PEMBANGUNAN SISTEM INFORMASI PRAKTIK PENGALAMAN LAPANGAN (PPL) BERBASIS ANDROID (STUDI PADA FAKULTAS ILMU KOMPUTER UNIVERSITAS BRAWIJAYA)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Rizky Wahyu Setiawan
NIM: 155150400111093

PROGRAM STUDI SISTEM INFORMASI
JURUSAN SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

PEMBANGUNAN SISTEM INFORMASI PRAKTIK PENGALAMAN LAPANGAN (PPL) BERBASIS ANDROID (STUDI PADA FAKULTAS ILMU KOMPUTER UNIVERSITAS BRAWIJAYA)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Rizky Wahyu Setiawan
NIM: 155150400111093

Skripsi ini telah diuji dan dinyatakan lulus pada
02 Januari 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing 1	Dosen Pembimbing 2
Satrio Agung Wicaksono, S.Kom, M.Kom	Admaja Dwi Herlambang , S.Pd., M.Pd
NIP: 19860521 201212 1 001	NIK: 2016098908021001

Mengetahui
Ketua Jurusan Sistem Informasi

Dr. Eng. Herman Tolle, S.T, M.T.
NIP: 19740823 200012 1 001

PERNYATAAN ORISINALITAS

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 referensi.

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

Malang, 04 Januari 2019

Rizky Wahyu Setiawan

NIM: 155150400111093

PRAKATA

Puji syukur penulis panjatkan kehadirat Allah SWT, karena berkat rahmat dan hidayah-Nya, penulis mampu menyelesaikan laporan skripsi dengan judul "Pembangunan Sistem Informasi Praktik Pengalaman Lapangan (PPL) Berbasis Android (Studi Pada Fakultas Ilmu Komputer Universitas Brawijaya)".

Dalam penulisan dan penyusunan laporan penulis mendapatkan banyak dukungan dan bantuan dari banyak pihak. Penulis ingin mengucapkan terimakasih yang sebesar-besarnya kepada :

1. Bapak Satrio Agung Wicaksono, S.Kom., M.Kom., selaku pembimbing 1 dan pembimbing akademik yang selalu memberikan masukan dan saran kepada penulis.
2. Bapak Admaja Dwi Herlambang, S.Pd., M. Pd., selaku pembimbing 2 yang selalu memberikan arahan dan pencerahan kepada penulis.
3. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya Malang. Serta semua dosen dan staff Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Herman Tolle, Dr. Eng., S.T, M.T selaku Ketua Jurusan Sistem Informasi Universitas Brawijaya Malang.
5. Bapak Yusi Tyroni Mursityo, S.Kom., M.AB selaku Ketua Program Studi Sistem Informasi Universitas Brawijaya Malang.
6. Seluruh jajaran dosen dan staff Fakultas Ilmu Komputer Universitas Brawijaya
7. Bapak dan Ibu saya sekeluarga atas segala dukungan dan do'a yang selalu diberikan. Teman-teman kontrakan Joyogrand boys, The-Ex *Community*, BCC Filkom, dan teman-teman 19 yang selalu memberikan informasi dan motivasi. Karunia Eka Safitri, Dimas, Ridho dan Christine yang selalu memberikan semangat dan motivasi. Semua teman-teman Keluarga Besar Sistem Informasi angkatan 2015. Semoga kelak menjadi angkatan emas, menjadi generasi emas Indonesia.

Penulis menyadari bahwa dalam penyusunan laporan skripsi ini masih banyak kekurangan. Maka dari itu, penulis sangat mengharapkan kritik dan saran yang membangun. Semoga laporan skripsi ini bermanfaat bagi pembaca dan penulis. Amin.

Malang, 04 Januari 2019

Penulis

rizkywahyu9999@gmail.com

ABSTRAK

Rizky Wahyu Setiawan, Pembangunan Sistem Informasi Praktik Pengalaman Lapangan (PPL) Berbasis Android (Studi Pada Fakultas Ilmu Komputer Universitas Brawijaya)

Pembimbing: Satrio Agung Wicaksono, S.Kom., M.Kom dan Admaja Dwi Herlambang, S.Pd., M.Pd

Kompetensi pedagogik merupakan syarat utama yang harus dimliki oleh seorang tenaga pengajar. Program Studi Pendidikan Teknologi Informasi, Fakultas Ilmu Komputer Universitas Brawijaya mewajibkan mahasiswanya untuk melakukan kegiatan Praktik Pengalaman Lapangan (PPL), untuk membekali lulusannya menjadi seorang tenaga pengajar yang memiliki kompetensi pedagogik. Didalam kegiatan PPL terdapat fase Pendaftaran PPL, Pelaksanaan PPL I, Pelaksanaan PPL II, dan Pelaporan PPL. Kegiatan PPL di lingkungan Fakultas Ilmu Komputer saat ini belum terdapat sistem informasi yang mendukung jalannya kegiatan PPL. Sehingga aktivitas pengurusan tahapan PPL menjadi kurang efektif. Penelitian ini bertujuan untuk membuat sistem informasi yang mampu membantu proses pengurusan kegiatan PPL bagi semua pihak terkait. Sistem yang dibuat memiliki fitur pendaftaran PPL bagi mahasiswa, pencatatan *logbook* bagi mahasiswa, persetujuan *logbook* bagi guru pamong, dan penilaian PPL bagi guru pamong. Dalam pengembangannya sistem dibangun diatas *platform Android* untuk meningkatkan portabilitas dan aksesibilitas pengguna. Metode pengembangan sistem yang digunakan adalah *extreme programming.* Kemudian sistem akan diuji dengan menggunakan metode *Black Box Testing* dan metode *Mobile (Android Platform) Compatibility Testing*. Hasilnya menunjukkan bahwa sistem 100% valid dan 100% kompatibel.

Kata kunci: *Pembangunan, Sistem Informasi, Android, Pendidikan, PPL*

ABSTRACT

Rizky Wahyu Setiawan, Development Information System of Educational Field Experience Placement Android Platform Based (Study at Faculty of Computer Science Brawijaya University)

Supervisors: Satrio Agung Wicaksono, S.Kom., M.Kom and Admaja Dwi Herlambang, S.Pd., M.Pd

Pedagogic competence is the main of terms that must be had for a teacher. in Information Technology Education Program Study, Faculty of Computer Science Brawijaya University, obligate their students to do Educational Field Experience Placement to equip their graduates become a teacher. PPL has several part, they are Registration of PPL, Practice PPL I, Practice PPL II, and PPL Report. However, the information system for this program has not supported yet in Faculty of Computer Science at the same time. So that, the activity of PPL become less effective. According to that problem, this research aim to develop an information system that can support process PPL for every stakeholder. The system that will be developed having four features, they are Registration PPL for student, Record Logbook for student, Approve Logbook for teacher, and Assessment PPL for teacher. This system will be developed using Android platform for increase portability and accesibility of system. In this development is used metodology Extreme Programming. Then system will be tested using Black Box and Mobile (Android Platform) Compatibility Testing. The result shows that the system is 100% valid and 100% compatible.

Keyword ***:*** *Development, Information System, Android, Education, PPL*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Teknologi informasi saat ini mulai menyebar dan masuk ke dalam berbagai aspek kehidupan, salah satunya adalah dunia pendidikan. Seperti contoh penggunaan *E-Learning* oleh tenaga pengajar untuk memberikan dan mengumpulkan tugas kepada mahasiswa, penggunaan sistem ujian berbasis komputer, penggunaan kelas *remote,* dll. Menurut Kepala Pusat Teknologi dan Komunikasi Kementrian dan Kebudayaan RI, Bapak Gatot Pramono, meyakini bahwa dengan menggunakan teknologi maka proses pendidikan di Indonesia akan menjadi lebih efisien (Dwiharianto, 2018). Berdasar peraturan Undang-Undang Nomor 20 Tahun 2003 Bab I ayat 11, menyatakan bahwa pendidikan formal di Indonesia terdiri dari beberapa tingkatan yang terstruktur dan berjenjang. Dimulai dari tingkat paling bawah adalah Pendidikan Dasar, Pendidikan Menengah dan Pendidikan Tinggi.

Salah satu tempat untuk mengenyam Pendidikan Tinggi di Indonesia adalah di Program Studi Pendidikan Teknologi Informasi yang merupakan program studi yang ada di Fakultas Ilmu Komputer Universitas Brawijaya. Program Studi Pendidikan Teknologi Informasi adalah suatu wujud jawaban Universitas Brawijaya atas tantangan global untuk mencetak generasi yang memiliki kompetensi sebagai tenaga pengajar sekaligus memiliki kompetensi tenaga profesional Teknologi Informasi dan jiwa *entrepreneur.* Misi Program Studi Pendidikan Teknologi Informasi adalah mendidik lulusan sebagai tenaga pendidik dalam bidang Teknologi Informasi, maka lulusan memiliki kompetensi utama bidang keguruan yang meliputi kompetensi pedagogik, kompetensi profesional bidang teknologi informasi, kompetensi sosial, dan kepribadian.

Untuk menjadi lulusan Program Studi Pendidikan Teknologi, seorang mahasiswa harus memiliki kompetensi pedagogik salah satunya. Kompetensi pedagogik adalah kompetensi khas yang harus dimiliki oleh seorang tenaga pengajar, serta kompetensi ini dapat menjadi penentu tingkat keberhasilan proses dan hasil pembelajaran peserta didiknya (Sapoetra, 2017). Kompetensi ini tidaklah secara *instant* diperoleh seorang tenaga pengajar. Melainkan harus didapatkan dengan cara belajar terus-menerus dan sistematis, baik pada masa pra jabatan (pendidikan calon guru) maupun selama masa jabatan, yang juga dipengaruhi oleh bakat, minat dan potensi keguruan lainnya dari masing-masing individu yang bersangkutan. Oleh karena itu, untuk memfasilitasi mahasiswa dalam mengasah kompetensi pedagogik, Program Studi Pendidikan Teknologi Informasi mewajibkan semua mahasiswa untuk mengambil mata kuliah wajib Praktik Pengalaman Lapangan.

Praktik Pengalaman Lapangan atau biasa disingkat PPL menurut Buku Panduan Penyelesaian dan Evaluasi Praktik Pengalaman Lapangan (PPL) Fakultas Ilmu Komputer Universitas Brawijaya Tahun 2018, merupakan salah satu mata kuliah wajib yang harus ditempuh oleh mahasiswa S1 Program Studi Pendidikan

Teknologi Informasi (PTI) Fakultas Ilmu Komputer Universitas Brawijaya untuk melatih kemampuannya sebagai calon tenaga pengajar pada bidang Teknologi Informasi dan Komunikasi (TIK). Didalam proses Praktik Pengalaman Lapangan, setidaknya mahasiswa dituntut untuk melakukan 5 praktik yaitu, praktik mengajar pada bidang ilmu TIK; Praktik mengenali kesulitan belajar siswa pada bidang TIK dan memberikan solusi terhadap hambatan pada proses pembelajaran TIK; Praktik manajemen sekolah berbasis TIK; Praktik pemanfaatan TIK untuk membantu para guru; Atau praktik tugas kependidikan yang lain dan masih relevan, seperti konseling TIK, dll. Kegiatan ini dilaksankan baik di kampus dan di sekolah mitra seperti Sekolah Menengah Kejuruan (SMK) atau Balai Latihan Kerja (BLK) pada bidang TIK (Tim Penyusun Pedoman, 2018).

Berdasarkan hasil wawancara dengan Ketua Program Studi Pendidikan Teknologi Informasi Fakultas Ilmu Komputer Universitas Brawijaya (Lampiran A.1) dan 4 mahasiswa program studi tersebut (Lampiran A.2), diketahui tahapan dalam kegiatan PPL ini memerlukan proses registrasi mahasiswa, pencatatan logbook oleh mahasiswa dan penilaian terhadap mahasiswa oleh guru pamong di sekolah mitra. Saat ini belum tersedia sistem informasi yang mendukung jalannya kegiatan PPL di lingkungan kampus. Sehingga aktivitas pengurusan tahapan PPL menjadi kurang efektif. Dari permasalahan tersebut, maka perlu dikembangkan sebuah sistem informasi berbasis *mobile* yang mampu membantu proses pengurusan kegiatan PPL bagi semua pihak terkait. Sebelum sistem dibangun, perlu dilakukan penggalian kebutuhan sistem dari berbagai pihak yang bersangkutan. Lalu hasil penggalian kebutuhan dianalisa dan dijadikan dasar perancangan sampai implementasi nantinya. Alasan pemilihan *platform mobile* adalah selain mengikuti *trend* saat ini juga dikarenakan lebih menekankan kemudahan akses bagi pengguna dan meningkatkan portabilitas sistem.

Berdasarkan uraian permasalahan yang telah dijelaskan, maka penulis ingin melakukan penelitian dengan judul Pembangunan Sistem Informasi Praktik Pengalaman Lapangan (PPL) Berbasis Android Studi Pada Program Studi Pendidikan Teknologi Informasi Fakultas Ilmu Komputer Universitas Brawijaya. Diharapkan dengan adanya sistem informasi ini dapat mempermudah mahasiswa program studi tersebut dalam melakukan registrasi dan pencatatan logbook harian PPL selama tahapan kedua PPL berlangsung. Serta dapat mempermudah guru pamong dalam melakukan penilaian terhadap mahasiswa yang bersangkutan selama melaksankan PPL di sekolah mitra guru pamong tersebut.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, muncul beberapa pertanyaan rumusan masalah pada penelitian ini, antara lain :

1. Bagaimana hasil analisis kebutuhan sistem informasi PPL Program Studi PTI Fakultas Ilmu Komputer Universitas Brawijaya ?
2. Bagaimana hasil perancangan dan implementasi sistem informasi PPL Program Studi PTI Fakultas Ilmu Komputer Universitas Brawijaya ?

3. Bagaimana hasil pengujian sistem informasi PPL Program Studi PTI Fakultas Ilmu Komputer Universitas Brawijaya ?

1.3 Tujuan

Berikut adalah beberapa tujuan dilakukannya penelitian ini,

1. Mengetahui hasil analisis kebutuhan sistem informasi PPL Program Studi PTI Fakultas Ilmu Komputer Universitas Brawijaya.
2. Merancang dan membuat sistem informasi yang mampu memenuhi semua kebutuhan proses kegiatan PPL Program Studi PTI Fakultas Ilmu Komputer Universitas Brawjiaya.
3. Mengetahui hasil pengujian sistem informasi PLL Program Studi PTI Fakultas Ilmu Komputer Universitas Brawijaya

1.4 Manfaat

Pada jangka panjang, hasil penelitian ini dapat bermanfaat untuk membantu proses kegiatan PPL yang dilakukan di lingkungan Program Studi PTI Fakultas Ilmu Komputer. Sistem dapat mempercepat proses atau bahkan dapat meningkatkan pelayanan pada sisi *staff.* Bagi mahasiswa yang bersangkutan dapat mempermudah proses administratif PPL. Sedangkan bagi peneliti bermanfaat sebagai sarana untuk meningkatkan keilmuan peneliti di bidang teknologi informasi dan ilmu komputer. Selain itu, diharapkan dapat menjadi bahan referensi atau pengembangan sistem untuk penelitian selanjutnya.

1.5 Batasan Masalah

Batasan masalah ditentukan supaya permasalahan yang diangkat dapat diselesaikan lebih terfokus dan tidak melebar. Adapun batasan masalah pada penelitian ini adalah,

1. Proses yang dimasukkan ke dalam sistem adalah proses pendaftaran, penjadwalan logbook dan penilaian mahasiswa di sekolah oleh guru pamong.
2. Sistem informasi dianggap selalu terhubung dengan internet, dikarenakan setiap transaksi data menggunakan API pada *web servive*.

1.6 Sistematika Pembahasan

Berikut adalah struktur dari penulisan skripsi pada penelitian ini,

BAB I Pendahuluan

Bab ini menjelaskan mengenai latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah serta sistematika penulisan laporan penelitian.

BAB II Landasan Kepustakaan

Bab ini menerangkan tinjauan pustaka, dasar-dasar teori dan konsep dari literatur ilmiah yang sesuai dengan tema serta mendukung dalam pembuatan sistem informasi pada penelitian ini. Pada bab ini juga terdapat beberapa penelitian sebelumnya yang memiliki beberapa kesamaan komponen penelitian yang mendukung topik yang peneliti angkat.

BAB III Metodologi

Bab ini membahas mengenai langkah-langkah selama melakukan penelitian secara sistematik dan langkah-langkah menyelesaikan masalah penelitian. Dalam bab ini juga membahas metode yang digunakan dalam membangun sistem informasi.

BAB IV Hasil dan Pembahasan

Didalam bab ini tediri dari tahapan analisis kebutuhan, perancangan sistem, implementasi sistem, pengujian sistem dan analisis hasil pengujian sistem. Pada tahap analisis kebutuhan menjelaskan mengenai pendekatan peneliti dalam menggali kebutuhan dari setiap *stakeholder*. Pada tahapan perancangan tedapat gambaran umum mengenai sistem yang akan dibangun yang direpresentasikan dalam bentuk diagram UML. Pada tahapan implementasi membahas mengenai proses implementasi sistem informasi PPL berbasi Android dengan menggunakan metode *Extreme Programming.* Proses dimulai dari pembuatan *user story* sampai tahap implementasi kode dan dilanjut dengan iterasi-iterasi sampai calon pengguna sistem terpenuhi semua kebutuhannya terhadap sistem. Pada tahapan pengujian dan analisis berisi tentang metode pengujian dan analisis berdasar hasil pengujian yang telah dilakukan

BAB V Penutup

Didalam bab ini terdapat kesimpulan yang diperoleh berdasar hasil perancangan, implementasi serta pengujian sistem. Didalam bab ini juga terdapat saran-saran untuk mempermudah pengembangan penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Kajian pustaka menjelaskan mengenai kajian penelitian yang telah dilakukan oleh peneliti lain dan memiliki kemiripan terhadap topik pembahasan yang diangkat oleh peneliti. Hasil kajian pustaka penelitian lain yang dilakukan oleh Faizal Johan Atletiko (Atletiko, 2017), menunjukkan proses pembangunan sistem informasi berbasis Android untuk memantau kurir obat pada suatu perusahaan farmasi. Pada sistem tersebut dibagi menjadi 3 modul, modul pertama digunakan oleh perusahaan distributor obat. Modul yang kedua digunakan oleh penjual atau *salesman* untuk memesan obat. Modul yang ketiga digunakan oleh kurir obat untuk mendapatkan informasi pemesanan dan lokasi pemesan. Alasan pembuatan sistem informasi ini dikarenakan selama ini proses pemesanan obat dari rumah sakit atau apotek masih manual menggunakan lembar permintaan dan dimasukkan dalam pembukuan kemudian diberikan kepada *salesman* ketika *salesman* berkunjung ke rumah sakit atau apotek. Sedangkan pemilihan *platform* Android dikarenakan kebutuhan untuk mengakses GPS guna mengetahui lokasi kurir dan lokasi pemesan obat. Sistem yang dikembangkan juga menggunakan API untuk menyimpan data obat pada *web service host.* Selain itu penulis tersebut juga menggunakan beberapa layanan Google seperti Google Cloud Messaging dan Google Maps. Hal yang ingin diadaptasi kedalam skripsi ini adalah konsep penyimpanan data menggunakan *web service* melalui aplikasi Android. Selain itu, aplikasi ini digunakan oleh beberapa aktor baik dari dalam organisasi maupun dari luar organisasi. Mirip dengan konsep yang penulis kembangkan nantinya.

Penelitian lain yang juga terkait adalah penelitian yang dilakukan oleh K. Kularbphettonga dkk. (Kularbphettonga dkk., 2015). Pada penelitian tersebut menunjukkan proses pembuatan *m-learning* untuk mempermudah proses pembelajaran matematika diskrit bagi siswa menggunakan *platform mobile.* Pada penelitian tersebut dibagi menjadi 2 tujuan penelitian, pertama untuk membuat aplikasi mobile bagi siswa dan kedua untuk melakukan pengujian serta evaluasi sistem menggunakan teknik *blackbox* untuk mengevaluasi kemampuan aplikasi dan menggunakan teknik kuisioner untuk mengukur tingkat kepuasan pengguna baik dari siswa maupun dari pakar. Pada penelitian ini, penulis skripsi ingin mengadaptasi proses pengujian aplikasi mobile menggunakan metode *blackbox testing* untuk menguji kehandalan aplikasi ketika digunakan.

Penelitian lain yang dilakukan oleh Christopher Dong dan Xing Liu (Dong & Liu, 2013), menunjukkan proses pembangunan aplikasi berbasis Android dengan tujuan untuk mempermudah proses belajar para siswa dalam mempelajari suatu bahasa. Aplikasi ini nantinya akan digunakan oleh lembaga kursus bahasa. Penggunaan aplikasi ini dapat mempermudah siswa dalam mengingat kosa kata baru didalam fikirannya. Kosa kata yang digunakan di dalam aplikasi mengacu pada buku cetak yang telah dipublikasikan secara umum. Siswa yang menggunakan aplikasi ini di lembaga kursusnya dapat melatih ulang kosa kata

yang dipelajari menggunakan fitur *virtual flashcard* dan kuis pilihan ganda. Penulis ingin mengadaptasi cara peneliti tersebut dalam menerapkan teknologi berbasis android didalam dunia pendidikan.

2.2 Profil Program Studi Pendidikan Teknologi Informasi Fakultas Ilmu Komputer Universitas Brawijaya

Sebagai salah satu perguruan tinggi terbesar di Indonesia, Universitas Brawijaya berusaha menjawab tantangan global yang bergerak terus secara dinamis. Pergerakan global tersebut membuat persaingan pasar yang semakin ketat sehingga menjadikan setiap individu juga harus berusaha lebih keras untuk bersaing di pasar global. Individu yang memiliki kemampuan menguasai informasi akan memiliki lebih besar peluang meraih kesuksesan. Maka dari itu, dunia pendidikan dan teknologi informasi ibarat dua mata pisau yang tidak bisa dipisahkan. Melalui Program Studi Pendidikan Teknologi Informasi, Universitas Brawijaya ingin mencetak dan mendidik insan-insan yang memiliki kompetensi sebagai seorang tenaga pengajar sekaligus memiliki kompetensi sebagai seorang tenaga profesional IT dan *entrepreneur*.

2.2.1 Visi Program Studi Pendidikan Teknologi Informasi Universitas Brawijaya

Menjadi program studi unggul dalam pendidikan dan pengembangan ilmu pengetahuan di bidang Pendidikan Teknologi Informasi di tingkat nasional dan internasional melalui integrasi Tri Dharma Perguruan Tinggi.

2.2.2 Misi Program Studi Pendidikan Teknologi Informasi Universitas Brawijaya

1. Menyelenggarakan sistem pendidikan yang efektif, efisien, akuntabel, dan berkelanjutan dalam rangka menghasilkan lulusan sarjana Pendidikan Teknologi Informasi.
2. Menghasilkan lulusan yang berkompeten sebagai tenaga pendidik di bidang Pendidikan Teknologi Informasi, berkompeten sebagai tenaga profesional di bidang Teknologi Informasi, dan berjiwa wirausaha (entrepreneur), serta dapat berperan positif di tingkat nasional dan internasional (world class).
3. Meningkatkan kontribusi dan kolaborasi dengan berbagai pihak dalam masyarakat dengan mengembangkan produk dan layanan dalam bidang Pendidikan Teknologi Informasi di tingkat regional, nasional, maupun internasional.

2.2.3 Tujuan Program Studi Pendidikan Teknologi Informasi Universitas Brawijaya

1. Menghasilkan lulusan yang berkompeten sebagai tenaga pendidik di bidang Pendidikan Teknologi Informasi, berkompeten sebagai tenaga profesional di bidang Teknologi Informasi, dan berjiwa wirausaha (entrepreneur), serta

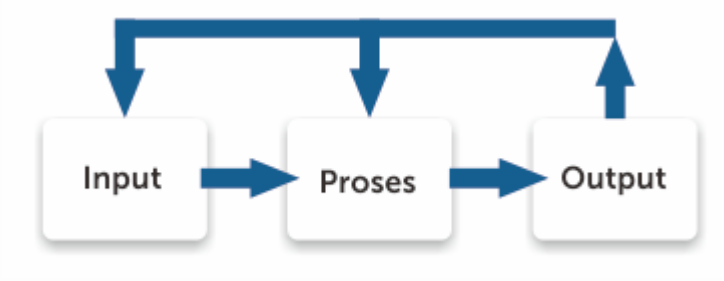
dapat dipercaya, sehingga mampu bekerjasama dan memberikan kontribusi di tingkat nasional dan internasional (world class), melalui kurikulum yang disusun dengan mempertimbangkan model kurikulum Pendidikan Teknologi Informasi pada tingkat nasional dan internasional.

2. Menjalankan sistem pendidikan dengan penjaminan mutu sesuai standar nasional dan internasional.
3. Melibatkan civitas akademika Program Studi Pendidikan Teknologi Informasi dalam penelitian yang dapat memperkaya khasanah keilmuan di bidang Pendidikan Teknologi Informasi yang berbasis pada moral dan etika dalam rangka mengisi dan menunjang pembangunan regional maupun nasional.
4. Melibatkan civitas akademika Prodi Teknologi Informasi dalam pengabdian masyarakat dalam bentuk pembinaan, bimbingan dan konsultasi dalam rangka meningkatkan peran serta masyarakat dalam pembangunan serta melakukan pemberdayaan masyarakat yang berbasis pada moral dan etika.
5. Meningkatkan kontribusi dan kolaborasi dengan berbagai pihak dalam masyarakat dengan mengembangkan produk dan layanan hasil inovasi dan kreasi dalam bidang Pendidikan Teknologi Informasi di tingkat regional, nasional maupun internasional.
6. Mengembangkan sertifikasi kompetensi di bidang Pendidikan Teknologi Informasi di tingkat regional, nasional maupun internasional.

2.3 Sistem Informasi

Sistem informasi adalah sekumpulan komponen yang berfungsi untuk mengumpulkan, menyimpan dan memproses data untuk menyajikan informasi pengetahuan serta produk digital (Zwass, 2017). Menurut Zwass, perusahaan dan organisasi mengandalkan sistem informasi untuk menjalankan dan mengelola operasional, berinteraksi dengan pelanggan dan *supplier,* serta untuk bersaing di dalam *marketplace*. Pengertian sistem informasi yang lain berdasar jurnal elektronik yang ditulis oleh Vojtech Democ, sistem informasi adalah perangkat yang penting bagi suatu perusahaan besar untuk mengelola organisasinya. Sedangkan menurut Stair dan Reynolds (2014), sistem informasi adalah sekumpulan kompenen yang saling berelasi yang mengumpulkan, memanipulasi, menyimpan dan menyebarkan data, sistem informasi nantinya kan memberikan informasi serta menyediakan umpan balik guna mencapai pada suatu tujuan.

Dalam sistem informasi tedapat sebuah alur yang harus dilewati secara urut. Alur tersebut adalah *input*, proses, *output* dan umpan balik. Setiap alur dalam sistem informasi akan mempengaruhi *output* informasi yang akan diberikan pada organisasi. Alur sistem informasi tersebut seperti yang terlihat pada Gambar 2.1,

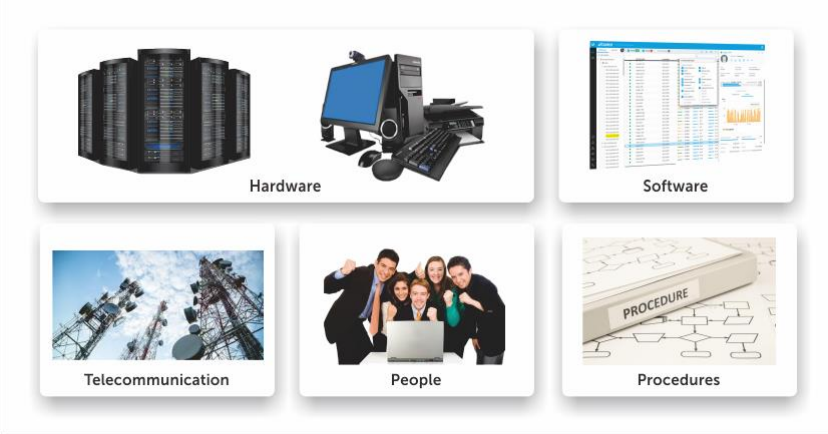


Gambar 2.1 Skema Alur Sistem Informasi

Alur tersebut memiliki tahapan-tahapan tersendiri yang didalamnya terdapat suatu aktifitas yang saling terkait, berikut uraian pada tiap tahap tersebut.

a. ***Input*** di dalam alur sistem informasi adalah aktifitas untuk mengumpulkan data mentah yang nantinya akan diolah pada tahap selanjutnya. Data mentah yang dikumpulkan harus valid, karena jika data yang dikumpulkan adalah data yang tidak valid atau data sampah maka hasil *output* juga akan bernilai sampah.
b. **Proses** adalah tahapan mengubah data mentah dan menyalurkan data ke hasil *output* yang lebih bermanfaat. Maksud dari bermanfaat adalah memiliki nilai yang dapat di manfaatkan oleh organisasi untuk mencapai tujuannya. Tahapan proses ini bisa berupa membuat perhitungan, membandingkan data, pengambilan keputusan alternative dan penyimpanan data untuk keperluan di masa mendatang.
c. ***Output*** adalah salah satu tahapan dalam skema alur sistem informasi yang tugasnya terlibat dalam menghasilkan informasi yang berharga, biasanya dalam bentuk laporan atau dokumen. *Output* juga dapat berupa laporan gaji pegawai, laporan manajer, serta laporan bagi pemegang saham, bank, dinas pemerintahan, atau kelompok lain. Namun dalam *case* lain, hasil *output* juga dapat menjadi bahan *input* bagi sistem yang lain. Contohnya adalah *output* dari sistem yang memproses permintaan penjualan dapat digunakan sebagai *input* bagi sistem *billing* pelanggan.
d. **Umpan balik** adalah tahapan dimana informasi dari sistem digunakan untuk memberikan perubahan pada *input* atau aktivitas proses yang lain. Contoh, *error* pada sistem terkadang membutuhkan perubahan data *input* ke yang benar atau membutuhkan perubahan pada proses didalamnya (Stair & Reynolds, 2014).

Sistem informasi sebenarnya bisa secara manual ataupun secara komputerisasi. Sistem informasi yang menggunakan komputer atau secara komputerisasi atau dalam bahasa Inggris disebut dengan *Computer-Based Information System* (CBIS), terdiri dari *hardware, software*, *databases,* telekomunikasi, organisasi, dan prosedur (SOP).



Gambar 2.2 Komponen Sistem Informasi Secara Komputersiasi

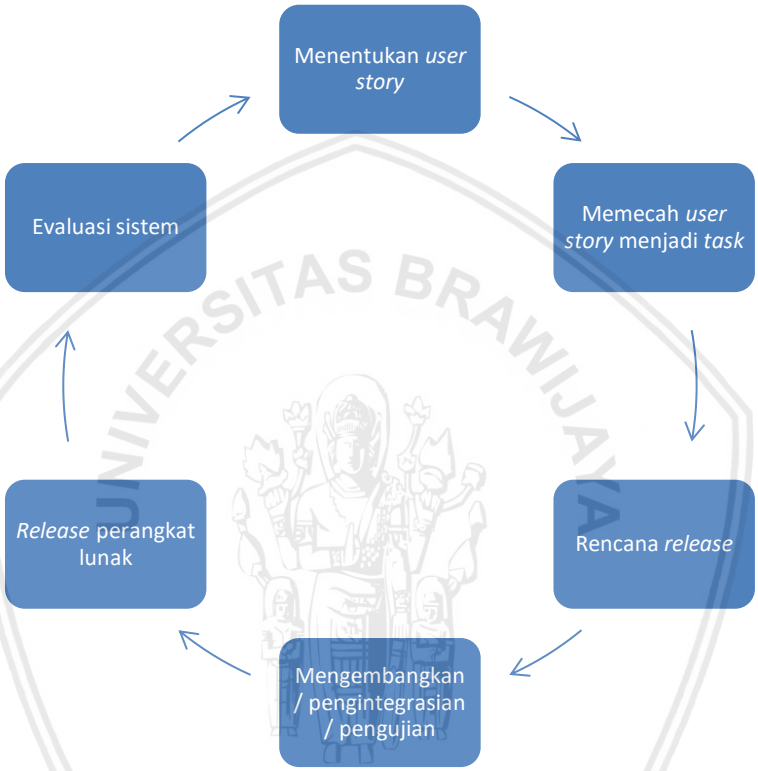
2.4 *Agile Software Development*

Agile software development adalah metode pengembangan perangkat lunak yang lebih menekankan pada proses pengembangan sistem jangka pendek dan mengulang terus menerus menyesuaikan perubahan dengan cepat (Alexandra, 2017). Metode *Agile* saat ini sedang banyak digunakan oleh banyak pengembang aplikasi dikarenakan metode ini merupakan metode jangka pendek, secepat mungkin memberikan hasil pengembangan kepada *client* dan menerima *feedback* yang nantinya digunakan untuk meningkatkan kualitas dan peforma perangkat lunak (Wells, 2009).

Keuntungan menggunakan metode *Agile* dalam mengembangkan suatu sistem atau perangkat lunak adalah tim menjadi lebih produktif, kualitas perangkat lunak menjadi meningkat, klien menjadi lebih puas, serta biaya yang dikeluarkan relatif lebih murah (Alexandra, 2017). Dalam metode pengembangan *Agile,* lebih penting interaksi antar anggota tim dari pada proses dan alat yang digunakan, lebih penting software yang berjalan sesuai kebutuhan daripada dokumentasi yang lengkap, lebih pentig kolaborasi dengan *client* daripada membahas mengenai negosiasi kontrak, serta lebih penting segera tanggap dengan perubahan daripada mengikuti rencana. Jadi intinya, metode *Agile* sangatlah membantu para pengembang sistem atau perangkat lunak dalam melakukan penyerahan produk secara tepat waktu dan tepat guna berdasar tahapan analisa dan desain. Dalam metode pengembangan *Agile* terdapat beberapa model tersebut antara lain *Acceptence Test Driven Development (ATDD), Agile Modeling, Adaptive Software Development (ASD), Agile Unified Process (AUP), Continuous integration (CI), Crystal Clear, Crystal Methods, Dynamic Systems Development Method (DSDM), Extreme Programming (XP), Feature Driven Development (FDD), Graphical System Design (GSD), Kanban, Lean software development, Rational Unified Process (RUP), Scrum, Scrum-ban, Story-driven modeling,Test-driven development (TDD), Velocity tracking, dan Software Development Rhythms.*

2.5 *Extreme Programming (XP)*

Extreme programming merupakan salah satu bagian dari Agile Software Development. Metode ini diperkenalkan oleh Beck pada tahun 2000. Metode ini adalah hasil dari pengembangan cara praktis pengembangan perangkat lunak yang telah dikenal, seperti metode *iterative*, menjadi level "ekstrim" (Sommerville, 2011). Contohnya adalah didalam XP, beberapa versi baru pada sistem bisa jadi dikembangkan oleh programmer yang berbeda, diintegrasikan dan diuji dalam satu hari.



