->c_akun->kirim_email($email, $subjek, $pesan);
46                  redirect ('c_mahasiswa/lihat_mahasiswa_belum_aktif',
'refresh');
47          }else{
48                  $data["nomor_induk"]= $id;
49                  $list = $this->m_user->get_mahasiswa();
50                  foreach($list as $row){
51                  if($data["nomor_induk"] == $row->nomor_induk){
52                      $data["nama_mahasiswa"] = $row->nama;
53                      $data["email"] = $row->email;
54                  }
55          }
56          $data['list'] = $this->m_user->get_dosen();
57          $this->load->view('akademik/v_aktivasi_mahasiswa', $data);
58          }
59          }
```

6 7 8 9 10 11 12 13

Pada Gambar 6.1 digambarkan *flowgraph* dari aktivasi mahasiswa yang didapatkan dari kode program yang telah dijabarkan. *Flowgraph* terdiri dari 13 node (N), 16 edge (E), dan 4 percabangan (P). Selanjutnya, dari flowgraph tersebut akan ditentukan *independent path* dan nilai *cyclomatic clompexity*. Dari jalur independen (*independen path*) yang telah didefinisikan tersebut maka dapat dibentuk kasus ujinya. Pengujian akan dilakukan pada beberapa path path sebagai *sample*.



Gambar 6.1 *Flowgraph* Aktivasi Mahasiswa

Dari *flowgraph* yang sudah dijelaskan pada Gambar 6.1 didapatkan *independent path* sebanyak lima path, diantaranya :

1. Path 1 : 1 – 2 – 3 – 4 – 5 – 13
2. Path 2 : 1 – 2 – 3 – 4 – 3 – 4 – 5 – 13
3. Path 3 : 1 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13
4. Path 4 : 1 – 6 – 7 – 8 – 9 – 10 – 11 – 8 – 9 – 10 – 11 – 12 – 13
5. Path 5 : 1 – 6 – 7 – 8 – 9 – 11 – 12 – 13

Dari *flowgraph* yang sudah dijelaskan pada Gambar 6.1 juga dapat dihitung nilai *cyclomatic clompexity* sebagai berikut:

V(G) = 5 regions

V(G) = E – N + 2

= 16 – 13 + 2

= 5

V(G) = P + 1

= 4 + 1

= 5

Berdasarkan 5 jalur independen yang telah didefinisikan tersebut maka dapat dibentuk kasus ujinya. Tabel 6.2 memaparkan kasus uji dari *method* aktivasi_mahasiswa.

Tabel 6.2 Kasus Uji *Method* aktivasi_mahasiswa

No	Jalur	Kelas	*Method*	Data *Input*	Hasil yang Diharapkan	Hasil yang Diperoleh	Status
1	1 – 2 – 3 – 4 – 5 – 13	C_Kelola_mahasiswa	aktivasi_mahasiswa()	id, judul_skripsi, tanggal_pengajuan, batas_akhir, dosen_pembimbing, dan laboratorium	Sistem mengubah status mahasiswa menjadi aktif. Dan menyimpan data input ke *database.*	Sistem mengubah status mahasiswa menjadi aktif. Dan menyimpan data input ke *database.*	Valid
2	1 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13	C_Kelola_mahasiswa	aktivasi_mahasiswa()	nomor_induk	Sistem hanya menampilkan halaman aktivasi mahasiswa	Sistem hanya menampilkan halaman aktivasi	Valid

						mahasiswa	
	1 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13	Kelola_mahasiswa	aktivasi_mahasiswa()	nomor_induk	Sistem hanya menampilkan halaman aktivasi mahasiswa	Sistem hanya menampilkan halaman aktivasi mahasiswa	Valid

2. Kasus Uji Melihat Detail Laporan Bimbingan

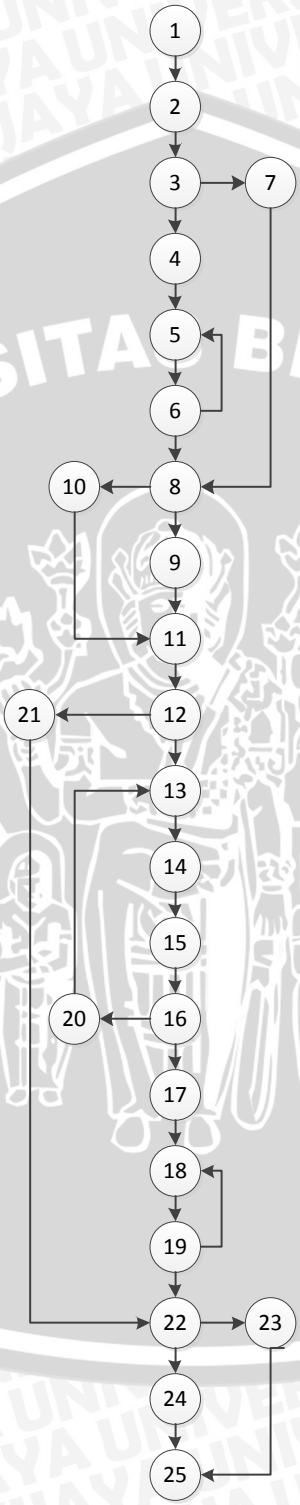
Tabel 6.3 memaparkan *method* detail_laporan beserta *node flowgraph.*

Tabel 6.3 *Method* Detail Laporan Bimbingan

```
1   public function detail_laporan($id){
2           session_start();
3           $user = $_SESSION["user_id"];
4           $data["nama"] = $_SESSION["nama"];
5           $data["foto"] = $_SESSION["foto"];
6           $data["nim"] = $_SESSION["user_id"];
7           $data['id_bimbingan'] = $this->input->post('id_bimbingan');
8           $data['tanggal_bimbingan'] = $this->input->post('tanggal_bimbingan');
9           $data['nama_dosen'] = $this->input->post('nama_dosen');
10          $data['bab'] = $this->input->post('bab');
11          $data['dokumen1'] = $this->input->post('dokumen_bimbingan');
12          $data['nim'] = $this->input->post('nim');
13          $data['pesan'] = $this->input->post('pesan');
14          $id2 = $this->m_cloud->get_file_sk_mhsi($data['dokumen1']);
15          if($id2 != NULL){
16                  $json_idr = json_decode($id2,  true);
17                  foreach($json_idr as $row ){
18                          $data['dokumen_bimbingan'] = $row['nama_file'];
19                  }
20          }else{
21                  $data['dokumen_bimbingan'] = "-";
22          }
23          if (isset($_POST['nim_mhs'])){
24                  $data['nim']= $this->input->post('nim_mhs') ;
25          }else{
```

```
26              $data['nim']=$user;
27          }
28          $list2 = $this->m_bimbingan->get_revisi($data['id_bimbingan']);
29          if($list2 != NULL){
30              foreach($list2 as $row2){
31                  $data["dokumen2"] = $row2->dokumen;
32                  $data["tanggal_revisi"] = $row2->tanggal;
33                  $data["komentar"] = $row2->komentar;
34                  $id3 = $this->m_cloud-
>get_file_skripsi($data['dokumen2']);
35                  if($id3 != NULL){
36                      $json_idr2 = json_decode($id3, true);
37                      foreach($json_idr2 as $row3 ){
38                      $data['dokumen_revisi'] = $ro...ia
39                      }
40                  }else{
41                      $data['dokumen_revisi'] = "-";
42                  }
43              }
44              }else{
45                  $data["dokumen_revisi"] = "-";
46                  $data["tanggal_revisi"] = "-";
47                  $data["komentar"] = "-";
48              }
49              if($id==1){
50                  $this->load->view('dosen/v_detail_laporan', $data);
51              }else{
52                  $this->load->view('bimbingan/v_detail_lapora...);
53              }
54          }
```

Pada Gambar 6.2 digambarkan *flowgraph* dari melihat detail laporan yang didapatkan dari kode program yang telah dijabarkan pada Tabel 6.3. *Flowgraph* terdiri dari 25 node (N), 31 edge (E), dan 7 percabangan (P). Selanjutnya, dari flowgraph tersebut akan ditentukan *independent path* dan nilai *cyclomatic clompexity*. Selanjutnya, dari jalur independen (*independen path*) yang telah didefinisikan tersebut maka dapat dibentuk kasus ujinya. Pengujian akan dilakukan pada beberapa path path sebagai *sample*.



