

LAPORAN PRAKTIK KERJA LAPANGAN (PKL)

PERUSAHAAN/INDUSTRI

**UNIT RESEARCH AND DEVELOPMENT REGIONAL PT. BANK NEGARA
INDONESIA (PERSERO), TBK. KANTOR WILAYAH MALANG**

**PENERAPAN FRAMEWORK LARAVEL DALAM PENGEMBANGAN SISTEM
INFORMASI LENDING BROMO BERBASIS WEB**

Diajukan untuk memenuhi sebagian persyaratan Kurikulum Sarjana



Disusun Oleh :

Muhammad Kevin Sandryan	175150700111003
Muhammad Rasyid Al Faruqi	175150700111007
Mohammad Raska	175150700111019

**PROGRAM STUDI TEKNOLOGI INFORMASI
JURUSAN SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2020**

PENGESAHAN

LAPORAN PRAKTIK KERJA LAPANGAN (PKL)
PERUSAHAAN/INDUSTRI
PT. BANK NEGARA INDONESIA (PERSERO), TBK. KANTOR WILAYAH
MALANG

PENERAPAN *FRAMEWORK LARAVEL DALAM PENGEMBANGAN SISTEM*
INFORMASI LENDING BROMO BERBASIS WEB

Diajukan untuk memenuhi sebagian persyaratan Kurikulum Sarjana
Program Studi Teknologi Informasi

Disusun Oleh :

Muhammad Kevin Sandryan	175150700111003
Muhammad Rasyid Al Faruqi	175150700111007
Mohammad Raska	175150700111019

Praktik Kerja Lapangan ini dilaksanakan pada
28 Januari 2020 sampai dengan 24 April 2020
Telah diperiksa dan disetujui oleh :

Dosen Pembimbing PKL



Issa Arwani, S.Kom., M.Sc.

NIP. 19830922 201212 1 003

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 laporan PKL ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain dalam kegiatan akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam laporan PKL ini terbukti terdapat unsur-unsur plagiasi, saya bersedia PKL ini digugurkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 19 Juni 2020



Mohammad Raska
NIM. 175150700111019

KATA PENGANTAR

Puji syukur senantiasa dipanjangkan kehadiran Tuhan Yang Maha Esa yang telah melimpahkan rahmat, hidayah, serta karunia-Nya sehingga penulis dapat menyelesaikan Praktik Kerja Lapangan dan penulisan laporan hasil Praktik Kerja Lapangan yang berjudul “Penerapan *Framework* Laravel dalam Pengembangan Sistem Informasi *Lending* BROMO Berbasis Web”.

Penulis menyadari bahwa banyak bimbingan dan bantuan yang didapat dari beberapa pihak sehingga penulisan Laporan Hasil Praktik Kerja Lapangan ini dapat berjalan dengan lancar. Oleh karena itu, penulis mengucapkan banyak terima kasih kepada :

1. Orang tua yang selalu memberikan dukungan moril dan materil, serta memberikan perhatian dan do'a dalam melakukan Praktik Kerja Lapangan.
2. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya yang telah memberikan kesempatan untuk melaksanakan Praktik Kerja Lapangan.
3. Bapak Dr. Eng. Herman Tolle, S.T., M.T. selaku Ketua Jurusan Sistem Informasi Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Issa Arwani, S.Kom., M.Sc selaku Kepala Program Studi Teknologi Informasi Fakultas Ilmu Komputer Universitas Brawijaya sekaligus selaku dosen pembimbing PKL yang senantiasa meluangkan waktu dan memberikan bimbingan dalam melakukan Praktik Kerja Lapangan.
5. Ibu Dyah Pangestuti, Ibu Kinanti Ramadhani, dan Bapak Amir Ibnu Alfian selaku pembimbing dari Divisi *Research and Development Region* selama Praktik Kerja Lapangan berlangsung.
6. Segenap karyawan PT. Bank Negara Indonesia Kantor Wilayah Malang dan semua pihak yang telah ikut membantu dalam pelaksanaan kegiatan Praktik Kerja Lapangan yang tidak dapat disebutkan satu persatu.

Penulis menyadari bahwa laporan ini masih jauh dari kata sempurna, sehingga penulis mengharapkan kritik dan saran bagi penulis. Akhir kata, semoga laporan ini dapat memberikan manfaat bagi penulis dan orang lain.

Malang, April 2020

Penulis

Muhammad Kevin Sandryan

Muhammad Rasyid Al Faruqi

Mohammad Raska

ABSTRAK

PT. Bank Negara Indonesia (Persero) adalah sebuah institusi bank tertua dalam sejarah Republik Indonesia yang dimiliki oleh BUMN (Badan Usaha Milik Negara). Saat ini BNI memiliki 1076 kantor cabang di Indonesia dan 5 di luar negeri. Salah satu kantor cabang terletak di kota Malang yang merupakan kantor wilayah. BNI Kantor Wilayah (Kanwil) Malang memiliki divisi *Research and Development Region* (RDR) yang berada di bawah Unit Pengelola Penelitian & Pengembangan di mana salah satu tugasnya adalah menunjang kegiatan kepegawaian secara internal maupun wilayah malang dalam lingkup IT. Bank Negara Indonesia Kantor Wilayah Malang melakukan proses sortir untuk calon debitur yang memiliki potensi gagal bayar maupun yang tidak. Dikarenakan proses pendataan calon debitur masih berjalan secara manual, komunikasi dan pelaporan antar petugas terkait sering terhambat dan proses yang berjalan tidak diketahui dengan baik. Oleh karena itu dikembangkan sistem informasi berbasis *web* untuk melakukan *monitoring* secara online yang mempermudah proses *lending* debitur sehingga meningkatkan efisiensi dari proses *lending* dan menghemat waktu dari petugas yang bersangkutan. Sistem Informasi *Lending* BROMO dikembangkan menggunakan metode SDLC waterfall. Implementasi *website* menggunakan *framework* Laravel 6.2 dan implementasi tampilan menggunakan *framework* Bootstrap 4.0. Kemudian dilakukan pengujian terhadap sistem menggunakan *black box testing* untuk mengetahui kebutuhan sistem telah terpenuhi. Hasil dari pengujian validasi menghasilkan semua kebutuhan telah terpenuhi, dan hasil pengujian *compatibility* dan *responsive* didapatkan sistem dapat berjalan pada 8 jenis browser dan *responsive* pada ukuran tablet dan desktop.

Kata Kunci : Sistem Informasi *Lending* , *Waterfall* , *Framework* , Laravel 6.2, PHP, MySQL

ABSTRACT

PT. Bank Negara Indonesia (Persero) is the oldest bank institution in the history of the Republic of Indonesia owned by BUMN (State-Owned Enterprises). At present BNI has 1076 branch offices in Indonesia and 5 abroad. One branch office is located in Malang which is a regional office. BNI Malang Regional Office has a Research and Development Region (RDR) Division which resides under Research & Development Management Unit where one of its tasks is to support both internal and Malang regional affairs within the IT scope. Bank Negara Indonesia Malang Regional Office conducts a sorting process for prospective borrowers who potentially failed to pay off loans. Because prospective borrowers' data collection process is still done manually, communication and reporting between related officers are often hampered and the running process is not monitored well. Therefore a web-based information system was developed to conduct online monitoring that simplifies the debtor lending process so as to increase the efficiency of the lending process and save time from the authorized officers. The BROMO Lending Information System was developed using the SDLC waterfall method. The website implementation uses the Laravel 6.2 framework and the display implementation uses the Bootstrap 4.0 framework. Then the system is tested using a black box testing to find out the system needs have been met. The results of the validation test resulted in all needs being met, and the results of compatibility and responsive testing found the system can run on 8 types of browsers and is responsive on tablet and desktop.

Keywords : *Lending Information System, Waterfall , Framework , Laravel 6.2, PHP, MySQL*

DAFTAR ISI

LAPORAN PRAKTIK KERJA LAPANGAN (PKL)	i
PENGESAHAN	ii
PERNYATAAN ORISINALITAS	iii
KATA PENGANTAR.....	iv
ABSTRAK.....	v
ABSTRACT.....	vi
DAFTAR ISI	vii
DAFTAR TABEL.....	xiii
DAFTAR GAMBAR	xv
DAFTAR LAMPIRAN	xvii
BAB 1 PENDAHULUAN.....	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Tujuan.....	2
1.4 Batasan Masalah	2
1.5 Manfaat	3
1.6 Waktu dan Tempat Pelaksanaan.....	3
1.7 Sistematika Penulisan.....	3
BAB 2 PROFIL OBJEK PKL	5
2.1 Profil PT. Bank Negara Indonesia (Persero), Tbk.	5
2.2 Visi & Misi PT. Bank Negara Indonesia (Persero), Tbk.	5
2.3 Struktur Organisasi PT. Bank Negara Indonesia Kantor Wilayah Malang.....	6
2.3.1 Divisi <i>Research and Development Region</i>	7
2.4 Proses Bisnis.....	7
BAB 3 TINJAUAN PUSTAKA.....	9
3.1 Sistem Informasi.....	9
3.1.1 Pengembangan Sistem Informasi.....	9
3.1.2 Rekayasa Kebutuhan	10
3.1.3 <i>Unified Modelling Language</i>	10
3.2 Teknologi Pengembangan Sistem	17

3.2.1 <i>System Development Life Cycle</i> (SDLC)	17
3.2.1.1 Metode Pengembangan <i>Waterfall</i>	17
3.2.2 Aplikasi Berbasis Web	19
3.2.3 Bahasa Pemrograman	19
3.2.3.1 HTML.....	19
3.2.3.2 PHP.....	19
3.2.3.3 CSS.....	19
3.2.3.4 Javascript.....	20
3.2.3.5 Database Management System	20
3.2.3.6 MySQL	20
3.2.4 <i>Software Pendukung</i>	20
3.2.4.1 Visual Studio Code	20
3.2.5 <i>Framework</i>	20
3.2.5.1 <i>Model-View-Controller</i>	21
3.2.5.2 Laravel.....	22
3.2.5.3 Bootstrap	25
3.6 Pengujian Perangkat Lunak	28
3.6.1 Black Box Testing.....	29
3.6.2 Compatibility Testing	29
3.6.3 Responsive Web Design (RWD) Testing	30
BAB 4 METODOLOGI	32
4.1 Diskusi dengan Pembimbing Lapangan.....	32
4.2 Studi Literatur.....	33
4.3 Analisis Kebutuhan	33
4.4 Perancangan Sistem	33
4.5 Implementasi.....	33
4.6 Pengujian Sistem	34
4.7 Kesimpulan dan Saran	34
BAB 5 ANALISIS KEBUTUHAN	35
5.1 Deskripsi Umum Sistem	35
5.2 Identifikasi Aktor	36
5.3 Analisis Kebutuhan Fungsional.....	37

5.4 Analisis Kebutuhan Non Fungsional	39
5.5 <i>Use Case Diagram</i>	39
5.6 Skenario <i>Use Case</i>	40
5.7 <i>Activity Diagram</i>	50
5.7.1 <i>Activity Diagram Login</i>	50
5.7.2 <i>Activity Diagram Input Data Lead Manual</i>	51
5.7.3 <i>Activity Diagram Input Data Lead Otomatis</i>	52
5.7.4 <i>Activity Diagram Solicit</i>	53
5.7.5 <i>Activity Diagram Change Solicit</i>	54
5.7.6 <i>Activity Diagram Be Prospect</i>	55
5.7.7 <i>Activity Diagram Approval Prospect</i>	56
5.7.8 <i>Activity Diagram Be Pipeline</i>	57
5.7.9 <i>Activity Diagram Approval Pipeline</i>	58
5.7.10 <i>Recheck</i>	59
5.7.11 <i>Refuse Data Lead</i>	60
5.7.12 Melihat <i>Refuse Data Lead</i>	61
5.7.13 <i>Activity Diagram Notification</i>	62
BAB 6 PERANCANGAN	64
6.1 Arsitektur Sistem	64
6.2 Pemodelan <i>Sequence Diagram</i>	64
6.2.1 <i>Sequence Diagram Login</i>	65
6.2.2 <i>Sequence Diagram Input Data Lead</i>	65
6.2.3 <i>Sequence Diagram Solicit</i>	66
6.2.4 <i>Sequence Diagram Be Prospect</i>	67
6.2.5 <i>Sequence Diagram Approval Prospect</i>	67
6.2.6 <i>Sequence Diagram Be Pipeline</i>	68
6.2.7 <i>Sequence Diagram Approval Pipeline</i>	69
6.3. Pemodelan <i>Class Diagram</i>	70
6.3.1. Pemodelan <i>Class Diagram</i> untuk <i>Auth Gateway</i>	70
6.3.2. Pemodelan <i>Class Diagram</i> untuk <i>Sales Services</i>	71
6.3.3. Pemodelan <i>Class Diagram</i> untuk <i>Penyelia Services</i>	72