Gambar 2.3 Siklus *Release Extreme Programming*

Di dalam *extreme programming,* kebutuhan sistem di wujudkan dalam bentuk skenario atau bisa disebut *user story,* yang diimplementaskan langsung menjadi serangkaian *task* atau list fitur yang diperlukan. Semua rangkaian tes harus berhasil ketika baris kode baru diintegrasikan ke dalam sistem. Dalam metode ini hanya ada jarak yang singkat sampai produk di-*release* kembali dengan fitur yang baru. *Extreme programming* melibatkan *client* atau calon pengguna untuk terlibat terus dalam proses pengembangan aplikasi. Sehingga *client* dapat memantau perkembangan atau mungkin memunculkan kebutuhan baru untuk meningkatkan kualitas aplikasi. Dengan menggunakan metode *extreme programming* perawatan sistem menjadi mudah karena setiap perulangan atau iterasi dilakukan analisis dan dokumentasi kebutuhan baru.

Dalam pembuatan *user story,* pengembang melibatkan *client* dalam menentukannya (Lucidchart Content Team, 2018). Setelah *user story* ditentukan, selanjutnya tim pengembang akan memberikan penilaian (*scoring*) terhadap *user*

story. Nilai *user story* yang paling besar dan beresiko tinggi akan diimplementasikan terlebih dahulu. Nantinya, hasil implementasi *user story* yang pertama akan dihitung kecepatan pengembangan *project* untuk dijadikan acuan seberapa cepat *project* akan dikerjakan. Hasil akhir dari pembuatan *user story* ini akan dipecah menjadi *task–task* kecil sehingga mirip dengan daftar kebutuhan fungsionalitas sistem. Format standar yang digunakan untuk membuat *user story* pada umumnya dan contoh *user story* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Format dan Contoh *User Story*

No.	***User Story***
1.	Format : Sebagai seorang [*calon pengguna*], saya menginginkan [*tujuan pengguna*] sehingga saya dapat [*alasan*]
2	Sebagai seorang Pelatih, saya ingin halaman *profile* saya memuat data diri saya secara lengkap. Sehingga saya bisa memperlihatkan pengalaman-pengalaman saya.
3	Sebagai seorang pengunjung *website,* saya dapat membaca berita terkini pada halaman utama web sehingga saya selalu mengetahui berita terbaru secara singkat dan cepat.

Sumber : Mountain Goat Software (2018)

Tahap selanjutnya yaitu tahap pemecahan *user story* kedalam *task-task* kecil. Pada tahap ini dilakukan analisis terhadap *user story* yang telah dibuat untuk mendapatkan poin-poin utama yang harus dipenuhi oleh sistem. Hasil akhir pada tahapan ini adalah daftar kebutuhan fungsionalitas dan kebutuhan non fungsionalitas sistem. Selain itu, pada tahap ini juga mulai terlihat gambaran umum sistem yang akan dibuat.

Selanjutnya pada tahap rencana *release,* berisi mengenai rencangan sistem nanti yang akan dibuat. *Extreme Programming* cenderung digunakan pengembang sistem untuk mengembangkan sistem berbasis objek (Michael, 2015). Pada tahap rencana *release* ini akan dibuatkan CRC *card* (*class-responsibility-collaboration*). CRC *card* ini memungkinkan semua anggota *project* tim untuk merancang sistem dan melihat bagaimana setiap objek saling berinteraksi. Tahapan ini mirip dengan perancangan *class diagram* pada pengembangan sistem pada umumnya. Gambar 2.4 merupakan *template standart* dan contoh dari penggunaan CRC *card.*

Pada tahap implementasi, pengembang akan mengumpulkan pembuat kode program. Tujuannya adalah untuk meninjau ulang kode program dan semua tim pengembang boleh menambahkan fungsionalitas, membenahi *bug,* atau melakukan refaktor kode. Pada proses implementasi, setiap tim pengembang harus melakukan pemilihan *system metaphor* atau melakukan standarisasi skema penamaan. Lalu pengembang juga harus melakukan *pair-programming.* Dua orang pengembang akan bekerja sama pada satu komputer, lalu menuliskan kode program disana dan mengirimkannya ke level *production.* Selain itu tim

pengembang juga harus mengintegrasikan *code* dan melakukan *commit* ke *repository* setiap beberapa jam sekali.

Gambar 2.4 *Template Standart* dan Contoh CRC *Card*

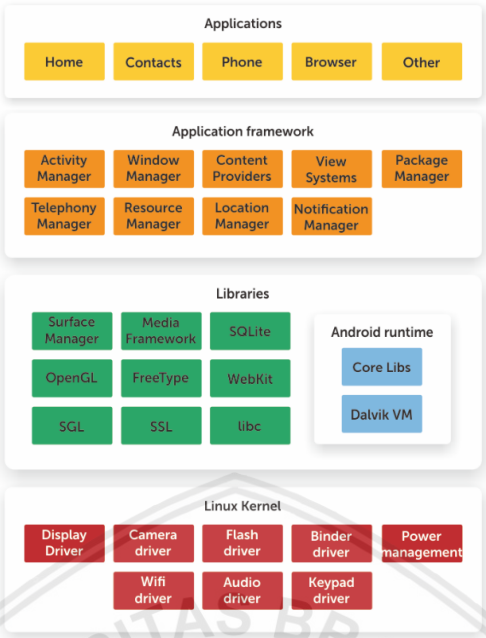
Sumber : Lucidchart Content Team (2018)

Selanjutnya pada tahap *release* sistem dan evaluasi, didalamnya adalah proses pengujian sistem di lingkungan *client.* Pada tahap ini biasanya juga dikenal sebagai *review* sistem. *Unit test* yang telah dibuat harus diimplementasikan menggunakan suatu *framework* dan diatur dalam universal testing suite. Selain itu, *unit test* juga harus diintegrasikan dan divalidasi setiap hari. *Customer test* atau *acceptence test* dilakukan oleh *client* dan fokus pada keseluruhan fitur dan fungsionalitas sistem. *Acceptence test* diperoleh dari *client stories* yang telah diimplementasikan sebagai bagian dari *software release.*

2.5.1 Android

Android adalah *platform* atau sistem operasi yang bersifat *open source* khusus bagi perangkat bergerak (*mobile*) (Gargenta & Nakamura, 2014). Android pada awalnya didirikan oleh perusahaan bernama Android.inc, lalu Google datang dan mengakuisisinya. Saat ini Android merupakan sistem operasi perangkat bergerak yang paling banyak digunakan di dunia, terhitung sudah jutaan *device* dipasangi sistem operasi ini. Salah satu penyebab android begitu disukai adalah karena android bersifat *open source,* hampir semua *stack*-nya dibuka untuk dikembangkan oleh siapapun. *Stack* yang dibuka meliputi *low-level native,* mesin virtual Dalvik, *framework* aplikasi, serta aplikasi *standart* bawaan.

Struktur sistem operasi android hampir mirip dengan kue, yang terdiri dari beberapa lapisan. Masing-masing lapisan memiliki karakteristik dan fungsi sendiri-sendiri. Namun, lapisan-lapisan tersebut tidak semuanya benar-benar terspisah satu sama lain. Melainkan ada yang tergabung kedalam lapisan yang lainnya. Gambar 2.3 menunjukkan lapisan-lapisan yang ada di dalam sistem operasi android.



Gambar 2.5 Lapisan Sistem Operasi Android

Sumber : Gargenta & Nakamura (2014)

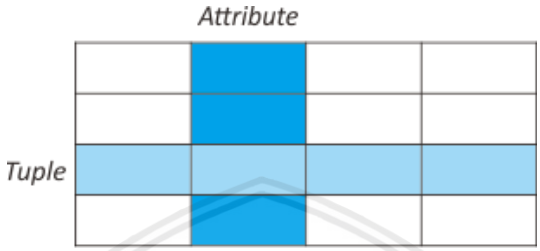
Android dibuat diatas Linux kernel. Sedangkan Linux adalah sistem operasi *open source* terbesar yang saat ini ada. Kernel linux sendiri terus dikembangkan dan diperkuat terus menerus sepanjang tahun oleh para *engineer* untuk meningkatkan performanya. Alasan android memilih lingkungan Linux sebagai dasarnya adalah karena portabilitasnya, keamanan dan fitur lengkap yang dimiliki Linux.

2.5.2 Web Service

Web service adalah sistem perangkat lunak yang dirancang untuk mendukung skema antar mesin yang dapat saling berkomunikasi melalui jaringan (Booth, 2004). *Web service* bisa disebut juga *messaging framework,* alasannya karena skema yang diberlakukan di *web service* adalah pertama *client* mengirimkan *request* ke *server*, lalu server mengirimkan kembali *respon* sesuai dengan *request* yang diminta *client.* Menurut artikel yang terdapat pada situs Tutorials Point, *web service* adalah sekumpulan *protocols* yang digunakan untuk bertukar data antar aplikasi atau antar sistem (tutorialspoint, 2018). *Web service* memiliki beberapa komponen standar yang umum digunakan, antara lain, SOAP (*Simple Object Access Protocol*), UDDI (*Universal Description, Discovery and Integration*) dan WSDL (*Web Service Description Language*). *Web service* memungkinkan untuk berkomunikasi antar aplikasi menggunakan format seperti JSON, XML, WSDL dan SOAP.

2.5.3 Relational Database

Database relational merupakan sekumpulan tabel yang saling terhubung satu sama lain dengan menggunakan *primary key* (Andre, 2017). Tabel dalam *database relational* juga disebut sebagai relasi. Selayaknya tabel pada umumnya, tabel dalam *database relational* juga memiliki kolom dan baris. Namun, kolom dan baris biasa disebut sebagai *attribute* (kolom) dan *tuple* (baris). Gambar 2.4 merupakan contoh tabel pada *relational database.*



Gambar 2.6 Tabel Dalam *Relational Database*

Sumber : Andre (2017)

Setiap baris didalam suatu tabel setidaknya harus memiliki satu kolom yang bersifat unik. Fungsinya adalah untuk menjadi identitas pada suatu baris. Sebagai contoh, pada tabel mahasiswa kolom NIM merupakan kandidat yang bagus untuk menjadi identitas mahasiswa. Hal ini dikarenakan tidak mungkin ada mahasiswa yang memiliki NIM yang sama dengan mahasiswa yang lain. Kolom unik yang berfungsi sebagai identitas baris dalam suatu tabel ini disebut juga sebagai *Primary Key.* Ciri-ciri dari *primary key* tidak boleh berulang dan tidak boleh kolom tersebut dikosongkan atau *null.* Dalam *relational database* juga terdapat istilah *foreign key, referential integrity* dan *index.*

2.5.4 *Use Case Scenario*

Use case scenario merupakan penjelasan secara rinci dari *use case diagram.* Menurut Bittner et.al, *use case diagram* adalah cara yang mudah untuk menjelaskan kebutuhan fungsionalitas sistem dan perilaku sistem (Bittner & Spence, 2003). *Use case scenario* juga merupakan urutan kegiatan yang harus dilakukan oleh aktor dan sistem untuk mencapai suatu tujuan tertentu. *Use case scenario* terdiri dari *use case name*, aktor, deskripsi singkat, aliran normal, aliran alternatif, *Pre-Condition* dan *Post-Condition* (Pressman, 2010). Secara umum format untuk menuliskan *use case scenario* dapat dilihat pada Tabel 2.2.

Tabel 2.2 Format *Use Case Scenario*

Use Case Name	
Objective	Deskripsi singkat
Aktor	Aktor yang menjalankan sistem
Pre-condition	Kondisi awal sebelum memulai langkah kerja
Main Flow	Alur utama yang harus dijalankan

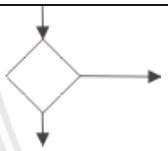
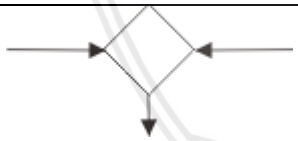
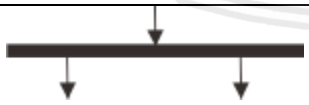
Alternative Flow	Ketika alur utama mengalami percabangan
Post-Condition	Kondisi sistem dan aktor setelah menjalankan aliran utama

Sumber : Pressman (2010)

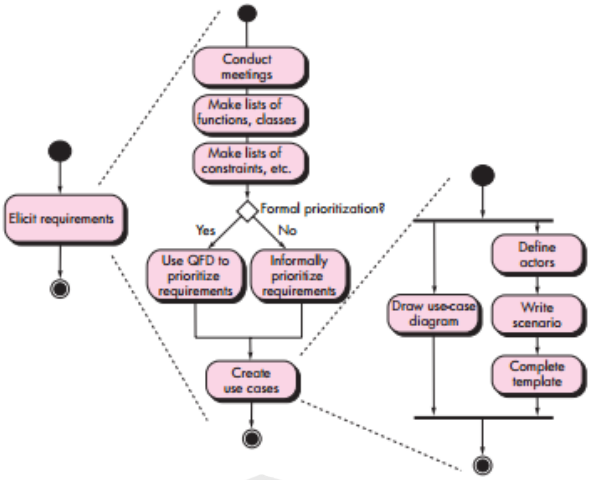
2.5.5 Activity Diagram

Activity Diagram atau dalam bahasa Indonesia diagram aktivitas merupakan diagram yang termasuk kedalam UML (*Unified Modelling Language)* dari sistem. Menurut Pressman, *Activity Diagarm* merupakan gambaran alur grafis berupa aliran dari interaksi yang berdasarkan suatu spesifik scenario (Pressman, 2010). Notasi-notasi yang digunakan pada diagram *activity* dapat dilihat pada Tabel 2.3 dan contoh penggunaan *activity diagram* dapat dilihat pada Gambar 2.7.

Tabel 2.3 Notasi Diagram *Activity*

No.	Simbol	Deskripsi
1.		*Initial Mode,* merupkan titik mulai suatu proses pada diagram aktifitas.
2.		*Actions,* merupakan notasi yang menggambarkan suatu langkah pada proses.
3.		*Flow,* notasi ini menggambarkan alur antar *action.*
4.		*Decision,* merupakan kondisi tertentu yang menentukan alur percabangan
5.		*Merge,* merupakan penggabungan alur dari dua alur yang berbeda menjadi satu
6.		*Fork,* merupakan bar hitam untuk percabangan aktivitas yang secara bersamaan.
7.		*Join*, untuk menyatukan kembali alur yang terpisah oleh *fork*
8.		*Activity Final,* merupakan titik akhir proses.

Sumber : Pressman (2010)

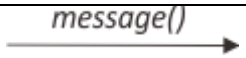
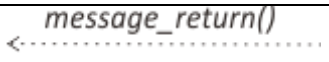


Gambar 2.7 Contoh Penggunaan Diagram *Activity*

2.5.6 Sequence Diagram

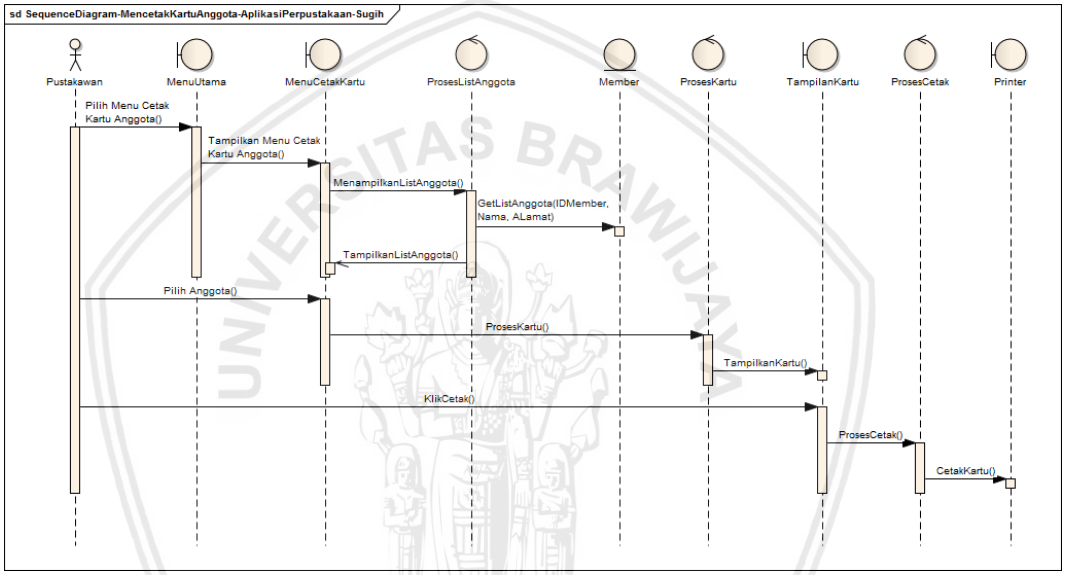
Menurut Rumbaugh, Jacobson & Booch (2005), *sequence diagram* digunakan untuk menggambarkan *dynamic view* dari suatu sistem. *Diagram Sequence* adalah diagram yang menunjukkan interaksi sistem dan menekankan pada permintaan pesan yang menunjukkan serangkaian peranan dan pesan yang dikirim dan diterima oleh bagian yang ada pada suatu peranan tersebut. Notasi pesan untuk panggilan terhadap suatu operasi dapat dijelaskan dalam bahasa UML atau dalam bahasa pemrogaman tertentu. Notasi yang terdapat pada diagram *Sequence* dapat dilihat pada Tabel 2.4 dan contoh penggunaannya dapat dilihat pada Gambar 2.8.

Tabel 2.4 Notasi Diagram *Sequence*

No.	Simbol	Deskripsi
1.		*Entity,* merupakan suatu entitas yang berfungsi untuk menyimpan data sistem.
2.		*Boundary,* merupakan *view* atau tampilan antar muka untuk menghubungkan pengguna dengan sistem.
3.		*Controller,* merupakan *objek* dalam sistem yang melakukan aktivitas fungsional
4.		*Message,* mendifinisikan alur pengiriman pesan.
5.		*Return Values,* merupakan definisi dari alur pengembalian data dari sistem

No.	Simbol	Deskripsi
6.		*Lifelines,* merupakan urutan dari garis hidup suatu komponen
7.		*Bar,* menunjukkan periode waktu ketika aktif kedalam sistem.

Sumber : Rumbaugh, Jacobson & Booch (2005)



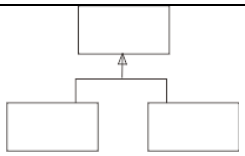
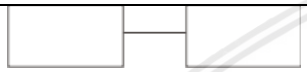
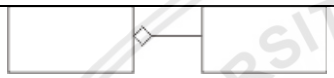
Gambar 2.8 Contoh Penggunaan Diagram *Sequence*

Sumber : Hartono (2012)

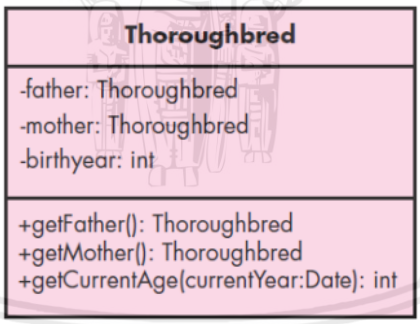
2.5.7 Class Diagram

Class Diagram merupakan suatu diagram yang memuat tiga elemen utama, yaitu elemen *class, interface* dan hubungan dependensi, generalisasi, serta asosiasi (Rumbaugh, Jacobson & Booch, 2005). Fungsi dari penggunaan *class diagram* adalah untuk menggambarkan tampilan statis dari sebuah sistem. Selain itu penggunaan diagram ini adalah untuk mendokumentasikan struktur sistem. Tabel 2.5 merupakan notasi dalam diagram *class.* Gambar 2.9 merupakan contoh penggunaan diagram *class*.

Tabel 2.5 Notasi Diagram *Class*

No.	Simbol	Deskripsi
1.		*Attribute,* menjelaskan mengenai sekumpulan data yang dimiliki oleh suatu objek *Behavior,* adalah sekumpulan perilaku yang dapat dilakukan suatu objek
2.		*Inheritance,* adalah garis yang melambangkan sutau kelas merupakan turuan dari kelas lain
3.		*Association,* merupakan hubungan antar kelas satu dengan yang lain.
4.		*Aggregation,* adalah hubungan antar kelas yang melambangkan kepemilikan
5.		*Composition* adalah hubungan antar kelas yang melambangkan bagian dari

Sumber : Rumbaugh, Jacobson & Booch (2005)



Gambar 2.9 Contoh Penggunaan Diagram *Class*

Sumber : Pressman (2010)

2.5.8 *Black Box Testing*

Pengujian menggunakan metode *black-box testing* merupakan pengujian yang dilakukan untuk mengetahui kesalahan / error yang terdapat pada perangkat lunak. *Error* yang dimaksud yaitu *error* hilangnya suatu fungsi, *error* antarmuka, *error* pada basisdata, *error* pada kinerja, serta *error* pada inisialisasi dan penghentian sistem (Pressman, 2010). Keunggulan yang dimiliki oleh metode ini adalah sebagai berikut,

1. Pengujian dilakukan berdasarkan *client angel* serta membantu dalam menemukan ketidaksesuaian dalam spesifikasi perangkat lunak.

2. Penguji tidak perlu mengetahui bagaiaman sistem diimplementasikan dan bahasa apa yang digunakan.
3. Pengujian dapat dilakukan oleh anggota dari pengembang yang bersifat objektif.
4. Kasus uji langsung dirancang setelah daftar spesifikasi sistem lengkap.

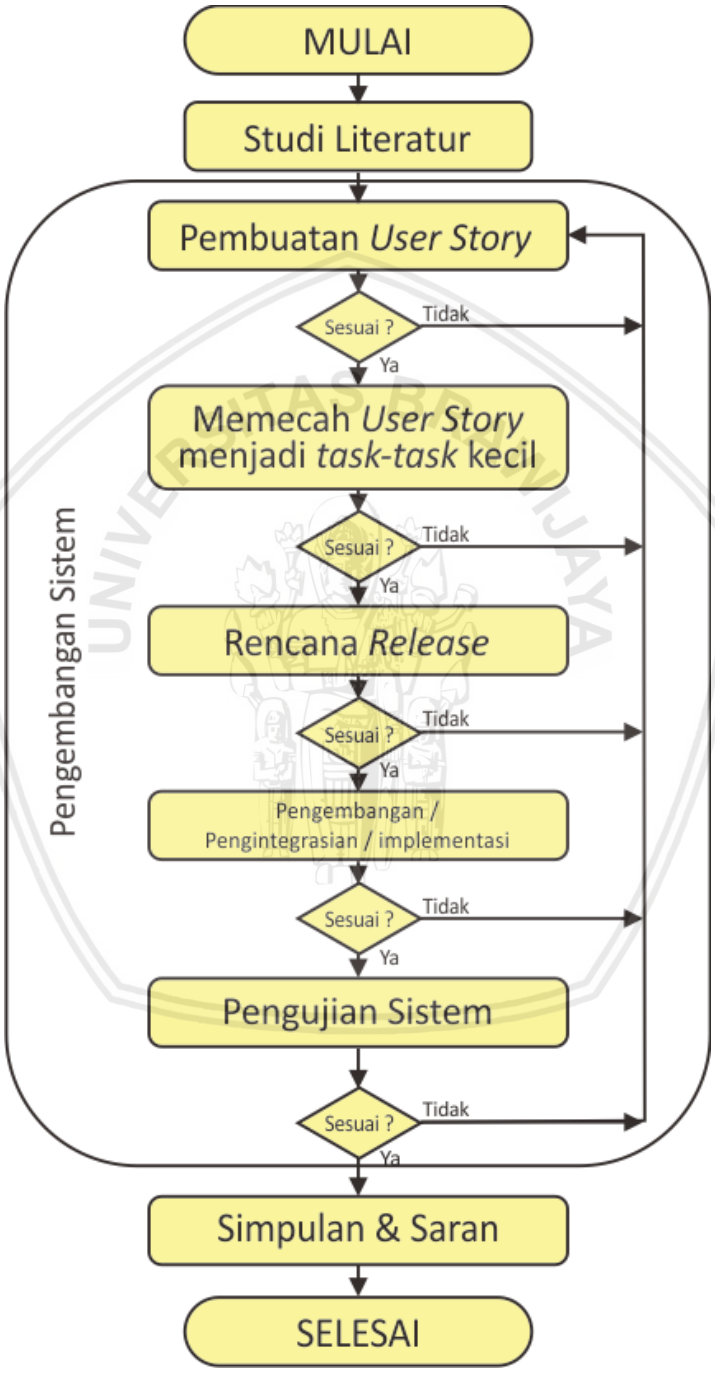
2.5.9 Compatibility Testing

Compatibility testing merupakan pengujian yang fokus pada kebutuhan non-fungsionalitas sistem guna untuk meningkatkan kepuasan penguna terhadap sistem (*Software Testing Help*, 2018). Pengujian ini akan mencoba menjalankan sistem yang telah dibuat pada lingkungan yang berbeda-beda seperti *browser* yang berbeda, *database* yang berbeda, *hardware* yang berbeda, sistem operasi yang berbeda, perangkat bergerak yang berbeda serta jaringan yang berbeda pula. Ketika diuji, seharusnya sistem tidak mengalami perubahan fungsi dan mampu berjalan seperti seharusnya pada berbagai perbedaan lingkungan operasional. Terdapat 2 jenis pada *compatibility testing* ini, yaitu,

1. *Backward Compatibility Testing,* pengujian sistem dengan kondisi lingkungan operasional dalam keadaan versi terdahulu.
2. *Forward Compatibility Testing,* pengujian sistem dengan kondisi lingkungan operasional dalam keadaan versi yang paling terbaru.

BAB 3 METODOLOGI

Berdasarkan latar belakang yang peneliti angkat, dapat dihasilkan kerangka kerja penelitian yang akan dilakukan sesuai dengan metode, yaitu :



Gambar 3.1 Diagram Alur Metode Penelitian

3.1 Studi Literatur

Tahapan studi literatur adalah tahapan mengumpulkan, mempelajari dan mencari hal-hal yang bisa diadaptasi dari jurnal / paper, buku serta penelitian yang terdahulu. Hasil dari studi literatur juga berguna untuk memperkuat dasar teori peneliitian pembangunan sistem. Dalam penelitian ini akan dijelaskan mengenai teori sistem informasi, metode pengembangan sistem *Extreme Programming, Android Platform*, dan *Web Service.*

3.2 Pembuatan *User Story*

Pembuatan *user story* dimulai dengan melakukan wawancara dengan *stakeholder.* Proses wawancara adalah proses dimana mencari informasi dan data yang ada pada organisasi yang dapat dimanfaatkan untuk menentukan rancangan sistem yang akan dibangun nantinya. Pada tahapan ini, bertujuan untuk mengidentifikasi bagaimana alur dari proses kegiatan Praktik Pengalaman Lapangan (PPL) nanti dengan menggunakan sistem. Penentuan dilakukan berdasar hasil analisis alur yang saat ini sedang mulai berlangsung. Selain itu, pada tahap ini juga dilakukan identifikasi siapa saja *stakeholder* yang terlibat di dalam sistem nantinya; dan menjelaskan bagaimana rancangan proses bisnis yang terjadi saat ini. Informasi yang ingin diperoleh dari wawancara ini adalah alur dari proses kegiatan PPL. Selain itu, tujuan dilakukan wawancara adalah untuk mendapatkan informasi sistem seperti apa yang nanti sesuai dengan tahapan kegiatan PPL.

Untuk dapat sampai pada tujuan tersebut, dilakukan wawancara pengambilan informasi dan data menggunakan metode wawancara bersama *stakeholder* terkait, yaitu Ketua Program Studi Pendidikan Teknologi Informasi Fakultas Ilmu Komputer Universitas Brawijaya, beberapa mahasiswa pada prodi yang sama, serta guru pamong yang berasal dari sekolah mitra. Metode wawancara ini dilakukan karena kegiatan PPL ini masih akan dilakukan pertama kali pada tahun ini, sehingga kurang cocok melakukan pengumpulan kebutuhan menggunakan metode pengamatan. Hasil dari proses wawancara ini adalah bisa memperoleh informasi dan data mengenai seperti apa alur kegiatan PPL yang saat ini sedang berlangsung pada tahap pendaftaran, serta seperti apa sistem yang sesuai nantinya dengan alur tersebut. Hasil akhir dari wawancara ini nanti akan digunakan sebagai dasar pembuatan *user story,* daftar kebutuhan sistem, fungsionalitas dan alur sistem yang akan dikembangkan.

3.3 Memecah *User Story* Menjadi *Task-Task* Kecil

Proses ini dilakukan setelah data hasil wawancara didapatkan dan *user story* telah berhasil dibuat, sehingga proses pemecahan *user story* kedalam *task-task* kecil dapat dilakukan. Pada tahap ini mirip dengan membuat daftar kebutuhan fungsionalitas dan non fungsionalitas sistem yang harus dipenuhi oleh sistem nantinya. Data dan informasi dari hasil wawancara dikumpulkan lalu dilakukan pengambilan informasi-informasi yang penting mengenai alur kegiatan PPL, apa saja fungsionalitas yang memungkinkan untuk dimasukkan kedalam sistem, dan apa saja *input* dan *output* yang diperlukan selama kegiatan PPL berlangsung.

Informasi yang didapatkan lalu digambarkan kedalam bentuk bagan dan diagram sehingga mudah untuk dipahami.

Analisis sistem lalu dilakukan berdasar pada informasi yang telah didapatkan dari tahapan wawancara. Dimulai dengan membuat diagram alur proses bisnis kegiatan PPL yang belum menggunakan sistem informasi. Lalu juga membuat diagram alur proses bisnis kegiatan PPL menggunakan sistem informasi menggunakan diagram BPMN. Pada tahapan ini juga dilakukan identifikasi siapa saja aktor yang terlibat dalam sistem nantinya.

3.4 Rencana *Release*

Tahap rencana *release* sistem dapat dilakukan setelah mendapatkan gambaran bagaimana alur proses bisnis dari sistem informasi yang akan dikembangkan. Pada tahap ini akan dilakukan identifikasi mengenai apa saja yang akan dibuat dalam perancangan sistem. Pada tahapan peracangan sistem ini juga akan didapatkan kebutuhan fungsional dan kebutuhan non-fungsional, yang kemudian dilakukan perancangan sistem. Dalam merancangkan sistem, akan digambarkan beberapa diagram untuk memperjelas rancangan, diagram tersebut antara lain, *use case diagram, activity diagram, sequence diagram, class diagram,* dan *physical data model diagram (PDM)*. Seleuruh diagram tersebut akan digambar berdasarkan kebutuhan fungsional dan kebutuhan non fungsional yang telah dianalisis terlebih dahulu. Tahapan berikutnya adalah melakukan perancangan *user interface (UI)* atau tampilan muka dari sistem informasi yang dibuat. Didalam perancangan UI akan menentukan tata letak dari setiap komponen yang ada di tampilan seperti tombol, kotak masukan, dll sebaik dan sesuai mungkin untuk diterapkan didalam sistem.

3.5 Pengembangan / Pengintegrasian / Implementasi Sistem

Pada tahapan pengembangan / pengintegrasian / implementasi sistem, dilakukan pengerjaan (*coding)* terhadap sistem yang telah dirancang. Setiap fungsi yang telah didefinisikan didalam diagram *use case* diimplementasikan kedalam sistem. Diagram yang lain seperti *class diagram* dan *physical data model diagram (PDM)* juga diimplementasikan dalam tahapan ini. Implementasi tampilan sistem juga dilakukan disini berdasarkan rancangan UI yang telah dibuat dengan menyambungkan antara komponen tampilan dan fungsi, contoh seperti menyambungkan komponene tombol dengan fungsi yang menangani kejadian ketika tombol ditekan.

3.6 Pengujian Sistem

Pada tahap ini dilakukan pengujian terhadap sistem yang telah dibuat. Pengujian dilakukan dengan melibatkan calon pengguna untuk menguji tingkat keberhasilan dari sistem apakah telah sesuai dengan rancangan di awal tahapan yang telah dibuat, dan menguji apakah sistem telah memenuhi kebutuhan dan ekspektasi *stakeholder* atau belum. Pengujian yang dilakukan menggunakan dua macam jenis teknik pengujian, yaitu *black box testing* dan *compatibility testing*.

Black box testing adalah pengujian sistem yang menguji fungsionalitas sistem ketika digunakan oleh pengguna dan metode ini juga bertujuan untuk menguji apakah telah memenuhi semua daftar kebutuhan sistem. Sedangkan pengujian menggunakan *Compatibility Testing* bertujuan untuk menguji apakah sistem dapat berjalan atau kompatibel di lingkungan sistem operasi yang bervariasi.

3.7 Kesimpulan dan Saran

Pada tahap yang terakhir setelah sistem selesai dibuat dan hasil pengujian menunjukkan keberhasilan sistem, dilakukan penarikan kesimpulan dan saran untuk mempermudah pengembangan sistem selanjutnya. Di dalam kesimpulan berisi kesimpulan hasil pada setiap tahapan pengembangan. Sedangkan saran berisi apa saja yang perlu dilakukan ketika peneliti lain ingin melanjutkan penelitian ini.

BAB 4 HASIL DAN PEMBAHASAN

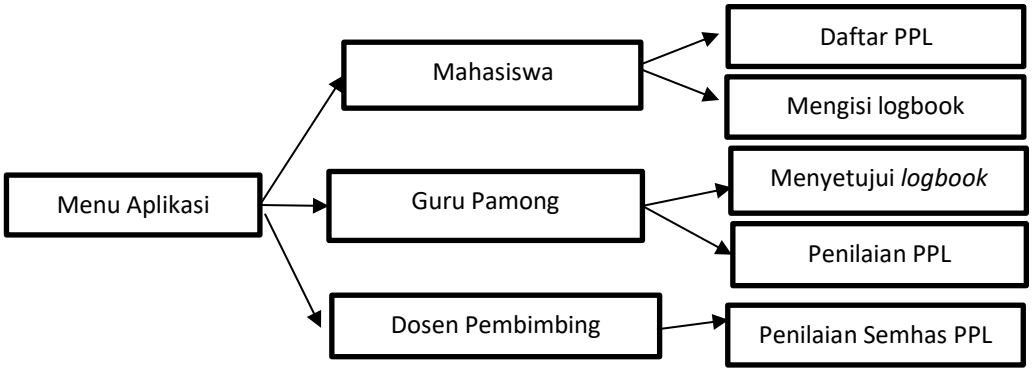
Pada bagian ini menjelaskan mengenai pendekatan penulis dalam menyelesaikan permasalahan penelitian. Pada Gambar 4.1 menunjukkan alur pengembangan sistem menggunakan metode *extreme programming* yang dialami penulis. Dalam prosesnya, penulis mengalami iterasi pada bagian pembuatan *use case story* 1 kali iterasi, pada bagian rencana *release* 2 kali iterasi dan pada bagian implementasi 1 kali iterasi.

Gambar 4.1 Pengembangan Sistem Menggunakan *Extreme Programming*

4.1 Iterasi Tahap Pembuatan *User Story*

4.1.1 Iterasi 1 Gambaran Umum Sistem

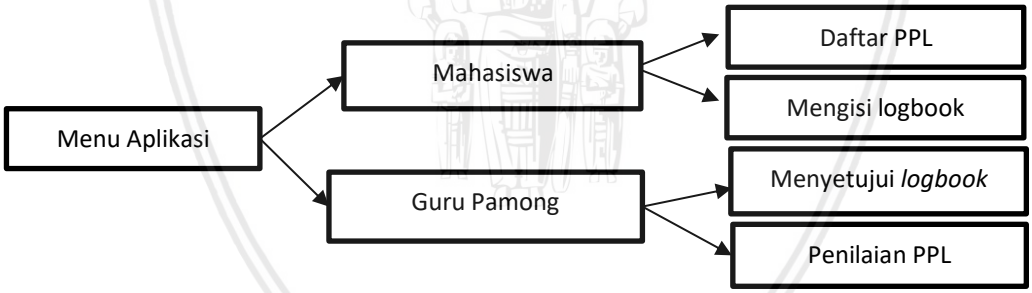
Iterasi pertama terjadi pada tahap pembuatan *user story.* Pada tahap ini pertama dilakukan wawancara dengan *stakeholder* untuk mendapatkan informasi mengenai kegiatan PPL dan analisis dokumen-dokumen pendukung kegiatan PPL. Hasil dari pembuatan *user story* digambarkan menjadi gambaran umum sistem yang dapat dilihat pada Gambar 4.2. Pada iterasi pertama ditemukan 3 aktor dan 5 fungsi utama. Aktor terdiri dari mahasiswa, guru pamong dan dosen pembimbing PPL. Sedangkan fitur untuk mahasiswa adalah daftar PPL dan mengisi logbook. Fitur untuk guru pamong adalah menyetujui logbook dan peniliaian PPL. Sedangkan fitur untuk dosen pembimbing adalah penilaian seminar hasil PPL.