Gambar 6.2 *Flowgraph* Melihat Detail laporan

Dari *flowgraph* yang sudah dijelaskan pada Gambar 6.2 didapatkan *independent path* sebanyak delapan *path*, diantaranya:

1. Path 1 : 1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22 – 24 – 25
2. Path 2 : 1 – 2 – 3 – 4 – 5 – 6 – 5 – 6 – 8 – 9 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22 – 24 – 25
3. Path 3 : 1 – 2 – 3 – 7 – 8 – 9 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22 – 24 – 25
4. Path 4 : 1 – 2 – 3 – 4 – 5 – 6 – 8 – 10 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22 – 24 – 25
5. Path 5 : 1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 11 – 12 – 21 – 22 – 24 – 25
6. Path 6 : 1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 11 – 12 – 13 – 14 – 15 – 16 – 20 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22 – 24 – 25
7. Path 7 : 1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 18 – 19 – 22 – 24 – 25
8. Path 8 : 1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22 – 23 – 25

Dari *flowgraph* yang sudah dijelaskan pada Gambar 6.2 juga dapat dihitung nilai *cyclomatic clompexity* sebagai berikut :

V(G) = 8 *regions*
V(G) = E – N + 2
= 31 – 25 + 2
= 8
V(G) = P + 1
= 7 + 1
= 8

Berdasarkan 8 jalur independen yang telah didefinisikan tersebut maka dapat dibentuk kasus ujinya. Tabel 6.4 memaparkan kasus uji dari *method* detail_laporan.

Tabel 6.4 Kasus Uji *Method* aktivasi_mahasiswa

No	Jalur	Kelas	*Method*	Data *Input*	Hasil yang Diharapkan	Hasil yang Diperoleh	Status
1	1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22 – 24 – 25	C_Mahasiswa	Detail_laporan()	id_bimbingan, tanggal_bimbingan, nama_dosen, bab, nim, id_laporan	Sistem menampilkan detail laporan bimbingan lengkap dengan dokumen laporan dan dokumen revisi.	Sistem menampilkan detail laporan bimbingan lengkap dengan dokumen laporan dan dokumen revisi.	Valid
2	: 1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 11 – 12 – 21 – 22 – 24 – 25	C_Mahasiswa	Detail_laporan()	id_bimbingan, tanggal_bimbingan, nama_dosen, bab, nim, id_laporan	Sistem menampilkan detail laporan bimbingan tanpa dokumen revisi.	Sistem menampilkan detail laporan bimbingan tanpa dokumen revisi.	Valid
3	1 – 2 – 3 – 4 – 5 – 6 –	C_Mahasiswa	Detail_laporan()	id_bimbingan, tanggal_	Sistem menampilkan detail laporan	Sistem menampilkan detail	Valid

	8 – 9 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22 – 23 – 25			bimbingan, nama_dosen, bab, nim, id_laporan	bimbingan beserta dokumen revisi pada halaman dosen.	laporan bimbingan beserta dokumen revisi pada halaman dosen.	

3. Kasus Uji Upload File

Tabel 6.5 memaparkan *method* add_file_proses beserta *node flowgraph.*

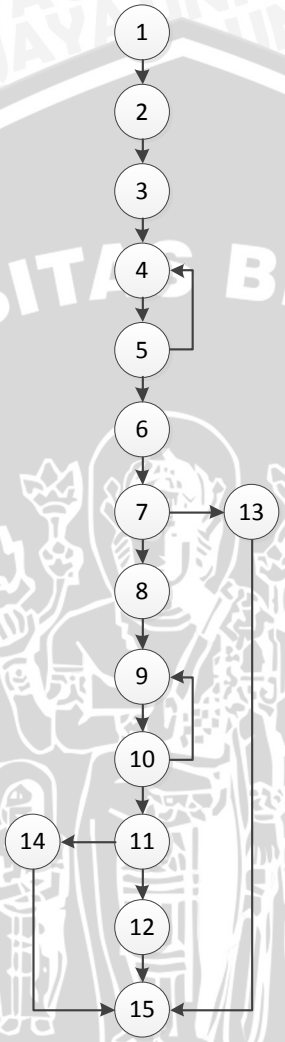
Tabel 6.5 Method *Upload File*

1	public function add_file_proses(){
2	session_start();
3	$user = $_SESSION["user_id"];
4	$data["nama"] = $_SESSION["nama"];
5	$data["foto"] = $_SESSION["foto"];
6	$data["nim"] = $_SESSION["user_id"];
7	$size = $this->m_cloud->get_space($user);
8	$json_idr = json_decode($size, true);
9	foreach($json_idr as $row){
10	$data['kapasitas_total'] = $row['kapasitas_total'];
11	$data['kapasitas_terpakai'] = $row['kapasitas_terpakai'];
12	}
13	$size = $_FILES['nama_file']['size'];
14	$data["kapasitas_upload"]=$size+$data['kapasitas_terpakai'];
15	if($data["kapasitas_upload"] <= $data['kapasitas_total']){
16	$tanggal = $this->m_cloud->get_ta();
17	foreach($tanggal as $row){
18	$tgl_sekarang= $row->tanggal;
19	}
20	if(isset($_SESSION["parent"]) and $_SESSION["parent"]

```
    !=0){
21                  $parent=$_SESSION["parent"];
22                  $data = array();
23                  $uploadDir = "file/";
24                  $RealTitleID = $_FILES['nama_file']['name'];
25                                                          12
26                  $data["size"] = $_FILES['nama_file']['size'];
27                  $data['file'] = new
    CurlFile($_FILES['nama_file']['tmp_name'],'file/exgpd',$RealTitleID);
28                  $data["parent"]=$parent;
29                  $data["id_user"]=$_SESSION["user_id"];
30                  $data["tanggal"]=$tgl_sekarang;
31                  $this->m_cloud->insert_file($data);
32                  redirect ('C_cloud/select_folder_direct','refresh');
33                  } else{
34                  $data = array();
35                  $parent =0;                             13
36                  $RealTitleID = $_FILES['nama_file']['name'];
37                  $data["size"] = $_FILES['nama_file']['size'];
38                  $data['file'] = new
    CurlFile($_FILES['nama_file']['tmp_name'],'file/exgpd',$RealTitleID);
39                  $data["parent"]=$parent;
40                  $data["id_user"]=$_SESSION["user_id"];
41                  $data["tanggal"]=$tgl_sekarang;
42                  $this->m_cloud->insert_file($data);
43                  redirect ('C_cloud/dasboard','refresh');
44                  }                                       14
45          }else{
46                  $this->load->view('cloud/v_upload_gagal', $data);
47          }                                   15
48  }
```

Pada Gambar 6.3 digambarkan *flowgraph upload file* yang didapatkan dari kode program yang telah dijabarkan. *Flowgraph* terdiri dari 15 node (N), 18 edge (E), dan 4 percabangan (P). Selanjutnya, dari flowgraph tersebut akan ditentukan

independent path dan nilai *cyclomatic clompexity*. Selanjutnya, dari flowgraph tersebut akan ditentukan *independent path* dan nilai *cyclomatic clompexity*.



Gambar 6.3 *Flowgraph Upload File*

Dari *flowgraph* yang sudah dijelaskan pada Gambar 6.3 didapatkan *independent path* sebanyak lima path, diantaranya :

1. Path 1 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 15
2. Path 2 : 1 – 2 – 3 – 4 – 5 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 15
3. Path 3 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 13 – 15
4. Path 4 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 9 – 10 – 11 – 12 – 15

5. Path 5 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 14 – 15

Dari *flowgraph* yang sudah dijelaskan pada Gambar 6.3 juga dapat dihitung nilai *cyclomatic clompexity* sebagai berikut :

V(G) = 5 *regions*
V(G) = E – N + 2
= 18 – 15 + 2
= 5
V(G) = P + 1
= 4 + 1
= 5

Berdasarkan 5 jalur independen yang telah didefinisikan tersebut maka dapat dibentuk kasus ujinya. Tabel 6.6 memaparkan kasus uji dari *method* add_file_proses.

Tabel 6.6 Kasus Uji *Method* add_file_proses

No	Jalur	Kelas	*Method*	Data *Input*	Hasil yang Diharapkan	Hasil yang Diperoleh	Status
1	1 – 2 – 3 – 4 – 5 – 6 – 7 – 13 – 15	C_Penyimpanan_berkas	add_file_proses	nama_file, size, parent, id_user, tanggal	*Upload file* gagal karena kapasitas penyimpanan yang tersedia tidak mencukupi.	*Upload file* gagal karena kapasitas penyimpanan yang tersedia tidak mencukupi.	Valid
2	1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 9 – 10 –	C_Penyimpanan_berkas	add_file_proses	nama_file, size, parent, id_user, tanggal	*Upload file* berhasil dilakukan	*Upload file* berhasil dilakukan	Valid

	11 – 12 – 15						

6.1.4 Analisis

Pada Tabel 6.7 akan ditunjukkan hasil dari pengujian *white box* pada semua kasus uji yang telah dilakukan.

Tabel 6.7 Analisa Hasil Pengujian *White Box*

Kasus Uji	Total Regions	Total Independent Path	V(G)
Kasus Uji Aktivasi Mahasiswa	5	5	5
Kasus Uji Melihat Detail Laporan	8	8	8
Kasus Uji *Upload File*	5	5	5

Dari Tabel 6.7 didapatkan bahwa dari kasus uji aktivasi mahasiswa dan *upload file* didapatkan nilai *cyclomatic complexity* sebesar 5 yang memiliki yang memiliki arti bahwa pada kasus uji ini merupakan prosedur yang *stable* dengan struktur yang baik dan dengan kerumitan untuk melakukan perbaikan *coding* adalah rendah. Sedangkan pada kasus uji melihat detail laporan, di didapatkan nilai *cyclomatic complexity* sebesar 8 yang memiliki yang memiliki arti bahwa struktur program sedikit lebih rumit dan kompleks.