6.3.4. Pemodelan <i>Class Diagram</i> untuk PBP Services	73
6.4. Pemodelan PDM (Physical Data Model)	74
6.4.1. Physical Data Model Proses Lending.....	75
6.4.1.1. Perancangan Tabel Debitur.....	75
6.4.1.2. Perancangan Tabel Data Lead.....	76
6.4.1.3. Perancangan Tabel Cdatalead.....	76
6.4.1.4. Perancangan Tabel Solicit	77
6.4.1.5. Perancangan Tabel Jkredit	77
6.4.1.6. Perancangan Tabel Dproduk.....	78
6.4.1.7. Perancangan Tabel User	78
6.4.1.8. Perancangan Tabel Status.....	79
6.4.1.9. Perancangan Tabel Outlet.....	79
6.4.1.10. Perancangan Tabel Jpekerjaan	79
6.4.2 <i>Physical Data Model</i> Lokasi	80
6.4.2.1. Perancangan Tabel Kabupaten	80
6.4.2.2. Perancangan Tabel Kecamatan.....	80
6.4.2.3. Perancangan Tabel Desa	80
6.4.3. Physical Data Model Notifikasi.....	81
6.4.3.1. Perancangan Tabel Notification.....	81
6.5. Perancangan <i>User Interface</i>	81
6.5.1. User Interface Login	82
6.5.2. <i>User Interface</i> Dashboard	82
6.5.3. <i>User Interface</i> Input Data Lead	83
6.5.4. <i>User Interface</i> Approval	84
BAB 7 IMPLEMENTASI	85
7.1 Implementasi <i>User Interface</i>	85
7.1.1 <i>User Interface</i> Login	85
7.1.2 <i>User Interface</i> Input Data Lead	86
7.1.3 <i>User Interface</i> List Data Lead	87
7.1.4 <i>User Interface</i> Solicit	88
7.1.5 <i>User Interface</i> Approval Prospect	89
7.1.6 <i>User Interface</i> Prospect.....	90
7.1.7 <i>User Interface</i> Approval Pipeline.....	92

7.1.8 <i>User Interface Pipeline</i>	93
7.2 Implementasi <i>Database</i>	94
7.2.1 Implementasi Table Users.....	96
7.2.2 Implementasi Tabel Debitur.....	96
7.2.3 Implementasi Tabel Data Lead.....	97
7.2.4 Implementasi Tabel Refuse Data Lead (cdatalead).....	97
7.2.5 Implementasi Tabel Solicit	98
7.2.6 Implementasi Tabel Jkredit	99
7.2.7 Implementasi Tabel Kabupaten	99
7.2.8 Implementasi Tabel Kecamatan	100
7.2.9 Implementasi Tabel Desa	100
7.2.10 Implementasi Tabel Outlet.....	100
7.2.11 Implementasi Tabel Dproduk	101
7.2.12 Implementasi Tabel Jpekerjaan	101
7.2.13 Implementasi Tabel Notification.....	101
7.2.14 Implementasi Tabel Status.....	102
7.2.15 Implementasi Tabel Kewenangan	102
7.3. Implementasi Kode Program.....	103
7.3.1. Implementasi <i>Login</i>	103
7.3.2. Implementasi <i>Input Datalead</i>	104
7.3.3. Implementasi <i>datalead</i> ke <i>Solicit</i>	105
7.3.4. Implementasi <i>Request Prospect</i>	106
7.3.5. Implementasi <i>Approval & Reject Prospect</i>	107
7.3.6. Implementasi <i>Request Pipeline</i>	109
7.3.7. Implementasi <i>Approval, Recheck & Reject Pipeline</i>	110
7.3.8. Implementasi <i>Input Datalead</i> Otomatis	113
BAB 8 PENGUJIAN.....	114
8.1 Hasil Pengujian Menggunakan <i>Black Box Testing</i>	114
8.2 Hasil Pengujian Menggunakan <i>Compatibility Testing</i>	124
8.3 Hasil Pengujian Menggunakan <i>Responsive Web Design (RWD) Testing</i>	125
BAB 9 PENUTUP.....	128

9.1 Kesimpulan	128
9.2 Saran.....	129
DAFTAR PUSTAKA.....	130
DAFTAR LAMPIRAN	133

DAFTAR TABEL

Tabel 3.1 Simbol-simbol pada <i>Flowchart</i>	11
Tabel 3.2 Komponen-komponen pada <i>Activity Diagram</i>	12
Tabel 3.3 Komponen-komponen pada <i>Sequence Diagram</i>	13
Tabel 3.4 Komponen-komponen pada <i>Use Case Diagram</i>	15
Tabel 3.5 Komponen-komponen pada <i>Class Diagram</i>	15
Tabel 3.6 Kode Program Employee.html	22
Tabel 3.7 Kode Program Employee.blade.php.....	23
Tabel 3.8 Contoh Penggunaan container	26
Tabel 3.9 Contoh Penggunaan row	26
Tabel 3.10 Contoh Penggunaan column	26
Tabel 3.11 Titik-Titik <i>Responsive Breakpoints</i>	27
Tabel 3.12 Contoh Penggunaan <i>Responsive Breakpoints</i>	27
Tabel 3.13 Nilai Standar Z-index pada kelas Bootstrap.....	28
Tabel 5.1 Identifikasi Aktor	37
Tabel 5.2 Kebutuhan Fungsional	37
Tabel 5.3 Kebutuhan Non Fungsional	39
Tabel 5.4 Skenario <i>Use Case Login</i>	40
Tabel 5.5 Skenario <i>Use Case Input Data Lead Manual</i>	41
Tabel 5.6 Skenario <i>Use Case Input Data Lead Otomatis</i>	42
Tabel 5.7 Skenario <i>Use Case Solicit</i>	43
Tabel 5.8 Skenario <i>Use Case Change Solicit</i>	43
Tabel 5.9 Skenario <i>Use Case Be Prospect</i>	44
Tabel 5.10 Skenario <i>Use Case Prospect</i>	45
Tabel 5.11 Skenario <i>Use Case Be Pipeline</i>	45
Tabel 5.12 Skenario <i>Use Case Approval Pipeline</i>	46
Tabel 5.13 Skenario <i>Use Case Recheck</i>	47
Tabel 5.14 Skenario <i>Use Case Refuse Data Lead</i>	48
Tabel 5.15 Skenario <i>Use Case Melihat Refuse Data Lead</i>	48
Tabel 5.16 Skenario <i>Use Case Approval Notification</i>	49
Tabel 5.17 Skenario <i>Use Case Request Notification</i>	49
Tabel 7.1 Kode Program Membuat Tabel Users	96
Tabel 7.2 Kode Program Membuat Tabel Debitur	96
Tabel 7.3 Kode Program Membuat Tabel Data lead	97
Tabel 7.4 Kode Program Membuat Tabel Refuse Data lead	97
Tabel 7.5 Kode Program Membuat Tabel Solicit.....	98
Tabel 7.6 Kode Program Membuat Tabel JKredit	99
Tabel 7.7 Kode Program Membuat Tabel Kabupaten.....	99
Tabel 7.8 Kode Program Membuat Tabel Kecamatan	100
Tabel 7.9 Kode Program Membuat Tabel Desa	100

Tabel 7.10 Kode Program Membuat Tabel Outlet	100
Tabel 7.11 Kode Program Membuat Tabel Detail Produk	101
Tabel 7.12 Kode Program Membuat Tabel Jenis Pekerjaan	101
Tabel 7.13 Kode Program Membuat Tabel Notifikasi	101
Tabel 7.14 Kode Program Membuat Tabel Status	102
Tabel 7.15 Kode Program Membuat Tabel Kewenangan	102
Tabel 7.16 Kode Program <i>Method postLogin</i> pada <i>Auth Gateway</i>	103
Tabel 7.17 Kode Program <i>Method inputManual</i> pada Sales, Penyelia, dan PBP	104
Tabel 7.18 Kode Program <i>Method create</i> pada Sales, Penyelia, dan PBP	105
Tabel 7.19 Kode Program <i>Method createSolicit</i> pada Sales	106
Tabel 7.20 Kode Program <i>Method beProspect</i> pada Sales	107
Tabel 7.21 Kode Program <i>Method approveProspect</i> pada Penyelia	108
Tabel 7.22 Kode Program <i>Method submitProspectToPbp</i> pada Sales	109
Tabel 7.23 Kode Program <i>Method approvePipeline</i> pada PBP	111
Tabel 7.24 Kode Program <i>Method import_excel</i> pada Penyelia dan PBP	113
Tabel 8.1 Pengujian Pada Proses Login	114
Tabel 8.2 Pengujian pada Proses Input Data Lead Manual.....	115
Tabel 8.3 Pengujian pada Proses Input Data Lead Otomatis.....	116
Tabel 8.4 Pengujian pada Proses Solicit	116
Tabel 8.5 Pengujian pada Proses Change Solicit.....	117
Tabel 8.6 Pengujian pada Proses Be Prospect	118
Tabel 8.7 Pengujian pada Proses Be Prospect	118
Tabel 8.8 Pengujian pada Proses Prospect	119
Tabel 8.9 Pengujian pada Proses Prospect	119
Tabel 8.10 Pengujian pada Proses Be Pipeline.....	120
Tabel 8.11 Pengujian pada Proses Approval Pipeline	120
Tabel 8.12 Pengujian pada Proses Recheck	121
Tabel 8.13 Pengujian pada Proses Reject (Prospect)	121
Tabel 8.14 Pengujian pada Proses Reject (Pipeline)	122
Tabel 8.15 Pengujian pada Proses Melihat Refuse Data Lead	123
Tabel 8.16 Pengujian pada Proses Approval Notification	123
Tabel 8.17 Pengujian pada Proses Request Notification.....	124
Tabel 8.18 Browser untuk Compatibility Testing	124
Tabel 8.19 Perangkat untuk <i>Responsive Web Design Testing</i>	125
Tabel 8.20 Hasil Responsive Web Design Testing pada PC.....	126
Tabel 8.21 Hasil <i>Responsive Web Design Testing</i> pada Tablet	126

DAFTAR GAMBAR

Gambar 2.1 Struktur Organisasi BNI Kantor Wilayah Malang	6
Gambar 3.1 <i>Systems Development Life Cycle</i>	9
Gambar 3.2 Alur Metode Pengembangan Waterfall	18
Gambar 3.3 Konsep MVC	21
Gambar 3.4 Grid System Bootstrap	25
Gambar 3.5 Ilustrasi Responsive Web Design dari Sistem Informasi Lending BROMO.....	30
Gambar 4.1 Diagram Alir Pembuatan Sistem Informasi <i>Lending</i> BROMO.....	32
Gambar 5.1 <i>Flowchart</i> fase-fase peminjaman pada Sistem Informasi <i>Lending</i> BROMO.....	36
Gambar 5.2 <i>Use Case Diagram</i> Sistem Informasi <i>Lending</i> BROMO	40
Gambar 5.3 <i>Activity Diagram</i> Login.....	51
Gambar 5.4 <i>Activity Diagram</i> Input Data Lead Manual	52
Gambar 5.5 <i>Activity Diagram</i> Input Data Lead Otomatis	53
Gambar 5.6 <i>Activity Diagram</i> Solicit	54
Gambar 5.7 <i>Activity Diagram</i> Change Solicit.....	55
Gambar 5.8 <i>Activity Diagram</i> Be Prospect.....	56
Gambar 5.9 <i>Activity Diagram</i> Approval Prospect	57
Gambar 5.10 <i>Activity Diagram</i> Be Pipeline	58
Gambar 5.11 <i>Activity Diagram</i> Approval Pipeline	59
Gambar 5.12 <i>Activity Diagram</i> Recheck	60
Gambar 5.13 <i>Activity Diagram</i> Refuse Data Lead	61
Gambar 5.14 <i>Activity Diagram</i> Melihat Refuse Data Lead	62
Gambar 5.15 <i>Activity Diagram</i> Notification	63
Gambar 6.1 Arsitektur Sistem Informasi <i>Lending</i> BROMO	64
Gambar 6.2 <i>Sequence Diagram</i> Login	65
Gambar 6.3 <i>Sequence Diagram</i> Input Data Lead	66
Gambar 6.4 <i>Sequence Diagram</i> Solicit	66
Gambar 6.5 <i>Sequence Diagram</i> Be Prospect	67
Gambar 6.6 <i>Sequence Diagram</i> Approval Prospect	68
Gambar 6.7 <i>Sequence Diagram</i> Be Pipeline.....	69
Gambar 6.8 <i>Sequence Diagram</i> Approval Pipeline	70
Gambar 6.9 Perancangan <i>Class Diagram</i> Auth Gateway.....	71
Gambar 6.10 Perancangan <i>Class Diagram</i> Sales Services.....	72
Gambar 6.11 Perancangan <i>Class Diagram</i> Penyelia Services	73
Gambar 6.12 Perancangan <i>Class Diagram</i> PBP Services.....	74
Gambar 6.13 <i>PDM</i> Proses <i>Lending</i>	75
Gambar 6.14 <i>PDM</i> Lokasi	80
Gambar 6.15 <i>PDM</i> Notifikasi.....	81

Gambar 6.16 Perancangan <i>User Interface Login</i>	82
Gambar 6.17 Perancangan <i>User Interface Dashboard</i>	83
Gambar 6.18 Perancangan <i>User Interface Data Lead</i>	83
Gambar 6.19 Perancangan <i>User Interface Approval</i>	84
Gambar 7.1 Halaman Login	85
Gambar 7.2 Dashboard	86
Gambar 7.3 Input Data Lead	86
Gambar 7.4 List Data Lead	87
Gambar 7.5 <i>Pop Up</i> Ketika Menekan Tombol <i>Solicit</i>	88
Gambar 7.6 All <i>Solicit</i>	89
Gambar 7.7 <i>Pop Up</i> Ketika Menekan Button <i>Be Prospect</i>	89
Gambar 7.8 Approval <i>Prospect</i>	90
Gambar 7.9 <i>Pop Up</i> Pada Halaman <i>Approval Prospect</i>	90
Gambar 7.10 <i>Prospect</i>	91
Gambar 7.11 <i>Pop Up</i> Pada Halaman <i>Prospect</i>	91
Gambar 7.12 <i>Approval Pipeline</i>	92
Gambar 7.13 <i>Pop Up</i> Pada Halaman <i>Approval Pipeline</i>	93
Gambar 7.14 <i>Pipeline</i>	94
Gambar 7.15 Implementasi <i>Database Sistem Informasi Lending BROMO</i>	95
Gambar 8.1 Hasil <i>Compatibility Testing</i>	125

DAFTAR LAMPIRAN

Gambar Lampiran 1 Foto kegiatan bersama BNI Wilayah Malang	133
Gambar Lampiran 2 Foto kegiatan bersama BNI Wilayah Malang	133
Gambar Lampiran 3 Daftar Kehadiran Muhammad Kevin Sandryan	134
Gambar Lampiran 4 Daftar Kehadiran Muhammad Kevin Sandryan	134
Gambar Lampiran 5 Daftar Kehadiran Muhammad Rasyid Al Faruqi.....	135
Gambar Lampiran 6 Daftar Kehadiran Muhammad Rasyid Al Faruqi.....	135
Gambar Lampiran 7 Daftar Kehadiran Mohammad Raska	136
Gambar Lampiran 8 Daftar Kehadiran Mohammad Raska	136

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Meningkatnya perkembangan teknologi informasi telah memberikan dampak yang besar bagi suatu perusahaan dan jajaran direksinya. Sistem informasi diyakini banyak pihak sebagai sebuah alat yang penting dalam perusahaan yang memiliki rutinitas yang tinggi, karena sistem informasi memberikan kontribusi terhadap kebutuhan untuk membangun keunggulan kompetitif melalui biaya yang rendah, kualitas yang lebih baik, dan peningkatan pelayanan terhadap konsumen.