Gambar 4.2 Gambaran Umum Sistem Iterasi 1

4.1.2 Iterasi 2 Gambaran Umum Sistem

Selanjutnya dilakukan iterasi kedua, dilakukan pengecekan kesesuaian hasil analisis dengan *stakeholder.* Hasilnya adalah aktor dosen pembimbing dihilangkan dan fitur penilaian semhas PPL juga dihilangkan. Alasan penghilangan aktor ini adalah untuk mempersingkat waktu *deliver* sistem kepada *stakeholder.* Sehingga saat ini hanya terdapat 2 aktor dan 4 fungsi utama sistem. Sistem yang akan dibangun nanti akan mencakup proses pendaftaran PPL, pencatatan *logbook* dan penilaian PPL mahasiswa oleh guru pamong. Selama masa pelaksanaan PPL di sekolah mitra, mahasiswa dapat memasukkan *logbook* harian kedalam sistem yang nanti dapat direkapitulasi di akhir kegiatan. Hasil Iterasi dapat dilihat pada Gambar 4.3.

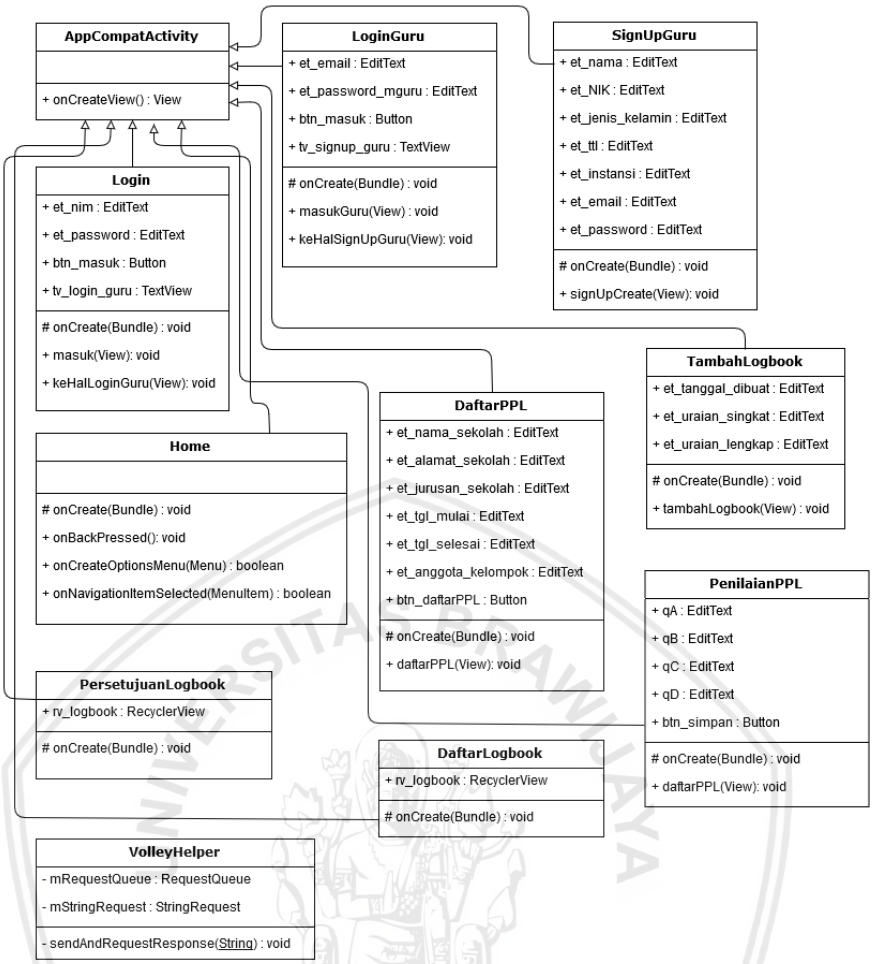


Gambar 4.3 Gambaran Umum Sistem Iterasi 2

4.2 Iterasi Tahap Rencana *Release*

4.2.1 Iterasi 1 Class Diagram

Pada iterasi pertama, *class diagram* yang dapat dilihat pada Gambar 4.4 memiliki 11 *class.* Semua *class* yang terdapat pada Gambar merupakan *class activity* yang digunakan didalam sistem nantinya. Setipa *class* merupakan turunan dari *class AppCompatActivity*. *Class AppCompatActivity* merupakan *class* yang membuat suatu halaman pada *smartphone* Android.



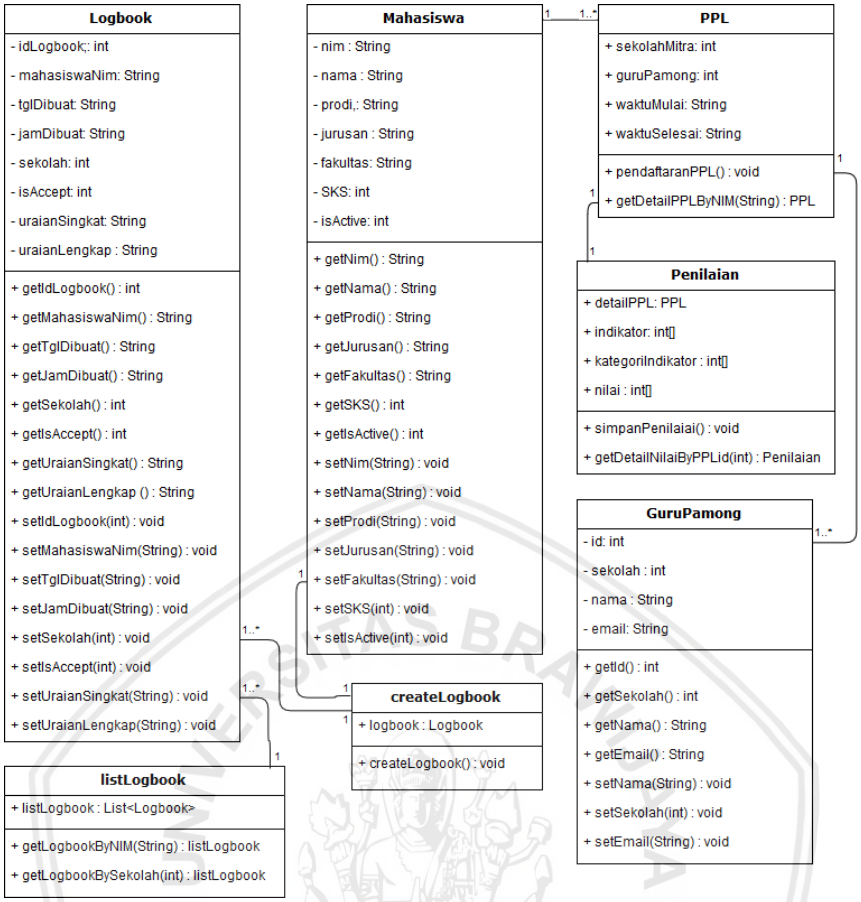
Gambar 4.4 *Class Diagram* Sistem Informasi PPL Iterasi 1

4.2.2 Iterasi 2 Class Diagram

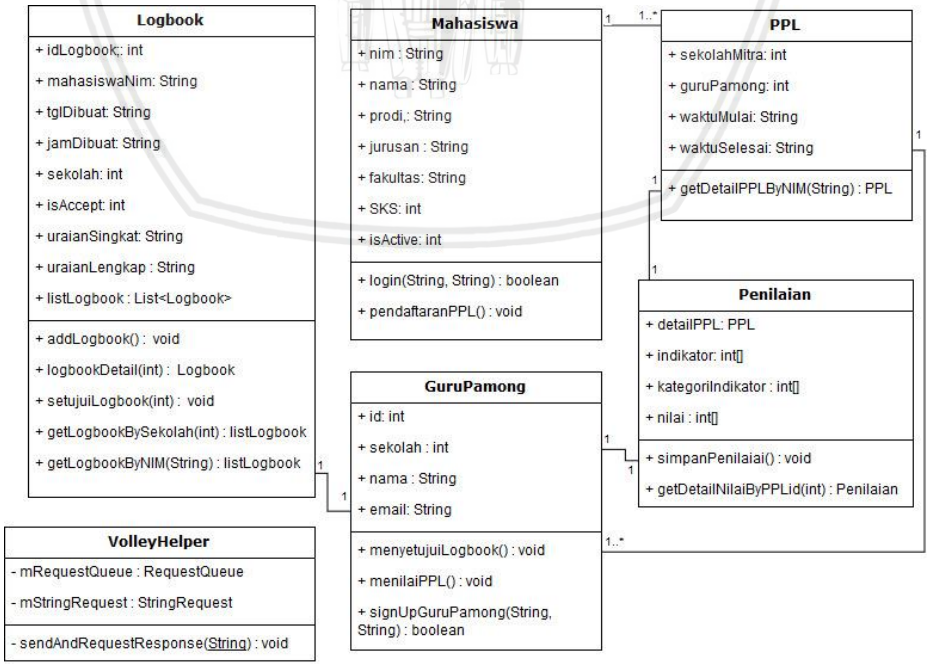
Pada tahap iterasi kedua *class diagram,* terjadi perubahan mendasar pada rancangan *class diagram.* Gambar 4.5 merepresentasikan semua *class* yang digunakan pada sistem. Perubahan terjadi karena pada rancangan *class diagram* yang pertama mengacu pada halaman / *activity* pada sistem seharusnya rancangan mengacu pada objek yang ada pada sistem. Terdapat *class* Logbook yang berfungsi sebagai model pada sistem. Data *logbook* yang diperoleh dari *response* API berupa JSON akan dimasukkan kedalam *ListArray* dengan tipe objek *class* Logbook. Salah satu penggunaan *class* ini adalah untuk menampilkan daftar logbook pada recyclerView pada menu Logbook.

4.2.3 Iterasi 3 Class Diagram

Pada tahap iterasi ketiga class diagram, terjadi perubahan pada rancangan *class diagram*. Perubahan disebabkan karena rancangan masih belum merepresentasikan objek didalam sistem. Gambar 4.6 menggambarkan semua *class* yang digunakan pada sistem setelah mengalami perubahan. *Class-class* tersebut antara lain *Logbook*, Mahasiswa, GuruPamong, PPL dan Penilaian.



Gambar 4.5 *Class Diagram* Sistem Informasi PPL Iterasi 2

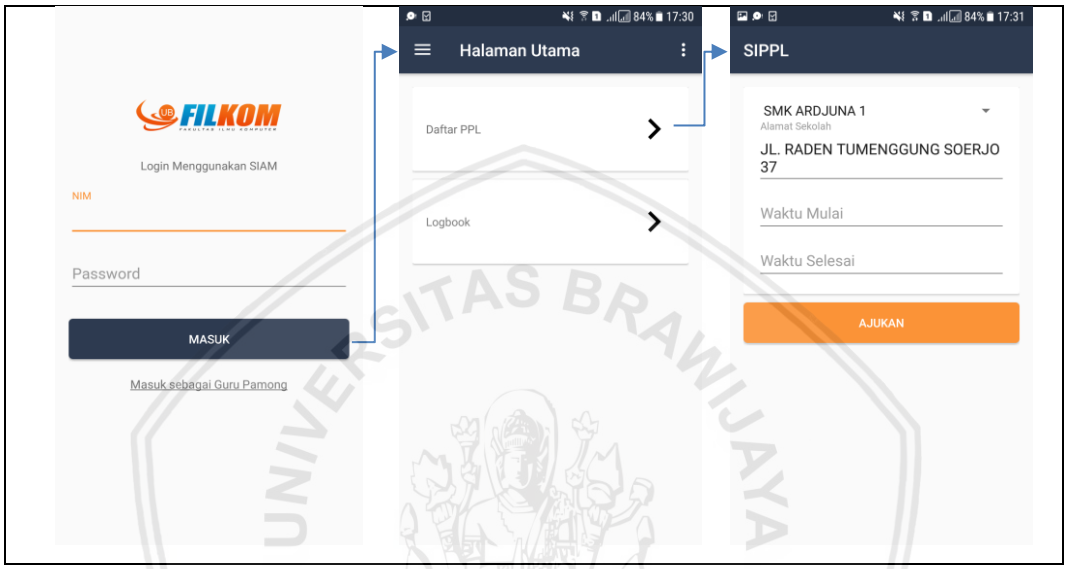


Gambar 4.6 *Class Diagram* Sistem Informasi PPL Iterasi 3

4.3 Iterasi Tahap Implementasi

4.3.1 Iterasi 1 Implementasi Pendaftaran PPL

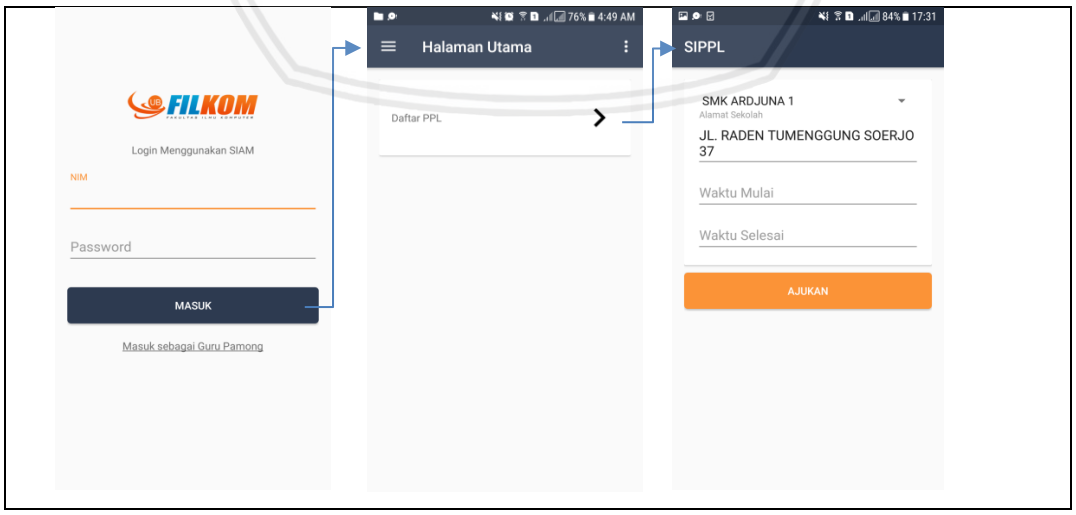
Pada iterasi pertama, menu *logbook* pada sistem muncul ketika mahasiswa belum melakukan pendaftaran PPL. Ketika dilakukan konfirmasi kepada *stakeholder*, *feedback adalah* ketika mahasiswa belum melakukan pendaftaran PPL maka menu *logbook* seharusnya tidak dimunculkan. Maka selanjutnya dilakukan Iterasi 2.



Gambar 4.7 Implementasi *Screen Flow* Pendaftaran PPL Iterasi 1

Iterasi 2 Implementasi Pendaftaran PPL

Pada iterasi kedua, menu *logbook* dihilangkan ketika mahasiswa belum melakukan pendaftaran PPL. Gambar 4.8 menunjukkan hasil iterasi 2, yaitu menu *logbook* tidak muncul ketika belum melakukan pendaftaran PPL.



Gambar 4.8 Implementasi *Screen Flow* Pendaftaran PPL Iterasi 2

4.4 Pembuatan *User Story*

Proses melakukan analisis kebutuhan berisikan tahapan-tahapan didalam mengumpulkan kebutuhan dalam penelitian yang sedang dilakukan. Penggalian kebutuhan dilakukan dengan cara *stakeholder interview* dengan pertanyaan yang bersifat *open-end question*. Pada bab ini akan terdapat beberapa tahapan analisis kebutuhan, diantarnya adalah gambaran umum sistem, identifikasi aktor, penentuan kebutuhan fungsional dan non-fungsional serta pemodelan kebutuhan dalam bentuk *use case* yang akan dijelaskan dalam *use case scenario.*

4.4.1 User Story

User story merupakan deskripsi singkat mengenai fitur sistem nantinya berdasarkan sudut pandang pengguna. Dalam pembuatan *user story* ini melibatkan *stakeholder*. Setelah membuat *user story,* selanjutnya dilakukan analisis berdasarkan sudut pandang bisnis . daftar *user story* ditunjukkan pada Tabel 4.1*.*

Tabel 4.1 Daftar *User Story*

No.	Kode	Skor	*User Story*
1.	US_01	8	Sebagai seorang mahasiswa, saya dapat mendaftarkan diri melaksanan kegiatan PPL secara online, sehingga saya menjadi lebih mudah dalam mendaftar.
2.	US _02	5	Sebagai seorang mahasiswa, saya dapat mencatat *logbook* harian melalui sistem, sehingga saya tidak perlu meminta tanda tangan persetujuan guru pamong setiap hari
3.	US _03	5	Sebagai seorang guru pamong, saya dapat menyetujui *logbook* mahasiswa melalui sistem, sehingga saya tidak perlu tanda tangan berkali-kali di lembar *logbook*
4.	US _04	8	Sebagai seorang guru pamong, saya dapat memberikan penilaian melalui sistem, sehingga penilaian menjadi lebih mudah dan orisinalitas terjaga

4.4.2 Business Perspective

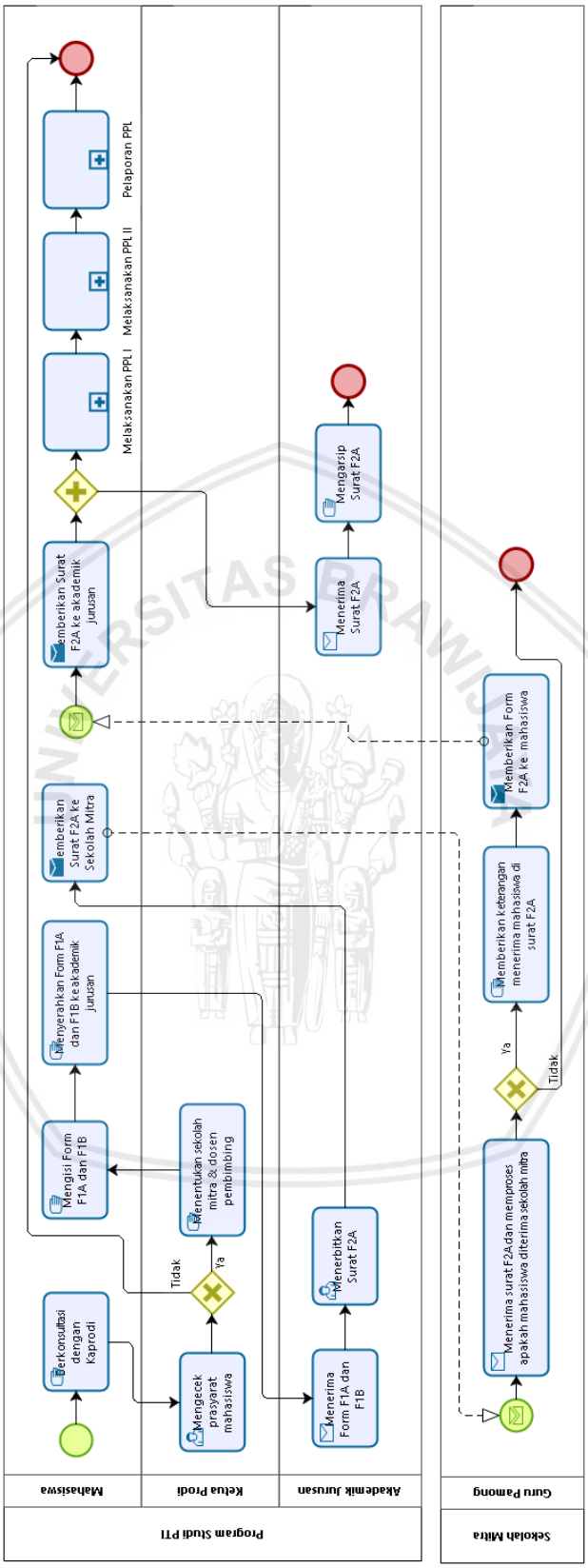
Pada proses *business perspective* ini akan dilakukan analisis mengenai kondisi atau alur kegiatan PPL yang saat ini sedang berjalan serta terdapat analisis mengenai alur yang akan digunakan setelah adanya sistem. Dalam tahap *bussiness perspective* ini menggunakan diagram BPMN atau *Business Process Model Notation.*

4.4.2.1 Identifikasi Proses Bisnis Pendaftaran PPL *as-is*

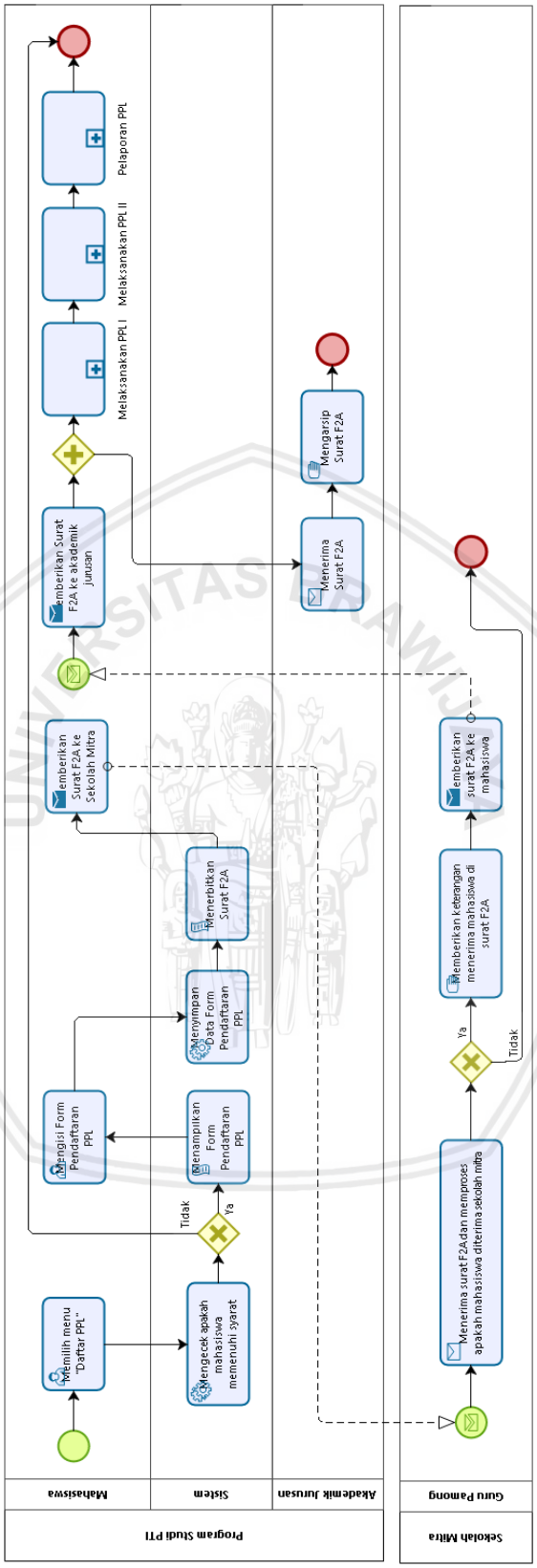
Berdasarkan hasil wawancara yang tertera dalam Lampiran 1 dan hasil analisis dokumen pendukung kegiatan PPL, diketahui bagaimana alur proses kegiatan PPL yang sedang berjalan saat ini. Kegiatan PPL dimulai dengan melakukan konsultasi terhadap Kaprodi PTI. Kaprodi PTI akan memeriksa apakah mahasiswa terkait sudah memenuhi syarat melaksanakan kegiatan PPL. Jika mahasiswa terkait telah memenuhi syarat maka Kaprodi akan menentukan sekolah mitra dan dosen pembimbing PPL. Selanjutnya, mahasiswa harus mengisi Form F1A dan F1B lalu menyerahkan form tersebut ke bagian akademik jurusan. Lalu bagian akademik jurusan memeriksa apakah data yang diisi sudah lengkap atau belum. Jika data telah lengkap maka bagian akademik mengeluarkan surat F2A untuk diserahkan mahasiswa kepada sekolah mitra sebagai tanda menerima mahasiswa melaksanakan PPL di sekolah tersebut. Jika sekolah mitra menerima mahasiswa, maka sekolah mitra memberikan keterangan diterima pada surat F2A. Selanjutnya mahasiswa harus mengembalikan surat F2A ke bagian akademik jurusan. Dengan begitu maka mahasiswa terkait dinyatakan resmi dapat memulai pelaksanaan PPL. Berdasarkan alur pendaftaran kegiatan PPL yang telah dijelaskan, maka dapat digambarkan BPMN alur pendaftaran kegiatan PPL sampai penilaian PPL. Gambar 4.9 merupakan BPMN alur pendaftaran kegiatan PPL saat ini.

4.4.2.2 Identifikasi Proses Bisnis Pendaftaran PPL to-be

Berdasarkan hasil analisis mengenai alur pendaftaran PPL saat ini, diketahui terdapat proses yang dapat disederhanakan menggunakan sistem. Gambar 4.10 berikut adalah BPMN pendaftaran PPL dengan menggunakan sistem. Proses pertama yang harus dilakukan oleh mahasiswa untuk mendaftarkan diri mengikuti kegiatan PPL adalah pertama mahasiswa harus login kedalam sistem. Lalu mahasiswa memilih tombol Daftar PPL. Pada halamana utama ini tombol *logbook* tidak dimunculkan terlebih dahulu sebelum mahasiswa mendaftarkan diri mengikuti kegiatan PPL. Setelah mahasiswa menekan tombol PPL, sistem akan menampilkan halaman Form F1A dan F1B. Selanjutnya, mahasiswa harus memilih sekolah yang akan dituju untuk melaksanakan PPL. Pemilihan sekolah ditentukan oleh kaprodi terlebih dahulu lalu mahasiswa memasukkan data ke sistem. Untuk data tanggal mulai dan tanggal selesai ditentukan oleh kesepakatan sekolah mitra dan mahasiswa ketika mahasiswa melakukan survey ke sekolah mitra. Selanjutnya mahasiswa harus menekan tombol Simpan. Sistem ketika mahasiswa menekan tombol simpan akan melakukan *request* ke *webservice* untuk menyimpan data pendaftaran PPL. Ketika penyimpanan data selesai, selanjutnya sistem akan menyediakan Form F2A untuk dicetak mahasiswa lalu diberikan ke sekolah mitra. Kemudian ketika sekolah mitra menerima mahasiswa, maka pihak sekolah mitra memberikan data konfirmasi kepada mahasiswa dan mahasiswa memberikan data konfirmasi kepada akademik kampus. Selanjutnya mahasiswa telah resmi dinyatakan terdaftar melaksanakan PPL.



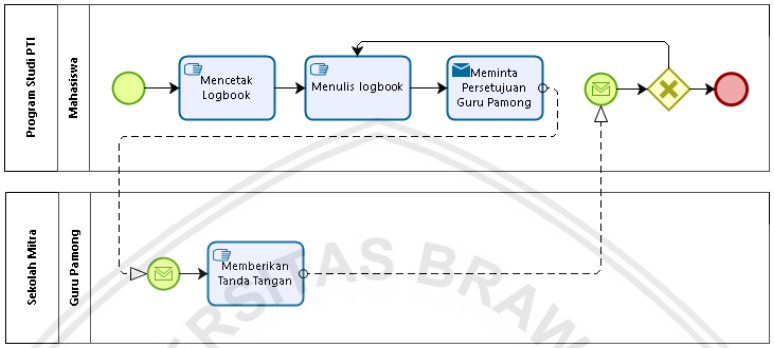
Gambar 4.9 BPMN Pendaftaran Kegiatan PPL *as-is*



Gambar 4.10 BPMN Pendaftaran Kegiatan PPL *to-be*

4.4.2.3 Identifikasi Proses Bisinis Pengisian Logbook as-is

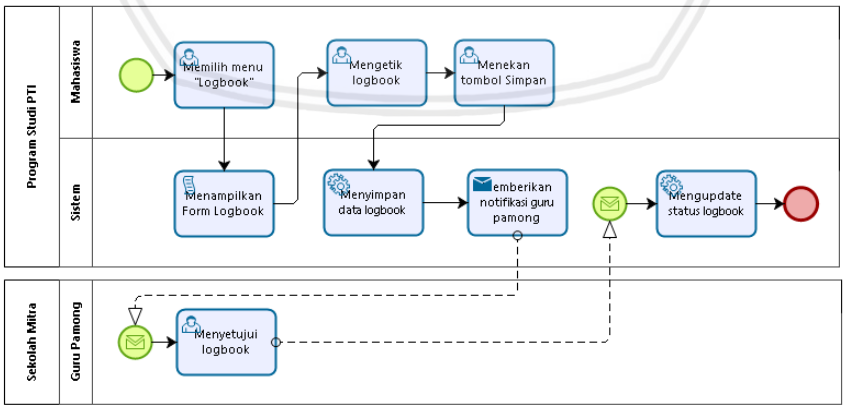
Berdasarkan analisis hasil wawancara dan dokumen pendukung kegiatan PPL. Proses pengisian logbook saat ini dilakukan dengan langkah pertama adalah mahasiswa harus mencetak format resmi *logbook* dari kampus. Lalu mahasiswa harus menuliskan kegiatan yang telah dilakukan selama kegiatan PPL berlangsung setiap hari. Lalu setiap hari mahasiswa harus meminta tanda tangan guru pamong untuk menyetujui *logbook* aktifitas mahasiswa. Gambar 4.11 menggambarkan proses pencatatan *logbook* yang saat ini berlangsung.



Gambar 4.11 BPMN Pencatatan *logbook* PPL *as-is*

4.4.2.4 Identifikasi Proses Bisnis Pencatatan logbook to-be

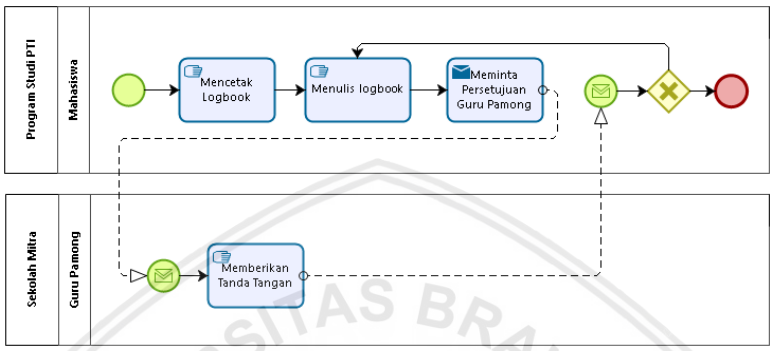
Berdasarkan hasil analisis mengenai alur pencatatan *logbook* saat ini, diketahui terdapat proses yang dapat ditambahkan menggunakan sistem untuk meningkatkan efisiensi kegiatan. Gambar 4.12 berikut adalah BPMN pencatatan *logbook* dengan menggunakan sistem.



Gambar 4.12 BPMN Pencatatan *logbook* PPL *to-be*

4.4.2.5 Identifikasi Proses Bisinis Persetujuan Logbook as-is

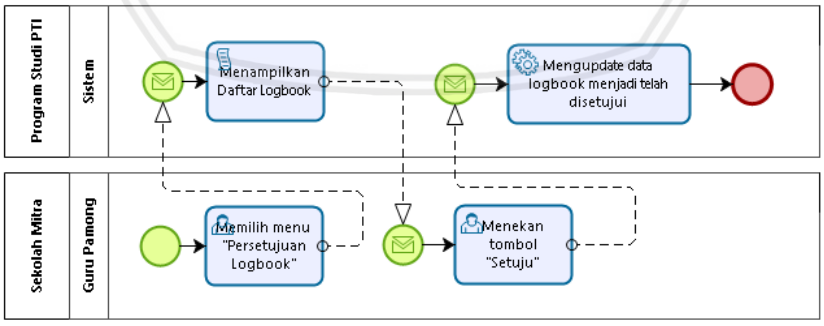
Berdasarkan analisis hasil wawancara dan analisis dokumen pendukung kegiatan PPL. Proses persetujuan logbook saat ini dilakukan dengan langkah pertama adalah mahasiswa harus menemui guru pamong setiap hari setelah kegiatan PPL pada suatu hari. Lalu guru pamong memeriksa *logbook* yang ditulis mahasiswa dan memberikan tanda-tangan pada lembar *logbook* mahasiswa. Gambar 4.13 menggambarkan proses persetujuan *logbook* saat ini.



Gambar 4.13 BPMN Persetujuan *logbook* PPL *as-is*

4.4.2.6 Identifikasi Proses Bisinis Persetujuan Logbook to-be

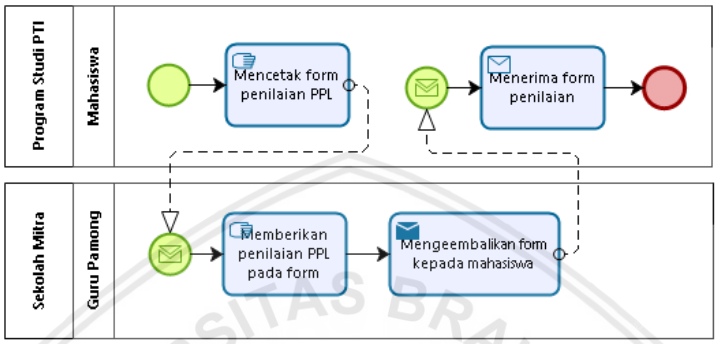
Berdasarkan hasil analisis mengenai alur persetujuan *logbook* saat ini, diketahui terdapat proses yang dapat ditambahkan menggunakan sistem untuk meningkatkan efisiensi kegiatan. Gambar 4.14 berikut adalah BPMN persetujuan *logbook* dengan menggunakan sistem.



Gambar 4.14 BPMN Persetujuan *logbook* PPL to-be

4.4.2.7 Identifikasi Proses Bisinis Penilaian Kegiatan PPL as-is

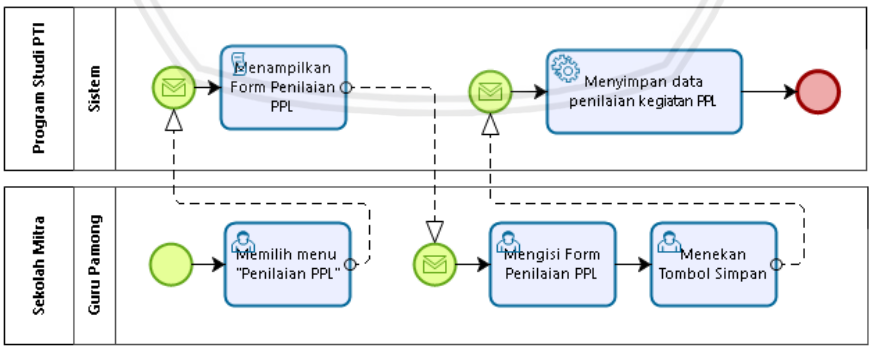
Berdasarkan analisis hasil wawancara dan analisis dokumen pendukung kegiatan PPL. Proses penilaian kegiatan PPL saat ini dilakukan dengan langkah pertama adalah mahasiswa harus mencetak form penilaian PPL lalu menyerahkan kepada guru pamong lalu guru pamong memberikan nilai pada form dan selanjutnya form dikembalikan kepada mahasiswa. Gambar 4.15 menggambarkan proses penilaian kegiatan PPL saat ini.