6.2 Pengujian Validasi dengan Metode *Black Box*

6.2.1 Tujuan

Tujuan dilakukannya pengujian validasi adalah untuk memastikan setiap spesifikasi kebutuhan perangkat lunak yang didefinisikan telah sesuai dengan yang diharapkan.

6.2.2 Mekanisme

Setiap spesifikasi kebutuhan perangkat lunak yang didefinisikan pada tahap analisis kebutuhan akan diujikan pada pengujian ini dengan cara mendefinisikan kasus uji terhadap setiap kebutuhan tersebut lalu membandingkannya dengan hasil yang diperoleh. Kasus uji pada sistem informasi bimbingan skripsi adalah sebagai berikut.

Tabel 6.8 menjelaskan kasus uji dari pengujian validasi pada fungsi *submit* dokumen skripsi.

Tabel 6.8 Kasus Uji Validasi *Submit* Dokumen Skripsi

Nomor Kasus Uji	VAL_001
Nama Kasus Uji	Kasus uji *submit* dokumen skripsi
Nomor *Use Case*	SRS_1_01
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan *submit* dokumen skripsi kepada dosen pembimbing
Prosedur Uji	1. Penguji melakukan *Upload* dokumen 2. Penguji melakukan *checklist* bab 3. Penguji mengisi Pesan 4. Penguji menekan tombol *submit*
Hasil yang Diharapkan	Dokumen bimbingan dikirim kepada dosen pembimbing. Sistem menampilkan pemberitahuan bahwa dokumen berhasil dikirim

Tabel 6.9 menjelaskan kasus uji dari pengujian validasi pada fungsi melihat dokumen skripsi.

Tabel 6.9 Kasus Uji Validasi Melihat Dokumen Skripsi

Nomor Kasus Uji	VAL_002
Nama Kasus Uji	Kasus uji melihat dokumen skripsi
Nomor *Use Case*	SRS_1_02
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan proses untuk melihat dokumen skripsi yang telah di-*submit* oleh mahasiswa
Prosedur Uji	1. Memilih mahasiswa 2. Menekan tombol rincian bimbingan
Hasil yang Diharapkan	Menampilkan dokumen bimbingan dan pesan dari mahasiswa

Tabel 6.10 menjelaskan kasus uji dari pengujian validasi pada fungsi melakukan ACC.

Tabel 6.10 Kasus Uji Validasi Melakukan ACC

Nomor Kasus Uji	VAL_003
Nama Kasus Uji	Kasus uji melakukan ACC
Nomor *Use Case*	SRS_1_03
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan proses ACC atas dokumen bimbingan mahasiswa
Prosedur Uji	1. Pengujia menekan tombol acc
Hasil yang Diharapkan	Status bimbingan mahasiswa berubah menjadi ACC

Tabel 6.11 menjelaskan kasus uji dari pengujian validasi pada fungsi submit revisi.

Tabel 6.11 Kasus Uji Validasi *Submit* Revisi

Nomor Kasus Uji	VAL_004
Nama Kasus Uji	Kasus uji *submit* revisi
Nomor *Use Case*	SRS_1_04
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan proses *submit* dokumen revisi kepada mahasiswa
Prosedur Uji	1. Penguji menekan tombol revisi 2. Penguji melakukan *Upload* dokumen revisi 3. Penguji menulis komentar 4. Penguji menekan tombol *submit*
Hasil yang Diharapkan	Revisi berhasil dikirim kepada mahasiswa. Menampilkan pemberitahuan bahwa revisi berhasil dikirim

Tabel 6.12 menjelaskan kasus uji dari pengujian validasi pada fungsi mengakses fitur penyimpanan *cloud*.

Tabel 6.12 Kasus Uji Validasi Mengakses Fitur Penyimpanan *Cloud*

Nomor Kasus Uji	VAL_005
Nama Kasus Uji	Kasus uji mengakses fitur penyimpanan *cloud*
Nomor *Use Case*	SRS_1_05
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan proses akses terhadap fitur penyimpanan *cloud*
Prosedur Uji	1. Penguji memilih menu *cloud storage*
Hasil yang Diharapkan	Menampilkan dokumen dan menu pada sistem penyimpanan *cloud*

Tabel 6.13 menjelaskan kasus uji dari pengujian validasi pada fungsi melihat laporan bimbingan.

Tabel 6.13 Kasus Uji Validasi Melihat Laporan Bimbingan

Nomor Kasus Uji	VAL_006
Nama Kasus Uji	Kasus uji melihat laporan bimbingan
Nomor *Use Case*	SRS_1_06
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan laporan bimbingan skripsi mahasiswa
Prosedur Uji	1. Penguji menekan tombol lihat laporan bimbingan
Hasil yang Diharapkan	Menampilkan laporan bimbingan beserta rinciannya

Sedangkan kasus uji pada sistem penyimpanan berbasis cloud adalah sebagai berikut.

Tabel 6.14 menjelaskan kasus uji dari pengujian validasi pada fungsi mengatur kapasitas *storage*.

Tabel 6.14 Kasus Uji Validasi Mengatur Kapasitas *Storage*

Nomor Kasus Uji	VAL_007
Nama Kasus Uji	Kasus uji mengatur kapasitas *storage*
Nomor *Use Case*	SRS_2_02
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan proses ubah terhadap kapasitas *storage user*
Prosedur Uji	Memilih kapasitas *storage* pada *drop down* menu yang telah disediakan
Hasil yang	Kapasitas berhasil diubah sesuai dengan pilihan *administrator*

Diharapkan	

6.2.3 Hasil Pengujian

Tabel 6.15 memaparkan hasil pengujian validasi yang telah dilakukan terhadap setiap kasus uji yang telah didefinisikan pada tahap sebelumnya.

Tabel 6.15 Hasil pengujian *black box* sistem informasi bimbingan skripsi

No	Nama Kasus Uji	Hasil yang Diharapkan	Hasil yang Diperoleh	Status
1	Submit Dokumen Skripsi	Dokumen bimbingan dikirim kepada dosen pembimbing. Sistem menampilkan pemberitahuan bahwa dokumen berhasil dikirim	Dokumen bimbingan dikirim kepada dosen pembimbing/ Sistem menampilkan pemberitahuan bahwa dokumen berhasil dikirim	Valid
2	Melihat Daftar Bimbingan	Sistem menampilkan daftar mahasiswa yang melaksanakan bimbingan	Sistem menampilkan daftar mahasiswa yang melaksanakan bimbingan	Valid
2	Melihat Dokumen Skripsi	Menampilkan dokumen bimbingan dan pesan dari mahasiswa	Menampilkan dokumen bimbingan dan pesan dari mahasiswa	Valid
3	Melakukan ACC	Status bimbingan mahasiswa berubah menjadi ACC	Status bimbingan mahasiswa berubah menjadi ACC	Valid
4	Submit Revisi	Revisi berhasil dikirim kepada mahasiswa	Revisi berhasil dikirim kepada mahasiswa	Valid
5	Melihat Revisi	Menampilkan dokumen revisi yang telah dipilih oleh mahasiswa	Menampilkan dokumen revisi yang telah dipilih oleh mahasiswa	Valid

6	Mengakses Fitur *Cloud*	Menampilkan dokumen dan menu pada sistem penyimpanan *cloud*	Menampilkan dokumen dan menu pada sistem penyimpanan *cloud*	Valid
7	Melihat Laporan Bimbingan	Menampilkan laporan bimbingan beserta rinciannya	Menampilkan laporan bimbingan beserta rinciannya	Valid
8	Melihat Data *User*	Menampilkan data *user*	Menampilkan data *user*	Valid
9	Mengaktifkan Akun Mahasiswa	Akun mahasiswa berhasil diaktivasi	Akun mahasiswa berhasil diaktivasi	Valid
10	Mengaktifkan Akun Dosen	Akun dosen berhasil diaktivasi	Akun dosen berhasil diaktivasi	Valid
11	Melakukan Registrasi	Menampilkan pemberitahuan bahwa registrasi berhasil	Menampilkan pemberitahuan bahwa registrasi berhasil	Valid
12	*Login*	*User* berhasil masuk ke sistem	*User* berhasil masuk ke sistem	Valid

Tabel 6.16 Hasil pengujian *black box* sistem penyimpanan berbasis *cloud*

No	Nama Kasus Uji	Hasil yang Diharapkan	Hasil yang Diperoleh	Status
1	Mengatur kapasitas storage	Kapasitas berhasil diubah sesuai dengan pilihan *administrator*	Kapasitas berhasil diubah sesuai dengan pilihan *administrator*	Valid
2	Melihat *user*	Menampilkan data *user*	Menampilkan data *user*	Valid
3	Menghapus *user*	*User* berhasil dihapus dari sistem	*User* berhasil dihapus dari sistem	Valid

6.2.4 Analisis

Berdasarkan hasil pengujian validasi, seluruh kasus uji yang didefinisikan telah valid atau berarti seluruh spesifikasi kebutuhan perangkat lunak telah terpenuhi. Sehingga dapat dikatakan sistem sistem informasi bimbingan skripsi dan sistem *cloud* lolos uji validasi.