Sistem informasi terus mengalami perkembangan dari waktu ke waktu dalam peningkatan cakupan sistem, fungsi, maupun teknologi yang digunakan. Teori pengembangan sistem informasi juga terus mengalami perkembangan sehingga terciptanya metodologi-metodologi baru yang bermunculan seperti *Agile* dan *DevOps* yang belum lama ditemukan dari metodologi-metodologi pendahulunya seperti SDLC (*Systems Development Life Cycle*), *Waterfall*, dan RAD (*Rapid Application Development*). Setiap metodologi pengembangan sistem informasi memiliki kelebihan dan kelemahannya masing-masing dan digunakan untuk proyek dengan sifat tertentu sehingga tidak ada metodologi yang sempurna untuk digunakan pada semua proyek. Model *Waterfall* sangat cocok digunakan pada proyek yang berukuran kecil dan gambaran dari sebuah produk sudah diurai secara jelas (Dimas Gustino, 2018).

Dalam istilah perbankan, *lending* dapat diartikan sebagai kegiatan dimana orang yang dianggap sebagai debitur mengajukan permohonan dana kepada pihak bank sebagai pinjaman. Proses *lending* di perbankan umumnya terdapat beberapa tahap dan pihak bank perlu melakukan tahapan registrasi dan verifikasi berdasarkan data dan rekam jejak dari debitur. Pada umumnya, terdapat 2 jenis *lending* yaitu *lending* produktif dan *lending* konsumtif. *Lending* produktif diberikan kepada debitur pelaku usaha atau bisnis, sementara *lending* konsumtif adalah pinjaman dana yang diberikan kepada debitur untuk keperluan pribadi seperti kredit motor dan KPR (Kredit Kepemilikan Rumah).

Proses *lending* produktif maupun *lending* konsumtif di PT. Bank Negara Indonesia (Persero) Tbk. Kantor Wilayah Malang selama ini masih dilakukan secara manual dan belum tersistematis dengan meminta laporan kepada masing-masing cabang pada setiap bulannya untuk diproses pada bulan selanjutnya. Data yang sedang diproses juga belum tercatat dengan baik dan belum memiliki pemantauan harian untuk mengetahui perkembangan proses pengajuan *lending*. Mengetahui kekurangan tersebut, dibutuhkan sistem informasi yang dapat melakukan pencatatan dan pemantauan proses *lending* secara harian.

Diharapkan dengan adanya Sistem Informasi *Lending* BROMO dapat mempermudah proses monitoring debitur sehingga meningkatkan efisiensi dari proses *lending* dan menghemat waktu dari petugas yang bersangkutan. Sistem Informasi *Lending* BROMO dikembangkan menggunakan platform *web* dengan bahasa pemrograman PHP menggunakan *framework* Laravel 6.2. Platform *web* merupakan pilihan dalam mengembangkan sistem informasi ini karena dianggap lebih mudah diakses dari PC, laptop, tablet, maupun *mobile*. *Framework* Laravel adalah *framework* pembuatan *web* yang menggunakan bahasa PHP, mudah dipelajari, dan memiliki banyak *community support* sehingga hal tersebut menjadi alasan kami untuk menggunakan dalam proyek ini. Hingga pada pembuatan laporan Praktik Kerja Lapangan ini, Sistem Informasi *Lending* BROMO masih dalam tahap pengembangan dikarenakan keterbatasan waktu dan bertambahnya *requirement* seiring dengan pelaksanaan Praktik Kerja Lapangan.

1.2 Rumusan Masalah

Dari latar belakang sebelumnya maka dapat dibuat rumusan masalah sebagai berikut :

1. Bagaimana analisis kebutuhan dari Sistem Informasi *Lending* BROMO berbasis web di PT. Bank Negara Indonesia Kantor Wilayah Malang?
2. Bagaimana rancangan dan implementasi dari Sistem Informasi *Lending* BROMO berbasis web di PT. Bank Negara Indonesia Kantor Wilayah Malang?
3. Bagaimana hasil pengujian dari Sistem Informasi *Lending* BROMO berbasis web di PT. Bank Negara Indonesia Kantor Wilayah Malang?

1.3 Tujuan

Berdasarkan rumusan masalah di atas, maka tujuan penulisan laporan ini akan menjawab permasalahan tersebut antara lain :

1. Mengetahui analisis kebutuhan dari Sistem Informasi *Lending* BROMO berbasis web di PT. Bank Negara Indonesia Kantor Wilayah Malang.
2. Mengetahui rancangan dan implementasi dari Sistem Informasi *Lending* BROMO berbasis web di PT. Bank Negara Indonesia Kantor Wilayah Malang.
3. Mengetahui hasil pengujian dari Sistem Informasi *Lending* BROMO berbasis web di PT. Bank Negara Indonesia Kantor Wilayah Malang.

1.4 Batasan Masalah

Adapun hal-hal yang menjadi batasan masalah dalam pelaksanaan Praktik Kerja Lapangan yaitu :

1. Sistem Informasi *Lending* BROMO tidak dipublikasikan untuk masyarakat umum tetapi digunakan oleh internal PT. Bank Negara Indonesia Kantor Wilayah Malang.
2. Sistem Informasi *Lending* BROMO merupakan aplikasi berbasis web yang dikembangkan menggunakan *framework* Laravel versi 6.2 dan *database* MySQL.
3. Sistem informasi ini hanya menyediakan proses pengajuan dana (*lending*) dan belum termasuk proses pengembalian dana (*funding*).

1.5 Manfaat

Adapun manfaat dari pengembangan Sistem Informasi *Lending* BROMO yang dikerjakan selama pelaksanaan Praktik Kerja Lapangan yaitu untuk menghasilkan sistem informasi yang dapat digunakan oleh PT. Bank Negara Indonesia Kantor Wilayah Malang.

1.6 Waktu dan Tempat Pelaksanaan

Praktik Kerja Lapangan (PKL) ini dilaksanakan pada tanggal 28 Januari 2020 – 24 April 2020 di PT. Bank Negara Indonesia Kantor Wilayah Malang yang berlokasi di Jl. Jendral Basuki Rahmat no.75-77, Kauman, Kec. Klojen, Kota Malang, Jawa Timur 65119.

1.7 Sistematika Penulisan

BAB 1 Pendahuluan

Bab ini membahas mengenai latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat, waktu dan tempat pelaksanaan Praktik Kerja Lapangan serta sistematika penulisan laporan Praktik Kerja Lapangan.

BAB 2 Profil Objek PKL

Bab ini menjelaskan gambaran profil perusahaan PT. Bank Negara Indonesia Kantor Wilayah Malang.

BAB 3 Tinjauan Pustaka

Bab ini akan menyajikan kumpulan teori yang membantu pelaksanaan Sistem Informasi *Lending* BROMO selama Praktik Kerja Lapangan.

BAB 4 Metodologi

Bab ini akan membahas metodologi yang digunakan dalam penyusunan laporan dan penggerjaan Sistem Informasi *Lending BROMO* selama Praktik Kerja Lapangan.

BAB 5 Analisis Kebutuhan

Bab ini membahas hal-hal terkait proses penggalian kebutuhan sistem dan identifikasi aktor dalam pengembangan Sistem Informasi *Lending BROMO*.

BAB 6 Perancangan

Bab ini akan menguraikan perancangan pengembangan Sistem Informasi *Lending BROMO* berdasarkan analisis kebutuhan yang telah dilakukan.

BAB 7 Implementasi

Bab ini membahas tentang implementasi dari perancangan sistem informasi dalam bentuk desain *user interface*, kode program, dan *database*. Pengembangan sistem informasi diimplementasikan berdasarkan rancangan model yang telah dilakukan sebelumnya.

BAB 8 Pengujian

Bab ini akan memuat tentang pengujian sistem yang telah dibangun dengan menggunakan metode *Black Box Testing*.

BAB 9 Penutup

Bab ini akan menguraikan kesimpulan dan saran berdasarkan hasil pengembangan Sistem Informasi *Lending BROMO*.

BAB 2 PROFIL OBJEK PKL

2.1 Profil PT. Bank Negara Indonesia (Persero), Tbk.

PT. Bank Negara Indonesia (Persero), Tbk (selanjutnya disebut "BNI") merupakan institusi perbankan yang dimiliki oleh BUMN (Badan Usaha Milik Negara). Pada awalnya, BNI didirikan sebagai bank sentral bernama "Bank Negara Indonesia" berdasarkan Peraturan Pemerintah Pengganti Undang-Undang No 2 tahun 1946 tanggal 5 Juli 1946. Selanjutnya, berdasarkan Undang-Undang No 17 tahun 1968, BNI ditetapkan menjadi "Bank Negara Indonesia 1946", dan statusnya menjadi Bank Umum Milik Negara. Peran BNI sebagai bank diberikan mandat untuk memperbaiki ekonomi rakyat dan ikut serta dalam pembangunan nasional dikukuhkan oleh UU No 17 tahun 1968 tentang Bank Negara Indonesia 1946.

BNI menjadi bank pertama yang menjadi perusahaan publik setelah mencatatkan sahamnya di Bursa Efek Jakarta dan Bursa Efek Surabaya pada tahun 1996. 60% saham BNI dimiliki oleh Pemerintah Republik Indonesia, sedangkan 40% dimiliki oleh masyarakat, baik oleh individu, institusi, domestik dan pihak asing. Dilihat dari total aset, total kredit maupun total dana pihak ketiga, BNI kini tercatat sebagai bank nasional terbesar ke-4 di Indonesia. Hingga akhir tahun 2019, BNI telah memiliki total aset sebesar Rp 845,61 triliun. Selain itu, jaringan layanan BNI berada di 1.585 outlet yang tersebar di seluruh Indonesia dan telah berhasil merambah hingga ke Singapura, Hong Kong, Tokyo, Osaka, Seoul, Yangon, London dan New York. BNI juga memiliki 8.227 unit ATM, 42.000 EDC serta fasilitas internet dan SMS banking yang dapat memanjakan nasabah. Perkembangan BNI juga dibantu melalui beberapa anak perusahaannya seperti Bank BNI Syariah, BNI Multi Finance, BNI Securities dan BNI Life Insurance.

BNI menawarkan layanan penyimpanan dana atau *funding* (Simpanan) maupun fasilitas pinjaman atau *lending* (Pinjaman) baik pada segmen korporasi, menengah, maupun kecil. Beberapa produk dan layanan terbaik telah disesuaikan dengan kebutuhan nasabah sejak kecil, remaja, dewasa, hingga pensiunan.

2.2 Visi & Misi PT. Bank Negara Indonesia (Persero), Tbk.

Visi :

Menjadi Lembaga Keuangan yang Unggul dalam Layanan dan Kinerja. BNI berupaya menjadi Bank yang menunjukkan kinerja unggul untuk memberikan nilai investasi yang memuaskan bagi para pemegang saham, menjadi *the bank of choice* dengan menyajikan kualitas layanan terbaik,

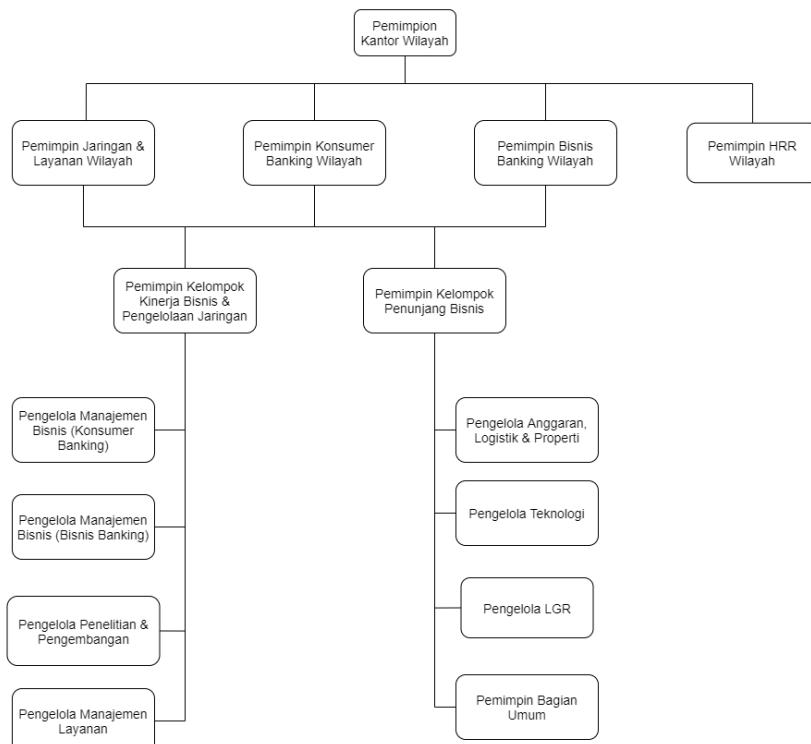
serta menjadi *dominant player (market leader)* dengan menyajikan produk dan jasa yang bernilai tinggi di segmen pasar yang dilayani.