Gambar 4.15 BPMN Penilaian Kegiatan PPL *as-is*

4.4.2.8 Identifikasi Proses Bisinis Penilaian Kegiatan PPL to-be

Berdasarkan hasil analisis mengenai alur penilaian kegiatan PPL saat ini, diketahui terdapat proses yang dapat ditambahkan menggunakan sistem untuk meningkatkan efisiensi kegiatan. Gambar 4.16 berikut adalah BPMN penilaian kegiatan PPL dengan menggunakan sistem.



Gambar 4.16 BPMN Penilaian Kegiatan PPL *to-be*

4.4.2.9 Identifikasi Daftar BPMN dan Aktifitas

Berdasarkan diagram BPMN yang telah dianalisis dan dibuat, dibuatkan daftar BPMN berupa tabel yang berisi kode, nama BPMN dan deskripsi. Tujuan pembuatan BPMN ini adalah untuk meningkatkan *traceability* analisis kebutuhan yang dilakukan. Tabel 4.2 merupakan tabel daftar BPMN pada penelitian ini.

Tabel 4.2 Daftar BPMN

No.	Kode BPMN	Kode *User Story*	Nama	Deskripsi
1.	BPMN_01	US_01	Pendaftaran PPL *to be*	Mahasiswa mendaftarkan diri melaksanakan kegiatan PPL melalui sistem
2.	BPMN_02	US_02	Mencatat *logbook to be*	Mahasiswa mencatat *logbook* melalui sistem
3.	BPMN_03	US_03	Menyetujui *logbook to be*	Guru pamong menyetujui *logbook* mahasiswa melalui sistem
4.	BPMN_04	US_04	Penilaian PPL *to be*	Guru pamong memberikan penilaian kegiatan PPL mahasiswa melalui sistem

Selain daftar BPMN, dibuatkan juga tabel aktifitas yang menggambarkan aktifitas yang dilakukan *stakeholder* dalam melaksanakan kegiatan PPL. Daftar aktifitas ini dibuat berdasarkan alur BPMN yang telah dibuat. Tabel 4.3 merupakan Tabel Aktifitas yang berisi aktifitas pendaftaran, pencatatan *logbook*, menyetujui *logbook* dan penilaian PPL.

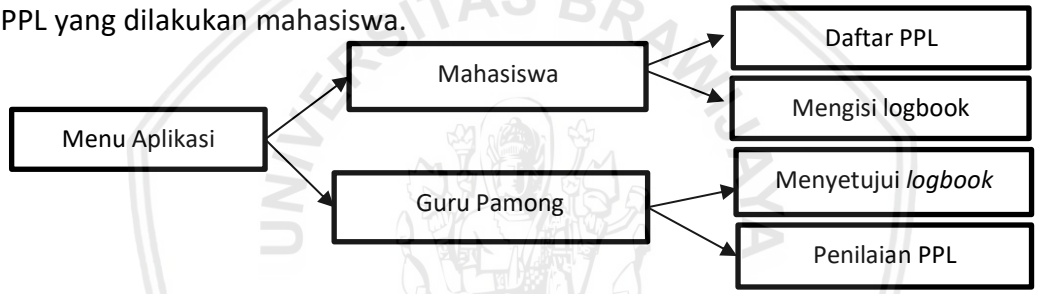
Tabel 4.3 Daftar Aktifitas

No.	Kode	Kode BPMN	Nama	Deskripsi
1.	AKT_01	BPMN_01	Pendaftaran PPL *to be*	Mahasiswa mendaftarkan diri melaksanakan kegiatan PPL melalui sistem
2.	AKT_02	BPMN_02	Mencatat *logbook to be*	Mahasiswa mencatat *logbook* setiap hari melalui sistem
3.	AKT_03	BPMN_03	Menyetujui *logbook to be*	Guru pamong menyetujui *logbook* mahasiswa melalui sistem

No.	Kode	Kode BPMN	Nama	Deskripsi
4.	AKT_04	BPMN_04	Penilaian PPL *to be*	Guru pamong memberikan penilaian kegiatan PPL mahasiswa melalui sistem

4.4.3 Gambaran Umum Sistem

Berdasrkan hasil iterasi kedua tahap pembuatan *user story,* sehingga saat ini hanya terdapat 2 aktor dan 4 fungsi utama sistem. Sistem yang akan dibangun digambarkan kedala gambaran umum yang dapat dilihat pada Gambar 4.17. Sistem nantinya akan mencakup proses pendaftaran PPL, pencatatan *logbook* dan penilaian PPL mahasiswa oleh guru pamong. Selama masa pelaksanaan PPL di sekolah mitra. Mahasiswa dapat memasukkan logbook harian kedalam sistem yang nanti dapat direkapitulasi di akhir kegiatan. Didalam sistem juga akan dibangun fitur untuk memberikan penilaian oleh guru pamong terhadap kegiatan PPL yang dilakukan mahasiswa.



Gambar 4.17 Gambaran Umum Sistem

4.5 Pemecahan *User Story* Menjadi *Task-Task* Kecil

4.5.1 Identifikasi Aktor

Pada tahapan ini akan dijelaskan mengenai siapa saja aktor yang akan berinteraksi dengan sistem. Pada sistem yang akan dibangun ini nantinya akan ada 2 aktor, yaitu aktor Mahasiswa dan Guru. Penjelasan lebih detil dari kedua aktor tersebut akan dijelaskan dalam Tabel 4.4.

Tabel 4.4 Identifikasi Aktor

Aktor	Penjelasan
Mahasiswa	Aktor Mahasiswa merupakan aktor yang bisa melakukan pendaftaran PPL dan pencatatan logbook.
Guru	Aktor Guru merupakan aktor yang bisa melakukan penilaian terhadap proses kegiatan PPL mahasiswa selama di sekolah mitra.

4.5.2 Aturan Penomoran

Pada tahap ini akan dijelaskan mengenai penomoran untuk mewakili daftar kebutuhan sistem yang akan digunakan selama penelitian. Penjelasan aturan penomoran dapat dilihat pada Tabel 4.5.

Tabel 4.5 *Index Random Consistency*

Kode Penomoran	**Penjelasan**
SIPPL	Nama Sistem
F	Kode yang melambangkan kebutuhan fungsional sistem
NF	Kode yang melambangkan kebutuhan non-fungsional sistem
NOMOR	Merupakan penomoran urutan kebutuhan

Contoh :

SIPPL_F_01 : Mahasiswa dapat mencatat logbook harian ke dalam sistem.

4.5.3 Kebutuhan Fungsional

Kebutuhan fungsional sistem adalah kebutuhan yang diharapkan harus ada oleh pengguna sistem. Berikut ini adalah penjelasan secara rinci mengenai kebutuhan fungsional sistem. Daftar kebutuhan funsional sistem dapat dilihat pada tabel Tabel 4.6.

Tabel 4.6 Kebutuhan Fungsional Sistem

No.	**Kode**	**Nama**	**Deskripsi**	**Aktor**
1.	SIPPL_F_01	*Login* kedalam sistem	Mahasiswa maupun guru pamong dapat *login* kedalam sistem	Mahasiswa, guru
2.	SIPPL_F_02	Mendaftarkan diri (*Sign Up*) sebagai guru pamong ke sistem	Guru pamong dapat mendaftarkan diri kedalam sistem menggunakan email	Guru
3.	SIPPL_F_03	Mendaftarkan diri melaksanakan kegiatan PPL	Mahasiswa dapat mendaftarkan diri melaksanakan PPL menggunakan sistem	Mahasiswa
4.	SIPPL_F_04	Menambahkan dan menyimpan	Mahasiswa dapat menambahkan dan menyimpan *logbook*	Mahasiswa

No.	Kode	Nama	Deskripsi	Aktor
		logbook harian	harian kedalam sistem	
5.	SIPPL_F_05	Menyetujui *logbook* harian mahasiswa	Guru pamong dapat menyetujui *logbook* harian yang telah dibuat oleh mahasiswa menggunakan sistem	Guru
6.	SIPPL_F_06	Memberikan penilaian PPL	Guru pamong dapat memberikan penilaian PPL yang dilakukan mahasiswa	Guru

4.5.4 Kebutuhan Non Fungsional

Kebutuhan non fungsional sistem adalah kebutuhan yang jika ada maka akan menambah nilai guna pada sistem. Pada bagian ini akan dijelaskan secara rinci mengenai kebutuhan non fungsional sistem informasi PPL yang akan dibangun. Kebutuhan non fungsional dapat dilihat pada Tabel 4.7.

Tabel 4.7 Kebutuhan Non Fungsional Sistem

No.	Kode	Nama	Deskripsi
1.	SIPPL_NF_01	*Compatibility*	Sistem dapat berjalan pada semua versi Android diatas versi 4.0 (*Ice Cream Sandwich)*

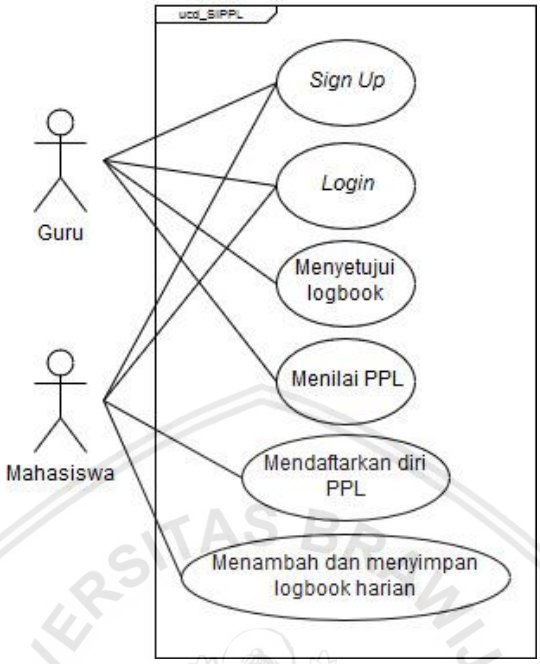
4.5.5 Pemodelan Kebutuhan

Tahap selanjutnya didalam perekayasaan kebutuhan adalah proses pemodelan kebutuhan. Pemodelan kebutuhan dibuat guna mempermudah pemahaman kebutuhan-kebutuhan yang telah didefinisikan. Pemodelan kebutuhan akan representasikan kedalam diagram *use case*. Diagram *use case* akan menjelaskan semua yang dapat dilakukan aktor terhadap sistem. Lalu selanjutnya diagram *use case* dijelaskan secara rinci pada bagian *use case scenario*.

4.5.5.1 Diagram *Use Case*

Berdasarkan hasil iterasi kedua, aktor dosen pembimbing dihilangkan sehingga pada *use case diagram* terdapat 2 aktor dan 6 *use case.* Gambar 4.18 menunjukkan *use case diagram* hasil iterasi kedua, terlihat aktor guru pamong memiliki fungsi untuk *sign up, login,* menyetujui *logbook* dan memberikan penilaian PPL. Sedangkan aktor mahasiswa memiliki fitur *login,* mendaftar PPL dan

mencatat *logbook.* Hasil pembuatan *use case diagram* ini nantinya akan dibuatkan *use case scenario.*



Gambar 4.18 Diagram *Use Case* Sistem Informasi PPL Iterasi 2

4.5.5.2 *Use Case Scenario*

Didalam *use case scenario*, semua yang terdapat pada diagram *use case* akan dijelaskan secara rinci mulai dari *pre-condiiton* sampai bagian *post-condition.* Berikut adalah *use case scenario* sistem informasi PPL yang dibuat bedasarkan diagram *use case* pada Gambar 4.18.

1. Melakukan login kedalam sistem informasi PPL (SIPPL_F_01)
   Use case scenario dari *use case* melakukan *login* kedalam sistem informasi PPL dapat dilihat pada Tabel 4.8.

Tabel 4.8 *Use Case Scenario* Melakukan *Login* Ke Dalam sistem

Melakukan *login* kedalam sistem informasi PPL	
Objective	Mahasiswa dan Guru dapat melakukan login sesuai dengan *role* masing-masing
Aktor	Mahasiswa dan Guru
Pre-condition	Mahasiswa dan Guru telah membuka aplikasi Sistem Informasi PPL (SIPPL) Mahasiswa atau Guru telah membuka halaman login sesuai masing-masing *role*
Main Flow	1. Mahasiswa memasukkan NIM dikolom NIM, sedangkan guru memasukkan email yang sudah didaftarkan kedalam sistem pada kolom email.

	2. Mahasiswa dan guru memasukkan *password* dikolom *password* pada halaman login masing-masing. 3. Mahasiswa dan guru menekan tombol "masuk" pada halaman login masing-masing.
Alternative Flow	-
Post-Condition	Sistem mengarahkan halaman ke halaman *home* ketika mahasiswa atau guru memasukkan kombinasi nim / email dan *password* secara benar. Jika kombinasi nim / email dan *passwordi* yang dimasukkan kedalam sistem salah, maka muncul pesan "Gagal Login".

2. Mendaftarkan diri (*Sign Up*) sebagai guru pamong ke sistem (SIPPL_F_02)

Use case scenario dari *use case* mendaftarkan diri (*Sign Up*) sebagai guru pamong ke sistem informasi PPL dapat dilihat pada Tabel 4.9.

Tabel 4.9 *Use Case Scenario* Mendaftarkan diri (*Sign Up*) sebagai guru pamong ke sistem

Mendaftarkan diri (*Sign Up*) sebagai guru pamong ke sistem	
Objective	Guru dapat melakukan *Sign Up* kedalam sistem
Aktor	Guru
Pre-condition	Guru telah membuka aplikasi Sistem Informasi PPL (SIPPL) Guru telah membuka halaman *Sign Up*
Main Flow	1. Guru memasukkan data diri sesuai form *Sign Up* yang terdapat pada sistem 2. Guru menekan tombol "Daftar" 3. Sistem mengirimkan email verifikasi 4. Guru mendapatkan email verifikasi 5. Guru melakkukan verifikasi email dengan membuka link unik yang ada didalam email
Alternative Flow	1. Guru memasukkan data diri yang kurang lengkap pada sistem 2. Tombol "Daftar" tidak bisa ditekan
Post-Condition	Sistem akan menyimpan data yang dimasukkan guru ke sistem. Setelah melakukan verifikasi email, guru dapat melakukan *Login* kedalam sistem seperti *use case scenario* pertama

3. Mendaftarkan diri melaksanakan Kegiatan PPL (SIPPL_F_03)

   Use case scenario dari *use case* mendaftarkan diri melaksanakan kegiatan PPL dapat dilihat di Tabel 4.10.

Tabel 4.10 *Use Case Scenario* Mendaftarkan Diri Melaksanakan Kegiatan PPL

Mendaftarkan diri melaksanakan kegiatan PPL	
Objective	Mahasiswa dapat diakui secara resmi telah mendaftarkan diri untuk melaksanakan kegiatan PPL
Aktor	Mahasiswa
Pre-condition	Mahasiswa telah membuka aplikasi Sistem Informasi PPL (SIPPL) Mahasiswa telah melakukan berhasil *Login* menggunakan NIM dan *Password* Mahasiswa berada pada halaman *Home*
Main Flow	1. Mahasiswa memilih menu "Daftar PPL" 2. Sistem akan mengecek apakah mahasiswa yang bersangkutan sudah memenuhi syarat untuk melaksanakan PPL atau belum. 3. Jika sudah memenuhi syarat, maka sistem akan menampilkan halaman form F1A dan form F1B 4. Lalu mahasiswa mengisi form F1A dan F1B yang terdapat pada sistem 5. Mahasiswa menekan tombol "Ajukan" 6. Sistem akan menyimpan data yang dimasukkan lalu sistem akan memberikan form F2A, F2B dan F2C untuk diberikan kepada sekolah mitra yang dituju mahasiswa
Alternative Flow	3. Mahasiswa belum memenuhi syarat melaksanakan kegiatan PPL 4. Sistem akan diarahkan ke menu *Home*
Post-Condition	Sistem akan menyimpan data yang dimasukkan mahasiswa kedalam sistem serta mencatat tanggal memasukkan data tersebut. Mahasiswa dinyatakan telah mendaftarkan diri melaksanakan PPL ketika form F2C diberikan ke bagian akademik jurusan.

4. Menambahkan dan menyimpan *logbook* harian PPL ke dalam sistem (SIPPL_F_04)

   Use case scenario dari *use case* menambahkan dan menyimpan *logbook* harian PPL kedalam sistem dapat dilihat di Tabel 4.11.

Tabel 4.11 *Use Case Scenario* Menambahkan dan Menyimpan *Logbook* Harian Kedalam Sistem

Menambahkan dan menyimpan *logbook* harian kedalam sistem	
Objective	Mahasiswa dapat melakukan pencatatan *logbook* harian dengan cara menambahkan dan menyimpan *logbook* kedalam sistem informasi PPL
Aktor	Mahasiswa
Pre-condition	Mahasiswa telah membuka aplikasi Sistem Informasi PPL (SIPPL) Mahasiswa telah melakukan berhasil *Login* menggunakan NIM dan *Password* Mahasiswa berada pada halaman *Home*
Main Flow	1. Mahasiswa memilih menu "*Logbook*" 2. Sistem akan menampilkan daftar logbook yang pernah dibuat sebelumnya 3. Mahasiswa menekan tombol "Tambah" untuk menambahkan *logbook* harian 4. Sistem menampilkan form *logbook* 5. Lalu mahasiswa mengisi form *logbook* yang terdapat pada sistem 6. Mahasiswa menekan tombol "Simpan"
Alternative Flow	4. Mahasiswa menekan tombol "Batal" 5. Sistem akan diarahkan ke halaman daftar *logbook* harian yang telah dibuat
Post-Condition	Sistem akan menyimpan data yang dimasukkan mahasiswa kedalam sistem serta mencatat tanggal memasukkan data tersebut. Halaman akan diarahkan ke halaman daftar *logbook* harian.

5. Menyetujui *logbook* harian yang telah dibuat oleh mahasiswa (SIPPL_F_05)

   Use case scenario dari *use case* menyetujui *logbook* harian yang telah dibuat mahasiswa dapat dilihat di Tabel 4.12.

Tabel 4.12 *Use Case Scenario* Menyetujui *Logbook* Harian Yang Telah Dibuat Oleh Mahasiswa

Menyetujui *logbook* harian yang telah dibuat oleh mahasiswa	
Objective	Guru dapat melakukan persetujuan *logbook* harian mahasiswa PPL
Aktor	Guru
Pre-condition	Guru telah membuka aplikasi Sistem Informasi PPL (SIPPL) Guru telah melakukan berhasil *Login* menggunakan Email dan *Password* pada halaman login guru pamong Guru berada pada halaman *Home*
Main Flow	1. Guru memilih menu "Persetujuan *Logbook*" 2. Sistem akan menampilkan daftar logbook yang telah dibuat oleh mahasiswa 3. Guru menekan tombol "Setuju" untuk melakukan persetujuan *logbook* harian mahasiswa
Alternative Flow	-
Post-Condition	Sistem akan memperbarui status *logbook* mahasiswa menjadi telah disetujui oleh guru pamong.

6. Memberikan penilaian PPL mahasiswa oleh guru pamong (SIPPL_F_06)

   Use case scenario dari *use case* memberikan penilaian PPL mahasiswa oleh guru pamong dapat dilihat di Tabel 4.13.

Tabel 4.13 *Use Case Scenario* Memberikan Penilaian PPL Mahasiswa Oleh Guru Pamong

Memberikan penilaian PPL mahasiswa oleh guru pamong	
Objective	Guru dapat melakukan persetujuan *logbook* harian mahasiswa PPL
Aktor	Guru
Pre-condition	Guru telah membuka aplikasi Sistem Informasi PPL (SIPPL) Guru telah melakukan berhasil *Login* menggunakan Email dan *Password* pada halaman login guru pamong Guru berada pada halaman *Home*
Main Flow	1. Guru memilih menu "Penilaian PPL" 2. Sistem akan menampilkan form penilaian PPL

	3. Guru mengisi form penilaian berdasar hasil kegiatan PPL yang telah dilakukan oleh mahasiswa 4. Guru menekan tombol “Simpan”
Alternative Flow	-
Post-Condition	Sistem akan menyimpan data penilaian. Sistem akan mengarahkan halaman ke halaman *Home.*

4.6 Rencana *Release*

Pada proses rencana *release* ini merupakan perancangan sistem yang terdiri dari 4 tahap, yaitu tahap perancangan UML, tahap perancangan arsitektur sistem, tahap perancangan basis data, dan tahap perancangan antarmuka.

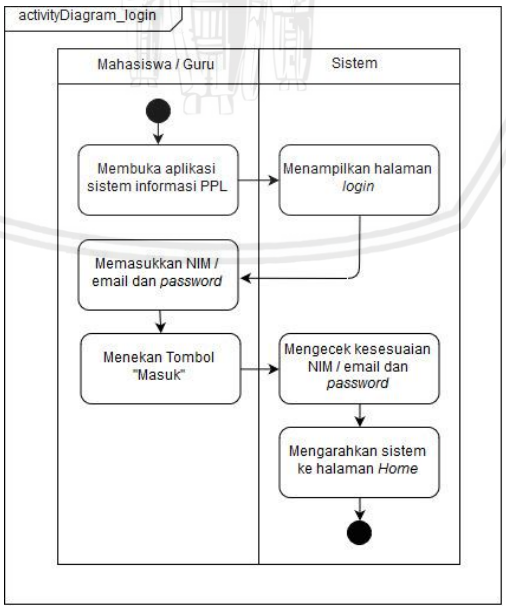
4.6.1 Perancangan UML

Didalam perancangan UML akan dibagi menjadi beberapa diagram, yaitu *activity diagram, class diagram* dan *sequence diagram*.

4.6.1.1 *Activity Diagram*

Activity diagram dibuat berdasarkan *use case scenario* yang telah dibuat. *Activiti diagram* berguna untuk memodelkan aktivitas pengguna terhadap sistem.

1. *Activity Diagram* melakukan *login* ke dalam sistem
   Alur dari melakukan *login* ke dalam sistem dapat dilihat pada Gambar 4.19.

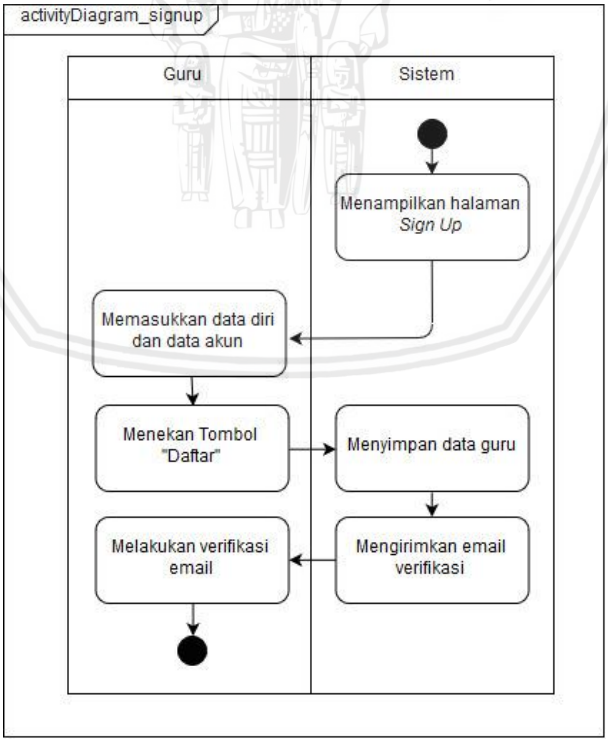


Gambar 4.19 *Activity Diagram* Melakukan *Login* kedalam sistem

Aktor mahasiswa atau guru pertama harus membuka sistem informasi PPL. Sistem pertama akan memunculkan halaman *login* ketika aktor tersebut belum tercatat telah *login.* Pada saat *login,* terdapat 2 halaman *login* yang berbeda. Satu halaman *login* digunakan khusus untuk *login* mahasiswa dan satu lagi adalah halaman *login* khusus digunakan untuk *login* guru pamong. Ketika mahasiswa atau guru telah mengisi NIM / email dan *password* sesuai yang telah terdaftarkan, selanjutnya aktor harus menekan tombol "masuk". Sistem akan mengecek apakah kombinasi NIM / email dan *password* cocok atau tidak. Jika cocok maka sistem akan diarahkan ke halaman *Home*. Sedangkan jika tidak cocok maka akan muncul pesan *error.*

2. *Activity Diagram* mendaftarkan diri (*Sign Up)* sebagai guru pamong ke sistem

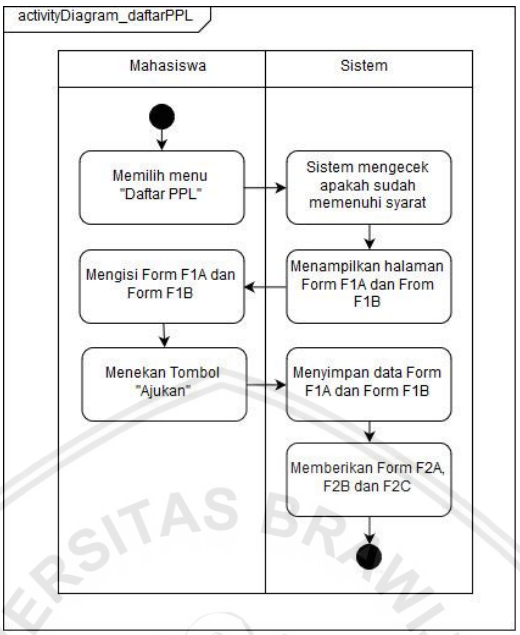
Alur dari melakukan *login* ke dalam sistem dapat dilihat pada Gambar 4.20. Sistem menampilkan halaman *Sign Up*, lalu guru pamong agar bisa melakukan *login* ke sistem harus memasukkan data diri dan data akun yang akan dibuat ke dalam sistem. Ketika aktor telah selesai mengisi semua data yang dibutuhkan oleh sistem, selanjutnya aktor harus menekan tombol "Daftar". Lalu sistem akan menyimpan data tersebut dan selanjutnya sistem akan mengirimkan *email* untuk melakukan verifikasi *email*. Didalam email tersebut terdapat link unik yang ketika dibuka otomatis akan mengubah status akun aktor menjadi telah terverifikasi.



Gambar 4.20 *Activity Diagram* Melakukan *Sign Up* Guru kedalam sistem

3. *Activity Diagram* mendaftarkan diri melaksanakan kegiatan PPL

Alur dari mendaftarkan diri melaksanakan kegiatan PPL dapat dilihat pada Gambar 4.21.

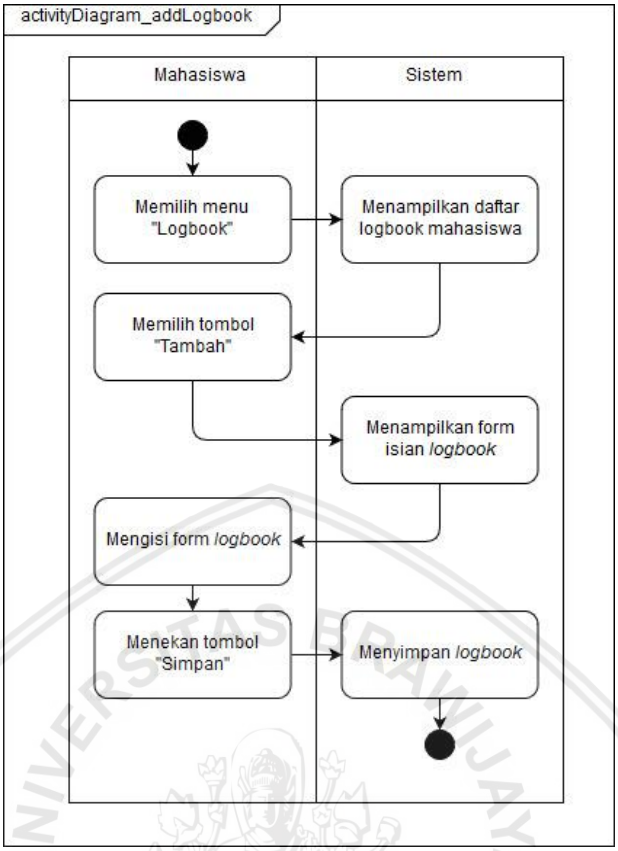


Gambar 4.21 *Activity Diagram* mendaftarkan diri melaksanaka kegiatan PPL

Mahasiswa dianggap sudah *login* dan sudah masuk ke halaman *Home*. Lalu mahasiswa memilih menu "Daftar PPL". Sistem akan melakukan pengecekan apakah mahasiswa yang bersangkutan sudah memenuhi syarat untuk melaksankan kegiatn PPL atau tidak dengan melihat informasi akun di SIAM. Selanjutnya jika mahasiswa telah memenuhi syarat melaksanakan PPL maka sistem akan menampilkan halaman Form F1A dan Form F1B. Untuk mendaftarkan diri, mahasiswa harus mengisi kedua form tersebut lalu ketika telah selesai ditekan tombol "Ajukan". Sistem akan menyimpan data dari form F1A dan F1B. Lalu sistem akan mengeluarkan form F2A, F2B dan F2C yang nantinya harus diberikan kepada sekolah mitra dan sekolah mitra harus mengembalikan form F2C kepada akadmik jurusan melalui mahasiswa sebagai tanda mahasiswa bersangkutan telah diterima di sekolah tersebut.

4. *Activity Diagram* menambahkan dan menyimpan *logbook* harian PPL ke dalam sistem

Alur dari mendaftarkan diri melaksanakan kegiatan PPL dapat dilihat pada Gambar 4.22.

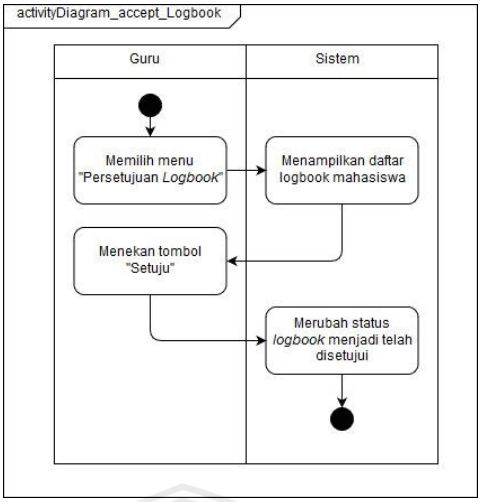


Gambar 4.22 *Activity Diagram* menambahkan dan menyimpan *logbook* harian PPL ke dalam sistem

Mahasiswa dianggap sudah *login* dan sudah masuk ke halaman *Home*. Lalu mahasiswa memilih menu "*Logbook*". Lalu sistem akan menampilkan daftar semua *logbook* yang pernah dibuat sebelumnya oleh mahasiswa tersebut. Untuk menambahkan *logbook*, mahasiswa harus menekan tombol "Tambah". Lalu sistem akan menampilkan form isian *logbook*. Mahasiswa harus mengisi semua *field* yang ada di form, lalu mahasiswa harus menekan tombol "Simpan" untuk menyimpan *logbook* ke dalam sistem. Selanjutnya sistem akan mengarahkan halaman ke halaman daftar *logbook* mahasiswa.

5. *Activity Diagram* menyetujui *logbook* harian yang telah dibuat oleh mahasiswa.

   Alur dari mendaftarkan diri melaksanakan kegiatan PPL dapat dilihat pada Gambar 4.23.

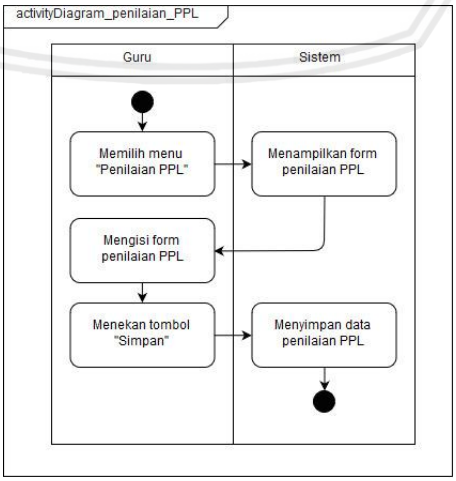


Gambar 4.23 *Activity Diagram* menyetujui *logbook* harian

Pertama guru pamong harus memilih menu "Persetujuan *Logbook".* Lalu sistem akan menampilkan daftar *logbook* mahasiswa. Guru pamong dapat melakukan persetujuan *logbook* dengan menekan tombol "Setuju" pada *logbook* yang ingin disetujui. Selanjutnya sistem akan merubah status *logbook* menjadi telah disetujui oleh guru pamong.

6. *Activity Diagram* penilaian PPL mahasiswa oleh guru pamong

Alur dari penilaian PPL mahasiswa oleh guru pamong dapat dilihat pada Gambar 4.24. Guru pamong memilih menu "Penilaian PPL" lalu sistem akan menampilkan form untuk memberikan penilaian. Guru mengisi form penilaian PPL pada sistem sesuai dengan hasil kegiatan PPL yang dilakukan mahasiswa di sekolah mitra. Ketika guru telah mengisi semua *field* pada form penilaian PPL, maka guru harus menekan tombol "Simpan". Kemudian sistem akan menyimpan data form penilaian PPL. Selanjutnya sistem akan mengarahkan ke halaman *Home.*

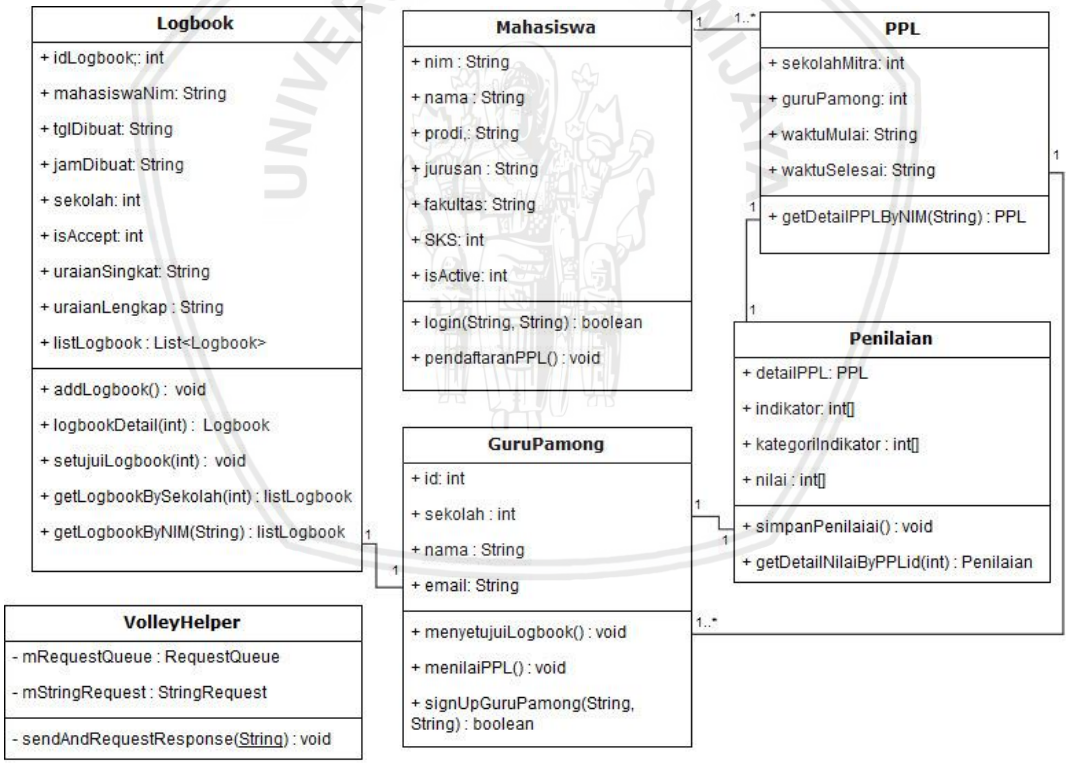


Gambar 4.24 *Activity Diagram* penilaian PPL oleh guru pamong

4.6.1.2 *Class Diagram*

Class diagram merupakan diagram yang akan menampilkan semua *class* yang digunakan dalam pengembangan sistem informasi ini. Selain itu, didalam diagram ini akan terlihat hubungan antara satu *class* dengan *class* yang lain. Didalam *class diagram* juga terdapat atribut dan method yang digunakan dalam sistem. Gambaran lengkap mengenai *class-class* beserta atribut dan methodnya dapat dilihat pada Gambar 4.25.