6.3 Pengujian REST *Web Service*

6.3.1 Tujuan

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. Dalam hal ini akan dilakukan pengujian untuk mengetahui kinerja API sistem dalam melayani *request* dan mengirim respon ke *user*.

6.3.2 Mekanisme

1. Kasus Uji *Get File*

Pada tahap ini, dilakukan pengujian REST *web service* pada menu yang berfungsi untuk menampilkan *file* di sistem *user*. Kasus uji pada *get file* ditunjukkan oleh Tabel 6.17.

Tabel 6.17 *Method Get File*

Nama Kasus Uji	Kasus Uji *Get File*
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa REST *Web Service* yang dibuat dapat memenuhi kebutuhan untuk melayani *request* dan mengirim *respons* ke *user*.
Data Masukan	ID file
Prosedur Uji	Menuliskan *service endpoint* dari *server cloud* dan menambahkan parameter berupa ID file. (**http://localhost/rest-server/api/cloud/folder/id/1**)
Hasil yang Diharapkan	Mengembalikan *status code* 200 (OK) yang menyatakan bahwa *request* HTTP berhasil

2. Kasus Uji *Post File*

Pada tahap ini, dilakukan pengujian REST *web service* pada menu yang berfungsi untuk melakukan *upload file* ke sistem penyimpanan *cloud*. Kasus uji pada *post file* ditunjukkan oleh Tabel 6.18.

Tabel 6.18 *Method Post File*

Nama Kasus Uji	Kasus Uji *Post File*
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa REST *Web Service* yang dibuat dapat memenuhi kebutuhan untuk melayani *request* dan mengirim *respons* ke *user*.
Data Masukan	1. Nama File 2. ID User 3. Tanggal 4. Ukuran File 5. ID Parent
Prosedur Uji	Menuliskan *service endpoint* dari *server cloud*. (**http://localhost/rest-server/api/cloud/addfile**)
Hasil yang Diharapkan	Mengembalikan *status code* 200 (OK) yang menyatakan bahwa *request* HTTP berhasil

3. Kasus Uji *Post User*

Pada tahap ini, dilakukan pengujian REST *web service* pada menu yang berfungsi untuk menambahkan *user* baru ke sistem penyimpanan *cloud*. Kasus uji pada *post user* ditunjukkan oleh Tabel 6.19.

Tabel 6.19 *Method Post User*

Nama Kasus Uji	Kasus Uji *Post User*
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa REST *Web Service* yang dibuat dapat memenuhi kebutuhan untuk melayani *request* dan mengirim *respons* ke *user*.

Data Masukan	1. Nama User 2. ID User 3. Status User 4. Kapasitas Total 5. Kapasitas Terpakai
Prosedur Uji	Menuliskan *service endpoint* dari *server cloud*. (**http://localhost/rest-server/api/cloud/adduser**)
Hasil yang Diharapkan	Mengembalikan *status code* 200 (OK) yang menyatakan bahwa *request* HTTP berhasil

4. Kasus Uji *Post Share*

Pada tahap ini, dilakukan pengujian REST *web service* pada menu yang berfungsi untuk melakukan *sharing file* dengan *user* lain pada sistem *cloud*. Kasus uji pada *post share* ditunjukkan oleh Tabel 6.20.

Tabel 6.20 *Method Post Share*

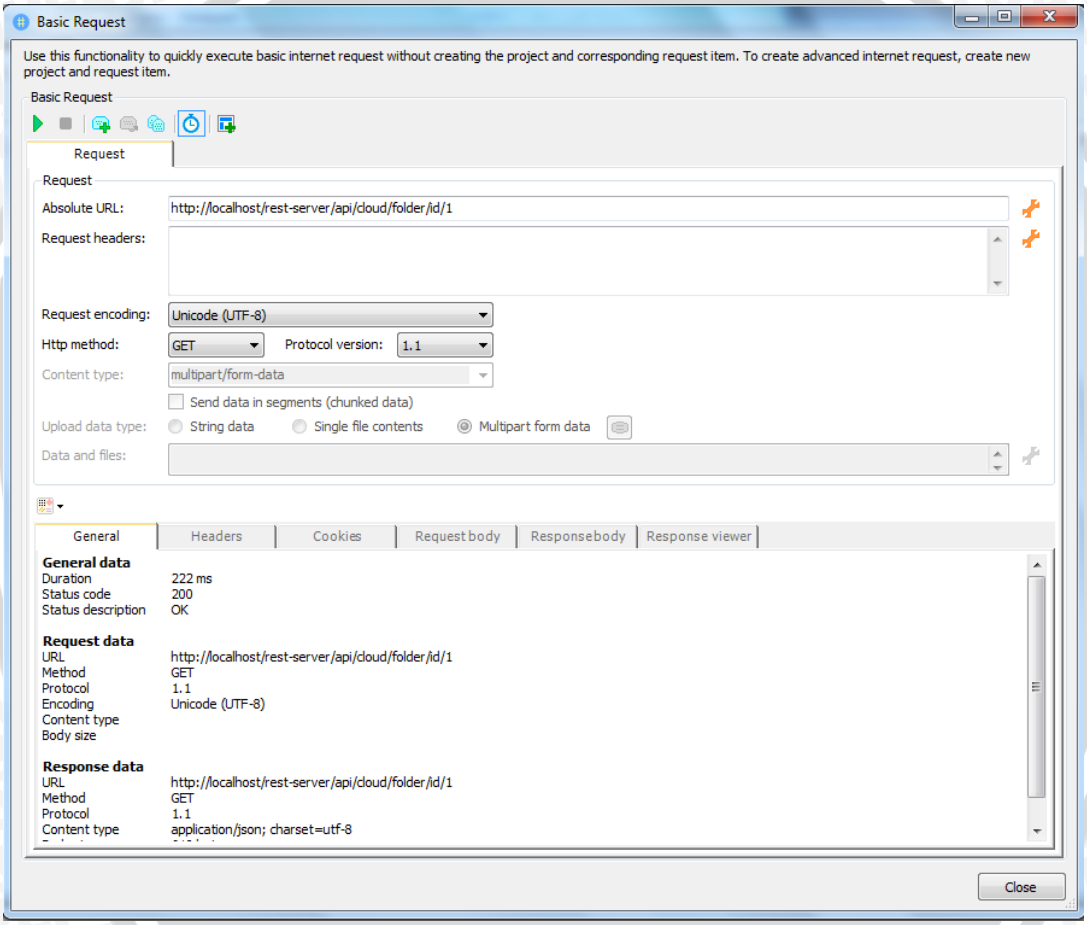
Nama Kasus Uji	Kasus Uji *Post Share*
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa REST *Web Service* yang dibuat dapat memenuhi kebutuhan untuk melayani *request* dan mengirim *respons* ke *user*.
Data Masukan	1. ID User 2. ID Folder
Prosedur Uji	Menuliskan *service endpoint* dari *server cloud*. (**http://localhost/rest-server/api/cloud/addSharingFolder**)
Hasil yang Diharapkan	Mengembalikan *status code* 200 (OK) yang menyatakan ba.hwa *request* HTTP berhasil

6.3.3 Hasil

Pengujian *web service* dilakukan dengan menggunakan software HTTPMaster. HTTMaster sendiri merupakan *software* yang secara spesifik digunakan untuk melakukan pengujian API dan *web service*.