Misi :

- Memberikan layanan prima dan solusi yang bernilai tambah kepada seluruh nasabah, dan selaku mitra pilihan utama.
- Meningkatkan nilai investasi yang unggul bagi investor.
- Menciptakan kondisi terbaik bagi karyawan sebagai tempat kebanggaan untuk berkarya dan berprestasi.
- Meningkatkan kepedulian dan tanggung jawab kepada lingkungan dan komunitas.
- Menjadi acuan pelaksanaan kepatuhan dan tata kelola perusahaan yang baik.

2.3 Struktur Organisasi PT. Bank Negara Indonesia Kantor Wilayah Malang

Gambar 2.1 menunjukkan jajaran pengurus dan Struktur Organisasi pada PT. Bank Negara Indonesia Kantor Wilayah Malang. Selama kami melakukan Praktik Kerja Lapangan, kami berada di Divisi *Research and Development Region* yang dikepalai oleh Unit Pengelola Penelitian dan Pengembangan.



Gambar 2.1 Struktur Organisasi BNI Kantor Wilayah Malang

Sumber: PT. Bank Negara Indonesia Kantor Wilayah Malang (2020)

2.3.1 Divisi *Research and Development Region*

Divisi *Research and Development Region* (Disingkat RDR) merupakan salah satu divisi yang ada di PT. Bank Negara Indonesia Kantor Wilayah Malang yang berada di bawah Unit Pengelola Penelitian & Pengembangan. Divisi RDR memiliki tugas untuk melakukan penelitian dan pengembangan dari produk yang dikeluarkan oleh BNI Kantor Wilayah Malang. RDR juga memiliki tugas untuk mengelola kebutuhan IT untuk menunjang kegiatan internal atau kepegawaian di BNI Kantor Wilayah Malang. Salah satunya adalah dengan membuat Sistem Informasi *Lending* BROMO berbasis *web* yang tujuannya mempermudah proses monitoring debitur sehingga meningkatkan efisiensi dari proses *lending* dan menghemat waktu dari petugas yang bersangkutan.

2.4 Proses Bisnis

Dalam mengajukan pinjaman (*lending*) di BNI Wilayah Malang, seorang calon debitur harus menghadapi lima fase yang harus dilewati. Fase-fase ini bertujuan untuk menyeleksi calon debitur yang sesuai dengan standar dari BNI Wilayah Malang untuk diberikan pinjaman. Fase-fase tersebut diantaranya yaitu Data Lead, Refuse Data Lead, Solicit, Prospect, dan Pipeline. Berikut adalah penjelasan masing-masing fase:

1. Data Lead

Fase ini merupakan fase awal di mana data debitur akan diinputkan ke dalam sistem oleh SCO, Penyelia, atau PBP secara manual dengan mengisi form di dalam *web* maupun otomatis dengan mengupload file dokumen excel. Data yang sudah diinputkan akan terdaftar ke dalam sistem sebagai Data Lead.

2. Refuse Data Lead

Data ini merupakan data debitur yang ditolak dan tidak dapat diajukan kembali. Sumber data dari Refuse Data Lead adalah data Solicit yang tidak mendapatkan persetujuan Penyelia dan data Prospect yang tidak mendapatkan persetujuan PBP.

3. Solicit

SCO dapat mengambil *Data Lead* untuk dilakukan pemrosesan lebih lanjut dengan melakukan *follow up* kepada debitur. Pada waktu ini, data memiliki status *Solicit*. Debitur yang setuju untuk melakukan *lending* akan diproses menuju tahap selanjutnya dengan persetujuan Penyelia (menjadi *Prospect*) atau dikembalikan menjadi *Data Lead* dengan catatan dan *tag* lokasi debitur. *Solicit* yang dikembalikan menjadi *Data Lead* dapat diambil kembali oleh SCO yang berbeda.

4. Prospect

Pada fase ini, SCO melakukan *interview* lebih lanjut ke debitur perihal kredit di bank lain, usaha/pekerjaan yang dijalankan dan memberikan informasi lebih lanjut tentang fasilitas kredit layanan maupun produk lain yang dimiliki BNI. Jika debitur memenuhi syarat kelayakan, data debitur akan diteruskan ke PBP untuk persetujuan ke proses selanjutnya (menjadi Pipeline).

5. Pipeline

Setelah Prospect disetujui oleh PBP, data akan menjadi berstatus Pipeline yang artinya debitur sudah akan menjalani proses *lending* dan proses *monitoring*.

Selama ini pemrosesan *lending* di BNI Wilayah Malang masih dilakukan secara manual dan belum tersistematis dengan meminta laporan kepada masing-masing cabang pada setiap bulannya untuk diproses pada bulan selanjutnya. Data yang sedang diproses juga belum tercatat dengan baik dan belum memiliki pemantauan harian untuk mengetahui perkembangan proses pengajuan lending. Mengetahui kekurangan tersebut, Unit Research and Development Regional memiliki peran untuk merancang dan membangun sebuah sistem informasi yang dapat melakukan pencatatan dan pemantauan proses *lending* secara harian.

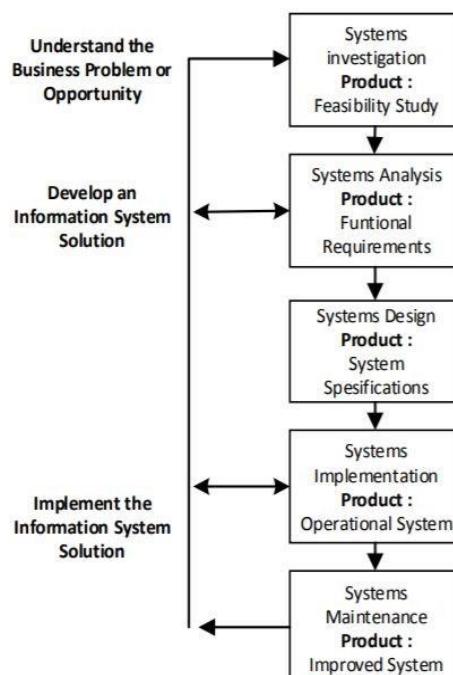
BAB 3 TINJAUAN PUSTAKA

3.1 Sistem Informasi

Sistem merupakan sekelompok komponen yang saling terintegrasi satu sama lain untuk mencapai tujuan bersama dengan menerima masukan (*input*) dan menghasilkan keluaran (*output*) dalam proses yang teratur (O'brien & Marakas, 2010). Sedangkan pengertian informasi merupakan data yang telah dikonversi ke dalam konteks yang memiliki makna tertentu sehingga berguna bagi pengguna tertentu (O'brien & Marakas, 2010). Dari pengertian-pengertian tersebut, dapat ditarik kesimpulan bahwa sistem informasi merupakan sistem yang melakukan pengelolaan terhadap sumber daya data dan menghasilkan informasi yang berguna dalam sebuah organisasi.

3.1.1 Pengembangan Sistem Informasi

Pengembangan sistem informasi merupakan serangkaian prosedur yang digunakan untuk mengembangkan dan memelihara sistem informasi atau perangkat lunak. Salah satu pendekatan metode pengembangan sistem informasi adalah *Systems Development Life Cycle* (SDLC). SDLC dapat dilihat sebagai suatu tahapan yang dilakukan secara iteratif (O'brien & Marakas, 2010). Pada SDLC, terdapat beberapa fase untuk mengembangkan perangkat lunak. Fase-fase tersebut dijelaskan pada Gambar 3.1.



Gambar 3.1 *Systems Development Life Cycle*

Sumber: O'Brien (2010)

Dari gambar di atas, dapat diketahui bahwa SDLC terdiri dari 5 fase. Fase pertama adalah *investigation*. Fase ini bertujuan untuk menemukan dan memahami proses bisnis yang dijalankan, serta masalah dan peluang-peluangnya. Fase kedua dan ketiga adalah *analysis* dan *design*. Kedua fase ini bertujuan untuk menganalisis dan membuat rancangan dari sistem informasi yang akan menunjang proses bisnis. Kemudian fase ketiga dan keempat adalah *implementation* dan *maintenance*. Di fase ini akan dilaksanakan implementasi rancangan sistem informasi ke dalam bentuk kode-kode program. Setelah proses implementasi selesai, maka akan dilakukan pemeliharaan sistem informasi tersebut.

3.1.2 Rekayasa Kebutuhan

Rekayasa kebutuhan digunakan sebagai alat untuk mencapai kesepakatan antara *developer*, *client*, dan pengguna akhir (*end user*) terhadap kebutuhan yang harus dipenuhi, untuk menyediakan dasar yang akurat bagi perancangan dan pengembangan perangkat lunak (Sommerville I, 2011). Selain itu rekayasa kebutuhan juga digunakan untuk memvalidasi perangkat lunak yang akan dibuat. Proses Rekayasa Kebutuhan terdiri dari analisis kebutuhan, spesifikasi kebutuhan, validasi dan verifikasi kebutuhan, dan kebutuhan manajemen.

a. Penggalian dan Analisis kebutuhan

Proses Analisis kebutuhan sistem melalui pengamatan sistem yang ada dan melibatkan pengembangan satu atau lebih model *prototype*.

b. Spesifikasi Kebutuhan

Spesifikasi Kebutuhan adalah kegiatan menerjemahkan informasi yang dikumpulkan ketika proses analisis sedang dilakukan. Terdapat dua jenis kebutuhan, yaitu kebutuhan *User* dan *System*.

c. Validasi Kebutuhan

Validasi Kebutuhan adalah proses pengecekan untuk memastikan dan menjamin pernyataan kebutuhan yang telah dijabarkan dan dispesifikasikan telah tepat.

3.1.3 Unified Modelling Language

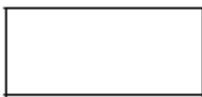
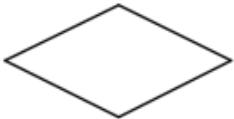
Unified Modelling Language (UML) adalah bahasa standar yang digunakan untuk memodelkan sistem atau perangkat lunak yang berparadigma berorientasi objek (Nugroho, 2010). UML dapat berguna sebagai *blue print* dari sebuah perangkat lunak sebab berisi tentang informasi detail mengenai kebutuhan-kebutuhan yang akan diimplementasikan. UML terdiri dari beberapa diagram untuk menggambarkan sistem, yaitu *flowchart*, *activity diagram*, *sequence diagram*, *use case diagram*, dan *class diagram*. Berikut diberikan penjelasan terkait model-model *diagram* UML.

a. *Flowchart*

Flowchart adalah gambar atau bagan yang memperlihatkan urutan atau langkah-langkah dari suatu program dan hubungan antar proses. *Flowchart* berguna sebagai fasilitas untuk berkomunikasi antara pemrogram yang bekerja dalam tim suatu proyek. Dalam pembuatan *flowchart* tidak ada rumus atau patokan yang bersifat mutlak karena *flowchart* merupakan gambaran hasil pemikiran dalam menganalisis suatu permasalahan yang nantinya akan diubah menjadi program komputer sehingga dapat bervariasi antara satu pemrogram dengan yang lainnya (Barakbah *et al.*, 2013).

Berikut adalah simbol-simbol *flowchart* standar yang dikeluarkan oleh ANSI dan ISO pada Tabel 3.1.

Tabel 3.1 Simbol-simbol pada *Flowchart*

Simbol	Nama	Keterangan
	<i>Terminator</i>	Simbol awal (<i>start</i>) / Simbol akhir (<i>end</i>)
	<i>Flow Line</i>	Simbol aliran / penghubung
	<i>Proses</i>	Perhitungan / pengolahan
	<i>Input / Output Data</i>	Mempresentasikan pembacaan data (<i>read</i>) / penulisan (<i>write</i>)
	<i>Decision</i>	<i>Simbol pernyataan pilihan, berisi suatu kondisi yang selalu menghasilkan 2 nilai keluaran yaitu benar atau salah</i>
	<i>Preparation</i>	Inisialisasi / pemberian nilai awal
	<i>Predefined Process (Subprogram)</i>	Proses menjalankan <i>subprogram/fungsi/prosedur</i>
	<i>On Page Connector</i>	Penghubung <i>Flowchart</i> pada suatu halaman

Simbol	Nama	Keterangan
	<i>Off Page Connector</i>	Penghubung <i>Flowchart</i> pada halaman berbeda

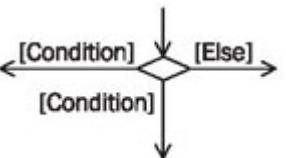
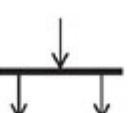
Sumber: Barakbah *et al* (2013)

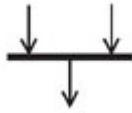
b. *Activity Diagram*

Activity Diagram merupakan diagram yang menggambarkan aktivitas yang terjadi dari awal hingga akhir pada sebuah sistem dan langkah-langkah dalam proses kerja sistem dari awal hingga akhir. *Activity Diagram* berfungsi sebagai memberikan gambaran mengenai proses bisnis dan urutan aktivitas yang dibuat berdasarkan *use case* pada *use case diagram* (Bentley, Whitten, & Bentley, 2007).