Berdasarkan hasil iterasi kedua *class diagram,* terjadi perubahan mendasar pada rancangan *class diagram.* Gambar 4.25 merepresentasikan semua *class* yang digunakan pada sistem. *Class-class* tersebut merupakan *object* pada sistem, terlihat semua *class* saling terhubung dan memiliki relasi sesuai fungsionalitasnya. Terdapat *class* Logbook yang berfungsi sebagai model pada sistem. Data *logbook* yang diperoleh dari *response* API berupa JSON akan dimasukkan kedalam *ListArray* dengan tipe objek *class* Logbook. Salah satu penggunaan *class* ini adalah untuk menampilkan daftar logbook pada recyclerView pada menu Logbook.



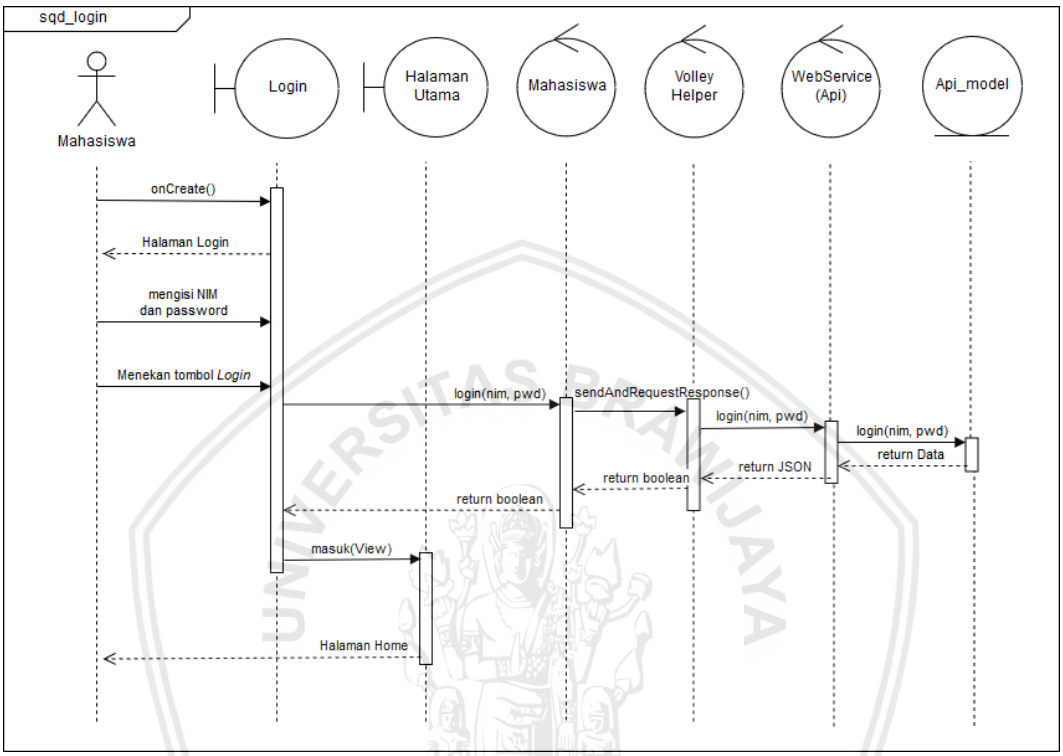
Gambar 4.25 *Class Diagram* Sistem Informasi PPL

4.6.1.3 *Sequence Diagram*

1. *Sequence Diagram* melakukan login kedalam sistem

   Sequence Diagram pada Gambar 4.26 merepresentasikan proses aktor mahasiswa dalam melakukan login kedalam sistem. Ketika aktor mahasiswa membuka sistem, pertama akan disajikan halaman (*activity)* Login. Didalam

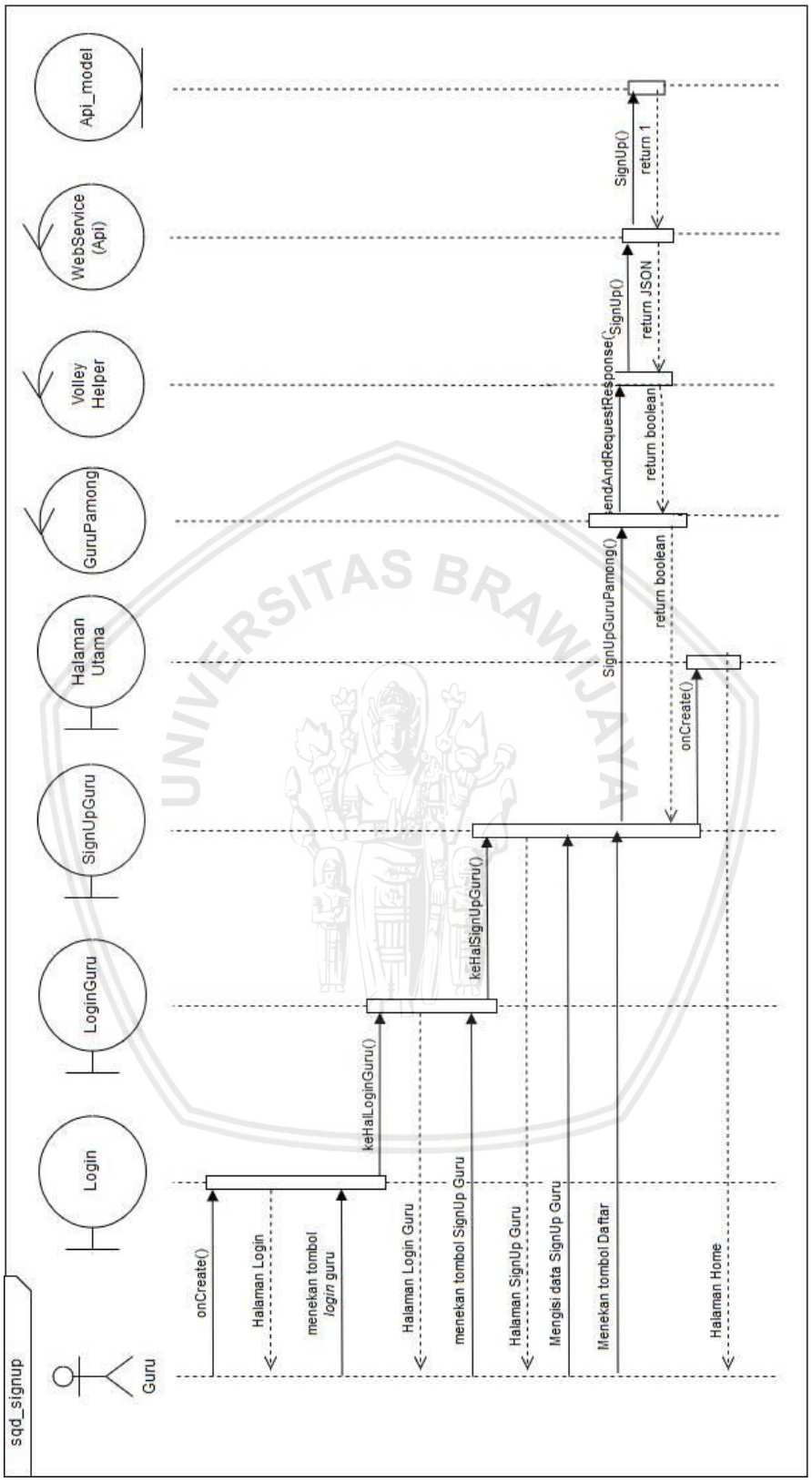
halaman Login aktor harus memasukkan NIM lalu memasukkan password. Selanjutnya ketika aktor menekan tombol *Login* maka sistem akan mengecek apakah kombinasi NIM dan password cocok atau tidak dengan menggunakan bantuan VolleyHelper. Penggunaan *Library* Volley dikarenakan pengecekan memanfaatkan API.



Gambar 4.26 *Sequence Diagram* proses aktor mahasiswa dalam melakukan login kedalam sistem

2. *Sequence Diagram* mendaftarkan diri (*Sign Up)* sebagai guru pamong ke sistem

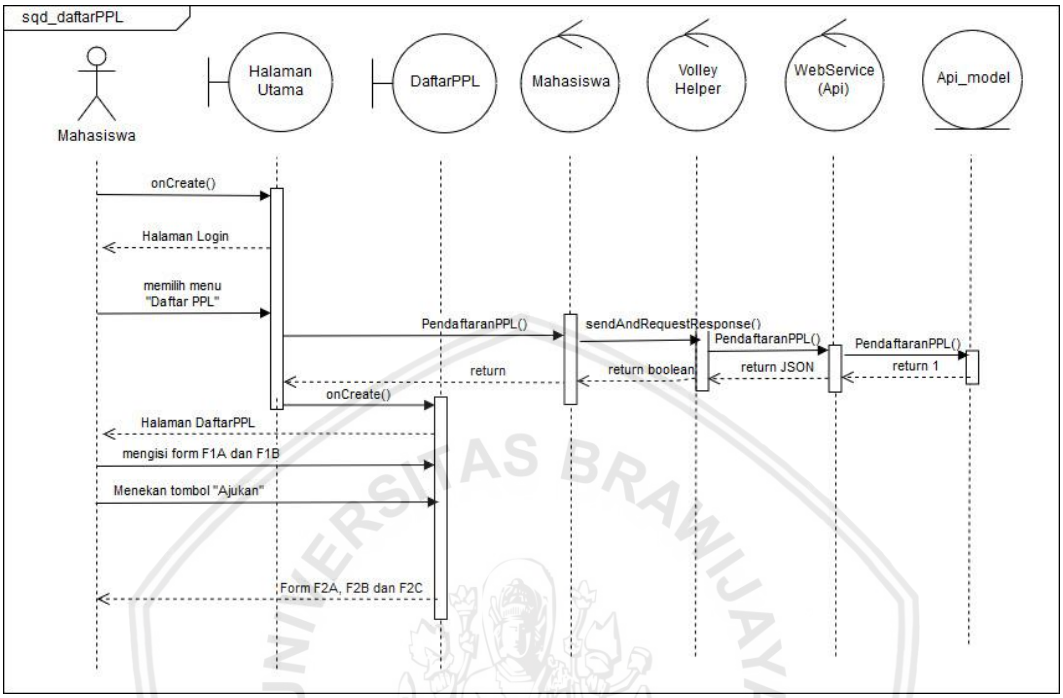
   Sequence Diagram pada Gambar 4.27 merepresentasikan proses pendaftaran diri (*Sign Up*) sebagai guru pamong ke sistem. Proses pendaftaran diri (*Sign Up*) sebagai guru pamong ke sistem dimulai dari halaman *Login.* Aktor guru pertama masuk ke halaman *login* guru. Lalu masuk ke halaman *Sign Up Guru* untuk memasukkan data diri dan data akun ke dalam sistem Setelah data terisi aktor harus menekan tombol Daftar. Aktor selanjutnya ketika telah berhasil mendaftar akan diarahkan langsung ke halaman *Login.*



Gambar 4.27 *Sequence Diagram Sign Up* sebagai guru pamong

3. *Sequence Diagram* mendaftarkan diri melaksanakan kegiatan PPL

Sequence Diagram pada Gambar 4.28 merepresentasikan proses mendaftarkan diri melaksanakan kegiatan PPL.

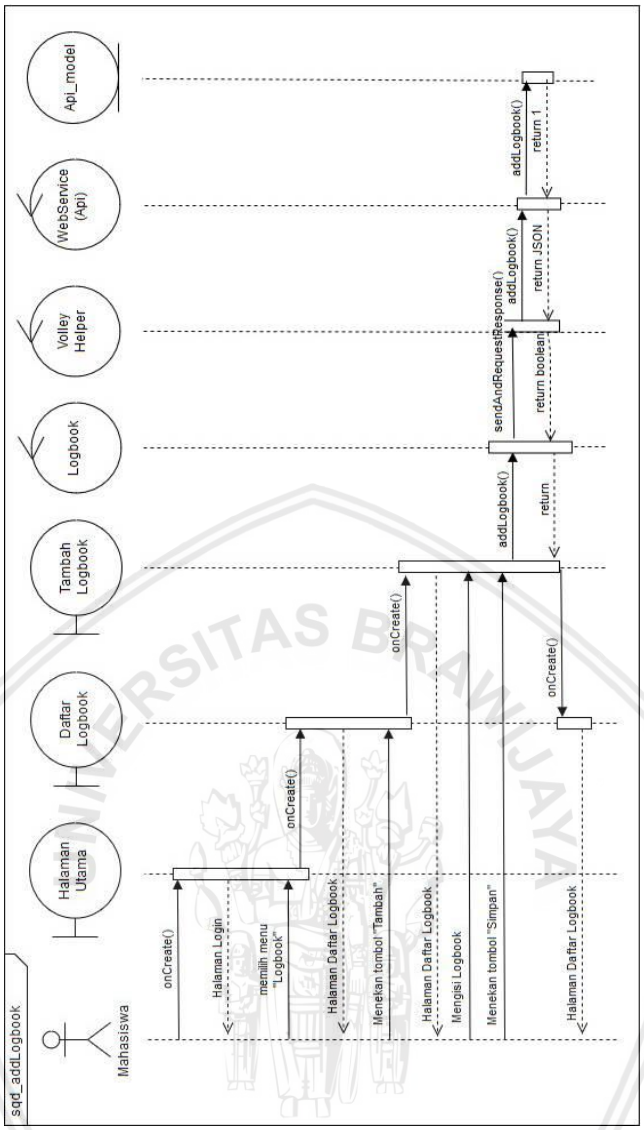


Gambar 4.28 *Sequence Diagram* proses mendaftarkan diri melaksanakan kegiatan PPL

Proses pendaftaran mahasiswa melaksanakan kegiatan PPL pertama dengan memilih menu “Daftar PPL”. Lalu sistem akan mengecek apakah mahasiswa yang bersangkutan sudah memenuhi syarat melaksanakan PPL atau belum dengan menggunakan VolleyHelper memanggil API pengecekan. Jika mahasiswa telah memenuhi syarat maka sistem akan mengarahkan halaman ke halaman form F1A dan F1B. Selanjutnya mahasiswa harus mengisi data yang diperlukan dalam form F1A dan form F1B. Setelah data terisi maka mahasiswa harus menekan tombol “Ajukan” untuk menyimpan data form. Selanjutnya sistem akan memberikan form F2A, F2B dan F2C kepada mahasiswa untuk diserahkan kepada sekolah mitra. Selanjutnya mahasiswa telah dinyatakan terdaftar melaksanakan PPL setelah sekolah mitra membalas form F2C kepada akademik jurusan melalui mahasiswa.

4. *Sequence Diagram* menambahkan dan menyimpan *logbook* harian PPL ke dalam sistem

Sequence Diagram pada Gambar 4.29 merepresentasikan proses menambahkan dan menyimpan *logbook* harian PPL ke dalam sistem.

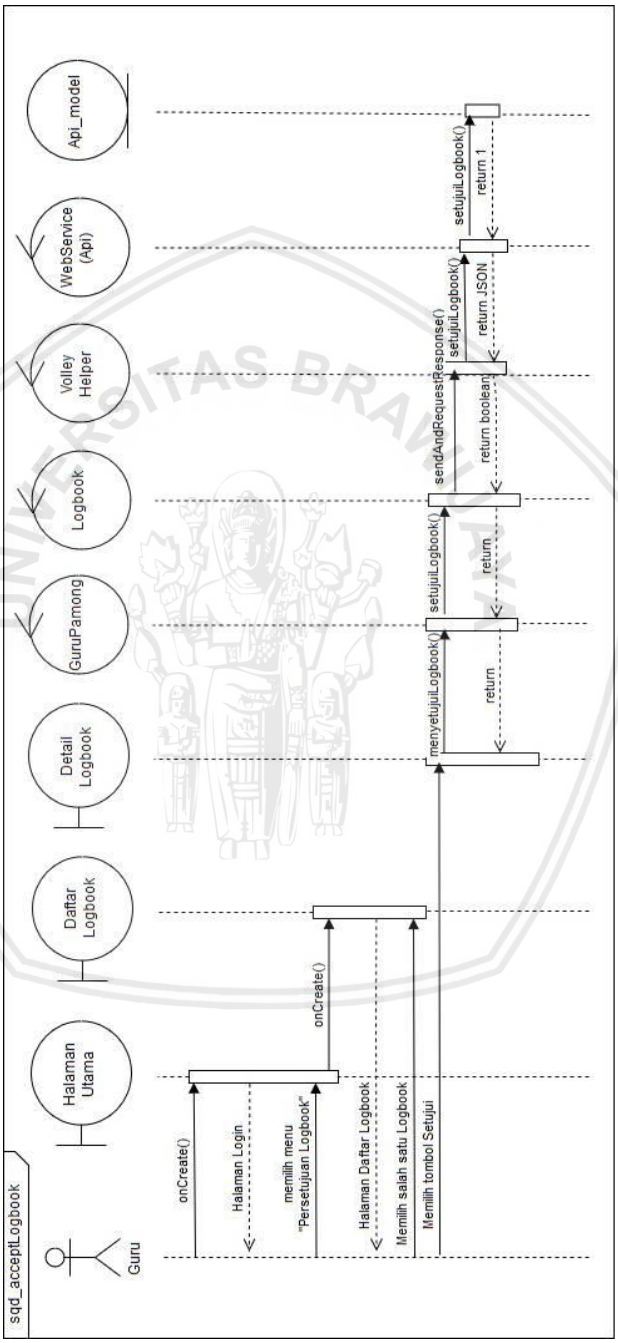


Gambar 4.29 *Sequence Diagram* menambahkan *logbook* PPL ke dalam sistem

Ketika mahasiswa akan menambahkan *logbook,* pertama sistem akan memanggil *method onCreate*() pada halaman *home* untuk menampilkan halaman *home.* Lalu mahasiswa harus memilih menu "*Logbook".* Sistem ketika ditekan menu "*Logbook"* akan memanggil halaman DaftarLogbook dengan cara memanggil *method* milik DaftarLogbook yaitu *onCreate*(). Didalam halaman ini akan dimunculkan semua daftar *logbook* yang telah dibuat oleh mahasiswa. Selanjutnya mahasiswa untuk menambah *logbook* harus menekan tombol "Tambah". Ketika tombol "Tambah" ditekan maka akan dipanggil *method onCreate*() milik TambahLogbook untuk menampilkan halaman tersebut. Selanjutnya mahasiswa harus mengisi isian *logbook* yang telah disediakan. Ketika telah selesai mengisi *logbook,* mahasiswa harus menekan tombol "Simpan". Lalu sistem akan menyimpan logbook dengan bantuan *method sendAndRequestResponse*() milik *class* VolleyHelper untuk melakukan *request* API ke *web server.*

5. *Sequence Diagram* menyetujui *logbook* harian yang telah dibuat oleh mahasiswa

Sequence Diagram pada Gambar 4.30 merepresentasikan proses persetujuan *logbook* harian yang telah dibuat oleh mahasiswa

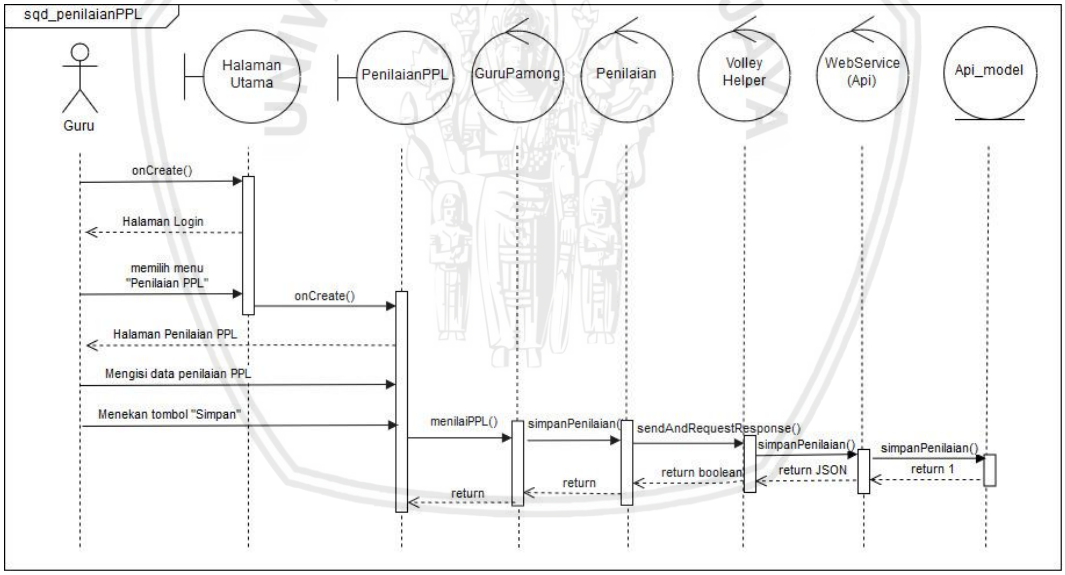


Gambar 4.30 *Sequence Diagram* proses menambahkan dan menyimpan *logbook* harian PPL ke dalam sistem

Mahasiswa yang telah membuat *logbook* akan tersimpan didalam sistem dan guru akan menerima *logbook* mahasiswa untuk disetujui didalam menu "Persetujuan Logbook". Ketika guru menekan tombol "Setuju" pada salah satu *logbook* mahasiswa, maka sistem akan mengirimkan *request update* data *logbook* mahasiswa menggunakan method *sendAndRequestResponse()* milik *class VolleyHelper* ke *web server.* Selanjutnya *logbook* mahasiswa statusnya akan berubah menjadi telah disetujui oleh guru pamong.

6. *Sequence Diagram* penilaian PPL mahasiswa oleh guru pamong

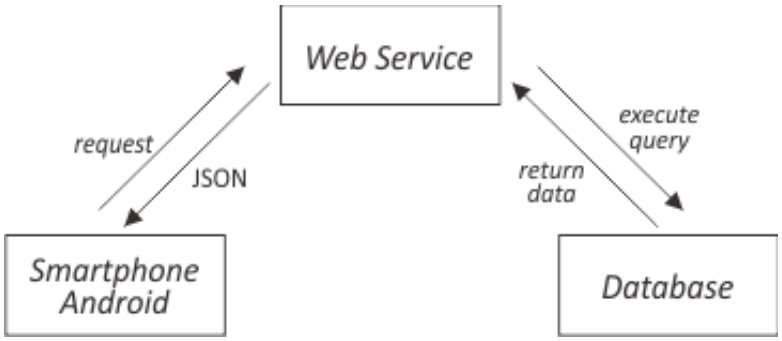
   Sequence Diagram pada Gambar 4.31 merepresentasikan proses penilaian kegiatan PPL mahasiswa oleh guru pamong. Ketika guru pamong akan memberikan penilaian terhadap kegiatan PPL yang dilakukan oleh mahasiswa, pertama guru pamong harus memilih menu "Penilaian PPL". Lalu sistem akan memanggil menu *onCreate()* yang dimiliki oleh *class* daftaLogbook untuk menampilkan halaman daftar *logbook.* Selanjutnya ketika guru telah selesai mengisi data penilaian dan menekan tombol Simpan, sistem akan melakukan *request* API ke *web server* menggunakan *method sendAndRequestResponse()* milik *class VolleyHelper.*



Gambar 4.31 *Sequence Diagram* proses penilaian kegiatan PPL mahasiswa oleh guru pamong

4.6.2 Perancangan Arsitektur Sistem

Pada perancangan arsitektur sistem akan digambarkan mengenai bagaimana alur sistem berjalan dari sisi *client* sampai sisi *server*. Penggunaan *web service* berfungsi untuk menerima dan memberi data yang dibutuhkan oleh sistem yang digunakan pada sisi *client* yang menggunakan *platform* Android. Selanjutnya *web service* yang akan mengakses *database* pada *server. Web Service* yang dibangun nantinya menggunakan *framework* CodeIgniter. Gambar 4.32 menggambarkan arsitektur sistem informasi yang akan dibangun.



Gambar 4.32 Arsitektur Sistem Informasi PPL

Arsitektur sistem terdiri dari 3 komponen utama, yaitu komponen *smartphone android,* komponen *web service* dan komponen *database.* Pertama *smartphone android* melakukan *request* terhadap *web service* dengan tipe POST. Lalu selanjutnya *web service* akan menjalankan eksekusi *query* SQL ke *database.* Selanjutnya *database* akan memberikan hasil eksekusi *query* kepada *web service.* Lalu *web service* akan mengubah data hasil kembalian *database* kedalam format JSON yang selanjutnya dikirimkan ke *smartphone android client* yang melakukan *request.* Format JSON yang dikirimkan oleh *web service* dapat dilihat pada Gambar 4.33.


```
{
    "ppl_id": "54",
    "nama": "Rizky Wahyu Setiawan",
    "nim": "155150400111093",
    "prodi": "Pendidikan Teknologi Informasi",
    "sekolah_mitra": "SMK ARDJUNA 1",
    "id_sekolah_mitra": "1",
    "waktu_mulai": "2019-01-03",
    "waktu_selesai": "2019-01-04"
}
```

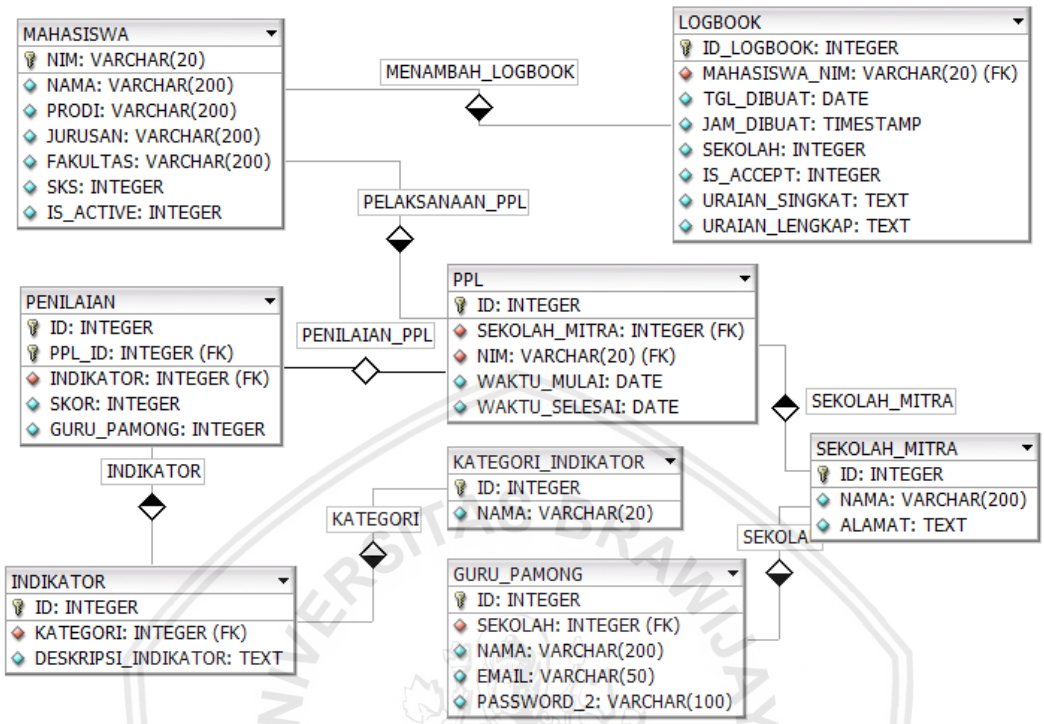

Gambar 4.33 Format JSON *Get* Data *Detail* PPL Mahasiswa

4.6.3 Perancangan Basis Data

Pada perancangan basis data, terdapat penjelasan mendetil mengenai struktur basis data untuk menyimpan data yang dibutuhkan sistem. Jenis basis data yang digunakan oleh sistem adalah *relational database*, dengan RDBMS MySQL. Dikarenakan berjenis *realtional database*, maka perancangan yang dibuat selain menampilkan struktur basis data berupa tabel juga menampilkan hubungan suatu tabel dengan tabel yang lain. Gambar 4.32 menunjukkan struktur basis data yang digunakan didalam sistem,

Gambar 4.34 menunjukkan hubungan antar tabel didalam sistem yang saling terhubung. Tabel Mahasiswa memiliki hubungan *one-to-many* terhadap tabel Logbook. Hal ini dikarenakan aktor mahasiswa dapat membuat logbook lebih dari satu. Lalu tabel Mahasiswa juga memiliki hubungan *one-to-one* terhadap tabel Penilaian. Hubungan ini disebabkan karena aktor mahasiswa hanya akan

mendapatkan satu penilaian dari guru pamong. Sedangkan guru hanya memberikan satu nilai pada satu mahasiswa.



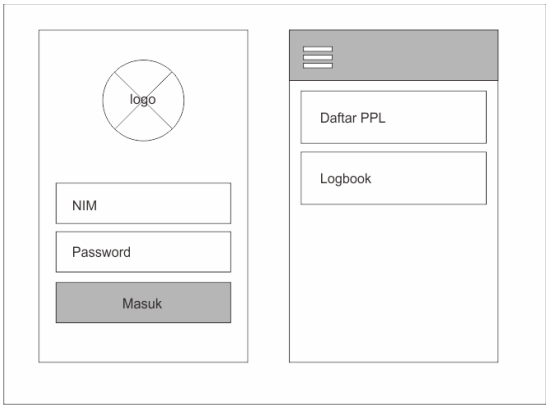
Gambar 4.34 Perancangan Struktur Basis Data Sistem

4.6.4 Perancangan Antarmuka

Pada tahap perancangan antarmuka terdapat rancangan antarmuka *low-fidelity* dari sistem yang akan dibangun. Penggunaan rancangan antarmuka mempermudah dalam memberikan gambaran umum sistem yang akan digunakan oleh aktor. Rancangan antarmuka dalam tahap ini mewakili seluruh proses yang akan digunakan oleh aktor.

4.6.4.1 Perancangan Antarmuka Proses Melakukan Login Ke Dalam Sistem

Rancangan antarmuka pada Gambar 4.35 merupakan rancangan halaman ketika aktor melakukan *login* dan masuk kedalam sistem.

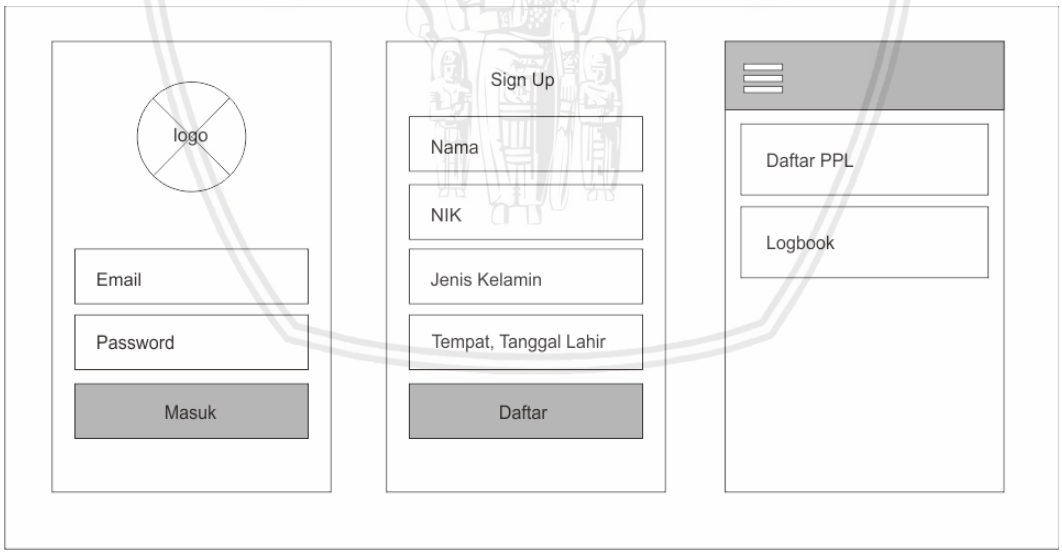


Gambar 4.35 Rancangan Antarmuka Proses Melakukan Login Ke Dalam Sistem

Gambar 4.35 adalah tampilan antarmuka ketika aktor melakukan login. Aktor dapat mengisikan kombinasi NIM/email dan passowrd pada field yang terdapat pada tampilan bagian kiri. Ketika aktor telah berhasil login maka tampilan akan diarahkan ke halaman *home* seperti pada tampilan dibagian kanan.

4.6.4.2 Perancangan Antarmuka Proses Mendaftarkan Diri (Sign Up) Sebagai Guru Pamong Ke Sistem

Rancangan antarmuka pada Gambar 4.36 merupakan rancangan halaman ketika aktor Mendaftarkan Diri (*Sign Up*) Sebagai Guru Pamong Ke Sistem.



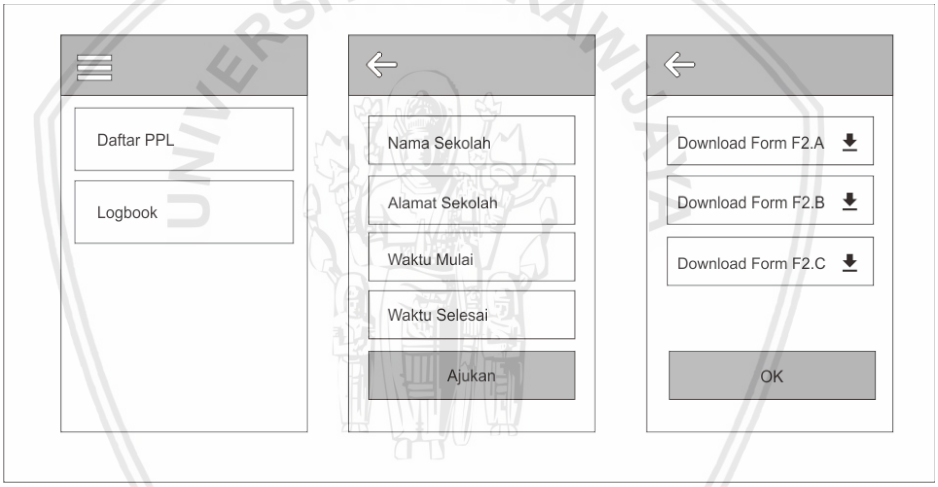
Gambar 4.36 Rancangan Antarmuka Proses Mendaftarkan Diri (*Sign Up*) Sebagai Guru Pamong Ke Sistem

Gambar 4.36 adalah tampilan antarmuka ketika aktor melakukan *Sign Up*. Aktor dapat mengisikan data diri dan data akun pada field yang terdapat pada tampilan bagian tengah. Lalu ketika aktor telah selesai mengisi data dan menekan

tombol daftar, tampilan akan diarahkan ke halaman *home* seperti pada tampilan dibagian kanan.