1. Hasil Pengujian *Get File*

Hasil pengujian pada *get file* ditunjukkan oleh gambar 6.4



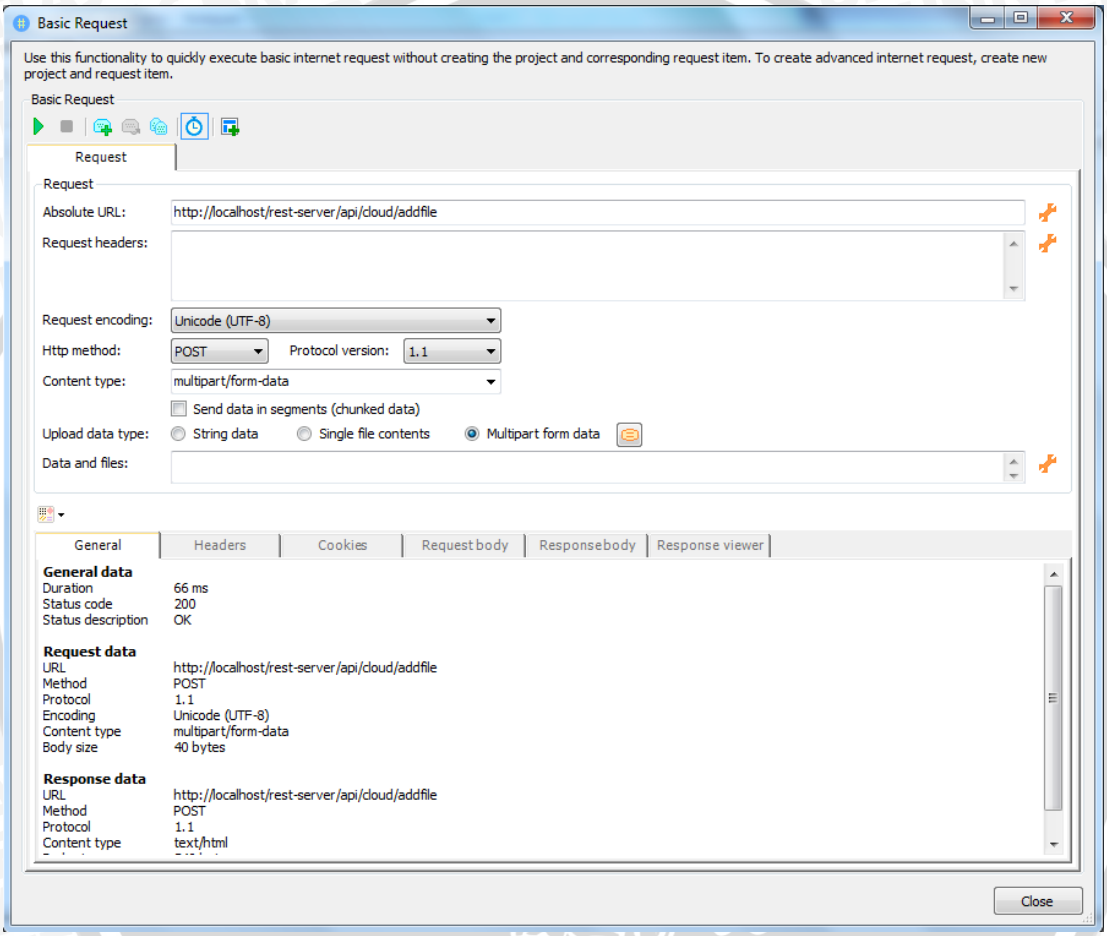
Gambar 6.4 Hasil Pengujian pada *Get File*

Dari gambar 6.4, *absolute* URL merupakan *service endpoint* dari *server cloud* untuk *get file*. HTPMethod merupakan method yang digunakan untuk melakukan *request*, dalam hal ini adalah *method GET*. Setelah request dieksekusi maka akan ditampilkan hasil dari request tersebut. Pada *General Data*, *duration* menunjukkan durasi atau waktu yang digunakan untuk melakukan request, *status code*

mengembalikan nilai 200 yang menyatakan bahwa *request* HTTP berhasil dengan *status description* OK.

2. Hasil Pengujian *Post File*

Hasil pengujian pada *post file* ditunjukkan oleh gambar 6.5.

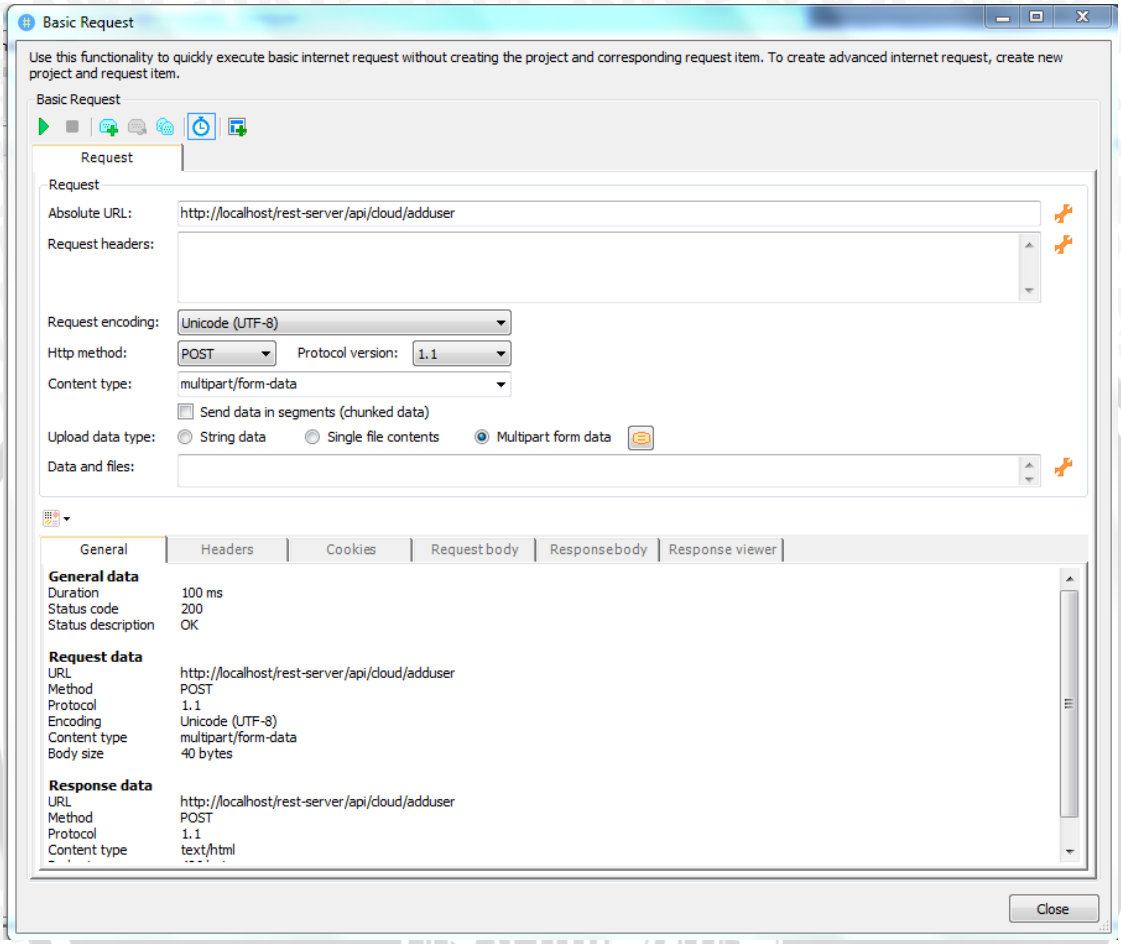


Gambar 6.5 Hasil Pengujian pada *Post File*

Dari gambar 6.4, *absolute* URL merupakan *service endpoint* dari *server cloud* untuk *get file*. HTPMethod merupakan method yang digunakan untuk melakukan *request*, dalam hal ini adalah *method POST*. Setelah request dieksekusi maka akan ditampilkan hasil dari request tersebut. Pada *General Data*, *duration* menunjukkan durasi atau waktu yang digunakan untuk melakukan request, *status code* mengembalikan nilai 200 yang menyatakan bahwa *request* HTTP berhasil dengan *status description* OK.

3. Hasil Pengujian *Post User*

Hasil pengujian pada *post user* ditunjukkan oleh gambar 6.6.