Simbol-simbol yang digunakan pada *activity diagram* akan dijelaskan pada Tabel 3.2.

Tabel 3.2 Komponen-komponen pada *Activity Diagram*

Simbol	Nama	Keterangan
	<i>Start Point (Initial Node)</i>	<i>Activity</i> dimulai
	<i>End Point (Activity Final Node)</i>	<i>Activity</i> berakhir
	<i>Activity</i>	Memperlihatkan bagaimana kelas-kelas antarmuka berinteraksi satu sama lain
	<i>Edge (Control Flow)</i>	Menghubungkan satu simbol dengan simbol lainnya.
	<i>Decision</i>	Percabangan seleksi-kondisi
	<i>Fork</i>	Membuat cabang dari satu <i>activity</i>

Simbol	Nama	Keterangan
	<i>Join</i>	Menggabungkan dua <i>activity</i>

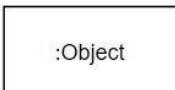
Sumber: Bentley, Whitten, & Bentley (2007)

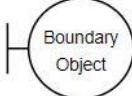
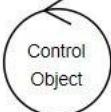
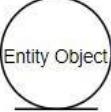
b. *Sequence Diagram*

Sequence Diagram merupakan diagram yang menggambarkan perilaku objek dengan mendeskripsikan waktu hidup dan pesan (*message*) yang dikirimkan dan diterima oleh objek yang satu dengan objek yang lainnya (Sukamto & Shalahuddin, 2013).

Penjelasan tentang komponen-komponen pada *Sequence Diagram* akan dijelaskan pada Tabel 3.3.

Tabel 3.3 Komponen-komponen pada Sequence Diagram

Simbol	Nama	Keterangan
	<i>Actor</i>	Subjek yang berinteraksi dengan sistem
	<i>Object</i>	Representasi dari class/object. Menunjukkan cara objek berperilaku pada sebuah sistem
	<i>Lifeline</i>	Umur hidup sebuah object
	<i>Activation</i>	Menunjukkan bahwa object sedang berinteraksi dengan object lain
	<i>Boundary Object</i>	Representasi dari batasan

Simbol	Nama	Keterangan
		sistem (<i>boundary</i>). Biasanya berupa <i>user interface</i>
	<i>Control Object</i>	Mengatur aliran informasi dari sebuah skenario
	<i>Entity Object</i>	Penyimpanan data atau informasi
	<i>Message Entry</i>	Menunjukkan <i>object</i> sedang berinteraksi dengan <i>object</i> lain
	<i>Message to Self/Return</i>	Mengembalikan <i>object</i> setelah berinteraksi dengan <i>object</i> lain
	<i>Self-Message</i>	Pesan yang mewakili jenis pesan yang hidup di <i>lifetime</i> yang sama

Sumber: Sukamto & Shalahuddin (2013)

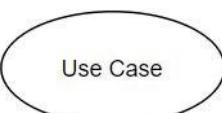
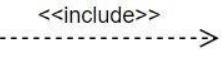
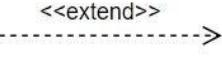
Objek pada *Sequence Diagram* diurutkan dari kiri ke kanan. *Sequence Diagram* terdiri dari 2 dimensi, yaitu vertikal dan horizontal. Dimensi vertikal merepresentasikan waktu, sedangkan dimensi horizontal merepresentasikan objek-objek yang ada pada sebuah *Sequence Diagram*.

c. Use Case Diagram

Use Case Diagram merupakan diagram yang menggambarkan kebutuhan-kebutuhan fungsional dari sistem yang akan dibuat. *Use Case Diagram* memaparkan cara aktor berinteraksi dengan sistem. *Use Case Diagram* berfungsi untuk menjelaskan bagaimana *end user* dapat berkomunikasi dengan sistem yang ada, kemudian memastikan *client* mendapatkan pemahaman yang tepat mengenai kebutuhan sistem. (Rosa & Shalahuddin, 2015)

Pada Tabel 3.4 akan dijelaskan komponen-komponen pada *Use Case Diagram*.

Tabel 3.4 Komponen-komponen pada Use Case Diagram

Simbol	Nama	Keterangan
 Actor	Actor	Subjek yang berinteraksi dengan sistem
	Use Case	Gambaran fungsional sistem yang akan dibuat
	Include	Proses yang harus terpenuhi agar proses selanjutnya dapat dijalankan
	Extend	Proses perluasan yang dapat dipanggil jika kondisi atau syarat terpenuhi

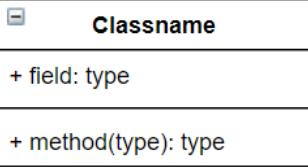
Sumber: Rosa & Shalahuddin (2015)

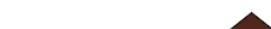
d. *Class Diagram*

Class Diagram merupakan diagram yang menggambarkan struktur dari sebuah sistem dengan cara mendefinisikan kelas-kelas yang dibangun dari sebuah sistem (Sukamto & Shalahuddin, 2013). Kelas-kelas tersebut merepresentasikan kumpulan fungsi yang sesuai dengan kebutuhan sistem.

Pada Tabel 3.5 akan dijelaskan komponen-komponen dari *Class Diagram*.

Tabel 3.5 Komponen-komponen pada Class Diagram

Simbol	Nama	Keterangan
	Class	Representasi dari objek yang mendefinisikan <i>attribute</i> dan <i>method</i>

Simbol	Nama	Keterangan
	<i>Association</i>	Hubungan antar kelas yang mereferensikan kelas yang lain, menggambarkan interaksi yang mungkin terjadi antara 1 kelas dengan kelas yang lain selama kelas tersebut tidak saling memiliki atau bukan bagian dari kelas tersebut
	<i>Directed Association</i>	Hubungan antar kelas dimana kelas yang satu digunakan oleh kelas yang lain
	<i>Generalization</i>	Hubungan antar kelas dimana sebuah kelas adalah kelas <i>child</i> yang lebih spesifik terhadap kelas <i>parentnya</i> . Kelas <i>child</i> memiliki kelas <i>parent</i> tetapi kelas <i>parent</i> tidak memiliki apa yang hanya dimiliki oleh kelas <i>child</i> . Jika hubungan di balik maka disebut <i>Inheritance</i> .
	<i>Aggregation</i>	Hubungan antar kelas dimana suatu kelas merupakan bagian dari kelas yang lain namun bersifat tidak wajib. Relasi ini juga menyatakan bahwa suatu kelas yang menjadi bagian dari kelas yang lain tidak akan dihapus meskipun kelas yang memiliki dihapus
	<i>Composition</i>	Hubungan antar kelas yang saling bergantung, dimana suatu kelas merupakan bagian dari kelas yang lain dan bersifat wajib. Relasi ini juga mengindikasikan bahwa suatu kelas yang menjadi bagian kelas

Simbol	Nama	Keterangan
		yang lain akan terhapus ketika kelas yang memilikinya dihapus
	<i>Dependency</i>	Hubungan antar kelas yang mengindikasikan ketergantungan sebuah kelas terhadap kelas yang lain

Sumber: Sukamto & Shalahuddin (2013)

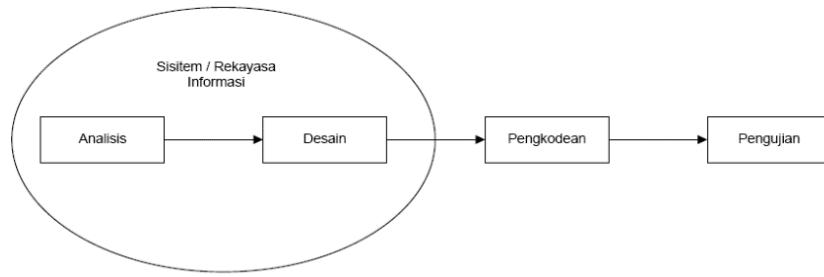
3.2 Teknologi Pengembangan Sistem

3.2.1 *System Development Life Cycle (SDLC)*

System Development Life Cycle (SDLC) merupakan sebuah model konseptual yang digunakan dalam pengelolaan proyek yang menggambarkan tahapan-tahapan yang dilibatkan dalam proyek pengembangan perangkat lunak. Proses tersebut terbagi menjadi analisa kebutuhan, desain, pengkodean, pengujian, instalasi, serta pemeliharaan. Semua aktivitas tersebut dijalankan dengan cara yang berbeda-beda sesuai dengan kebutuhan klien (Kumar, Zadgaonkar, & Shukla, 2013).

3.2.1.1 Metode Pengembangan *Waterfall*

Metode Pengembangan *Waterfall* merupakan metode pengembangan perangkat lunak yang berurutan, dipandang sebagai terus mengalir kebawah (seperti air terjun) melewati fase-fase perencanaan, pemodelan, implementasi, dan pengujian (Rosa & Shalahuddin, 2015). Metode *Waterfall* sangat cocok digunakan untuk pengembangan perangkat lunak yang jarang berubah-ubah (Nugroho, 2010). Alur dari Metode Pengembangan *Waterfall* dipaparkan pada Gambar 3.2.



Gambar 3.2 Alur Metode Pengembangan Waterfall

Sumber: Rosa & Salahuddin (2015)

Metode *waterfall* terdiri dari beberapa tahapan dalam sistematika pelaksanaan modelnya. Tahapan *waterfall* yang dimaksud, yaitu (Rosa & Shalahuddin, 2015):

1. **Analisis kebutuhan perangkat lunak**
Sebelum sistem dibuat, diperlukan suatu analisis sebagai dasar untuk mengetahui kebutuhan sistem kedepannya. Analisis kebutuhan sistem terdiri dari analisis kebutuhan fungsional yang bertujuan untuk mengetahui kebutuhan fungsi sistem dan analisis kebutuhan non fungsional untuk mengetahui perangkat keras dan perangkat lunak yang dibutuhkan serta kriteria pengguna sistem.
2. **Desain**
Desain berfungsi sebagai dasar perancangan yang mengubah data-data yang didapat dari analisis menjadi sebuah rancangan yang terdiri dari desain struktur data, struktur navigasi, dan rancangan antar muka.
3. **Pembuatan kode program**
Tahapan ini merupakan lanjutan dari tahapan desain, yaitu mentranslasikan desain menjadi sebuah program. Tahap ini menghasilkan suatu program yang sesuai dengan desain.
4. **Pengujian**
Program yang telah dibuat wajib diuji terlebih dahulu untuk memastikan bahwa program layak digunakan dari segi logika maupun fungsional. Pengujian ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan diinginkan.
5. **Pendukung (*support*) atau pemeliharaan (*maintenance*)**
Program yang telah diuji dapat mengalami perubahan ketika sudah dikirimkan ke pengguna. Perubahan dapat terjadi karena terjadi kesalahan yang tidak terdeteksi saat pengujian program harus beradaptasi dengan lingkungan baru (*hardware* baru). Tahap pendukung atau pemeliharaan bertujuan untuk menjaga stabilitas program yang telah dibuat tanpa harus membuat program yang baru.

3.2.2 Aplikasi Berbasis Web

Aplikasi berbasis web merupakan sebuah program yang *source code*-nya disimpan di dalam *server* dan dapat diakses melalui jaringan internet menggunakan *web browser* (Sagala & Entik, 2014). Aplikasi web dapat diakses melalui *web browser* dari perangkat PC ataupun perangkat *mobile*. Aplikasi web dibangun menggunakan HTML, CSS, dan bahasa pemrograman yang mendukung pengembangan aplikasi web seperti PHP, Ruby, Python, dan lain-lain.

3.2.3 Bahasa Pemrograman

3.2.3.1 HTML

HTML adalah bahasa pemformatan teks untuk dokumen-dokumen pada jaringan komputer yang sering disebut sebagai *world wide web* (Nugroho, 2006). HTML dapat dibaca oleh berbagai macam *platform*. HTML juga merupakan bahasa pemrograman yang fleksibel dan dapat digabungkan dengan bahasa pemrograman lain seperti PHP, Javascript, Ruby, maupun Python.

Beberapa *tag* dalam dokumen-dokumen HTML menentukan bagaimana teks diformat. *Tag-tag* yang lain memberitahukan komputer bagaimana menanggapi aksi-aksi yang datang dari pengguna. Kemudian *tag* lain yang penting adalah *link* yang mengandung *Uniform Resource Locator* (URL), yang merujuk pada dokumen lain di *server* yang sama atau komputer lain yang ada di global jaringan internet.

3.2.3.2 PHP

PHP merupakan bahasa *server side scripting* yang berarti sintaksnya atau perintah-perintahnya di eksekusi di *server* kemudian di kirim ke *browser* dalam format HTML. Sintaks PHP hanya dapat diterjemahkan pada *server* yang sudah ter-*install* PHP (Rudianto, 2011). PHP pada awalnya adalah singkatan dari *Personal Home Page*, namun sekarang lebih dikenal dengan *Hypertext Preprocessor*. PHP sendiri dapat digunakan untuk mengembangkan web statis, dinamis, maupun aplikasi berbasis web.