4.6.4.3 Perancangan Antarmuka Proses Mendaftarkan Diri Melaksanakan Kegiatan PPL

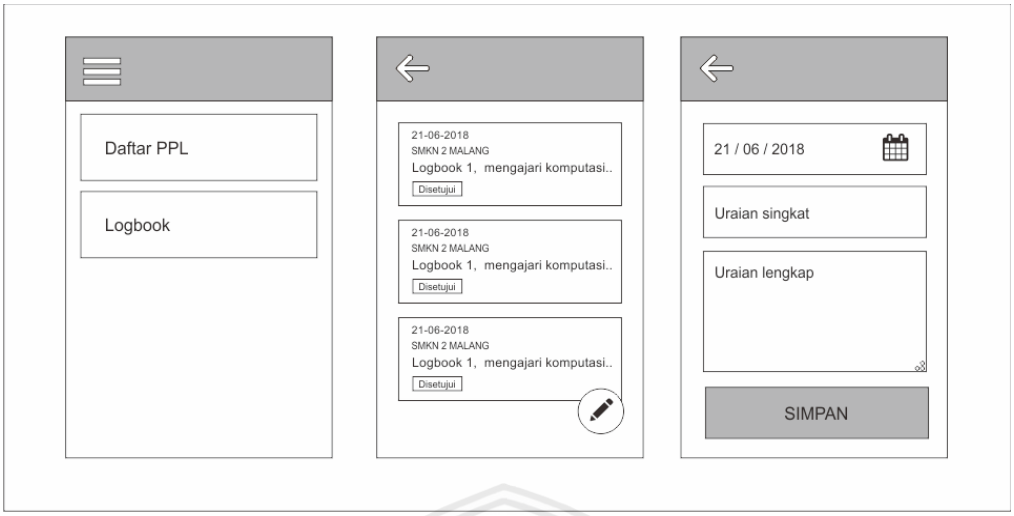
Rancangan antarmuka pada Gambar 4.37 merupakan rancangan halaman ketika mahasiswa mendaftarkan diri melaksanakan kegiatan PPL. Gambar 4.37 merupakan rancangan antarmuka proses mahasiswa mendaftarkan diri melaksanakan kegiatan PPL. Pertama mahasiswa memilih menu “Daftar PPL” lalu mahasiswa mengisi data sekolah mitra dan data kelompok PPL nya kedalam sistem. Ketika telah selesai disimpan maka sistem akan memberikan form F2A, F2B dan F2C yang nanti ditujukan kepada sekolah mitra. Selanjutnya mahasiswa dinyatakan telah resmi memasuki fase pelaksanaan PPL ketika Form F2C dibalas oleh sekolah mitra dan dikembalikan kepada akademik jurusan melalui mahasiswa yang terkait.



Gambar 4.37 Rancangan Antarmuka Proses Mendaftarkan Diri Melaksanakan Kegiatan PPL

4.6.4.4 Perancangan Antarmuka Proses Menambahkan Dan Menyimpan Logbook Harian PPL Ke Dalam Sistem

Rancangan antarmuka pada Gambar 4.38 merupakan rancangan halaman ketika mahasiswa menambahkan dan menyimpan logbook harian ppl ke dalam sistem.

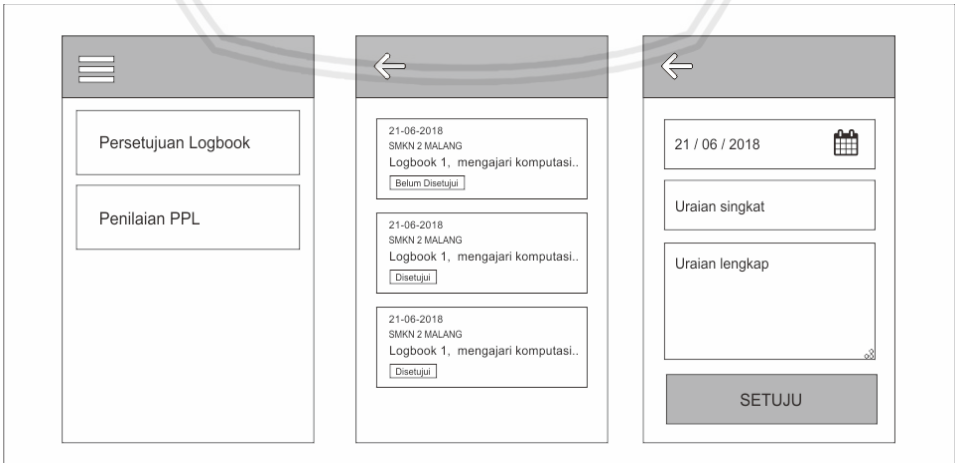


Gambar 4.38 Rancangan Antarmuka Proses Menambahkan dan Menyimpan *Logbook* Harian PPL Ke Dalam Sistem

Gambar 4.38 merupakan rancangan antarmuka proses menambahkan dan menyimpan *logbook* PPL ke dalam sistem yang dilakukan oleh mahasiswa. Pertama mahasiswa harus menekan menu *Logbook*. Lalu sistem menampilkan semua daftar *logbook* yang telah dibuat sebelumnya. Untuk menambahkan *logbook* baru, tekan tombol Tambah yang diwakili dengna simbol *pencil*. Setelah itu aka muncul form untuk menambahkan *logbook* baru. Setelah terisi semua, maka tekan tombol simpan untuk menyimpan data *logbook* yang baru dibuat seperti terlihat pada gambar paling kanan.

4.6.4.5 Perancangan Antarmuka Proses Menyetujui Logbook Harian Yang Telah Dibuat Oleh Mahasiswa.

Rancangan antarmuka pada Gambar 4.39 merupakan rancangan halaman ketika guru pamong menyetujui *logbook* harian yang telah diibuat oleh mahasiswa.

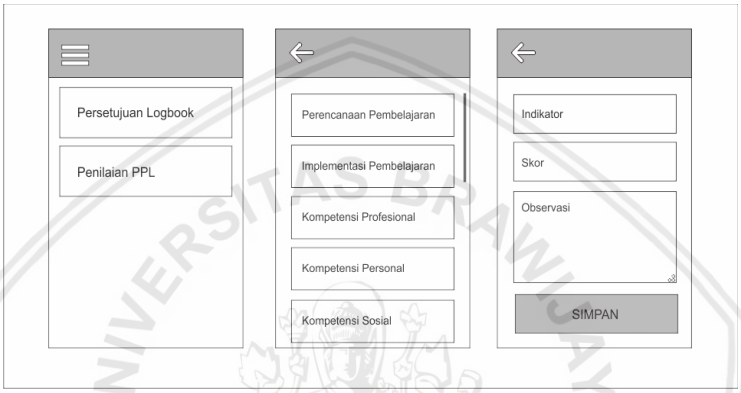


Gambar 4.39 Rancangan Antarmuka Proses Menyetujui *Logbook* Harian yang Telah Dibuat Oleh Mahasiswa

Gambar 4.39 diatas merupakan rancang antarmuka proses persetujuan *logbook* mahasiswa oleh guru pamong sekolah mitra. Pertama guru pamong yang telah *login* memilih menu "Persetujuan *Logbook*". Lalu sistem akan menampilkan semua daftar *logbook* yang telah dibuat oleh mahasiswa. Guru dapat menyetujui *logbook* tersebut dengan menekan tombol setuju pada *logbook* yang dipilih.

4.6.4.6 Perancangan Antarmuka Proses Penilaian PPL Mahasiswa Oleh Guru Pamong

Rancangan antarmuka pada Gambar 4.40 merupakan rancangan halaman ketika guru pamong memberikan penilaian terhadap kegiatan PPL yang dilakukan oleh mahasiswa.



Gambar 4.40 Rancangan Antarmuka Penilaian PPL Oleh Guru Pamong

Gambar 4.40 merupakan rancang antarmuka proses penilaian PPL mahasiswa oleh guru pamong di sekolah mitra. Pertama guru harus login untuk sampai pada halaman *Home*. Ketika sudah pada halaman *home,* guru menekan menu Penilaian PPL untuk masuk pada halaman pengisian form. Ketika telah semua data penilaian terisi, guru pamong harus menekan tombol simpan untuk menyimpan semua data penilaian.

4.7 Implementasi Sistem

Pada bagian ini menjelaskan mengenai implementasi yang dilakukan penulis untuk membangun Sistem Informasi Praktik Pengalaman Lapangan. Pembahasan implementasi sistem terdiri dari spesifikasi lingkungan implementasi, batasan implementasi, implementasi basis data yang digunakan sistem, implementasi kode program sistem, dan implementasi antarmuka sstem.

4.7.1 Spesifikasi Lingkungan Implementasi

Implementasi sistem ini memerlukan perangkat keras dan perangkat lunak yang dapat meng-*compile* kode java serta dapat menjalankan sistem pada perangkat untuk proses *debugging*. Untuk menulis kode program penulis menggunakan komputer dan untuk menjalankan sistem penulis menggunakan perangkat bergerak *smartphone* bersistem operasi Android. Berikut rincian spesifikasi perangkat keras dan perangkat lunak yang penulis gunakan selama proses pembangunan sistem.

4.7.1.1 Spesifikasi Perangkat Keras

Dalam membangun sistem ini, penulis menggunakan komputer dengan spesifikasi perangkat keras yang tertulis pada Tabel 4.14,

Tabel 4.14 Spesifikasi Perangkat Keras Komputer

Nama Komponen	**Spesifikasi**
Model Sistem	Laptop ASUS X455DG-WX027D
Processor	AMD A10-8700P Radeon R6, 10 Compute Cores 4C+6G (4 CPUs), ~1.8GHz
Storage (HDD)	1 TB HDD
Memory (RAM)	4096MB RAM
Graphic Card	AMD Radeon(TM) R6 Graphics

Lalu spesifikasi perangkat keras *smartphone* Android yang digunakan untuk proses *debug*, instalasi dan pengujian sistem penulis menggunakan *smartphone* dengan spesifikasi yang tertulis pada Tabel 4.15,

Tabel 4.15 Spesifikasi Perangkat Keras *Smartphone*

Nama Komponen	**Spesifikasi**
Model Sistem	Samsung Galaxy A5 2016
Processor	Exynos 7580 Octa-core 1.6 GHz Cortex-A53
Storage (Internal)	16GB
Storage (Memory Card)	4GB
Memory (RAM)	2GB RAM
Layar	Super AMOLED 5.2 inci (1.920 x 1.080 piksel)
WLAN	Wi-Fi 802.11 a/b/g/n, dual-band, WiFi Direct, hotspot

4.7.1.2 Spesifikasi Perangkat Lunak

Dalam membangun sistem ini, penulis menggunakan komputer dengan spesifikasi perangkat lunak yang tertulis pada Tabel 4.16

Tabel 4.16 Spesifikasi Perangkat Lunak Komputer

Nama Komponen	Spesifikasi
Sistem Operasi	Windows 10 Pro 64-bit (10.0, Build 16299)
Bahasa Pemrogaman	Java, XML
Software Development Kit	Java SE Development Kit 8 (64 bit)
Programming Environment	Java Runtime Environment 8
Android SDK	API 16
Editor	Android Studio

Spesifikasi perangkat lunak *smartphone* Android yang digunakan sebagai proses *debug,* instalasi dan pengujian tertulis pada Tabel 4.17,

Tabel 4.17 Spesifikasi Perangkat Lunak *Smartphone*

Nama Komponen	Spesifikasi
Sistem Operasi	Android 7.0 (Nougat)

4.7.2 Batasan Implementasi

Beberapa batasan dalam mengimplementasikan sistem adalah sebagai berikut :

1. Sistem Informasi harus selalu terhubung ke internet sebagai media pertukaran data dengan *web service*.
2. Implementasi *web service* di-*hosting* pada *free server hosting* yang memiliki batasan ruang penyimpanan dan batasan *life-time.*
3. Sistem hanya dapat digunakan pada lingkungan sistem operasi Android

4.7.3 Implementasi Basis Data

Implementasi basis data yang telah dirancang diimplementasikan pada *web service* dengan menggunakan RDBMS MySQL. Implementasi query basis data dapat ditunjukkan pada Tabel 4.18

Tabel 4.18 Implementasi / *Query Create* Database Sistem

```
1.  CREATE TABLE KATEGORI_INDIKATOR (
2.    ID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
3.    NAMA VARCHAR(20) NULL,
4.    PRIMARY KEY(ID)
5.  );
6.
7.  CREATE TABLE MAHASISWA (
8.    NIM VARCHAR(20) NOT NULL,
```

```
  NAMA VARCHAR(200) NULL,
  PRODI VARCHAR(200) NULL,
  JURUSAN VARCHAR(200) NULL,
  FAKULTAS VARCHAR(200) NULL,
  SKS INTEGER UNSIGNED NULL,
  IS_ACTIVE INTEGER UNSIGNED NULL,
  PRIMARY KEY(NIM)
);

CREATE TABLE SEKOLAH_MITRA (
  ID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  NAMA VARCHAR(200) NULL,
  ALAMAT TEXT NULL,
  PRIMARY KEY(ID)
);

CREATE TABLE LOGBOOK (
  ID_LOGBOOK INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  MAHASISWA_NIM VARCHAR(20) NOT NULL,
  TGL_DIBUAT DATE NULL,
  JAM_DIBUAT TIMESTAMP NULL,
  SEKOLAH INTEGER UNSIGNED NULL,
  IS_ACCEPT INTEGER UNSIGNED NULL,
  URAIAN_SINGKAT TEXT NULL,
  URAIAN_LENGKAP TEXT NULL,
  PRIMARY KEY(ID_LOGBOOK),
  INDEX LOGBOOK_FKIndex1(MAHASISWA_NIM),
  FOREIGN KEY(MAHASISWA_NIM)
    REFERENCES MAHASISWA(NIM)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
);

CREATE TABLE GURU_PAMONG (
  ID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  SEKOLAH INTEGER UNSIGNED NOT NULL,
  NAMA VARCHAR(200) NULL,
  EMAIL VARCHAR(50) NULL,
  PASSWORD_2 VARCHAR(100) NULL,
  PRIMARY KEY(ID),
  INDEX GURU_PAMONG_FKIndex1(SEKOLAH),
  FOREIGN KEY(SEKOLAH)
    REFERENCES SEKOLAH_MITRA(ID)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
);

CREATE TABLE INDIKATOR (
  ID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  KATEGORI INTEGER UNSIGNED NOT NULL,
  DESKRIPSI_INDIKATOR TEXT NULL,
  PRIMARY KEY(ID),
  INDEX INDIKATOR_FKIndex1(KATEGORI),
  FOREIGN KEY(KATEGORI)
    REFERENCES KATEGORI_INDIKATOR(ID)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
);

CREATE TABLE PPL (
  ID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  SEKOLAH_MITRA INTEGER UNSIGNED NOT NULL,
  NIM VARCHAR(20) NOT NULL,
  WAKTU_MULAI DATE NULL,
  WAKTU_SELESAI DATE NULL,
```

```
74.     PRIMARY KEY(ID),
75.     INDEX PPL_FKIndex1(NIM),
76.     INDEX PPL_FKIndex3(SEKOLAH_MITRA),
77.     FOREIGN KEY(NIM)
78.       REFERENCES MAHASISWA(NIM)
79.         ON DELETE NO ACTION
80.         ON UPDATE NO ACTION,
81.     FOREIGN KEY(SEKOLAH_MITRA)
82.       REFERENCES SEKOLAH_MITRA(ID)
83.         ON DELETE NO ACTION
84.         ON UPDATE NO ACTION
85.   );
86.
87.   CREATE TABLE PENILAIAN (
88.     ID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
89.     PPL_ID INTEGER UNSIGNED NOT NULL,
90.     INDIKATOR INTEGER UNSIGNED NOT NULL,
91.     SKOR INTEGER UNSIGNED NULL,
92.     GURU_PAMONG INTEGER UNSIGNED NULL,
93.     PRIMARY KEY(ID, PPL_ID),
94.     INDEX PENILAIAN_FKIndex1(INDIKATOR),
95.     INDEX PENILAIAN_FKIndex2(PPL_ID),
96.     FOREIGN KEY(INDIKATOR)
97.       REFERENCES INDIKATOR(ID)
98.         ON DELETE NO ACTION
99.         ON UPDATE NO ACTION,
100.    FOREIGN KEY(PPL_ID)
101.      REFERENCES PPL(ID)
102.        ON DELETE NO ACTION
103.        ON UPDATE NO ACTION
104.  );
```

Penjelasan *query* Kode 4.18 adalah sebagai berikut :

1. Baris 1 – 23 terlihat proses *create table* yang pertama dilakukan adalah dengan mendahulukan *table child* yang tidak memiliki *foreign key.* Pada rancangan database, *table* yang tidak memiliki *foreign key* adalah *table* kategori_indikator, mahasiswa, dan sekolah_mitra.
2. Baris 25 – 104 merupakan proses *create table* yang memiliki *foreign key* dan tentunya juga memiliki relasi dengan tabel yang lain dengan menggunakan *foreign key* ini. Alasan mendahulukan *table-table* yang tidak memiliki *foreign key* adalah untuk menghindari *error* sebab ketika *query create table* dengan *foreign key* dijalankan maka dia akan mencari *key* yang dia butuhkan pada semua *table* yang telah ada. Ketika *key* tidak ditemukan maka *table* dengan *foreign key* gagal dibuat.

4.7.4 Implementasi Kode Program Sistem

Pada tahap Implementasi kode program ini terdapat proses penulisan kode program berdasarkan fitur utama yang terdapat pada Sistem Informasi Praktik Pengalaman Lapangan yang diberi nama SIPPL. Terdapat 4 fitur utama pada sistem ini yang dibagi menjadi 2 sisi, yaitu sisi Mahasiswa dan sisi Guru Pamong. Fitur yang terdapat pada sisi Mahasiswa adalah fitur Pendaftaran PPL dan fitur Pencatatan Logbook. Sedangkan pada sisi Guru Pamong, terdapat fitur

Persetujuan Logbook dan fitur Penilaian Mahasiswa. Implementasi kode program ini berdasarkan rancangan *class diagram* dan *sequence diagram* pada sub-bab sebelumnya. Kode program ditulis menggunakan bahasa pemrogaman Java dan XML. Pada subbab selanjutnya akan diperlihatkan potongan *source code* beserta penjelasan pada setiap fitur utama SIPPL.

4.7.4.1 Kode Program Fitur *Login*

Potongan kode program pada Tabel 4.19 merupakan potongan kode program untuk menjalankan fitur *Login*. Pertama pengguna membuka aplikasi SIPPL pada *smartphone.* Lalu sistem akan menampilkan halaman *login* untuk mahasiswa. Jika ingin *login* sebagai guru pamong maka ketuk tulisan masuk sebagai guru pamong.

Tabel 4.19 Kode Program Fitur *Login*

```
 1.  public class Mahasiswa extends AppCompatActivity {
 2.      private static final String URL_LOGIN =
 3.  "http://arpulsabizz.000webhostapp.com/api/trans_data/cek_mahasis
 4.  wa_is_active";
 5.      ProgressDialog progressDialog;
 6.      private static final String TAG = "Login";
 7.      private RequestQueue mRequestQueue;
 8.      private StringRequest mStringRequest;
 9.      EditText et_nim_login, et_pwd_login;
10.      SharedPreferences sharedPreferences;
11.      public static final String myPreference = "cache_sippl";
12.      @Override
13.      protected void onCreate(Bundle savedInstanceState) {
14.          super.onCreate(savedInstanceState);
15.          requestWindowFeature(Window.FEATURE_NO_TITLE);
16.  getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
17.  WindowManager.LayoutParams.FLAG_FULLSCREEN);
18.          setContentView(R.layout.activity_login);
19.          sharedPreferences = getSharedPreferences(myPreference,
20.  Context.MODE_PRIVATE);
21.          if(sharedPreferences.contains("role")){
22.              Intent i = new Intent(getApplicationContext(),
23.  HalamanUtama.class);
24.              startActivity(i);
25.              finish();        }
26.          progressDialog = new ProgressDialog(this);
27.          et_nim_login = findViewById(R.id.et_nim_login);
28.          et_pwd_login =  findViewById(R.id.et_password_login);
29.      }
30.      public void keLoginDosenGuru(View v){
31.          Intent i = new Intent(getApplicationContext(),
32.  LoginGuru.class);
33.          startActivity(i);
34.          finish();
35.      }
36.      public void MasukMahasiswa(View v){
37.          String nim = et_nim_login.getText().toString();
38.          String password = et_pwd_login.getText().toString();
39.          login(nim, password);
40.      }
41.      private void login(final String nim, final String password)
42.  {
43.          progressDialog.setMessage("Loading...");
44.          progressDialog.show();
45.          //RequestQueue initialized
46.          mRequestQueue = Volley.newRequestQueue(this);
47.          //String Request initialized
```

```
48.         mStringRequest = new StringRequest(Request.Method.POST,
49.  URL_LOGIN, new Response.Listener<String>() {
50.              @Override
51.              public void onResponse(String response) {
52.                  progressDialog.hide();
53.                  try {
54.                      JSONObject obj = new JSONObject(response);
55.                      String nama = obj.getString("NAMA");
56.                      String nim = obj.getString("NIM");
57.                      String jurusan = obj.getString("JURUSAN");
58.                      String prodi = obj.getString("PRODI");
59.                      String fakultas = obj.getString("FAKULTAS");
60.                      int sks = obj.getInt("SKS");
61.                      int is_active = obj.getInt("IS_ACTIVE");
62.                     SharedPreferences.Editor editor =
63.  sharedPreferences.edit();
64.                      editor.putString("nama", nama);
65.                      editor.putString("nim", nim);
66.                      editor.putString("jurusan", jurusan);
67.                      editor.putString("prodi", prodi);
68.                      editor.putString("fakultas", fakultas);
69.                      editor.putInt("sks", sks);
70.                      editor.putInt("is_active", is_active);
71.                      editor.putInt("role", 1); //role 1 for
72.  mahasiswa ; 2 is guru pamong
73.                      editor.commit();
74.                      Intent i = new
75.  Intent(getApplicationContext(), HalamanUtama.class);
76.                      startActivity(i);
77.                      finish();
78.                  } catch (JSONException e) {
79.                      Toast.makeText(Login.this, "terjadi
80.  kesalahan pada data", Toast.LENGTH_SHORT).show();
81.                      e.printStackTrace();
82.                  }
83.              }
84.          }, new Response.ErrorListener() {
85.              @Override
86.              public void onErrorResponse(VolleyError error) {
87.                  progressDialog.hide();
88.                  Toast.makeText(Login.this, "nim atau password
89.  anda salah", Toast.LENGTH_SHORT).show();
90.                  Log.i(TAG,"Error :" + error.toString());
91.              }
92.          })
93.          {
94.              @Override
95.              public byte[] getBody() throws AuthFailureError {
96.                  HashMap<String, String> params2 = new
97.  HashMap<String, String>();
98.                  params2.put("nim", nim);
99.                  return new
100. JSONObject(params2).toString().getBytes();
101.             }
102.             @Override
103.             public String getBodyContentType() {
104.                 return "application/json";
105.             }
106.         };
107.         ;
108.        mRequestQueue.add(mStringRequest);  }}
```

Penjelasan Tabel Kode 4.19 adalah sebagai berikut :

1. Baris 13 - 31 merupakan *method onCreate()* yang berfungsi mirip dengan *Constructor,* yaitu sebagai *method* yang pertama dipanggil oleh program dan didalamnya terdapat instansiasi komponen-komponen yang terdapat pada XML bedasarkan id nya serta terdapat pengecekan jika pengguna telah login sebelumnya maka langsung di-*redirect* ke halaman utama.
2. Baris 32 – 36 merupakan *method* yang digunakan untuk menangani *event* ketika pengguna menekan tulisan Masuk Sebagai Guru Pamong. Maka sistem akan menuju halaman login guru dengan menggunakan Intent.
3. Baris 38 – 108 merupakan *method* untuk menangani *event* ketika pengguna mahasiswa menekan tombol Masuk. Pertama sistem akan mengambil data NIM dan *password.* Selanjutnya sistem akan melakukan *request* API ke *web services* untuk mengecek apakah kombinasi NIM dan *password* telah sesuai. Jika sesuai maka data kembalian dari *web service* berupa JSON akan disimpan setiap *value*-nya kedalam *SharedPreferences* untuk digunakan nantinya dan berlanjut sistem akan mengarahkan halaman ke halaman utama. Jika kombinasi tidak sesuai maka akan muncul *Toast* dengan isi pesan "nim atau *password* anda salah".

4.7.4.2 Kode Program Fitur *SignUp* Guru Pamong

Potongan kode program pada Tabel Kode 4.20 merupakan potongan kode program untuk menjalankan fitur *SignUp*. Pertama pengguna membuka aplikasi SIPPL pada *smartphone.* Lalu sistem akan menampilkan halaman *login* untuk mahasiswa. Lalu masuk ke halaman *Login* guru pamong. Selanjutnya masuk ke halaman *SignUp* dengan menekan tulisan "Buat Akun Guru Pamong".

Tabel 4.20 Kode Program Fitur *SignUp* Guru Pamong

```
1.  public class GuruPamong extends AppCompatActivity {
2.      private static final String URL_GET_SEKOLAH =
3.  "http://arpulsabizz.000webhostapp.com/api/trans_data/get_sekolah
4.  _mitra";
5.      private static final String URL_SIGNUP_GURU =
6.  "http://arpulsabizz.000webhostapp.com/api/trans_data/sign_up_gur
7.  u";
8.      ProgressDialog progressDialog;
9.      private static final String TAG = "SignUpGuru";
10.     private RequestQueue mRequestQueue;
11.     private StringRequest mStringRequest;
12.     Spinner dynamicSpinner;
13.     String[] nama_sekolah;
14.     EditText et_nama_signup, et_email_signup,
15. et_password_signup_dosen_guru;
16.     @Override
17.     protected void onCreate(Bundle savedInstanceState) {
18.         super.onCreate(savedInstanceState);
19.         requestWindowFeature(Window.FEATURE_NO_TITLE);
20. getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
21. WindowManager.LayoutParams.FLAG_FULLSCREEN);
22.         setContentView(R.layout.activity_sign_up_guru);
23.         progressDialog = new ProgressDialog(this);
24.         dynamicSpinner = (Spinner)
25. findViewById(R.id.sp_instansi_signup);
26.         et_nama_signup = findViewById(R.id.et_nama_signup);
```

```
        et_email_signup = findViewById(R.id.et_email_signup);
        et_password_signup_dosen_guru =
findViewById(R.id.et_password_signup_dosen_guru);
        sendAndRequestResponse();
    }
    public void SignUpGuru(View v){
        int sekolah =
dynamicSpinner.getSelectedItemPosition()+1;
        String nama = et_nama_signup.getText().toString();
        String email = et_email_signup.getText().toString();
        String password =
et_password_signup_dosen_guru.getText().toString();
        password = MD5(password);
        signUpGuruPamong(nama, sekolah, email, password);
    }
    public String MD5(String md5) {
        try {
            java.security.MessageDigest md =
java.security.MessageDigest.getInstance("MD5");
            byte[] array = md.digest(md5.getBytes());
            StringBuffer sb = new StringBuffer();
            for (int i = 0; i < array.length; ++i) {
                sb.append(Integer.toHexString((array[i] & 0xFF)
| 0x100).substring(1,3));
            }
            return sb.toString();
        } catch (java.security.NoSuchAlgorithmException e) {
        }
        return null;
    }
    private void sendAndRequestResponse() {
        //RequestQueue initialized
        mRequestQueue = Volley.newRequestQueue(this);
        //String Request initialized
        mStringRequest = new StringRequest(Request.Method.POST,
URL_GET_SEKOLAH, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    JSONArray respon_array = new
JSONArray(response);
                    nama_sekolah = new
String[respon_array.length()];
                    JSONObject respon_obj ;
                    for (int i = 0; i < respon_array.length() ;
i++) {
                        String obj =
respon_array.get(i).toString();
                        respon_obj = new JSONObject(obj);
                        nama_sekolah[i] =
respon_obj.getString("NAMA");
                    }
                    ArrayAdapter<String> adapter = new
ArrayAdapter<String>(getApplicationContext(),
android.R.layout.simple_spinner_item, nama_sekolah);
                   dynamicSpinner.setAdapter(adapter);
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(SignUpGuru.this, "Jaringan
terputus", Toast.LENGTH_SHORT).show();
```

```
92.                 Log.i(TAG,"Error :" + error.toString());
93.             }
94.         })
95.         {
96.             @Override
97.             public String getBodyContentType() {
98.                 return "application/json";
99.             }
100.        };
101.        mRequestQueue.add(mStringRequest);
102.    }
103.    private void signUpGuruPamong(final String nama, final int
104. sekolah, final String email, String password) {
105.        progressDialog.setMessage("Loading...");
106.        progressDialog.show();
107.        JSONObject jsonBody = new JSONObject();
108.        try {
109.            jsonBody.put("nama", nama);
110.            jsonBody.put("instansi", sekolah);
111.            jsonBody.put("email", email);
112.            jsonBody.put("password", password);
113.        } catch (JSONException e) {
114.            e.printStackTrace();
115.        }
116.        final String mRequestBody = jsonBody.toString();
117.        //RequestQueue initialized
118.        mRequestQueue = Volley.newRequestQueue(this);
119.        //String Request initialized
120.        mStringRequest = new StringRequest(Request.Method.POST,
121. URL_SIGNUP_GURU, new Response.Listener<String>() {
122.            @Override
123.            public void onResponse(String response) {
124.                Intent i = new Intent(getApplicationContext(),
125. LoginGuru.class);
126.                startActivity(i);
127.                finish();
128.            }
129.        }, new Response.ErrorListener() {
130.            @Override
131.            public void onErrorResponse(VolleyError error) {
132.                progressDialog.hide();
133.                Toast.makeText(SignUpGuru.this, "Jaringan
134. terputus", Toast.LENGTH_SHORT).show();
135.                Log.i(TAG,"Error :" + error.toString());
136.            }
137.        })
138.        {
139.            @Override
140.            public byte[] getBody() throws AuthFailureError {
141. //                HashMap<String, String> params2 = new
142. HashMap<String, String>();
143.                try{
144.                    return mRequestBody == null ? null :
145. mRequestBody.getBytes("utf-8");
146.                }catch (UnsupportedEncodingException uee){
147.        VolleyLog.wtf("Encoding tidak terdukung -Rizky");
148.                    return null;
149.                }
150.            }
151.            @Override
152.            public String getBodyContentType() {
153.                return "application/json";
154.            }
155.        };
156.        mRequestQueue.add(mStringRequest);
```

```
157.     }
158. }
159.
160.
161.
162.
```

Penjelasan Kode 4.20 adalah sebagai berikut :

1. Baris 17 - 31 merupakan *method onCreate()* yang berfungsi mirip dengan *Constructor,* yaitu sebagai *method* yang pertama dipanggil oleh program dan didalamnya terdapat instansiasi komponen-komponen yang terdapat pada XML bedasarkan id nya serta terdapat *request* ke *web service* untuk mendapatkan nama-nama sekolah SMK di Malang yang selanjutnya datanya akan dimasukkan ke dalam *Spinner.*
2. Baris 32 – 42 merupakan *method* yang digunakan untuk menangani *event* ketika pengguna menekan tombol Daftar. Pertama sistem akan mengambil data yang telah dimasukkan pengguna pada form *SignUp* lalu sistem akan melakukan *request* ke *web service* untuk mendaftarkan data guru pamong.
3. Baris 43 – 56 merupakan *method* untuk merubah data *password* menjadi *hash* md5 atau di enkripsi dengan metode md5.
4. Baris 58 – 160 merupakan *method* untuk melakukan *request* ke *web service* untuk mendapatkan nama sekolah dan untuk memasukkan data *signup* guru pamong ke database melalui API *web service.*

4.7.4.3 Kode Program Fitur Pendaftaran PPL

Potongan kode program pada Tabel Kode 4.21 merupakan potongan kode program untuk menjalankan fitur pendaftaran PPL pada sisi Mahasiswa. Pertama mahasiswa harus memilih sekolah yang ditelah ditentukan oleh Kaprodi lalu memasukkan tanggal mulai dan tanggal selesai sesuai dengan arahan guru pamong dan selanjutnya mahasiswa menekan tombol Ajukan untuk menyimpan pengajuan PPL.