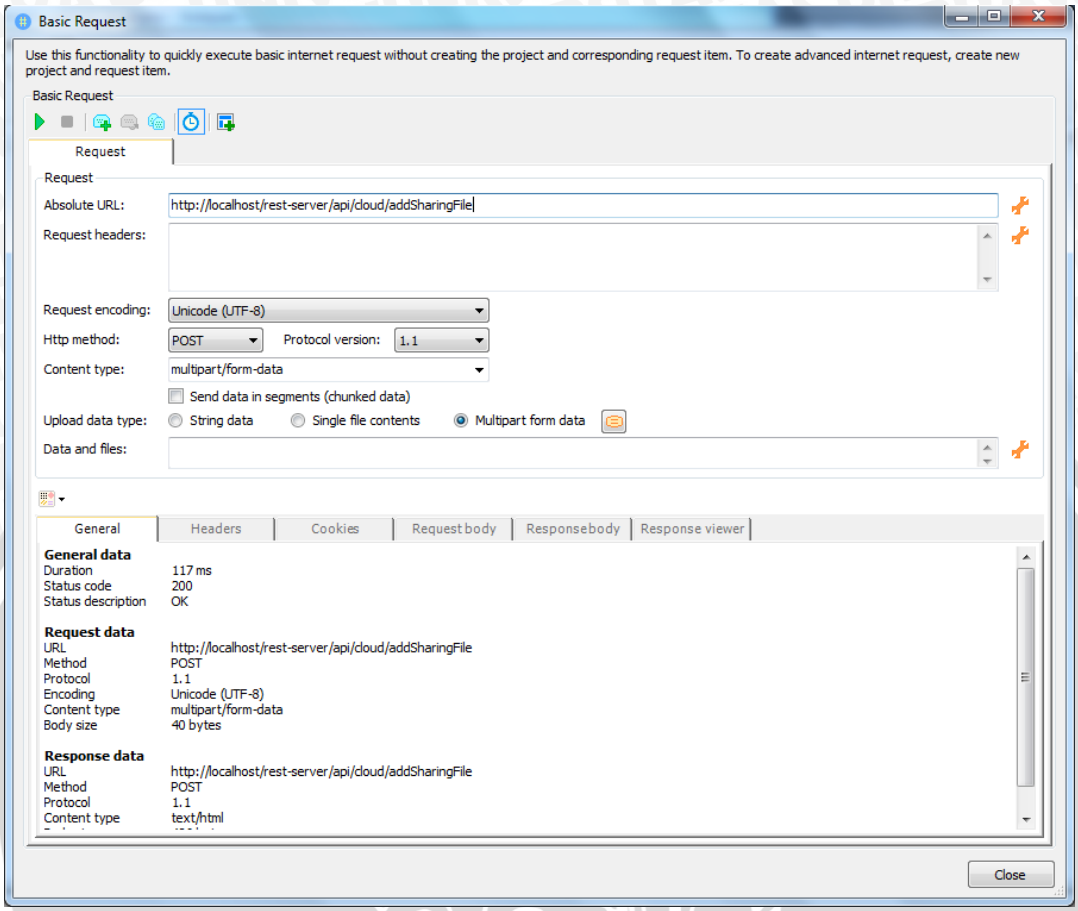


Gambar 6.6 Hasil Pengujian pada *Post User*

Dari gambar 6.6, *absolute* URL merupakan *service endpoint* dari *server cloud* untuk *get file*. HTPMethod merupakan method yang digunakan untuk melakukan *request*, dalam hal ini adalah *method POST*. Setelah request dieksekusi maka akan ditampilkan hasil dari request tersebut. Pada *General Data*, *duration* menunjukkan durasi atau waktu yang digunakan untuk melakukan request, *status code* mengembalikan nilai 200 yang menyatakan bahwa *request* HTTP berhasil dengan *status description* OK.

4. Hasil Pengujian *Post Share*

Hasil pengujian pada *post share* ditunjukkan oleh gambar 6.7.



Gambar 6.7 Hasil Pengujian pada *Post Share*

Dari gambar 6.7, *absolute* URL merupakan *service endpoint* dari *server cloud* untuk *get file*. HTPMethod merupakan method yang digunakan untuk melakukan *request*, dalam hal ini adalah *method POST*. Setelah request dieksekusi maka akan ditampilkan hasil dari request tersebut. Pada *General Data*, *duration* menunjukkan durasi atau waktu yang digunakan untuk melakukan request, *status code* mengembalikan nilai 200 yang menyatakan bahwa *request* HTTP berhasil dengan *status description* OK.

6.3.4 Analisis

Mengacu pada hasil pengujian REST *Web Service*, sistem telah dapat melayani http *request* dengan baik. Hail ini dapat dilihat dari *response* yang diberikan dimana dari semua kasus uji baik untuk *method get* mapun *post*, masing-masing

megembalikan status *code* 200 yang merupakan *response* standar bahwa http *request* berhasil. Dengan demikian dapat disimpulkan bahwa REST API telah mampu bekerja dengan baik.

BAB VII
PENUTUP

7.1 Kesimpulan

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

1. Proses analisis dalam membangun sistem informasi bimbingan skripsi dilakukan dengan menggunakan analisa terkait pelaksanaan bimbingan skripsi di PTIIK, melakukan studi literatur, serta kuisioner untuk mendeskripsikan kebutuhan fungsional dan non-fungsional sistem informasi.
2. Proses perancangan sistem informasi bimbingan skripsi dilakukan dengan membuat beberapa perancangan yang sesuai dengan tahap analisa sebelumnya, seperti perancangan arsitektural, perancangan basis data, perancangan *use case*, *class diagram*, perancangan *activity diagram*, perancangan *sequence diagram*, dan perancangan antarmuka sistem yang akan dibuat. Dan dari rancangan tersebut akan dilakukan implementasi melalui pembuatan kode program dalam bentuk *class-class* program serta melakukan implementasi basis data menggunakan MySQL.
3. Fitur penyimpanan berkas berbasis *cloud* memiliki API berupa REST *web service* dengan menggunakan format pertukaran data JSON. Sistem informasi bimbingan skripsi akan melakukan *request* data serta layanan, selanjutnya REST *web service* akan mengembalikan data yang diminta sesuai dengan parameter yang diberikan.
4. Untuk mengetahui validitas sistem, dilakukan pengujian dengan menggunakan metode *white box*, *black box*, dan pengujian REST *web service*. Dan dari hasil pengujian yang telah dilakukan dapat diketahui bahwa sistem 100% valid.

7.2 Saran

Saran yang dapat diberikan setelah menyelesaikan penelitian skripsi ini adalah:

1. Pengembangan sistem dapat dilakukan dengan penambahan fitur untuk melakukan *edit* dokumen secara *real-time*.
2. Pengembangan sistem dapat dilakukan dengan penambahan fitur untuk *upload* dokumen skripsi dalam format .docx.
3. Pengembangan sistem dapat dilakukan dengan penambahan fitur untuk mendeteksi tingkat plagiasi terhadap dokumen skripsi mahasiswa.
4. Untuk menangani terjadinya *bug* ketika nama *file* yang di-*upload* sama, maka dapat digunakan teknik *hash*.

DAFTAR PUSTAKA

[ADE-14] O, Eletta Adeola., A, Ayeni Adeseko., Philip, Simon. 2014. *Cloud Based Document Storage System for a Higher Institution*. IJASCSE, Volume 3, Issue 12.

[ALF-07] Al Fatta, Hanif. 2007. *Analisis dan Perancangan Sistem Informasi untuk Keunggulan Bersaing Perusahaan & Organisasi Modern*. Yogyakarta: Andi Offset.

[APR-15] Dwiyani, Aprillita. *Perancangan Sistem Pendukung Bimbingan Online Tugas Akhir Mahasiswa Program Studi Teknik Informatika*. *Online*. http://download.portalgaruda.org/article.php?article=112039&val=2313. 8 April 2015.

[AZI-08] Aziz, M Farid. 2005. *Object Oriented Programming dengan PHP5*. Jakarta: Elex Media Komputindo.

[CAN-15] Candra R, Resa. 2015. *User Guide Manajemen Data Antar Cloud Storage dengan Mover*. Bandung: Modula.

[ELC-12] Tim Elcom. 2012. *Cloud Computing – Aplikasi berbasis web yang mengubah cara kerja dan kolaborasi Anda secara online*. Yogyakarta: Andi Offset.

[HER-04] Hermawan, Julius. 2004. *Analisa Desain & Pemrograman Berorientasi Obyek dengan UML dan Visual Basic NET*. Yogyakarta: Andi Offset.

[HAR-04] Hariyanto, Bambang. 2004. *Rekayasa Sistem Berorientasi Objek*. Bandung: Informatika.

[HUT-14] Hutahaeon Jeperson.2014. *Konsep Sistem Informasi*. Yogyakata: Deepublish

[JON-10] Jones, M Tim. 2010. *Anatomy of a cloud storage infrastructure models, features, and internals*. *Online*. http://www.ibm.com/developerworks/cloud/library/cl-cloudstorage/cl-cloudstorage-pdf.pdf. 27 Pebruari 2015

[KBB-15] Kamus Besar Bahasa Indonesia (KBBI) Kamus Versi *Online*. *Online*. http://kbbi.web.id/bimbing. 18 Mei 2015.

[LEE-12] Sunguk, Lee. 2012. *Unified Modeling Language (UML) for Database Systems and Computer Applications*. International Journal of Database Theory and Application Vol. 5, No. 1.

[MEL-11] Mell, Peter., Grance, Timothy. 2011. *The NIST Definition of Cloud Computing*. United States: National Institute of Standars and Technology.

[NUG-09] Nugroho, Adi. 2009. *Rekayasa Perangkat Lunak Menggunakan UML dan Java*. Yogyakarta : Andi Offset.

[NGH-11] Nugroho, Adi. 2011. *Perancangan dan Implementasi Sistem Basis Data*. Yogyakarta : Andi Offset.

[OBJ-11] Object Management Group. 2011. *Business Process Modelling and Notation BPMN*. Needhan: OMG.

[PRE-01] Pressman, Roger S. 2001. *Software Engineering : A Practitioner's Approach, Fifth Edition*. McGraw Hill.

[RIZ-11] Rizky, Soetam. 2011. *Konsep Dasar Rekayasa Perangkat Lunak*. Jakarta: Prestasi Pustaka.

[ROS-14] A. S, Rosa., Shalahudin, M. 2011. *Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek)*. Bandung: Modula.

[SDL-15] Tutorials Point. *Software Development Life Cycle (SDLC)*. *Online*. http://www.tutorialspoint.com/sdlc/sdlc_tutorial.pdf. 5 Pebruari 2015.