3.2.3.3 CSS

CSS (*Cascading Style Sheet*) merupakan *stylesheet language* yang digunakan untuk menggambarkan penyajian dari dokumen yang dibuat oleh *markup language* (Binarso *et al.*, 2012). CSS merupakan *script* yang berguna untuk membuat halaman *web* menjadi bentuk *web* yang lebih indah dan menarik.

3.2.3.4 Javascript

Javascript merupakan bahasa pemrograman yang digunakan pada aplikasi web yang berjalan pada sisi klien (*client side*). Javascript memberikan fitur tambahan diluar kemampuan HTML dan CSS yang dapat menambah interaksi antara halaman web dengan pengguna (Binarso *et al.*, 2012).

3.2.3.5 Database Management System

Database atau basis data adalah kumpulan data yang disimpan secara tersistematis dan dapat ditambahkan, dimanipulasi, maupun dihapus melalui sebuah sistem atau aplikasi. *Database* sangat penting digunakan dalam sebuah sistem informasi karena mampu menyimpan banyak data dan mengolahnya menjadi informasi yang bernilai. Penggunaan *database* dapat mengurangi duplikasi data dan menyederhanakan kinerja dari pengelolaan informasi.

Database Management System (Disingkat DBMS) merupakan sistem pengelolaan basis data yang memungkinkan *user* untuk memelihara, mengontrol, dan melakukan manipulasi data secara efisien.

3.2.3.6 MySQL

MySQL (*My Structured Query Language*) adalah *Database Management System* yang dikembangkan oleh Oracle Corporation. MySQL merupakan sebuah program pembuat *database* yang bersifat *open source*, artinya siapa saja dapat menggunakananya secara bebas (Nugroho, 2004).

3.2.4 Software Pendukung

3.2.4.1 Visual Studio Code

Visual Studio Code merupakan aplikasi *source-code editor* yang dikembangkan oleh Microsoft untuk Windows, Linux, dan macOS. Visual Studio Code mendukung berbagai hampir semua bahasa pemrograman dengan berbagai fitur yang dimilikinya. Visual Studio Code memiliki fitur *extensions* yang berfungsi untuk menambah fungsionalitas dari Visual Studio Code agar memudahkan pengembangan perangkat lunak dan penulisan kode.

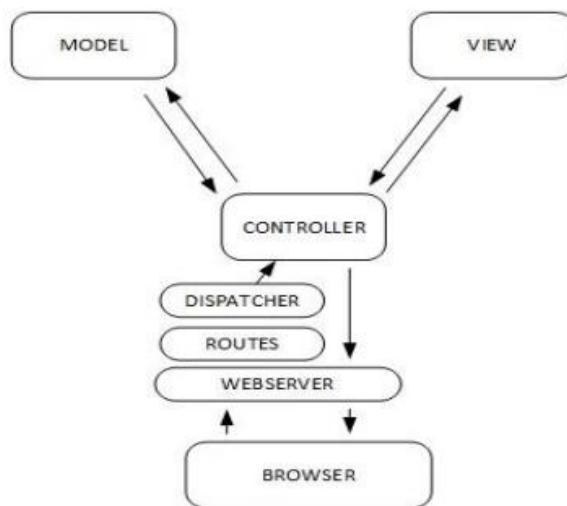
3.2.5 Framework

Framework merupakan suatu kerangka kerja yang berisikan kumpulan *script* yang dapat membantu pengembangan aplikasi dalam menangani berbagai masalah pemrograman seperti koneksi ke database, pemanggilan *variable* , dan *file* (Rosmala *et al.*, 2011).

Menulis kode program dapat memakan waktu berjam-jam hingga beratus-ratus baris kode. *Framework* memfasilitasi pengembang aplikasi *web* untuk dapat meningkatkan produktivitas, karena *framework* memiliki fungsi bawaan yang dapat dipakai oleh pengembang, sehingga pengembang tidak perlu menulis kode program yang sama berulang-ulang. Menggunakan *framework* juga berarti dapat meningkatkan keamanan aplikasi *web* karena *framework* sudah tersedia fitur-fitur keamanan, sehingga pengembang tidak perlu membuat dari awal sistem keamanan aplikasi *web* yang dikembangkan (Prokofyeva & Boltunova, 2016).

3.2.5.1 Model-View-Controller

Model-View-Controller (atau disingkat MVC) merupakan paradigma pada pengembangan perangkat lunak yang memisahkan *source code* aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi, seperti manipulasi data (*model*), antarmuka pengguna (*view*), dan proses bisnis yang mengontrol aplikasi (*controller*) (Handika & Purbasari, 2018). MVC bertujuan untuk memisahkan proses bisnis dari pertimbangan antarmuka user agar para pengembang bisa lebih mudah mengembangkan salah satu bagian dari aplikasi sehingga tidak memengaruhi bagian yang lain (Badiyanto, 2013). Alur kerja sebuah MVC dipaparkan pada Gambar 3.3.



Gambar 3.3 Konsep MVC

Sumber: Badiyanto (2013)

Dengan menggunakan konsep MVC pada pengembangan perangkat lunak, maka pemeliharaan program akan lebih mudah dan teratur, mengingat semua komponen program telah dipetakan dalam struktur yang jelas, sehingga apabila terdapat perubahan pada desain perangkat lunak, maka proses bisnis pada perangkat lunak tidak ikut berubah. Pengubahan pada proses bisnis juga akan lebih mudah karena dapat dilakukan di bagian program yang terpisah.

3.2.5.2 Laravel

Laravel merupakan *framework* aplikasi berbasis web dengan bahasa PHP. Laravel dirilis di bawah lisensi MIT dengan kode sumber yang sudah disediakan oleh Github. Sama seperti *framework* yang lain, Laravel dibangun dengan konsep MVC (*Model-View-Controller*), kemudian Laravel dilengkapi juga *command line tool* yang bernama “Artisan” yang bisa digunakan untuk *packaging bundle* dan instalasi *bundle* melalui *command prompt* (Aminudin, 2015).

Laravel memiliki fitur-fitur yang dapat membantu pengembang dalam membuat aplikasi. Fitur yang digunakan pada pengembangan sistem ini adalah *Blade*, *Eloquent ORM*, *Middleware*, dan Artisan. Berikut adalah penjelasan mengenai fitur-fitur tersebut:

1. *Blade*

Blade merupakan *template engine* yang disediakan oleh Laravel (Laravel Docs 6.2). Pada dasarnya *Blade* adalah View dengan bahasa HTML, namun pengembang dapat memasukkan perintah percabangan, perulangan, serta *array*. Maka dari itu, *Blade* akan sangat membantu pengembang dalam mengatur tampilan *website* dan menampilkan data. Untuk membuat file *Blade* yaitu dengan menambah ekstensi .blade.php pada nama file. Berikut adalah dua contoh kode program untuk membandingkan file HTML biasa dengan file *Blade*.

Tabel 3.6 Kode Program Employee.html

Employee.html	
1	<div id="Employee"> 2 <h1>Employee List</h1> 3 <?php if(!empty(\$employee)): ?> 4 5 <?php foreach(\$employee as \$emp): ?> 6 7 <?= \$emp ?> 8 <?php endforeach ?> 9 10 <?php else: ?> 11 <p>Empty.</p> 12 <?php endif ?> 13 </div>

Tabel 3.7 Kode Program Employee.blade.php

Employee.blade.php	
1	<div id="Employee">
2	<h1>Employee List</h1>
3	@if(!empty(\$employee))
4	
5	@foreach(\$employee as \$emp)
6	{{\$emp}}
7	@endforeach
8	
9	@else
10	<p>Empty.</p>
11	@endif
12	</div>
13	

Dari kedua contoh program diatas dapat disimpulkan jika dengan menggunakan file Blade maka penulisan program akan menjadi lebih singkat dan rapi.

2. Eloquent ORM

Eloquent ORM adalah implementasi dari *ActiveRecord* yang digunakan untuk mengatur relasi antar tabel di *database*. Pada *framework* Laravel, Setiap tabel pada *database* memiliki kelas Model yang digunakan untuk berinteraksi dengan tabel yang sesuai (Laravel Docs 6.2). *Eloquent* ORM diimplementasikan pada setiap Model dalam bentuk kelas dan data yang tersimpan di dalam tabel direpresentasikan dalam bentuk objek. Relasi yang dapat diatur menggunakan Eloquent ORM adalah sebagai berikut :

- a. One-to-One. Yaitu relasi satu ke satu. Pada relasi ini digunakan method `hasOne` dan `belongsTo`.
- b. One-to-Many. Yaitu relasi satu ke banyak. Pada relasi ini digunakan method `hasMany` dan `belongsToMany`.
- c. Many-to-One. Yaitu relasi banyak ke satu. Pada relasi ini digunakan method `belongsTo` dan `hasMany`.
- d. Many-to-Many. Yaitu relasi banyak ke banyak. Pada relasi ini digunakan method `belongsToMany`.

3. Middleware

Middleware adalah fitur yang menyediakan mekanisme untuk melakukan penyaringan (*filtering*) terhadap HTTP request yang masuk ke aplikasi. Laravel memiliki beberapa Middleware yaitu `Authenticate`, `EncryptCookies`, `RedirectIfAuthenticated`, dan `VerifyCsrfToken`.

Salah satu Middleware yang digunakan pada pengembangan system ini adalah Middleware `Authenticate`. Middleware tersebut akan memeriksa apakah user sudah login atau belum. Jika user sudah login maka request akan dilanjutkan ke halaman yang dikehendaki oleh user. Tetapi jika user belum login maka

Middleware Authenticate akan mengarahkan user ke halaman login. Selain itu Middleware Authenticate juga dapat membatasi hak akses pada setiap user.

Jika Middleware yang sudah ada pada Laravel kurang sesuai dengan kebutuhan ataupun tidak sesuai dengan kebutuhan maka dapat dibuat sendiri Middleware yang sesuai dengan kebutuhan.

4. Artisan

Artisan merupakan *command-line interface* yang disediakan oleh Laravel (Laravel Docs 6.2). Artisan memberikan sejumlah perintah yang dilakukan secara otomatis seperti menjalankan *local environment* dan pembuatan kelas sehingga membantu pengembang dalam mengembangkan aplikasi karena apabila dikerjakan secara manual, akan menghabiskan banyak waktu serta memperbesar kesalahan dalam mengerjakannya. Berikut adalah beberapa fungsi artisan yang sering digunakan :

a. Serve

Command : php artisan serve

Command serve akan menjalankan *development server* bawaan PHP. Seacara *default*, perintah *serve* akan menjalankan *server* pada *port* 8000, namun jika *port* tersebut sudah digunakan dapat menambahkan perintah sebagai berikut :

php artisan serve --port=8080

b. Make:Controller

Command : php artisan make:controller namaController

Command ini akan membuatkan controller baru secara otomatis lengkap dengan *method* yang dibutuhkan.

c. Make:Model

Command : php artisan make:model namaModel

Command ini akan secara otomatis membuatkan kelas model *eloquent*.

d. Make:Migration

Command : php artisan make:migration nama_tabel

Command migration berfungsi untuk membuat *blueprint* dari database. *Blueprint* tersebut digunakan sebagai struktur dari database yang dibangun dan dapat digunakan oleh anggota tim proyek untuk membuat database yang sama tanpa perlu membangunnya dari awal. Untuk mengaplikasikan *migration* yang sudah dibuat, dapat menjalankan perintah berikut :

php artisan migrate

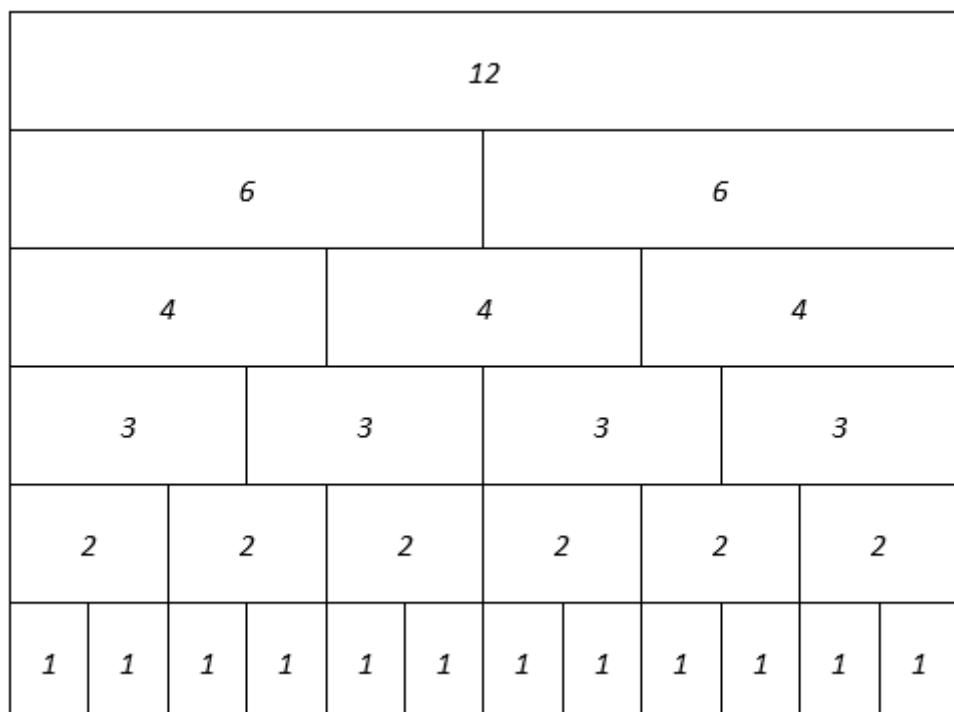
3.2.5.3 Bootstrap

Bootstrap merupakan *framework CSS* yang digunakan untuk mendesain halaman *web* menjadi lebih sederhana dan mudah. Bootstrap memiliki slogan “*Sleek, intuitive, and powerful front-end framework for faster and easier web development*”, yang artinya dengan menggunakan bootstrap, pengembang aplikasi *web* dapat mendesain sebuah *website* dengan lebih cepat, rapi, dan mudah. Bootstrap juga *responsive* terhadap banyak *platform*, artinya tampilan halaman *website* yang menggunakan Bootstrap ini akan tampak tetap rapi, baik versi *mobile* maupun *desktop* (Zakir, 2016).