Tabel 4.21 Kode Program Fitur Pendaftaran PPL

```
1.  public class Mahasiswa extends AppCompatActivity {
2.      final Calendar myCalendar = Calendar.getInstance();
3.      EditText editText_mulai, editText_selesai, et_alamat;
4.      Spinner dynamicSpinner;
5.      private static final String URL_LOGIN =
6.  "http://arpulsabizz.000webhostapp.com/api/trans_data/get_sekolah
7.  _mitra";
8.      private static final String URL_GET_ALAMAT_SEKOLAH =
9.  "http://arpulsabizz.000webhostapp.com/api/trans_data/get_sekolah
10. _mitra_alamat";
11.     private static final String URL_DAFTAR_PPL =
12. "http://arpulsabizz.000webhostapp.com/api/trans_data/create_pend
13. aftaran_PPL";
14.     ProgressDialog progressDialog;
15.     private static final String TAG = "DaftarPPL";
16.     private RequestQueue mRequestQueue;
17.     private StringRequest mStringRequest;
18.     String[] nama_sekolah;
19.     SharedPreferences sharedPreferences;
20.     public static final String myPreference = "cache_sippl";
21.     @Override
```

```
22.     protected void onCreate(Bundle savedInstanceState) {
23.         super.onCreate(savedInstanceState);
24.         setContentView(R.layout.activity_daftar_ppl);
25.         progressDialog = new ProgressDialog(this);
26.         editText_mulai = (EditText)
27. findViewById(R.id.et_waktu_mulai);
28.         editText_selesai = (EditText)
29. findViewById(R.id.et_waktu_selesai);
30.         dynamicSpinner = (Spinner)
31. findViewById(R.id.sp_nama_sekolah);
32.         et_alamat = findViewById(R.id.et_alamat_sekolah);
33.         sharedPreferences = getSharedPreferences(myPreference,
34. Context.MODE_PRIVATE);
35.         sendAndRequestResponse();
36.         final DatePickerDialog.OnDateSetListener date = new
37. DatePickerDialog.OnDateSetListener() {
38.             @Override
39.             public void onDateSet(DatePicker view, int year, int
40. monthOfYear,
41.                                   int dayOfMonth) {
42.                 // TODO Auto-generated method stub
43.                 myCalendar.set(Calendar.YEAR, year);
44.                 myCalendar.set(Calendar.MONTH, monthOfYear);
45.                 myCalendar.set(Calendar.DAY_OF_MONTH,
46. dayOfMonth);
47.                 updateLabel("mulai");
48.             }
49.         };
50.         final DatePickerDialog.OnDateSetListener date1 = new
51. DatePickerDialog.OnDateSetListener() {
52.             @Override
53.             public void onDateSet(DatePicker view, int year, int
54. monthOfYear,
55.                                   int dayOfMonth) {
56.                 // TODO Auto-generated method stub
57.                 myCalendar.set(Calendar.YEAR, year);
58.                 myCalendar.set(Calendar.MONTH, monthOfYear);
59.                 myCalendar.set(Calendar.DAY_OF_MONTH,
60. dayOfMonth);
61.                 updateLabel("selesai");
62.             }
63.         };
64.         editText_mulai.setOnClickListener(new
65. View.OnClickListener() {
66.             @Override
67.             public void onClick(View v) {
68.                 // TODO Auto-generated method stub
69.                 new DatePickerDialog(DaftarPPL.this, date,
70. myCalendar
71.                         .get(Calendar.YEAR),
72. myCalendar.get(Calendar.MONTH),
73.
74. myCalendar.get(Calendar.DAY_OF_MONTH)).show();
75.             }
76.         });
77.
78.         editText_selesai.setOnClickListener(new
79. View.OnClickListener() {
80.             @Override
81.             public void onClick(View v) {
82.                 // TODO Auto-generated method stub
83.                 new DatePickerDialog(DaftarPPL.this, date1,
84. myCalendar
85.                         .get(Calendar.YEAR),
86. myCalendar.get(Calendar.MONTH),
```

```
87.
88.  myCalendar.get(Calendar.DAY_OF_MONTH)).show();
89.              }
90.          });
91.      }
92.      private void updateLabel(String label) {
93.          String myFormat = "dd-MM-yyyy"; //In which you need put
94.  here
95.          SimpleDateFormat sdf = new SimpleDateFormat(myFormat,
96.  Locale.US);
97.          switch (label){
98.              case "mulai" :
99.  editText_mulai.setText(sdf.format(myCalendar.getTime()));
100.                 break;
101.             case "selesai" :
102. editText_selesai.setText(sdf.format(myCalendar.getTime()));
103.                 break;
104.         }
105.     }
106.
107.     private void sendAndRequestResponse() {
108.         progressDialog.setMessage("Loading...");
109.         progressDialog.show();
110.         //RequestQueue initialized
111.         mRequestQueue = Volley.newRequestQueue(this);
112.         //String Request initialized
113.         mStringRequest = new StringRequest(Request.Method.POST,
114. URL_LOGIN, new Response.Listener<String>() {
115.             @Override
116.             public void onResponse(String response) {
117.                 progressDialog.hide();
118.                 try {
119.                     JSONArray respon_array = new
120. JSONArray(response);
121.                     nama_sekolah = new
122. String[respon_array.length()];
123.                     JSONObject respon_obj ;
124.                     for (int i = 0; i < respon_array.length() ;
125. i++) {
126.                         String obj =
127. respon_array.get(i).toString();
128.                         respon_obj = new JSONObject(obj);
129.                         nama_sekolah[i] =
130. respon_obj.getString("NAMA");
131.                     }
132.                     ArrayAdapter<String> adapter = new
133. ArrayAdapter<String>(getApplicationContext(),
134. android.R.layout.simple_spinner_item, nama_sekolah);
135.                     dynamicSpinner.setAdapter(adapter);
136.                     dynamicSpinner.setOnItemSelectedListener(new
137. AdapterView.OnItemSelectedListener() {
138.                         @Override
139.                         public void
140. onItemSelected(AdapterView<?> parent, View view,
141.                                    int position, long id) {
142.                             sendAndRequestResponse1(position+1);
143.                         }
144.                         @Override
145.                         public void
146. onNothingSelected(AdapterView<?> parent) {
147.                             // TODO Auto-generated method stub
148.                         }
149.                     });
150.                 } catch (JSONException e) {
151.                     e.printStackTrace();
```

```
                }
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                progressDialog.hide();
                Toast.makeText(DaftarPPL.this, "Jaringan
terputus", Toast.LENGTH_SHORT).show();
                Log.i(TAG,"Error :" + error.toString());
            }
        })
        {
            @Override
            public String getBodyContentType() {
                return "application/json";
            }
        };
        mRequestQueue.add(mStringRequest);
    }
    private void sendAndRequestResponse1(final int id_sekolah) {
        JSONObject jsonBody = new JSONObject();
        try {
            jsonBody.put("id_sekolah_mitra", id_sekolah);
        } catch (JSONException e) {
            e.printStackTrace();
        }

        final String mRequestBody = jsonBody.toString();
        //RequestQueue initialized
        mRequestQueue = Volley.newRequestQueue(this);
        //String Request initialized
        mStringRequest = new StringRequest(Request.Method.POST,
URL_GET_ALAMAT_SEKOLAH, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    JSONObject obj = new JSONObject(response);
                    String alamat = obj.getString("alamat");
                    et_alamat.setText(alamat);

                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(DaftarPPL.this, "Jaringan
terputus", Toast.LENGTH_SHORT).show();
                Log.i(TAG,"Error :" + error.toString());
            }
        })
        {
            @Override
            public byte[] getBody() throws AuthFailureError {
                try{
                    return mRequestBody == null ? null :
mRequestBody.getBytes("utf-8");
                }catch (UnsupportedEncodingException uee){
                    VolleyLog.wtf("Encoding tidak terdukung -
Rizky");
                    return null;
                }
            }

```

```
            @Override
            public String getBodyContentType() {
                return "application/json";
            }
        };

        mRequestQueue.add(mStringRequest);
    }
    public void pendaftaranPPL(View v){
        String nim = sharedPreferences.getString("nim", "");
        int sekolah =
dynamicSpinner.getSelectedItemPosition()+1;
        String waktu_mulai =
editText_mulai.getText().toString();
        String[] array_temp = waktu_mulai.split("-");
        waktu_mulai = array_temp[2]+"-"+array_temp[1]+"-
"+array_temp[0];
        String waktu_selesai =
editText_selesai.getText().toString();
        array_temp = waktu_selesai.split("-");
        waktu_selesai = array_temp[2]+"-"+array_temp[1]+"-
"+array_temp[0];

        sendAndRequestResponse_daftar_ppl(nim, sekolah,
waktu_mulai, waktu_selesai);
    }
    private void sendAndRequestResponse_daftar_ppl(final String
nim, final int sekolah, final String waktu_mulai, String
waktu_selesai) {
        progressDialog.setMessage("Loading...");
        progressDialog.show();
        JSONObject jsonBody = new JSONObject();
        try {
            jsonBody.put("nim", nim);
            jsonBody.put("sekolah_mitra", sekolah);
            jsonBody.put("waktu_mulai", waktu_mulai);
            jsonBody.put("waktu_selesai", waktu_selesai);
        } catch (JSONException e) {
            e.printStackTrace();
        }

        final String mRequestBody = jsonBody.toString();

        //RequestQueue initialized
        mRequestQueue = Volley.newRequestQueue(this);

        //String Request initialized
        mStringRequest = new StringRequest(Request.Method.POST,
URL_DAFTAR_PPL, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                Intent i = new Intent(getApplicationContext(),
SelesaiDaftarPPL.class);
                startActivity(i);
                finish();
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                progressDialog.hide();
                Toast.makeText(DaftarPPL.this, "Jaringan
terputus", Toast.LENGTH_SHORT).show();
                Log.i(TAG,"Error :" + error.toString());
            }
        })
```

```
282.         {
283.             @Override
284.             public byte[] getBody() throws AuthFailureError {
285. //                 HashMap<String, String> params2 = new
286. HashMap<String, String>();
287.                 try{
288.                     return mRequestBody == null ? null :
289. mRequestBody.getBytes("utf-8");
290.                 }catch (UnsupportedEncodingException uee){
291.                     VolleyLog.wtf("Encoding tidak terdukung -
292. Rizky");
293.                     return null;
294.                 }
295.             }
296.             @Override
297.             public String getBodyContentType() {
298.                 return "application/json";
299.             }
300.         };
301.         mRequestQueue.add(mStringRequest);
302.     }
303.
304. }
```

Penjelasan Tabel Kode 4.21 adalah sebagai berikut :

1. Baris 22 - 91 merupakan *method onCreate()* yang berfungsi mirip dengan *Constructor,* yaitu sebagai *method* yang pertama oleh program dan didalamnya terdapat instansiasi komponen-komponen yang terdapat pada XML bedasarkan id nya serta pemberian *listenerAction* pada komponen tanggal_mulai dan tanggal_selesai.
2. Baris 92 – 105 merupakan *method* untuk mengeset tanggal pada *EditText* tanggal_mulai dan tanggal_selesai. Ketika *DatePickerDialog* telah selesai dipilih, maka method ini akan mengambil *value DatePickerDialog* dan memasangnya pada *EditText.*
3. Baris 107 – 224 merupakan *method* untuk melakukan *request* API ke *web services* untuk mendapatkan nama-nama sekolah dan alamatnya yang terdapat di Malang. Pertama sistem akan mengambil data nama-nama sekolah lalu dimasukkan keadalam Spinner (Dropdown). Lalu ketika mahasiswa memilih salah satu sekolah yang terdapat pada spinner, sistem akan mengambil alamat sekolah lalu memasangnya ke kolom alamat yaitu komponen EditText et_alamat.
4. Baris 225 – 302 merupakan baris kode untuk menangani *event* mahasiswa ketika telah memilih sekolah dan menentukan waktu mulai dan waktu selesai PPL. Sistem pertama akan mengambil *value* pada setiap kolom input lalu melakukan *request insert* data pendaftaran PPL ke *web service.*

4.7.4.4 Kode Program Fitur Pencatatan *Logbook*

Potongan kode program pada Tabel Kode 4.22 merupakan potongan kode program untuk menjalankan fitur pencatatan *logbook* pada sisi mahasiswa. Pertama mahasiswa harus memilih tanggal *logbook* lalu mengisi uraian singkat

logbook singkat dan selanjutnya mengi bsi uraian lengkap lalu selanjutnya mahasiswa menekan tombol simpan.

Tabel 4.22 Kode Program Fitur Pencatatan *logbook*

```
 public class Logbook extends AppCompatActivity {
    final Calendar myCalendar = Calendar.getInstance();
    EditText editText_tanggalDibuat, et_uraian_singkat,
et_uraian_lengkap;
    private static final String URL_LOGIN =
"http://arpulsabizz.000webhostapp.com/api/trans_data/create_logb
ook";
    ProgressDialog progressDialog;
    private static final String TAG = "Create Logbook";
    private RequestQueue mRequestQueue;
    private StringRequest mStringRequest;
    SharedPreferences sharedPreferences;
    public static final String myPreference = "cache_sippl";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_logbook);
        sharedPreferences = getSharedPreferences(myPreference,
Context.MODE_PRIVATE);
        progressDialog = new ProgressDialog(this);
        editText_tanggalDibuat = (EditText)
findViewById(R.id.et_logbook_date_create);
        et_uraian_singkat =
findViewById(R.id.et_uraian_singkat);
        et_uraian_lengkap =
findViewById(R.id.et_uraian_lengkap);

        final DatePickerDialog.OnDateSetListener date = new
DatePickerDialog.OnDateSetListener() {

            @Override
            public void onDateSet(DatePicker view, int year, int
monthOfYear,
                                  int dayOfMonth) {
                // TODO Auto-generated method stub
                myCalendar.set(Calendar.YEAR, year);
                myCalendar.set(Calendar.MONTH, monthOfYear);
                myCalendar.set(Calendar.DAY_OF_MONTH,
dayOfMonth);
                updateLabel();
            }
        };
        editText_tanggalDibuat.setOnClickListener(new
View.OnClickListener() {

            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub
                new DatePickerDialog(CreateLogbook.this, date,
myCalendar
                        .get(Calendar.YEAR),
myCalendar.get(Calendar.MONTH),

myCalendar.get(Calendar.DAY_OF_MONTH)).show();
            }
        });
    }

    private void updateLabel() {
```

```
61.         String myFormat = "dd-MM-yyyy"; //In which you need put
62.  here
63.         SimpleDateFormat sdf = new SimpleDateFormat(myFormat,
64.  Locale.US);
65.
66.  editText_tanggalDibuat.setText(sdf.format(myCalendar.getTime()))
67.  ;
68.      }
69.
70.      public void simpan_logbook(View v){
71.          String tgl_logbook =
72.  editText_tanggalDibuat.getText().toString();
73.          String[] tgl_array = tgl_logbook.split("-");
74.          tgl_logbook = tgl_array[2]+"-"+tgl_array[1]+"-
75.  "+tgl_array[0];
76.          String uraian_singkat =
77.  et_uraian_singkat.getText().toString();
78.          String uraian_lengkap =
79.  et_uraian_lengkap.getText().toString();
80.          String nim = sharedPreferences.getString("nim", "");
81.          //sekolah masih hardcode;
82.          int sekolah = 1;
83.          addLogbook(nim, tgl_logbook, sekolah, uraian_singkat,
84.  uraian_lengkap);
85.      }
86.
87.      private void addLogbook(final String nim, final String
88.  tgl_logbook, final int sekolah, String uraian_singkat, String
89.  uraian_lengkap) {
90.          progressDialog.setMessage("Loading...");
91.          progressDialog.show();
92.          JSONObject jsonBody = new JSONObject();
93.          try {
94.              jsonBody.put("nim", nim);
95.              jsonBody.put("tgl_dibuat", tgl_logbook);
96.              jsonBody.put("sekolah", sekolah);
97.              jsonBody.put("uraian_singkat", uraian_singkat);
98.              jsonBody.put("uraian_lengkap", uraian_lengkap);
99.          } catch (JSONException e) {
100.             e.printStackTrace();
101.         }
102.
103.         final String mRequestBody = jsonBody.toString();
104.         //RequestQueue initialized
105.         mRequestQueue = Volley.newRequestQueue(this);
106.         //String Request initialized
107.         mStringRequest = new StringRequest(Request.Method.POST,
108. URL_LOGIN, new Response.Listener<String>() {
109.             @Override
110.             public void onResponse(String response) {
111.                 progressDialog.hide();
112.                 finish();
113.             }
114.         }, new Response.ErrorListener() {
115.             @Override
116.             public void onErrorResponse(VolleyError error) {
117.                 progressDialog.hide();
118.                 Toast.makeText(CreateLogbook.this, "Jaringan
119. terputus", Toast.LENGTH_SHORT).show();
120.                 Log.i(TAG,"Error :" + error.toString());
121.             }
122.         })
123.         {
124.             @Override
125.             public byte[] getBody() throws AuthFailureError {
```

```
126.  //                HashMap<String, String> params2 = new
127.  HashMap<String, String>();
128.                  try{
129.                      return mRequestBody == null ? null :
130.  mRequestBody.getBytes("utf-8");
131.                  }catch (UnsupportedEncodingException uee){
132.                      VolleyLog.wtf("Encoding tidak terdukung -
133.  Rizky");
134.                      return null;
135.                  }
136.              }
137.              @Override
138.              public String getBodyContentType() {
139.                  return "application/json";
140.              }
141.          };
142.          ;
143.          mRequestQueue.add(mStringRequest);
144.      }
145.  }
146.
```

Penjelasan Tabel Kode 4.22 adalah sebagai berikut :

1. Baris 19 - 59 merupakan *method onCreate()* yang berfungsi mirip dengan *Constructor,* yaitu sebagai *method* yang pertama oleh program dan didalamnya terdapat instansiasi komponen-komponen yang terdapat pada XML bedasarkan id nya serta pemberian *listenerAction* pada komponen editText_tanggalDibuat.
2. Baris 71 – 146 merupakan baris kode untuk menangani *event* mahasiswa ketika telah memilih tanggal *logbook* dan mengisi uraian singkat dan uraian lengkap. Selanjutnya sistem pertama akan mengambil *value* pada setiap kolom input lalu sistem akan melakukan *request insert* data *logbook* ke *web service.*

4.7.4.5 Kode Program Fitur Persetujuan *Logbook*

Potongan kode program pada Tabel Kode 4.23 merupakan potongan kode program untuk menjalankan fitur persetujuan *logbook* pada sisi guru pamong. Pertama guru pamong memilih salah satu *logbook* yang terdapat di list, lalu setelah guru pamong mengecek apakah *logbook* sesuai, jika telah sesuai maka guru pamong dapat menekan tombol setujui, maka *logbook* telah disetujui dan status *logbook* berubah.

Tabel 4.23 Kode Program Fitur Persetujuan *logbook*

```
1.   public class GuruPamong extends AppCompatActivity {
2.      private static final String URL_GET_LOGBOOK_DETAIL =
3.  "http://arpulsabizz.000webhostapp.com/api/trans_data/get_logbook
4.  _detail";
5.      private static final String URL_ACCEPT_LOGBOOK =
6.  "http://arpulsabizz.000webhostapp.com/api/trans_data/accept_logb
7.  ook";
8.      private static final String TAG = "DaftarPPL";
9.      private RequestQueue mRequestQueue;
10.     private StringRequest mStringRequest;
11.
12.
```

```
    TextView tv_tanggal_logbook_detail,
tv_status_logbook_detail, tv_uraian_singkat_logbook_detail,
tv_uraian_lengkap_logbook_detail;
    SharedPreferences sharedPreferences;
    public static final String myPreference = "cache_sippl";
    int role;
    Button btn_setuju_logbook;
    int id_logbook = 0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_logbook_detail);
        tv_tanggal_logbook_detail =
findViewById(R.id.tv_tanggal_logbook_detail);
        tv_status_logbook_detail =
findViewById(R.id.tv_status_logbook_detail);
        tv_uraian_singkat_logbook_detail =
findViewById(R.id.tv_uraian_singkat_logbook_detail);
        tv_uraian_lengkap_logbook_detail =
findViewById(R.id.tv_uraian_lengkap_logbook_detail);
        sharedPreferences = getSharedPreferences(myPreference,
Context.MODE_PRIVATE);
        role = sharedPreferences.getInt("role", 0);
        btn_setuju_logbook =
findViewById(R.id.btn_setuju_logbook);
        Intent i = getIntent();
        Bundle b = i.getExtras();
        if(b != null){
            id_logbook = b.getInt("id_logbook");
            sendAndRequestResponse(id_logbook);
        }
    }
    private void sendAndRequestResponse(final int id_logbook) {
        JSONObject jsonBody = new JSONObject();
        try {
            jsonBody.put("id_logbook", id_logbook);
        } catch (JSONException e) {
            e.printStackTrace();
        }
        final String mRequestBody = jsonBody.toString();
        //RequestQueue initialized
        mRequestQueue = Volley.newRequestQueue(this);
        //String Request initialized
        mStringRequest = new StringRequest(Request.Method.POST,
URL_GET_LOGBOOK_DETAIL, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    JSONObject obj = new JSONObject(response);
                    int id = obj.getInt("id_logbook");
                    String tgl_dibuat =
obj.getString("tgl_dibuat");
                    String nama_sekolah =
obj.getString("nama_sekolah");
                    String uraian_singkat =
obj.getString("uraian_singkat");
                    String uraian_lengkap =
obj.getString("uraian_lengkap");
                    String status = obj.getString("status");
                    if(!status.equalsIgnoreCase("belum
disetujui") && role == 2 ){
              btn_setuju_logbook.setVisibility(View.GONE);
                    } else if(role == 2){
              btn_setuju_logbook.setVisibility(View.VISIBLE);
                    }
```

```
78.  tv_tanggal_logbook_detail.setText(tgl_dibuat);
79.                      tv_status_logbook_detail.setText(status);
80.  tv_uraian_singkat_logbook_detail.setText(uraian_singkat);
81.  tv_uraian_lengkap_logbook_detail.setText(uraian_lengkap);
82.                  } catch (JSONException e) {
83.                      e.printStackTrace();
84.                  }
85.              }
86.          }, new Response.ErrorListener() {
87.              @Override
88.              public void onErrorResponse(VolleyError error) {
89.                  Toast.makeText(LogbookDetail.this, "Jaringan
90.  terputus", Toast.LENGTH_SHORT).show();
91.                  Log.i(TAG,"Error :" + error.toString());
92.              }
93.          })
94.          {
95.              @Override
96.              public byte[] getBody() throws AuthFailureError {
97.                  try{
98.                      return mRequestBody == null ? null :
99.  mRequestBody.getBytes("utf-8");
100.                 }catch (UnsupportedEncodingException uee){
101.          VolleyLog.wtf("Encoding tidak terdukung -Rizky");
102.                     return null;
103.                 }
104.             }
105.             @Override
106.             public String getBodyContentType() {
107.                 return "application/json";
108.             }
109.         };
110.         mRequestQueue.add(mStringRequest);
111.     }
112.     public void setujuiLogbook(View v){
113.         sendAndRequestResponse_setujui_logbook(id_logbook);
114.     }
115.     private void menyetujuiLogbook(final int id_logbook) {
116.         JSONObject jsonBody = new JSONObject();
117.         try {
118.             jsonBody.put("id_logbook", id_logbook);
119.         } catch (JSONException e) {
120.             e.printStackTrace();
121.         }
122.         final String mRequestBody = jsonBody.toString();
123.         //RequestQueue initialized
124.         mRequestQueue = Volley.newRequestQueue(this);
125.         //String Request initialized
126.         mStringRequest = new StringRequest(Request.Method.POST,
127.  URL_ACCEPT_LOGBOOK, new Response.Listener<String>() {
128.             @Override
129.             public void onResponse(String response) {
130.                 Toast.makeText(LogbookDetail.this, "Logbook
131.  Berhasil Disetujui", Toast.LENGTH_SHORT).show();
132.                 finish();
133.             }
134.         }, new Response.ErrorListener() {
135.             @Override
136.             public void onErrorResponse(VolleyError error) {
137.                 Toast.makeText(LogbookDetail.this, "Jaringan
138.  terputus", Toast.LENGTH_SHORT).show();
139.                 Log.i(TAG,"Error :" + error.toString());
140.             }
141.         })
142.         {
```

```
143.             @Override
144.             public byte[] getBody() throws AuthFailureError {
145.                 try{
146. return mRequestBody == null ? null : mRequestBody.getBytes("utf-
147. 8");
148.                 }catch (UnsupportedEncodingException uee){
149.             VolleyLog.wtf("Encoding tidak terdukung -Rizky");
150.                     return null;
151.                 }
152.             }
153.             @Override
154.             public String getBodyContentType() {
155.                 return "application/json";
156.             }
157.         };
158.         mRequestQueue.add(mStringRequest);
159.     }
160. }
```

Penjelasan Tabel Kode 4.23 adalah sebagai berikut :

1. Baris 20 – 110 merupakan *method onCreate()* yang berfungsi mirip dengan *Constructor,* yaitu sebagai *method* yang pertama dipanggil oleh program dan didalamnya terdapat instansiasi komponen-komponen yang terdapat pada XML bedasarkan id nya serta pemberian nilai pada setiap TextView data *logbook* seperti tanggal, nama mahasiswa, uraian singkat dan uraian lengkap.
2. Baris 111 – 159 merupakan baris kode yang berfungsi untuk menangani *event* guru pamong ketika guru pamong menekan tombol setujui. Alur pertama yang dijalankan sistem adalah dengan mengambil *variable global* id_logbook dan mengirimnya sebagai parameter pada method sendAndRequestResponse _setujui_logbook. Pada method tersebut ketika telah mengembalikan *response* maka akan men-*destroy activity* dan kembali ke *activity* sebelumnya.

4.7.4.6 Kode Program Fitur Penilaian PPL

Potongan kode program pada Tabel Kode 4.24 merupakan potongan kode program untuk menjalankan fitur penilaian PPL pada sisi guru pamong. Pertama guru pamong memilih salah satu nama mahasiswa yang terdapat di list, lalu setelah guru pamong memberikan nilai mahasiswa pada setiap indikator. Selanjutnya guru dapat mengirim nilai PPL mahasiswa dengan cara menekan tombol simpan dan selanjutnya akan diarahkan ke halaman Detail Mahasiswa untuk melihat hasil penilaian.

Tabel 4.24 Kode Program Fitur Penilaian PPL

```
1.  public class GuruPamong extends AppCompatActivity {
2.
3.      private static final String URL_SAVE_NILAI_MAHASISWA =
4.  "http://arpulsabizz.000webhostapp.com/api/trans_data/insert_nila
5.  i_ppl ";
6.
7.      private static final String TAG = "KategoriPenilaian";
8.      private RequestQueue mRequestQueue;
9.      private StringRequest mStringRequest;
10.
11.     SharedPreferences sharedPreferences;
12.     public static final String myPreference = "cache_sippl";
```

```
13.     Button btn_simpan_nilai_ppl;
14.
15.     String arrayIndikator[] = {
16.             "a1","a2", "a3","a4" ,"a5", "a6" ,"a7",
17.             "b1", "b2", "b3" ,"b4" ,"b5","b6" ,"b7"
18. ,"b8","b9","b10",
19.             "c1", "c2", "c3",
20.             "d1", "d2" ,"d3","d4" , "d5",
21.             "e1" , "e2" ,"e3" ,"e4"
22.     };
23.
24.     String nim = "";
25.     ImageView iva, ivb, ivc, ivd, ive;
26.
27.     @Override
28.     protected void onCreate(Bundle savedInstanceState) {
29.         super.onCreate(savedInstanceState);
30.
31. setContentView(R.layout.activity_kategori_penilaian_ppl);
32.
33.         btn_simpan_nilai_ppl =
34. findViewById(R.id.btn_simpan_nilai_ppl);
35.
36.         iva = findViewById(R.id.iva);
37.         ivb = findViewById(R.id.ivb);
38.         ivc = findViewById(R.id.ivc);
39.         ivd = findViewById(R.id.ivd);
40.         ive = findViewById(R.id.ive);
41.
42.
43.         sharedPreferences = getSharedPreferences(myPreference,
44. Context.MODE_PRIVATE);
45.         int sample_a = sharedPreferences.getInt("a1", 0);
46.         int sample_b = sharedPreferences.getInt("b1", 0);
47.         int sample_c = sharedPreferences.getInt("c1", 0);
48.         int sample_d = sharedPreferences.getInt("d1", 0);
49.         int sample_e = sharedPreferences.getInt("e1", 0);
50.
51.
52.
53.         Intent i = getIntent();
54.         Bundle b = i.getExtras();
55.         if(b != null){
56.             nim = b.getString("nim");
57.         }
58.         btn_simpan_nilai_ppl.setEnabled(true);
59.
60. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol)
61. ;
62.
63.         if(sample_a > 0 ){
64.
65. iva.setImageResource(R.drawable.ic_check_circle_black_24dp);
66.         }else{
67.             btn_simpan_nilai_ppl.setEnabled(false);
68.
69. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
70. abu);
71.         }
72.
73.         if(sample_b > 0 ){
74.
75. ivb.setImageResource(R.drawable.ic_check_circle_black_24dp);
76.         }else{
77.             btn_simpan_nilai_ppl.setEnabled(false);
```

```
78.
79.  btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
80.  abu);
81.          }
82.
83.          if(sample_c > 0 ){
84.
85.  ivc.setImageResource(R.drawable.ic_check_circle_black_24dp);
86.          }else{
87.              btn_simpan_nilai_ppl.setEnabled(false);
88.
89.  btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
90.  abu);
91.          }
92.
93.          if(sample_d > 0 ){
94.
95.  ivd.setImageResource(R.drawable.ic_check_circle_black_24dp);
96.          }else{
97.              btn_simpan_nilai_ppl.setEnabled(false);
98.
99.  btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
100. abu);
101.         }
102.
103.         if(sample_e > 0 ){
104.
105. ive.setImageResource(R.drawable.ic_check_circle_black_24dp);
106.         }else{
107.             btn_simpan_nilai_ppl.setEnabled(false);
108.
109. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
110. abu);
111.         }
112.
113.
114.
115.     }
116.
117.
118.     @Override
119.     protected void onResume() {
120.         super.onResume();
121.         int sample_a = sharedPreferences.getInt("a1", 0);
122.         int sample_b = sharedPreferences.getInt("b1", 0);
123.         int sample_c = sharedPreferences.getInt("c1", 0);
124.         int sample_d = sharedPreferences.getInt("d1", 0);
125.         int sample_e = sharedPreferences.getInt("e1", 0);
126.
127.         btn_simpan_nilai_ppl.setEnabled(true);
128.
129. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol)
130. ;
131.
132.         if(sample_a > 0 ){
133.
134. iva.setImageResource(R.drawable.ic_check_circle_black_24dp);
135.         }else{
136.             btn_simpan_nilai_ppl.setEnabled(false);
137.
138. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
139. abu);
140.         }
141.
142.         if(sample_b > 0 ){
```

```
143.
144. ivb.setImageResource(R.drawable.ic_check_circle_black_24dp);
145.         }else{
146.             btn_simpan_nilai_ppl.setEnabled(false);
147.
148. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
149. abu);
150.         }
151.
152.         if(sample_c > 0 ){
153.
154. ivc.setImageResource(R.drawable.ic_check_circle_black_24dp);
155.         }else{
156.             btn_simpan_nilai_ppl.setEnabled(false);
157.
158. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
159. abu);
160.         }
161.
162.         if(sample_d > 0 ){
163.
164. ivd.setImageResource(R.drawable.ic_check_circle_black_24dp);
165.         }else{
166.             btn_simpan_nilai_ppl.setEnabled(false);
167.
168. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
169. abu);
170.         }
171.
172.         if(sample_e > 0 ){
173.
174. ive.setImageResource(R.drawable.ic_check_circle_black_24dp);
175.         }else{
176.             btn_simpan_nilai_ppl.setEnabled(false);
177.
178. btn_simpan_nilai_ppl.setBackgroundResource(R.drawable.bg_tombol_
179. abu);
180.         }
181.
182.     }
183.
184.     public void keA(View v){
185.         Intent i = new Intent(getApplicationContext(),
186. PenilaianPerencanaanPembelajaran.class);
187.         i.putExtra("nim", nim);
188.         startActivity(i);
189.     }
190.
191.     public void keB(View v){
192.         Intent i = new Intent(getApplicationContext(),
193. PenilaianImplementasiPembelajaran.class);
194.         i.putExtra("nim", nim);
195.         startActivity(i);
196.     }
197.
198.     public void keC(View v){
199.         Intent i = new Intent(getApplicationContext(),
200. PenilaianKompetensiProfesional.class);
201.         i.putExtra("nim", nim);
202.         startActivity(i);
203.     }
204.
205.     public void keD(View v){
206.         Intent i = new Intent(getApplicationContext(),
207. PenilaianKompetensiPersonal.class);
```

```
208.         i.putExtra("nim", nim);
209.         startActivity(i);
210.     }
211.
212.     public void keE(View v){
213.         Intent i = new Intent(getApplicationContext(),
214. PenilaianKompetensiSosial.class);
215.         i.putExtra("nim", nim);
216.         startActivity(i);
217.     }
218.
219.
220.     public void simpanNilaiPPLMahasiswa(View v){
221.         int[] nilai = new int[arrayIndikator.length];
222.         for (int i = 0; i < arrayIndikator.length; i++)  {
223.             nilai[i] =
224. sharedPreferences.getInt(arrayIndikator[i], 0);
225.         }
226.         int ppl_id = sharedPreferences.getInt( "ppl_id" , 0);
227.         int id_guru_pamong = sharedPreferences.getInt( "id" ,
228. 0);
229.
230.         menilaiPPL(nilai, ppl_id, id_guru_pamong);
231.     }
232.
233.     private void menilaiPPL(final int[] nilai, int ppl_id, int
234. id_guru_pamong) {
235.         JSONObject jsonBody = new JSONObject();
236.         try {
237.             for (int i = 0; i < arrayIndikator.length; i++) {
238.                 jsonBody.put(arrayIndikator[i], nilai[i]);
239.             }
240.             jsonBody.put("ppl_id", ppl_id);
241.             jsonBody.put("id_guru_pamong", id_guru_pamong);
242.
243.         } catch (JSONException e) {
244.             e.printStackTrace();
245.         }
246.
247.         final String mRequestBody = jsonBody.toString();
248.
249.         //RequestQueue initialized
250.         mRequestQueue = Volley.newRequestQueue(this);
251.
252.         //String Request initialized
253.         mStringRequest = new StringRequest(Request.Method.POST,
254. URL_SAVE_NILAI_MAHASISWA, new Response.Listener<String>() {
255.             @Override
256.             public void onResponse(String response) {
257.                 finish();
258.             }
259.         }, new Response.ErrorListener() {
260.             @Override
261.             public void onErrorResponse(VolleyError error) {
262.                 Toast.makeText(KategoriPenilaianPPL.this,
263. "Jaringan terputus", Toast.LENGTH_SHORT).show();
264.                 Log.i(TAG,"Error :" + error.toString());
265.             }
266.         })
267.         {
268.             @Override
269.             public byte[] getBody() throws AuthFailureError {
270.                 try{
271.                     return mRequestBody == null ? null :
272. mRequestBody.getBytes("utf-8");
```

```
273.                 }catch (UnsupportedEncodingException uee){
274.                     VolleyLog.wtf("Encoding tidak terdukung -
275. Rizky");
276.                     return null;
277.                 }
278.             }
279.
280.             @Override
281.             public String getBodyContentType() {
282.                 return "application/json";
283.             }
284.         };
285.
286.         mRequestQueue.add(mStringRequest);
287.     }
288. }
289.
290.
```

Penjelasan Tabel Kode 4.24 adalah sebagai berikut :

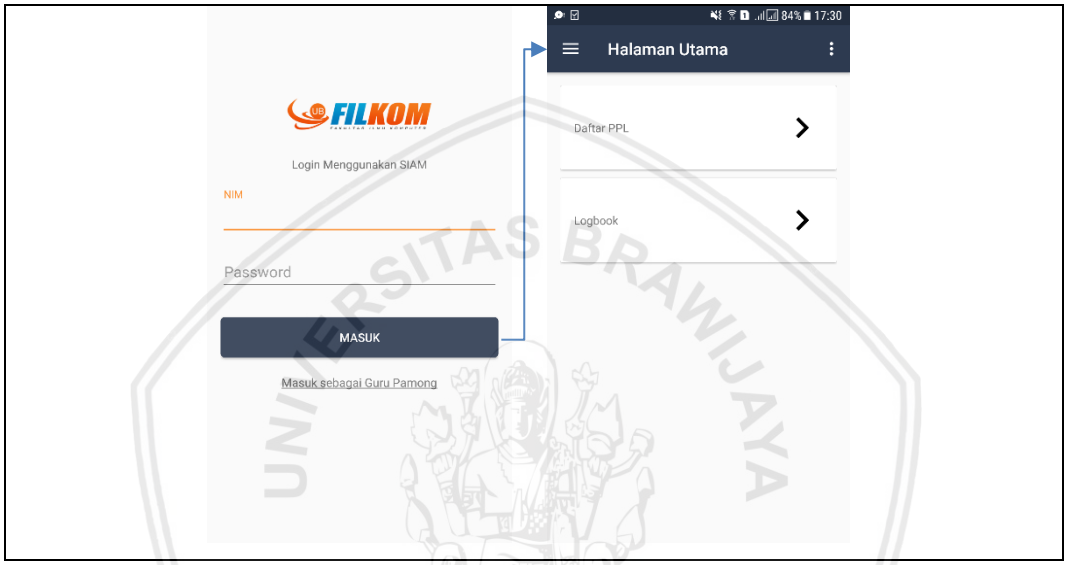
1. Baris 28 – 115 merupakan *method onCreate()* yang berfungsi mirip dengan *Constructor,* yaitu sebagai *method* yang pertama dipanggil oleh program dan didalamnya terdapat instansiasi komponen-komponen yang terdapat pada XML bedasarkan id nya. Pada *method* ini juga terdapat kode untuk pengecekan apakah suatu kategori indikator nilai sudah diisi apa belum. Jika kategori indikator sudah diisi, maka *icon* akan berubah menjadi centang kuning.
2. Baris 119 – 182 merupakan *method* turunan bernama *onResume()* yang akan dijalankan ketika activity ini kembali dari *activity* setelahnya. Selanjutnya didalam *method* ini akan dilakukan pengambilan data penilaian pada *sharedPreferences* untuk nantinya digunakan pengecekan apakah indikator telah diisi nilainya atau belum. Jika sudah maka *icon* akan berubah menjadi centan berwarna kuning.
3. Baris 184 – 217 merupakan baris kode untuk mengalihkan halaman atau *activity* guna memasukkan nilai pada setiap kategori indikator. Seperti contoh ketika kategori indikator Kompetensi Profesional ditekan maka sistem akan mengalihkan halaman ke halaman PenilaianKompetensiProfesional. Begitu pada setiap kategori indikator.
4. Baris 220 – 289 merupakan baris kode yang berfungsi untuk mengirimkan data nilai ke *web service* untuk disimpan pada database. Baris ini akan dieksekusi sistem ketika guru pamong telah selesai memberikan nilai dan menekan tombol simpan. Langkah pertama yang dilakukan sistem adalah dengan mengambil semua data nilai pada *sharedPreferences* lalu mengirimnya ke *web service* dengan memanggil *method sendAndRequestResponse()* yang mana didalam *method* tersebut terdapat perintah yang menggunakan *library* Volley yang berfungsi untuk mempermudah *request* API.