[SIS-15] Wikipedia Bahasa Indonesia Definisi Sistem Informasi. *Online*. https://id.wikipedia.org/wiki/Sistem_informasi. 5 Mei 2015.

[SOM-03] Sommerville, Ian. 2003. *Software Engineering : Rekayasa Perangkat Lunak, Edisi Enam, Jilid Satu*, (Alih Bahasa, Dra. Yuhilza Hanum M.Eng). Jakarta: Erlangga.

[TRI-12] Triandini, Evi., Suardika, I Gede. 2012. *Step by Step Desain Proyek Menggunakan UML*. Yogyakarta: Andi Offset.

[WAH-10] Wahana Komputer. 2010. *Shourt Course Pengembangan Aplikasi Database Berbasis JavaDB dengan Netbeans*. Yogyakarta: Andi Offset.

[WAL-12] Waloeyo, Yohan Jati. 2012. *Cloud Computing Aplikasin berbasis web yang mengubah cara kerja dan kolaborasi anda secara online*. Yogyakarta: Andi Offset.

[WIK-15] Wikipedia Bahasa Indonesia Definisi Skripsi. *Online*. http://id.wikipedia.org/wiki/Skripsi. 18 Mei 2015.

[YUH-08] Yuhefizard. 2008. *Database Management Menggunakan Microsoft Access 2003*. Jakarta: Elex Media Komputindo.

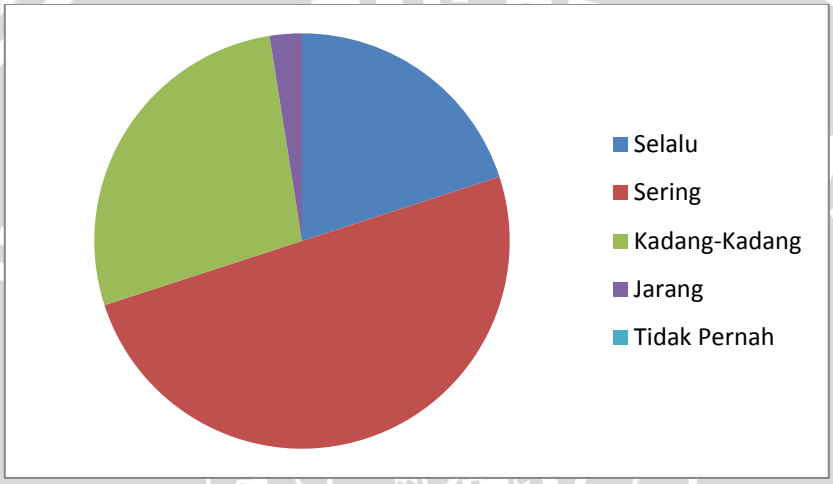
[YUS-15] Yusron, Mayang Laily, et al. *Rancang Bangun REST Web Service Pada Aplikasi Penentuan Portofolio Saham Menggunakan Model Makrowitz Dan JQuery Mobile*. Online. 14 Maret 2015.

LAMPIRAN

Lampiran 1 hasil kuisioner

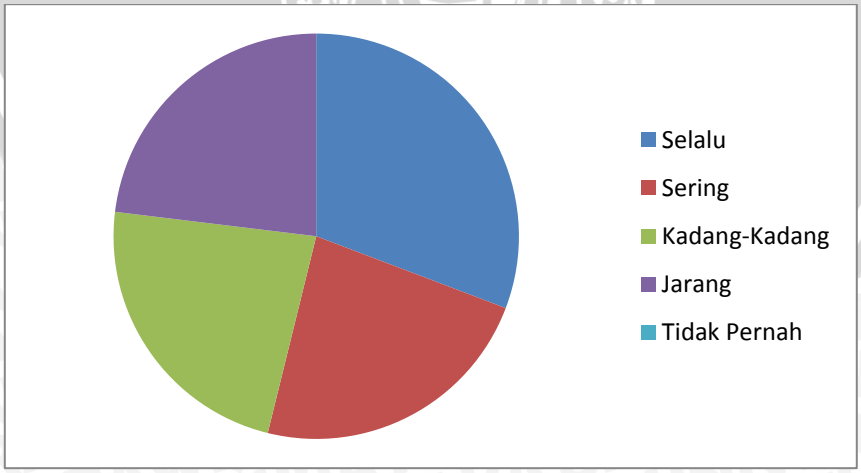
1. Dalam melaksanakan bimbingan skripsi, apakah saudara merasa kesulitan dalam mencari keberadaan dosen pembimbing?

Selalu	Sering	Kadang-Kadang	Jarang	Tidak Pernah
8	20	11	1	0



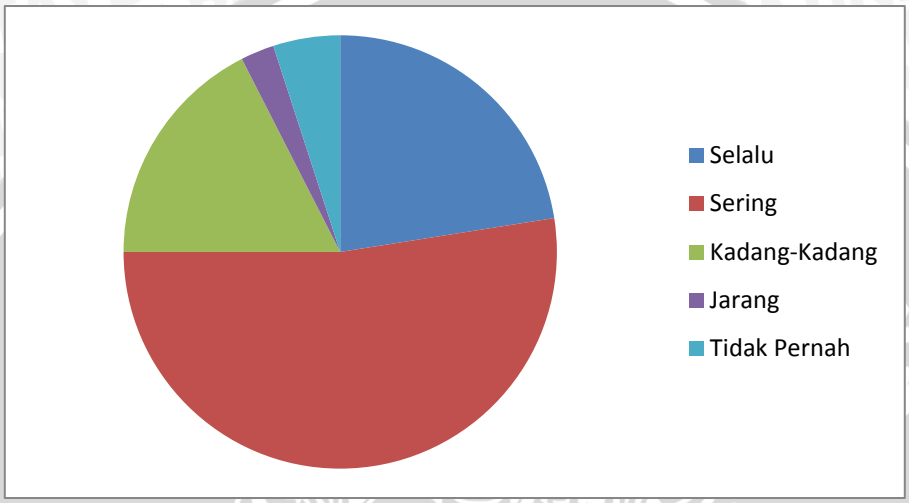
2. Dalam melaksanakan bimbinga skripsi, apakah saudara perlu mengantri ?

Selalu	Sering	Kadang-Kadang	Jarang	Tidak Pernah
16	12	12	12	0



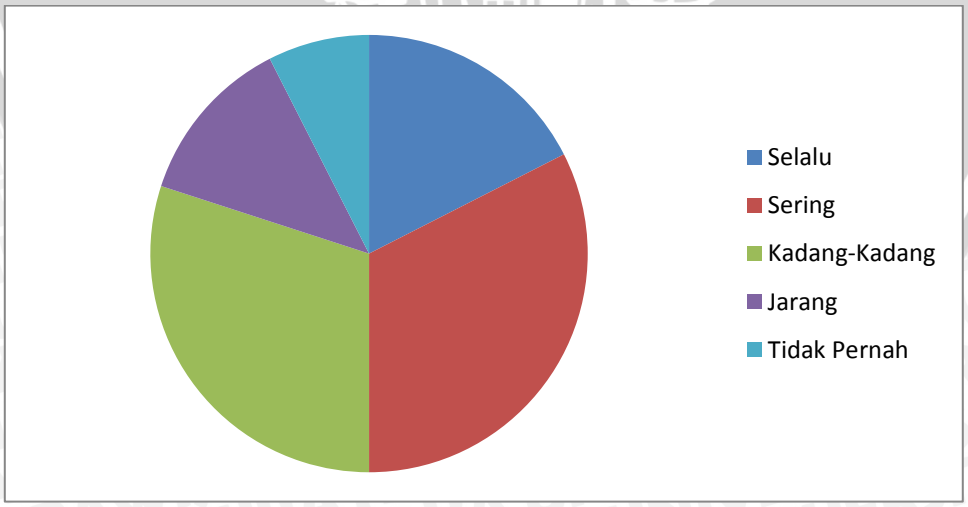
3. Apakah setiap akan melaksanakan bimbingan skripsi saudara mencetak ulang dokumen skripsi saudara akibat adanya revisi?

Selalu	Sering	Kadang-Kadang	Jarang	Tidak Pernah
9	21	7	1	2



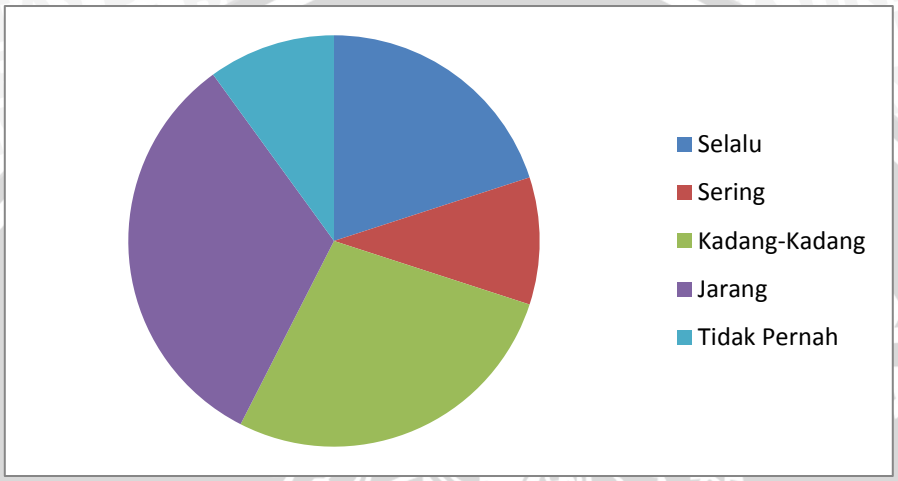
4. Apakah proses bimbingan skripsi saudara terbengkalai akibat adanya perbedaan jarak dan jadwal antara suadara dan dosen pembimbing?