Di dalam Bootstrap terdapat fitur-fitur utama yang membantu desain halaman *web* agar tampak rapi. Fitur utama tersebut adalah *Grid System*, *Responsive Breakpoints*, dan *Z-index*. Berikut adalah penjelasan dari ketiga fitur tersebut :

1. *Grid System*

Grid System adalah sistem pada Bootstrap untuk mengatur lebar dari setiap komponen html yang kita buat dan juga mengatur bagaimana tampilan akan menyesuaikan pada perangkat yang berbeda. *Grid System* terdiri dari total 12 *grid* Bootstrap. Gambar 3.4 menunjukkan ilustrasi dari *Grid System*.



Gambar 3.4 Grid System Bootstrap

Sumber : Bootstrap Docs 4.0

Terdapat 3 elemen penting yang membangun *Grid System*, yaitu :

a. *Container*

Container adalah elemen utama dalam *Grid System* yang wajib digunakan sebelum menggunakan elemen lainnya dalam Bootstrap. *Container* berfungsi untuk menyesuaikan lebar elemen. Terdapat 2 kelas pada *Container* yaitu *container* dan *container-fluid*. Kelas *container* akan menyesuaikan lebar isi elemen sesuai *minimum viewport* pada *media queries*. Sementara itu, kelas *container-fluid* akan mengisi elemen hingga memenuhi layar. Berikut adalah contoh penggunaan kedua kelas tersebut :

Tabel 3.8 Contoh Penggunaan container

```
1 <div class="container">
2     <!-- Konten Website -->
3 </div>
4
5 <div class="container-fluid">
6     <!-- Konten Website -->
7 </div>
```

b. *Row*

Row adalah baris yang terdapat di dalam *Container* dan berfungsi untuk memberi jarak dan menampung *column*. Berikut adalah contoh penggunaan *row* :

Tabel 3.9 Contoh Penggunaan row

```
1 <div class="container">
2     <div class="row"
3
4         </div>
5     </div>
```

c. *Column*

Column adalah tempat yang menyimpan elemen-elemen website. Berikut adalah contoh penggunaan *column* yang membagi *grid* menjadi 3 *column* :

Tabel 3.10 Contoh Penggunaan column

```
1 <div class="container">
2     <div class="row"
3         <div class="col-md-4"></div>
4         <div class="col-md-4"></div>
5         <div class="col-md-4"></div>
6     </div>
7 </div>
```

2. *Responsive Breakpoints*

Responsive Breakpoints adalah titik yang menjadi acuan kapan tampilan suatu elemen pada website akan berubah menyesuaikan ukuran dari layar perangkat (Bootstrap Docs 4.0). Berikut adalah titik-titik *responsive breakpoints* yang dijelaskan dalam tabel 3.5 :

Tabel 3.11 Titik-Titik *Responsive Breakpoints*

	Extra Small $< 576px$	Small $\geq 576px$	Medium $\geq 768px$	Large $\geq 992px$	Extra Large $\geq 1200px$
Nama Kelas	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
Mode	Portrait Phone	Landscape Phone	Tablet	Laptop	Desktop
Jumlah Grid	12				

Sumber : Bootstrap Docs 4.0

Berikut adalah contoh penggunaan *Responsive Breakpoints* menggunakan ukuran *small*, yaitu menyesuaikan ukuran terhadap layar *landscape phone*, namun jika dibuka dengan layar yang lebih lebar seperti tablet, laptop atau desktop, elemen tabel akan otomatis turun sejajar secara vertikal :

Tabel 3.12 Contoh Penggunaan *Responsive Breakpoints*

1	<div class="container">
2	<div class="row">
3	<div class="col-md">col</div>
4	<div class="col-md">col</div>
5	<div class="col-md">col</div>
6	</div>
7	</div>

3. *Z-index*

Z-index adalah fitur pada CSS yang mendefinisikan *stack order* dari elemen yang bertumpukan dengan menggunakan sumbu ketiga yaitu "z". Pada Bootstrap, didefinisikan standar *z-index* agar navigasi, *popup*, *modal*, dan elemen-elemen lain dapat bekerja dengan semestinya (Bootstrap Docs 4.0). Nilai standar *z-index* pada Bootstrap diberikan pada tabel 3.6 :

Tabel 3.13 Nilai Standar Z-index pada kelas Bootstrap

Kelas	Nilai z-index
zindex-dropdown:	1000
zindex-sticky:	1020
zindex-fixed:	1030
zindex-modal-backdrop:	1040
zindex-modal:	1050
zindex-popover:	1060
zindex-tooltip:	1070

Sumber : Bootstrap Docs 4.0

Berikut adalah cara mendefinisikan z-index pada Bootstrap beserta penjelasannya :

.z-index-{n2|n1|0|1|2|master}

- a. .z-index-n2 merepresentasikan z-index: -2
- b. .z-index-n1 merepresentasikan z-index: -1
- c. .z-index-0 merepresentasikan z-index: 0
- d. .z-index-1 merepresentasikan z-index: 1
- e. .z-index-2 merepresentasikan z-index: 2
- f. .z-index-master merepresentasikan z-index: 1090 (nilai z-index terbesar)

3.6 Pengujian Perangkat Lunak

Pengujian perangkat lunak merupakan kegiatan untuk memvalidasi atau mengetahui informasi mengenai kualitas maupun investigasi kekurangan dari perangkat lunak yang sedang diuji (Kaner, 2006). Terdapat beberapa aturan yang berfungsi sebagai sasaran pengujian pada perangkat lunak (Sukamto, 2009) :

1. Pengujian adalah proses eksekusi suatu program dengan maksud menemukan kesalahan.

2. *Test case* yang baik adalah *test case* yang memiliki probabilitas tinggi untuk menemukan kesalahan yang belum pernah ditemukan sebelumnya.
3. Pengujian yang sukses adalah pengujian yang mengungkap semua kesalahan yang belum pernah ditemukan sebelumnya.

Karakteristik umum dari pengujian perangkat lunak adalah sebagai berikut (Sukamto, 2009):

1. Pengujian dimulai pada level modul dan bekerja keluar ke arah integrasi pada sistem berbasiskan komputer.
2. Teknik pengujian yang berbeda sesuai dengan poin-poin yang berbeda pada waktunya.
3. Pengujian diadakan oleh *software developer* dan untuk proyek yang besar oleh *group testing* yang independen.
4. *Testing* dan *Debugging* adalah aktivitas yang berbeda tetapi *debugging* harus diakomodasikan pada setiap strategi *testing*

Terdapat 3 jenis metode pengujian perangkat lunak, yaitu (Sukamto, 2009):

1. *White Box* - pengujian operasi
2. *Black Box* - untuk menguji sistem
3. *Use case* - untuk membuat *input* dalam perancangan *black box* dan pengujian *state based*

3.6.1 Black Box Testing

Black Box Testing adalah teknik pengujian perangkat lunak yang berfokus pada spesifikasi fungsional tanpa menguji desain dan kode program (Shalahuddin & Rosa, 2011). Tujuan dari *Black box testing* adalah untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. *Black box testing* dilakukan dengan membuat skenario uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian *black box testing* harus dibuat dengan kasus benar dan kasus salah.

3.6.2 Compatibility Testing

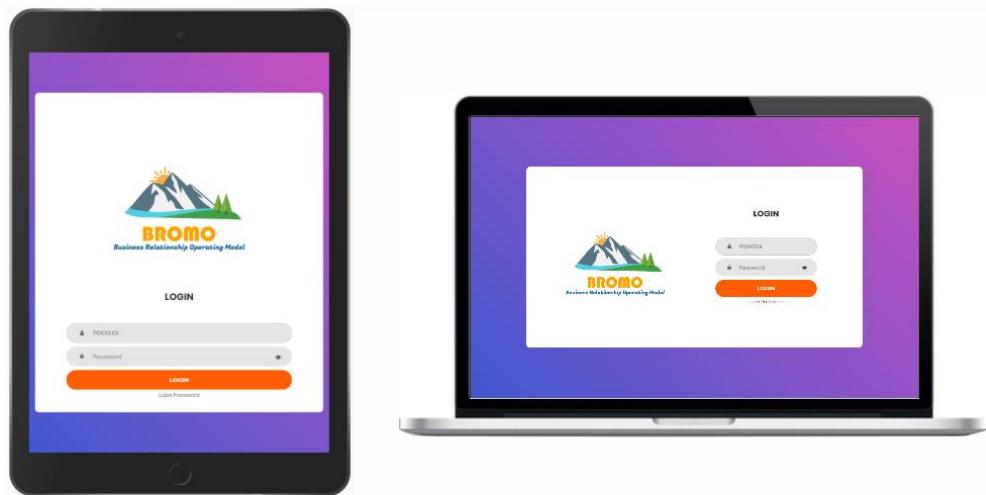
Compatibility Testing adalah teknik pengujian perangkat lunak yang berfokus untuk mengukur sejauh mana suatu perangkat lunak mendukung dengan perangkat lunak lain dalam lingkungan tempat perangkat lunak tersebut dijalankan (Anne Mette Hass, 2014). *Compatibility testing* dimaksudkan untuk memeriksa apakah suatu perangkat lunak dapat dijalankan pada perangkat keras, perangkat lunak, *database*, sistem operasi, *browser*, ataupun lingkungan

jaringan yang berbeda. *Compatibility testing* berguna untuk menentukan set lingkungan yang diharapkan dapat menjalankan perangkat lunak yang dikembangkan. Semakin beragam lingkungan yang dapat berjalan pada suatu perangkat lunak, semakin baik aspek kompatibilitasnya.

Compatibility test pada penelitian ini berfokus pada *browser compatibility*, yaitu menguji *browser* apa saja yang dapat digunakan untuk menjalankan perangkat lunak yang dikembangkan. Alat yang digunakan untuk pengujian adalah SortSite, yaitu perangkat lunak yang dapat digunakan sebagai alat pengujian yang menjalankan ratusan pemeriksaan kualitas pada setiap halaman dalam sebuah situs. Setiap halaman dilakukan pengujian kualitas menggunakan lebih dari 450 tempat pemeriksaan (PowerMapper, 2020).

3.6.3 Responsive Web Design (RWD) Testing

Responsive Web Design Testing adalah teknik pengujian perangkat lunak yang bertujuan untuk menguji responsifitas dari tampilan sebuah *website* pada berbagai ukuran layar dan perangkat seperti desktop dan tablet. Desain *website* yang *responsive* adalah *website* yang dapat menyesuaikan tampilannya pada semua macam ukuran layar tanpa memerhatikan perangkat yang digunakan. Hal ini berkaitan dengan *media queries* untuk mengubah *style* berdasarkan parameter seperti tipe layar, lebar, tinggi, dan lain-lain (Eugene, 2020). Berikut adalah ilustrasi *Responsive Web Design* Sistem Informasi *Lending* BROMO pada Gambar 3.4.



Gambar 3.5 Ilustrasi Responsive Web Design dari Sistem Informasi Lending BROMO

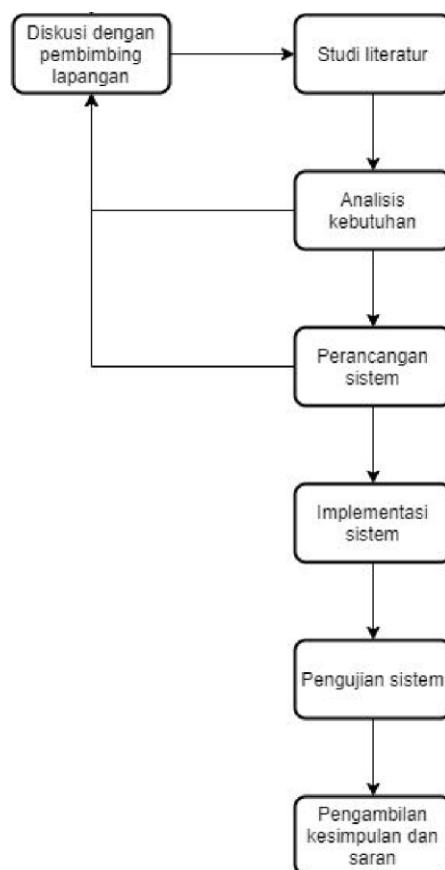
Terdapat 2 metode yang dapat digunakan untuk melakukan pengujian RWD (Eugene, 2020) :

1. Menggunakan *tools* pada *browser* untuk memeriksa *response* dari *website* ketika mengubah *viewport* dengan resolusi dan *zoom* pada perangkat yang berbeda.
2. Menggunakan *tools* khusus yang terpisah untuk menguji responsivitas konten *website* pada sebuah perangkat.