4.7.5 Implementasi Antarmuka Sistem

Implementasi antarmuka ini merupakan tahap pembuatan antar muka yang dilakukan berdasar perancangan antarmuka yang telah di jelaskan pada sub bab sebelumnya. Implementasi antarmuka akan disusun berdasarkan susunan *screen flow* pada perancangan.

4.7.5.1 Implementasi *Screen Flow Login*

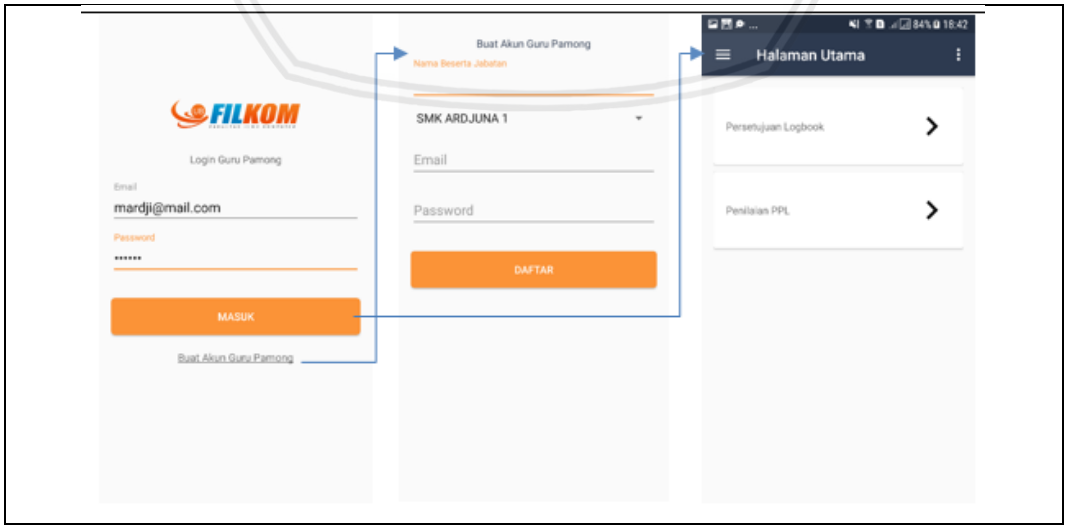
Screen flow implementasi fitur *login* dapat dilihat pada Gambar 4.41 dimana *screen flow* ini memiliki penjelasan yang sama dengan penjelasan di perancangan.



Gambar 4.41 Implementasi *Screen Flow* Login Sistem Informasi PPL

4.7.5.2 Implementasi Screen Flow SignUp

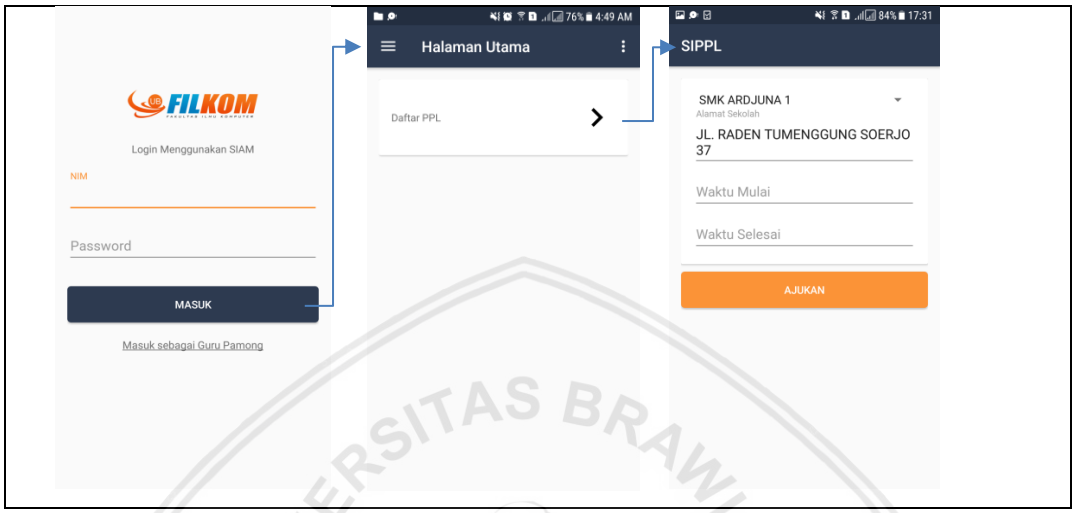
Screen flow implementasi fitur *ini* dapat dilihat pada Gambar 4.42 dimana *screen flow* ini memiliki penjelasan yang sama dengan penjelasan di perancangan.



Gambar 4.42 Implementasi *Screen Flow SignUp* Guru Pamong

4.7.5.3 Implementasi Screen Flow Pendaftaran PPL

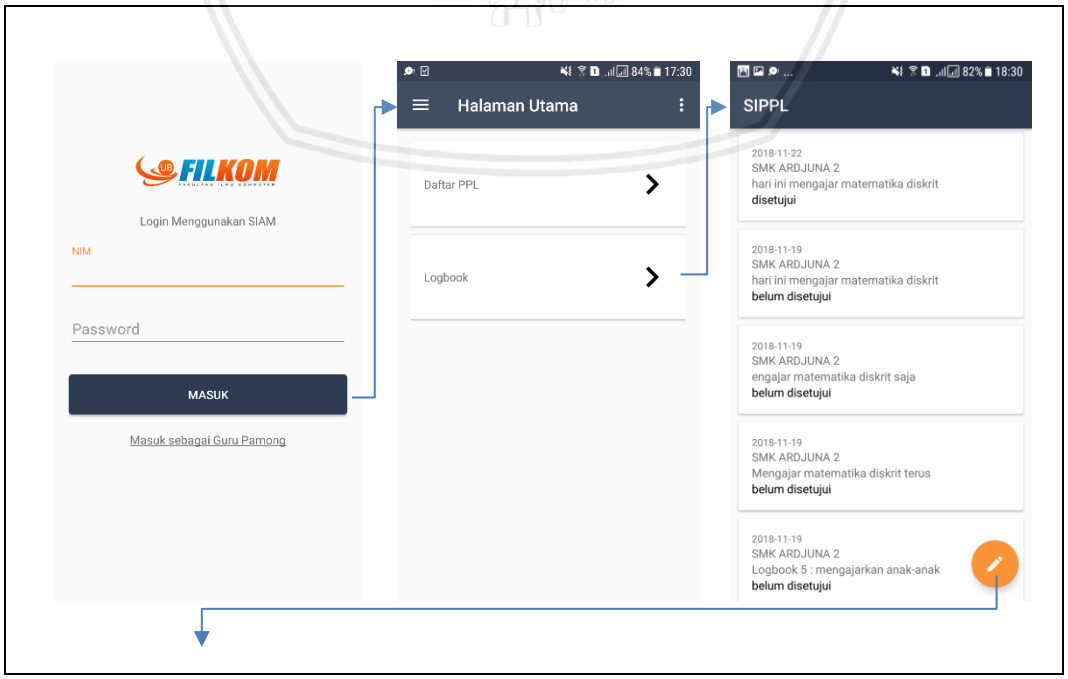
Berdasrakan hasil pada iterasi kedua, menu *logbook* dihilangkan ketika mahasiswa belum melakukan pendaftaran PPL. Gambar 4.43 menunjukkan hasil iterasi 2, yaitu menu *logbook* tidak muncul ketika belum melakukan pendaftaran PPL.

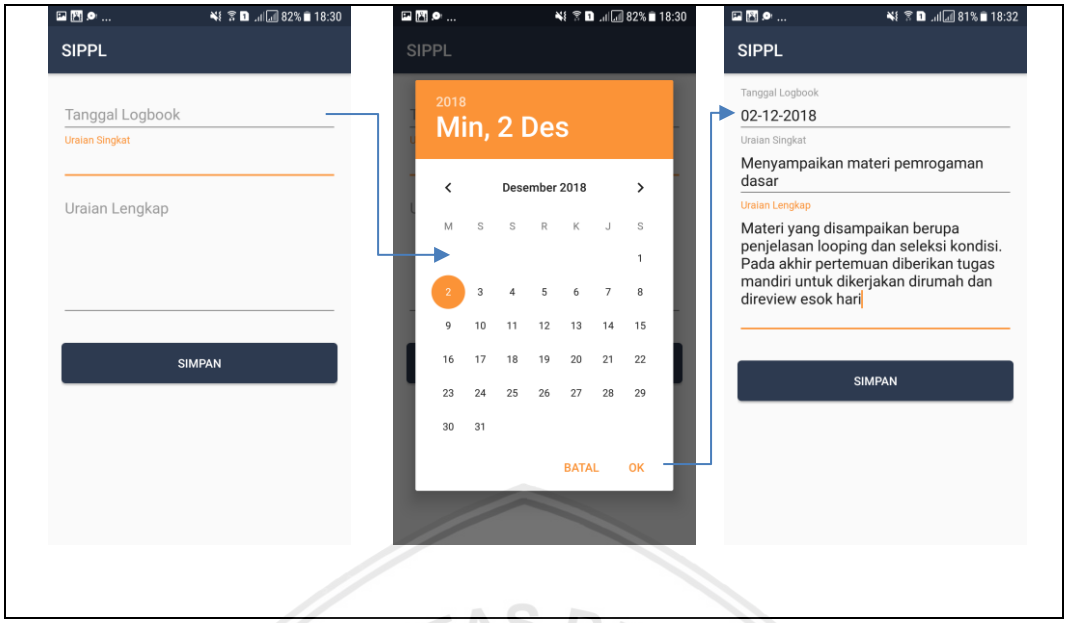


Gambar 4.43 Implementasi *Screen Flow* Pendaftaran PPL Iterasi 2

4.7.5.4 Implementasi Screen Flow Pencatatan Logbook

Screen flow implementasi pencatatan *logbook* dapat dilihat pada Gambar 4.44 dimana *screen flow* ini memiliki penjelasan yang sama dengan penjelasan di perancangan antarmuka.

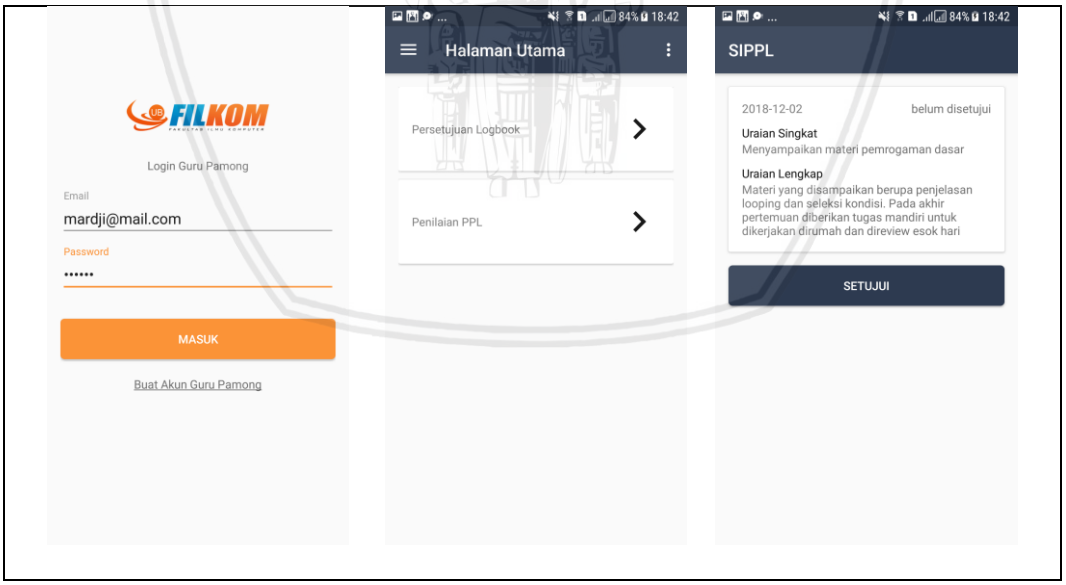




Gambar 4.44 Implementasi *Screen Flow* Pencatatan *Logbook*

4.7.5.5 Implementasi Screen Flow Persetujuan Logbook

Screen flow implementasi Persetujuan *logbook* dapat dilihat pada Gambar 4.45 dimana *screen flow* ini memiliki penjelasan yang sama dengan penjelasan di perancangan antarmuka.



Gambar 4.45 Implementasi *Screen Flow* Persetujuan *Logbook*

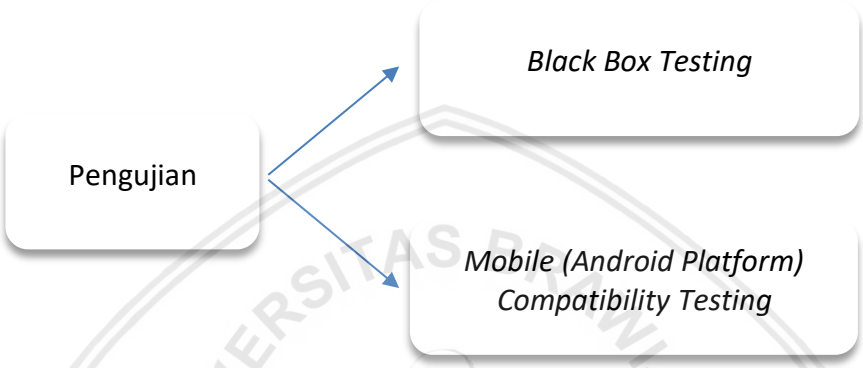
4.7.5.6 Implementasi Screen Flow Penilaian PPL

Screen flow implementasi penilaian PPL dapat dilihat pada Gambar 4.46 penjelasan *screen flow* ini sama dengan penjelasan di perancangan antarmuka.

Gambar 4.46 Implementasi *Screen Flow* Penilaian PPL

4.8 Pengujian Sistem

Pada bagian ini dijelaskan proses pengujian terhadap sistem informasi SIPPL. Proses pengujian dilakukan dengan tujuan untuk mengetahui seberapa tinggi tingkat kualitas sistem informasi yang dibangun. Dalam prosesnya, pengujian sistem dilakukan dengan menggunakan metode pengujian *Black-Box Testing* dan *Mobile (Android Platform) Compatibility Testing.* Struktur pengujian dapat dilihat pada Gambar 4.47.



Gambar 4.47 Struktur Pengujian Sistem SIPPL

4.8.1 Black Box Testing

Black box testing atau dalam bahasa indonesia berarti pengujian kotak hitam merupakan pengujian yang didasarkan pada verifikasi dan validasi kebutuhan fungsional dari sistem perangkat lunak. Pengujian ini juga biasa dikenal dengan nama pengujian perilaku. Proses *testing* yang dilakukan menggunakan metode ini dimulai dengan membuat perancangan kasus uji lalu selanjutnya melakukan pengujian berdasar skenario kasus uji yang telah dibuat. Perancangan kasus uji dilakukan dengan berpedoman pada hasil analisa kebutuhan sistem yang telah dilakukan pada bagian sebelumnya. Daftar kasus uji yang digunakan dalam proses pengujian ini dapat dilihat pada Tabel 4.25 sampai Tabel 4.30.

Tabel 4.25 Kasus Uji Melakukan Proses *Login*

Melakukan *login* kedalam sistem informasi PPL	
Nomor Kasus Uji	SIPPL_BBT_01
Nama Kasus Uji	Melakukan *login* kedalam sistem
Objek Uji	SIPPL_F_01
Tujuan Pengujian	Memastikan sistem memungkinkan pengguna dapat melakukan login sesuai dengan *role* masing-masing
Prosedur Uji	1. Membuka aplikasi Sistem Informasi PPL (SIPPL)

	2. Membuka halaman login sesuai masing-masing *role* 3. Bagi role Mahasiswa memasukkan NIM dikolom NIM, sedangkan role guru memasukkan email yang sudah didaftarkan kedalam sistem pada kolom email. 4. Pengguna memasukkan *password* dikolom *password* pada halaman login masing-masing. 5. Pengguna menekan tombol "masuk" pada halaman login masing-masing.
Hasil yang Diharapkan	Pengguna dapat melakukan *login* kedalam sistem

Tabel 4.26 Kasus Uji Melakukan Proses *SignUp*

Melakukan *SignUp* kedalam sistem informasi PPL	
Nomor Kasus Uji	SIPPL_BBT_02
Nama Kasus Uji	Melakukan *SignUp* kedalam sistem
Objek Uji	SIPPL_F_02
Tujuan Pengujian	Memastikan sistem memungkinkan guru pamong dapat melakukan *SignUp*
Prosedur Uji	1. Membuka aplikasi Sistem Informasi PPL (SIPPL) 2. Membuka halaman *Sign Up* 3. Memasukkan data diri sesuai form *Sign Up* yang terdapat pada sistem. 4. Guru menekan tombol "Daftar". 5. Sistem mengirimkan email verifikasi 6. Mendapatkan email verifikasi dan melakkukan verifikasi email dengan membuka link unik yang ada didalam email
Hasil yang Diharapkan	Guru pamong dapat melakukan *SignUp* kedalam sistem

Tabel 4.27 Kasus Uji Melakukan Proses Pendaftaran PPL

Melakukan Pendaftarn PPL pada sistem informasi SIPPL	
Nomor Kasus Uji	SIPPL_BBT_03
Nama Kasus Uji	Melakukan pendaftaran PPL kedalam sistem
Objek Uji	SIPPL_F_03
Tujuan Pengujian	Memastikan sistem memungkinkan mahasiswa dapat melakukan pendaftaran PPL
Prosedur Uji	1. Login sebagai mahasiswa 2. Memilih menu “Daftar PPL” 3. Sistem akan mengecek apakah mahasiswa yang bersangkutan sudah memenuhi syarat untuk melaksanakan PPL atau belum. 4. Jika sudah memenuhi syarat, maka sistem akan menampilkan halaman form F1A dan form F1B 5. Lalu mahasiswa mengisi form F1A dan F1B yang terdapat pada sistem 6. Mahasiswa menekan tombol “Ajukan” 7. Sistem akan menyimpan data yang dimasukkan lalu sistem akan memberikan form F2A, F2B dan F2C untuk diberikan kepada sekolah mitra yang dituju mahasiswa
Hasil yang Diharapkan	Mahasiswa dapat melakukan Pendaftaran PPL melalui sistem

Tabel 4.28 Kasus Uji Melakukan Proses Pencatatan *Logbook*

Melakukan Pencatatan *Logbook* pada sistem informasi SIPPL	
Nomor Kasus Uji	SIPPL_BBT_04
Nama Kasus Uji	Melakukan pencatatan *Logbook* kedalam sistem
Objek Uji	SIPPL_F_04
Tujuan Pengujian	Memastikan sistem memungkinkan mahasiswa dapat melakukan pencatatan *logbook*
Prosedur Uji	1. Login sebagai mahasiswa 2. Memilih menu “*Logbook*”

	3. Sistem akan menampilkan daftar logbook yang pernah dibuat sebelumnya 4. Menekan tombol "Tambah" untuk menambahkan *logbook* harian 5. Sistem menampilkan form *logbook* 6. Mengisi form *logbook* yang terdapat pada sistem 7. Menekan tombol "Simpan"
Hasil yang Diharapkan	Dapat melakukan Pencatatan *Logbook* melalui sistem

Tabel 4.29 Kasus Uji Melakukan Proses Persetujuan *Logbook*

Melakukan Persetujuan *Logbook* pada sistem informasi SIPPL	
Nomor Kasus Uji	SIPPL_BBT_05
Nama Kasus Uji	Melakukan persetujuan *Logbook* kedalam sistem
Objek Uji	SIPPL_F_05
Tujuan Pengujian	Memastikan sistem memungkinkan guru pamong dapat melakukan persetujuan *logbook*
Prosedur Uji	1. Login sebagai guru pamong 2. Memilih menu "Persetujuan *Logbook*" 3. Sistem akan menampilkan daftar logbook yang telah dibuat oleh mahasiswa 4. Memilih salah satu *logbook* yang ingin disetujui 5. Menekan tombol "Setuju" untuk melakukan persetujuan *logbook* harian mahasiswa
Hasil yang Diharapkan	Dapat melakukan persetujuan *Logbook* melalui sistem

Tabel 4.30 Kasus Uji Melakukan Proses Penilaian PPL

Melakukan Penilaian PPL pada sistem informasi SIPPL	
Nomor Kasus Uji	SIPPL_BBT_06
Nama Kasus Uji	Melakukan Penilaian PPL melalui sistem
Objek Uji	SIPPL_F_06

Tujuan Pengujian	Memastikan sistem memungkinkan guru pamong dapat melakukan penilaian PPL
Prosedur Uji	1. Login sebagai guru pamong 2. Memilih menu “Penilaian PPL” 3. Memilih salah satu nama mahasiswa 4. Menekan tombol “Masukkan Nilai” 5. Guru mengisi form penilaian berdasar hasil kegiatan PPL yang telah dilakukan oleh mahasiswa 6. Guru menekan tombol “Simpan”
Hasil yang Diharapkan	Dapat melakukan penilaian PPL melalui sistem

4.8.2 Mobile (Android Platform) Compatiblity Testing

Mobile (Android Platform) compatibility testing merupakan sebuah metode yang berguna untuk mengetahui tingkat ketergantungan sistem aplikasi android terhadap perbedaan versi sistem operasi Android pada perangkat yang menjalankannya. Pada pengujian ini dilakukan pengecekan seluruh fungsionalitas aplikasi saat dijalankan pada beberapa versi Android yang berbeda. Daftar kasus uji yang digunakan dalam proses pengujian dapat dilihat melalui Tabel 4.31 sampai Tabel 4.33.

Tabel 4.31 Android 5.0 (*Lollipop*) *Compatibility Testing*

Nomor Kasus Uji	SIPPL_MCT_01
Nama Kasus Uji	Android 5.0 (*Lollipop*) *Compatibility Testing*
Objek Uji	SIPPL_NF_01
Tujuan Pengujian	Memastikan seluruh fitur dapat berjalan dan bekerja sesuai *use case scenario* pada sistem operasi Android 5.0
Prosedur Uji	Mencoba seluruh fitur yang terdapat pada aplikasi

Tabel 4.32 Android 7.0 (*Nougat*) *Compatibility Testing*

Nomor Kasus Uji	SIPPL_MCT_02
Nama Kasus Uji	Android 7.0 (*Nougat*) *Compatibility Testing*
Objek Uji	SIPPL_NF_01
Tujuan Pengujian	Memastikan seluruh fitur dapat berjalan dan bekerja sesuai *use case scenario* pada sistem operasi Android 7.0

Prosedur Uji	Mencoba seluruh fitur yang terdapat pada aplikasi
Hasil yang Diharapkan	Seluruh fitur yang dimiliki aplikasi dapat berjalan dana bekerja sesuai *use case scenario* pada sistem operasi Android 7.0 (*Nougat)*

Tabel 4.33 Android 8.0 (*Oreo*) *Compatibility Testing*

Nomor Kasus Uji	SIPPL_MCT_03
Nama Kasus Uji	Android 8.0 (*Oreo*) *Compatibility Testing*
Objek Uji	SIPPL_NF_01
Tujuan Pengujian	Memastikan seluruh fitur dapat berjalan dan bekerja sesuai *use case scenario* pada sistem operasi Android 8.0
Prosedur Uji	Mencoba seluruh fitur yang terdapat pada aplikasi
Hasil yang Diharapkan	Seluruh fitur yang dimiliki aplikasi dapat berjalan dana bekerja sesuai *use case scenario* pada sistem operasi Android 8.0 (*Oreo)*

4.9 Analisis Hasil Pengujian

Pada tahapan ini akan diuraikan proses analisis terhadap seluruh hasil pengujian yang telah dilakukan pada bagian sebelumnya. Analisis dilakukan pada setiap metode pengujian untuk mengetahui apakah sistem berhasil dan benar dibuat. Bagian ini terdiri dari analisis hasil *Black Box Testing* dan *Mobile (Android Framework) Compatibility Testing.*

4.9.1 Analisis *Black Box Testing*

Analisis *Black Box Testing* diperoleh berdasarkan pelaksanaan kasus uji untuk pengujian dengan metode *Black Box Testing* seperti yang ditunjukkan pada Tabel 4.25 sampai Tabel 4.30. Berdasarkan kasus uji yang telah dilakukan, selanjutnya dilakukan pencocokan antara hasil yang diharapkan dengan hasil yang didapatkan sehingga dapat diperoleh validitas untuk setiap fungsional aplikasi yang ada. Hasil analisis *Black Box Testing* dapat dilihat pada Tabel 4.34.

Tabel 4.34 Hasil Analisis *Back Box Testing*

Nomor Kasus Uji	**Hasil yang Diharapkan**	**Hasil yang Didapatkan**	**Validitas**
SIPPL_BBT_01	Pengguna dapat melakukan *login* kedalam sistem	Pengguna dapat melakukan *login* kedalam sistem	Valid

SIPPL_BBT_02	Guru pamong dapat melakukan *SignUp* kedalam sistem	Guru pamong dapat melakukan *SignUp* kedalam sistem	Valid
SIPPL_BBT_03	Dapat melakukan Pendaftaran PPL melalui sistem	Dapat melakukan Pendaftaran PPL melalui sistem	Valid
SIPPL_BBT_04	Dapat melakukan Pencatatan *Logbook* melalui sistem	Dapat melakukan Pencatatan *Logbook* melalui sistem	Valid
SIPPL_BBT_05	Dapat melakukan persetujuan *Logbook* melalui sistem	Dapat melakukan persetujuan *Logbook* melalui sistem	Valid
SIPPL_BBT_06	Dapat melakukan penilaian PPL melalui sistem	Dapat melakukan penilaian PPL melalui sistem	Valid

4.9.2 Analisis *Mobile* (*Android Platform) Compatibility Testing*

Analisis *Mobile (Android Platform) Compatibility Testing* didapatkan berdasarkan pelaksanaan kasus uji untuk pengujian dengan metode ini seperti yang ditunjukkan pada Tabel 4.31 sampai Tabel 4.33. Berdasarkan kasus uji yang telah dilakukan, selanjutnya dilakukan pencocokan antara hasil yang diharapkan dengan hasil yang didapatkan. Sehingga berdasarkan hal tersebut dapat diperoleh kompatibilitas untuk setiap versi Android yang dijadikan rujukan. Hasil analisis *Mobile* (*Android Platform) Compatibility Testing* dapat dilihat pada Tabel 4.35.

Tabel 4.35 Hasil Analisis *Mobile (Android Platform) Compatibility Testing*

Nomor Kasus Uji	**Hasil yang Diharapkan**	**Hasil yang Didapatkan**	**Validitas**
SIPPL_MCT_01	Seluruh fitur yang dimiliki aplikasi dapat berjalan dan bekerja sesuai *use case scenario* pada sistem operasi Android 4.0 (*Ice Cream Sandwich)*	Seluruh fitur yang dimiliki aplikasi dapat berjalan dan bekerja sesuai *use case scenario* pada sistem operasi Android 4.0 (*Ice Cream Sandwich)*	Valid
SIPPL_MCT_02	Seluruh fitur yang dimiliki aplikasi dapat berjalan dan bekerja sesuai *use case scenario* pada sistem	Seluruh fitur yang dimiliki aplikasi dapat berjalan dan bekerja sesuai *use case scenario* pada sistem	Valid

	operasi Android 7.0 (*Nougat)*	operasi Android 7.0 (*Nougat)*	
SIPPL_MCT_03	Seluruh fitur yang dimiliki aplikasi dapat berjalan dan bekerja sesuai *use case scenario* pada sistem operasi Android 8.0 (*Oreo)*	Seluruh fitur yang dimiliki aplikasi dapat berjalan dan bekerja sesuai *use case scenario* pada sistem operasi Android 8.0 (*Oreo)*	Valid

BAB 5 PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, dapat diambil kesimpulan sebagai berikut :

1. Hasil analisis kebutuhan sistem informasi PPL dari dokumen pedoman dan aturan PPL Fakultas Ilmu Komputer Universitas Brawijaya, serta dari hasil wawancara Kaprodi PTI diperoleh 6 kebutuhan fungsionalitas, 1 kebutuhan fungsionalitas dan 2 aktor yaitu aktor mahasiswa dan aktor guru pamong. Kebutuhan terhadap sistem yang diperoleh yaitu sistem dapat memfasilitiasi pendaftaran PPL bagi mahasiswa, Sistem dapat mencatat *logbook* harian yang dibuat oleh mahasiswa, Sistem dapat menyetujui *logbook* mahasiswa dari *role* guru pamong, dan Sistem dapat mencatat penilaian kegiatan PPL mahasiswa dari *role* guru pamong.
2. Hasil perancangan sistem, didapatkan total terdapat 13 *class.* Hasil analisis kelas tersebut berdasarkan dari proses analisis kebutuhan yang digabungkan antara keseluruhan komponen menjadi satu analisis kelas. Untuk bagian perancangan basis data, terdapat 8 *table* untuk menampung data sistem informasi. Keseluruhan perancangan diimplementasikan menggunakan *framework* Android dan untuk membantu proses penyimpanan data kedalam *database* dibuatkan *web service* yang dibangun menggunakan *framework CodeIgniter.*
3. Hasil pengujian menggunakan metode *Black Box Testing* menunjukkan semua kebutuhan yang telah terdefinisikan pada Tabel Kebutuhan Fungsionalitas 100% dapat dipenuhi oleh sistem yang telah diimplementasikan. Selain itu untuk menjawab kebutuhan yang terdapat di Tabel Non-Fungsionalitas dilakukan pengujian *Mobile (Android Framework) Compatibility Testing.* Metod ini dilakukan dengan cara mencoba aplikasi sitem informasi pada beberapa *device* Android versi berbeda-beda diatas versi 4 yang hasilnya menunjukkan bahwa sistem telah 100% *Compatible* atau dapat berjalan normal pada semua versi Android diatas versi 4.

5.2 Saran

Berikut adalah saran yang bisa dilakukan untuk mengembangkan penelitian ini lebih lanjut :

1. Terdapat beberapa proses lain yang bisa ditambahkan kedalam fitur sistem informasi ini, seperti fitur *Notification,* fitur *Lesson Study,* fitur *Profile* dan fitur Panduan.
2. Diperlukan proses analisis UI/UX dan *usability testing* untuk mengetahui seberapa tinggi tingkat penerimaan pengguna terhadap sistem informasi PPL.

DAFTAR PUSTAKA

Alexandra, J., 2017. *Agile Development Methods.* Tersedia di: <http://sis.binus.ac.id/2017/05/08/agile-development-methods/> [Diakses pada 03 Agustus 2018].

Andre, 2017. *Tutorial Belajar MySQL Part 4: Pengertian Relational Database*. Tersedia di: < https://www.duniailkom.com/tutorial-mysql-pengertian-relational-database/ > [Diakses pada 03 Januari 2018]

Atletiko F. J., 2017. *Development of Android Application for Courier Monitoring System.* [e-jurnal] Tersedia melalui ScienceDirect < https://www.sciencedirect.com/science/article/pii/S1877050917329836 > [Diakses 16 Agustus 2018].

Bittner, K. & Spence, I., 2003. *Use Case Modeling.* United States : Pearson Education Inc.

Dong C. & Liu X., 2013. *Development of Android Application for Language Studies.* [e-jurnal] Tersedia melalui ScienceDirect <https://www.sciencedirect.com/science/article/pii/S2212667813000063> [Diakses 16 Agustus 2018].

Dwiharianto, I., 2018. *Pentingnya Teknologi Dalam Dunia Pendidikan di Indonesia.* Tersedia di: <https://m.kumparan.com/@millennial/pentingnya-teknologi-dalam-dunia-pendidikan-di-indonesia> [Diakses 23 Agustus 2018].

Gargenta, M. & Nakamura, M., 2014. *Second Edition Learning Android: Develop Mobile Apps Using Java and Eclipse.* Sebastopool, USA : O'Reilly [e-book] Tersedia di <https://books.google.co.id/books?id=vhWMAgAAQBAJ&printsec=frontcover&dq=android&hl=en&sa=X&ved=0ahUKEwix35-jspbdAhUR3o8KHcvZB0UQ6AEIKTAA#v=onepage&q&f=false> [Diakses 31 Agustus 2018].

Hartono, S., 2012. *Sequence Diagram Aplikasi Perpustakaan.* Tersedia di: < http://www.sugihhartono.com/2012/04/sequence-diagram-aplikasi-perpustakaan.html > [Diakases 03 Januari 2018]

Kularbphettonga, K., 2015. *Developing of mLearning for Discrete Mathematics based on Android Platform.* [e-jurnal] Tersedia melalui ScienceDirect <https://www.sciencedirect.com/science/article/pii/S1877042815041853> [Diakses 16 Agustus 18].

Lucidchart Content Team, 2018. *What Is Extreme Programming? An Overview of XP Rules and Values.* Tersedia di : < https://www.lucidchart.com/blog/what-is-extreme-programming > [Diakses pada 18 Desember 2018]

Michael, 2015. *Extreme Programming.* Tersedia di : < https://medium.com/@mikesebastian/extreme-programming-c715e6b8e0e9 > [Diakses pada 18 Desember 2018]

Nuroji, 2017. *Metode-metode Pengembangan Sistem Informasi.* [online] Tersedia di <http://nuroji.uhamka.ac.id/1641-2/> [Diakses pada 03 Agustus 2018].

Pressman, R. S., 2010. *Software Engineering A Practitioner's Approach. 7th Edition*. New York : McGraw-Hill.

Rumbaugh, J., Jacobson, I. & Booch, G., 2005. *The Unified Modeling Language User Guide. 2nd Edition*. Boston : Addison-Wesley.

Sapoetra, J., 2017. *Kompetensi Pedagogik.* Tersedia di : <https://pgsd.binus.ac.id/2017/12/31/kompetensi-pedagogik/> [Diakses 27 Agustus 2018].

Sommerville, I., 2011. *Software Engineering.* 9th Ed. London: Addison-Wesley.

Software Testing Help, 2018. *What is Software Compatibility Testing?.* Tersedia di: < https://www.softwaretestinghelp.com/software-compatibility-testing/> [Diakses pada 02 Januari 2018]

Stair, R. M. & Reynolds, G. W., 2014. *Fundamentals of Information Systems.* 8th Ed. Boston: Cengage Learning.

Tim Penyusun Pedoman, 2018. *Panduan Penyelesaian dan Evaluasi Praktik Pengalaman Lapangan (PPL).* Malang. Fakultas Ilmu Komputer Universitas Brawijaya.

Wells, D., 2009. *Agile Software Development: A gentle introduction.* Tersedia di < http://www.agile-process.org/ > [Diakses 01 September 2018].

Zwass, V., 2017. *Information system.* Encyclopædia Britannica, inc. Tersedia di : < https://www.britannica.com/topic/information-system > [Diakses 28 Agustus 2018].