Selalu	Sering	Kadang-Kadang	Jarang	Tidak Pernah
7	13	12	5	3



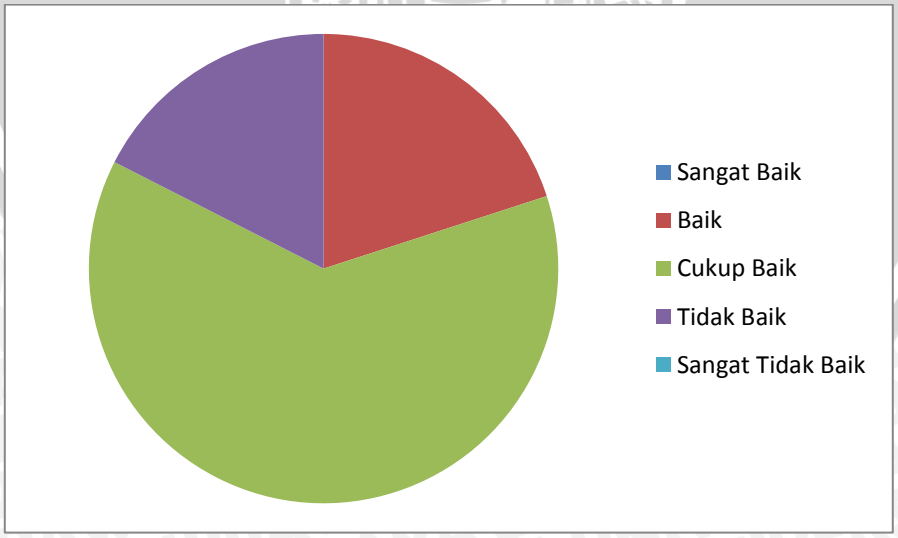
5. Apakah setiap selesai melaksanakan bimbingan skripsi saudara mencatat laporan pada kartu kendali bimbingan skripsi?

Selalu	Sering	Kadang-Kadang	Jarang	Tidak Pernah
8	4	11	13	4



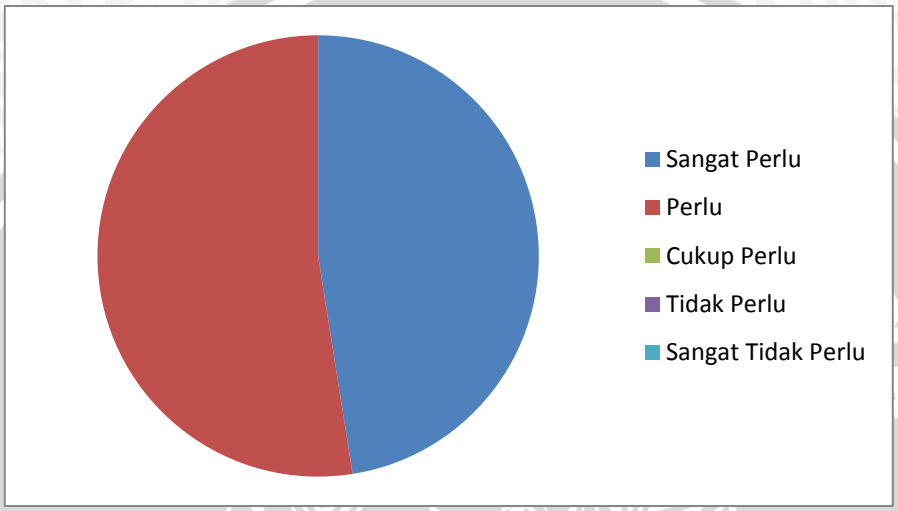
6. Apakah selama ini dokumen skripsi dan dokumen revisi saudara sudah terorganisir dengan baik dan sistematis?

Sangat Baik	Baik	Cukup Baik	Tidak Baik	Sangat Tidak Baik
0	8	25	7	0



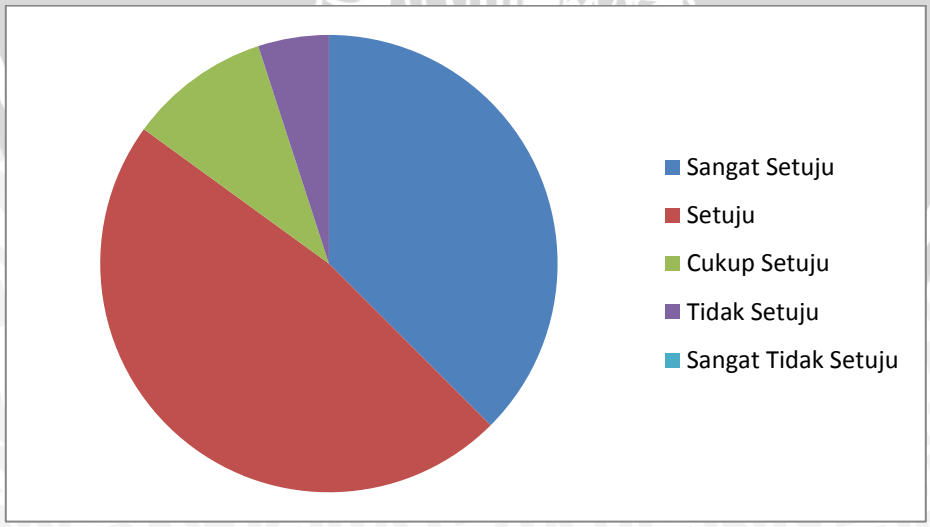
7. Apakah saudara merasa memerlukan media penyimpanan berkas yang aman untuk menyimpan dokumen skripsi saudara?

Sangat Perlu	Perlu	Cukup Perlu	Tidak Perlu	Sangat Tidak Perlu
19	21	0	0	0



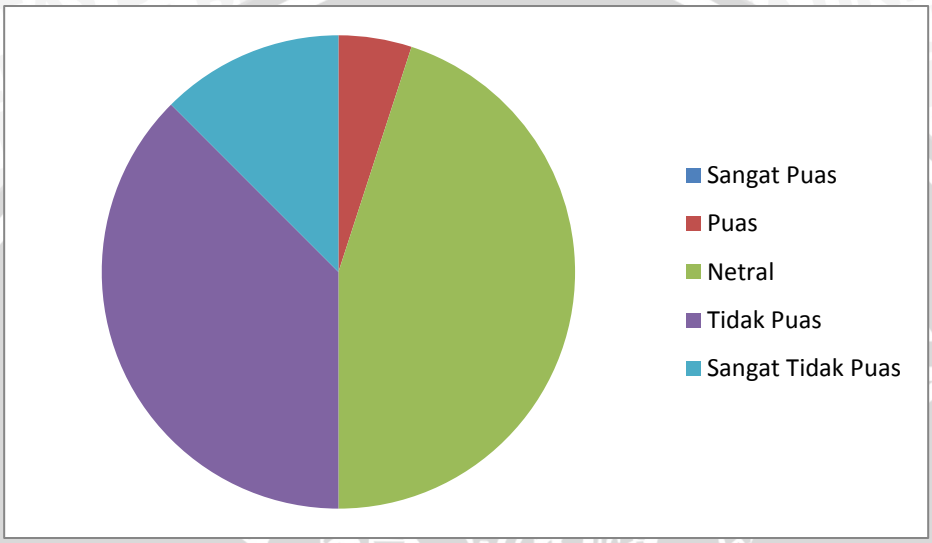
8. Apakah sistem penyimpanan berkas secara *online* merupakan media yang tepat dan aman bagi saudara untuk menyimpan dokumen skripsi?

Sangat Setuju	Setuju	Cukup Setuju	Tidak Setuju	Sangat Tidak Setuju
15	19	4	2	0



9. Bagaimana penilaian suadara terhadap manual prosedur pelaksanaan bimbingan skripsi yang ada di PTIIK saat ini?

Sangat Puas	Puas	Netral	Tidak Puas	Sangat Tidak Puas
0	2	18	15	5



10. Apakah saudara merasa bahwa diperlukan suatu sistem untuk meningkatkan pelayanan dan memberikan kemudahan dalam pelaksanaan bimbingan skripsi?

Sangat Perlu	Perlu	Cukup Perlu	Tidak Perlu	Sangat Tidak Perlu
18	19	3	0	0