Alat yang dipilih untuk melakukan pengujian RWD adalah Google Chrome DevTools yang menyediakan *tools* untuk memberi gambaran tampilan *website* yang dijalankan pada ukuran layar dan perangkat yang diinginkan.

BAB 4 METODOLOGI

Bab ini menjelaskan tahap-tahap yang dilakukan dalam pembuatan Sistem Informasi *Lending* BROMO. Metode yang dilakukan diantaranya adalah diskusi dengan pembimbing lapangan, studi literatur, analisis kebutuhan, perancangan sistem, implementasi, pengujian, dan penarikan kesimpulan dan saran. Berikut dipaparkan diagram alir metode penelitian pada Gambar 4.1.



Gambar 4.1 Diagram Alir Pembuatan Sistem Informasi *Lending* BROMO

4.1 Diskusi dengan Pembimbing Lapangan

Diskusi dengan pembimbing praktik kerja lapangan pertama kali dilakukan pada saat hari kedua pelaksanaan praktik kerja lapangan dengan cara melihat program yang telah dibangun sebelumnya dan diskusi untuk pengembangan selanjutnya. Pembimbing lapangan terdiri dari tiga orang yang merupakan anggota dari Unit Research and Development Region (RDR) yang terdiri dari satu orang *programmer*, satu orang *system analyst*, dan satu orang kepala unit sementara. Kegiatan diskusi dan laporan progres dilakukan setiap hari agar mengetahui kendala apa saja yang sedang dihadapi. Jika ada improvisasi ataupun saran tentang sistem, langsung dapat disampaikan dan didiskusikan lebih lanjut untuk langkah implementasinya.

4.2 Studi Literatur

Studi literatur merupakan tahap untuk mempelajari, mencari dan pengumpulan referensi dari buku, *e-book*, jurnal ataupun dokumentasi untuk memperoleh penjelasan teori yang berhubungan dengan penelitian. Sehingga teori tersebut diharapkan dapat mempermudah dan membantu dalam melaksanakan penelitian.

4.3 Analisis Kebutuhan

Tahapan ini dilakukan pada tahap awal dari pengembangan Sistem Informasi *Lending* BROMO. Analisis kebutuhan bertujuan untuk mengidentifikasi pengguna sistem serta kebutuhan dari pengguna maupun sistem dalam pengembangan Sistem Informasi *Lending* BROMO. Kebutuhan Sistem Informasi *Lending* BROMO telah dicantumkan oleh pembimbing lapangan dalam bentuk rancangan kebutuhan dan spesifikasi teknis. Analisis kebutuhan pada sistem ini digambarkan dalam *use case diagram* dan *activity diagram* yang dibuat menggunakan Draw.io.

4.4 Perancangan Sistem

Setelah semua kebutuhan untuk pengembangan sistem didapatkan melalui tahap analisis kebutuhan, tahap selanjutnya adalah melakukan perancangan pada Sistem Informasi *Lending* BROMO. Perancangan dengan berorientasi objek yang terdiri dari beberapa tahapan yaitu *sequence diagram*, *class diagram*, *user interface*, dan perancangan *physical data model*. Tools yang digunakan sebagai penunjang dalam perancangan sistem adalah Draw.io (untuk merancang *sequence diagram* dan *class diagram*), Figma (untuk merancang *user interface*), dan PowerDesigner

4.5 Implementasi

Pada tahap ini akan dilakukan proses pembangunan sistem dengan menerapkan rancangan sistem. Implementasi pada Sistem Informasi *Lending* BROMO meliputi :

1. Implementasi *user interface*

Pada tahap implementasi *user interface* dilakukan implementasi untuk membuat tampilan berdasarkan perancangan *user interface* menggunakan HTML, CSS dan Javascript dengan *framework* Bootstrap.

2. Implementasi database

Pada tahap implementasi database dilakukan implementasi rancangan database menggunakan MySQL sebagai *tools database management system* dan menggunakan bahasa SQL.

3. Implementasi kode program

Pada tahap implementasi kode program dilakukan implementasi dari perancangan sistem menggunakan bahasa PHP dengan *framework* Laravel.

4.6 Pengujian Sistem

Pengujian dilakukan untuk menguji apakah semua kebutuhan sudah sesuai dengan permintaan *user* dan menemukan kesalahan pada perangkat lunak. Bentuk pengujian yang akan dilakukan adalah pengujian validasi dengan menggunakan metode *Black Box Testing*, sementara itu untuk *tools* penunjang pengujian *compatibility* menggunakan *SortSite* dan untuk pengujian *responsive web design* (RWD) menggunakan Google Chrome DevTools. Jika selama pengujian ditemukan kesalahan ataupun ada kebutuhan yang tidak berjalan dengan semestinya, maka akan dilakukan perbaikan dari kesalahan tersebut. Jika sudah dilakukan perbaikan, maka akan diuji lagi apakah kesalahan tersebut sudah diselesaikan seperti semestinya.

4.7 Kesimpulan dan Saran

Kesimpulan diambil setelah semua tahapan perancangan dan implementasi sistem aplikasi telah selesai dilakukan. Sistem aplikasi yang dibangun dianalisis untuk ditarik kesimpulannya. Pada akhir penulisan dituliskan saran yang bertujuan untuk memberikan masukan serta memperbaiki kekurangan dari pengembangan sistem maupun cara penulisan yang telah dilakukan.

BAB 5 ANALISIS KEBUTUHAN

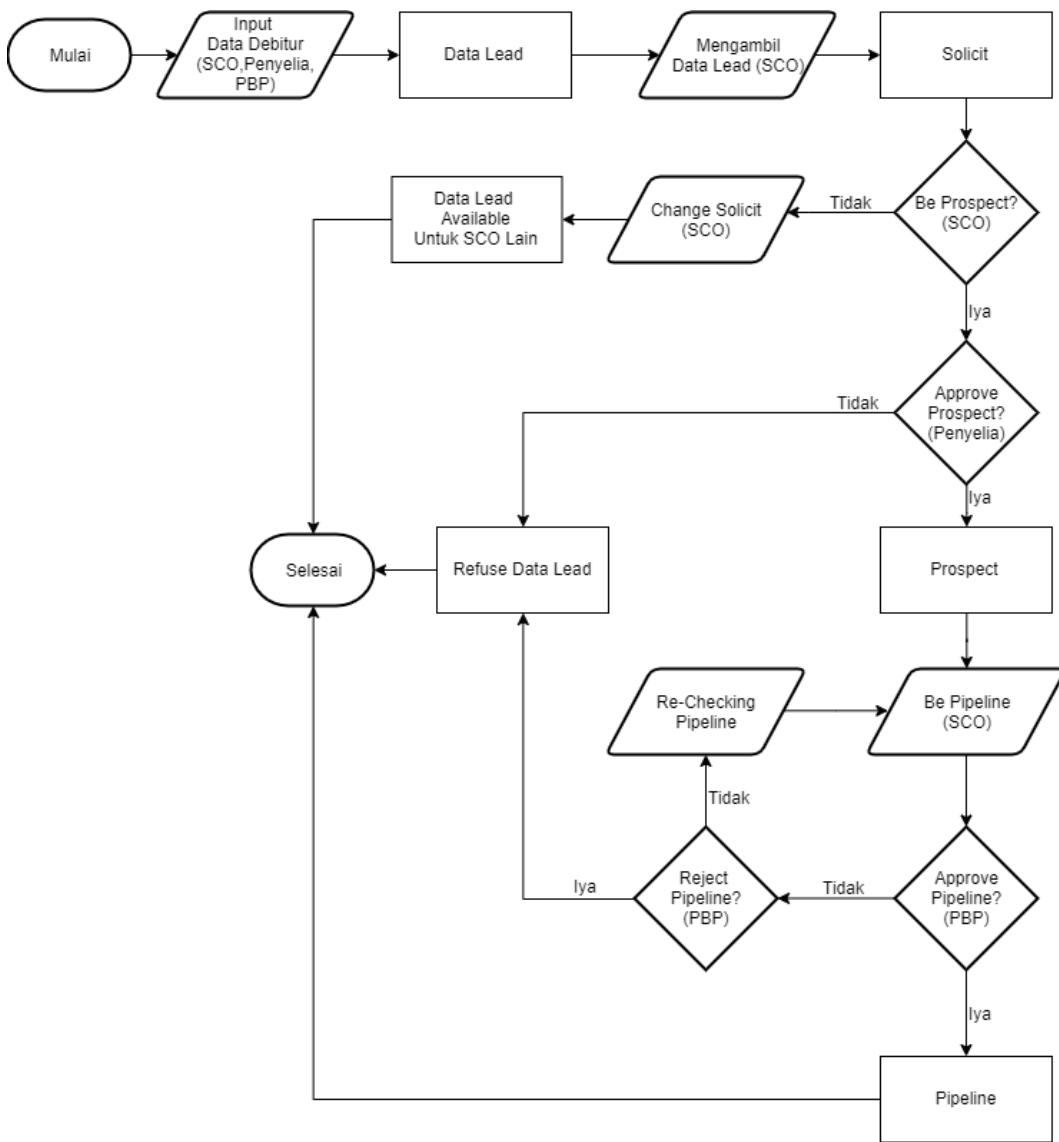
Bab ini bertujuan untuk menguraikan tahap-tahap analisis kebutuhan sistem. Analisis kebutuhan sistem perlu dilakukan dalam tahap pengembangan sistem untuk membantu membuat spesifikasi kebutuhan yang harus disediakan oleh sistem. Spesifikasi kebutuhan sistem akan dijadikan acuan untuk proses pengembangan perangkat lunak ke tahap selanjutnya. Analisis kebutuhan pada bab ini terdiri dari deskripsi umum sistem, identifikasi aktor, identifikasi kebutuhan baik fungsional maupun non fungsional, *use case diagram*, skenario *use case*, dan *activity diagram*.

Spesifikasi kebutuhan ditulis dengan format F_SILSCO_**, F_SILPIA_**, atau F_SIL_PBP_** untuk mendefinisikan kebutuhan fungsional. Format NF_SIL_** mendefinisikan kebutuhan non fungsional. F adalah singkatan untuk Fungsional, NF adalah singkatan untuk Non Fungsional, SIL adalah singkatan untuk Sistem Informasi *Lending* BROMO, SCO adalah singkatan untuk *Sales Company*, PIA adalah singkatan untuk Penyelia, PBP adalah singkatan untuk Pemimpin Bidang Pemasaran, dan simbol dua bintang menjelaskan angka sesuai dengan urutan kebutuhan yang dituliskan.

5.1 Deskripsi Umum Sistem

Sistem Informasi *Lending* BROMO adalah sistem informasi yang berfungsi untuk melakukan manajemen dan monitoring peminjaman produktif dan konsumtif debitur. Sistem Informasi *Lending* BROMO yang dikembangkan pada Praktik Kerja Lapangan akan digunakan oleh BNI Kantor Wilayah Malang dan Cabang Malang. *User* dalam Sistem Informasi *Lending* BROMO memiliki beberapa kewenangan diantaranya SCO (*Sales Company*), Penyelia, dan PBP (Pemimpin Bidang Pemasaran). SCO adalah petugas lapangan yang bertugas untuk melakukan pendataan dan *follow-up* debitur. Penyelia dan PBP adalah petugas otoritas yang bertugas untuk melakukan monitoring dan persetujuan dari proses *lending*.

Fase-fase peminjaman dijelaskan lebih detail dalam *flowchart* pada Gambar 5.1.



Gambar 5.1 Flowchart fase-fase peminjaman pada Sistem Informasi *Lending BROMO*

5.2 Identifikasi Aktor

Aktor dari sistem ini adalah *Sales Company* (selanjutnya disebut SCO), Penyelia, dan Pemimpin Bidang Pemasaran (selanjutnya disebut PBP). Karakteristik setiap aktor akan dipaparkan pada Tabel 5.1.

Tabel 5.1 Identifikasi Aktor

No.	Identifikasi Aktor	Karakteristik
1.	Sales Company (SCO)	Orang yang dapat mengakses sistem dengan masuk terlebih dahulu agar dapat memiliki hak akses fitur yang ada pada sistem. Memiliki kewenangan untuk melakukan pendataan debitur.
2.	Penyelia	Orang yang dapat mengakses sistem dengan masuk terlebih dahulu agar dapat memiliki hak akses fitur yang ada pada sistem. Memiliki kewenangan untuk melakukan monitoring dan persetujuan dari proses <i>lending</i> .
3.	Pemimpin Bidang Pemasaran (PBP)	Orang yang dapat mengakses sistem dengan masuk terlebih dahulu agar dapat memiliki hak akses fitur yang ada pada sistem. Memiliki kewenangan untuk melakukan monitoring dan persetujuan dari proses <i>lending</i> .

5.3 Analisis Kebutuhan Fungsional

Analisis Kebutuhan fungsional bertujuan untuk menganalisis fungsi-fungsi apa saja yang harus dilakukan oleh sistem yang dikembangkan. Kebutuhan fungsional pada sistem ini dijelaskan pada Tabel 5.2.

Tabel 5.2 Kebutuhan Fungsional

No.	Kode Fungsi	Nama Fungsi	Deskripsi
1.	F_SIL SCO_01, F_SIL PIA_01, F_SIL PBP_01	<i>Login</i>	SCO, Penyelia, dan PBP dapat masuk ke sistem menggunakan <i>username</i> dan <i>password</i> yang telah disediakan oleh pihak BNI Wilayah Malang
2.	F_SIL SCO_02, F_SIL PIA_02, F_SIL PBP_02	<i>Input Data Lead Manual</i>	SCO, Penyelia, dan PBP dapat mengisi data debitur untuk masuk ke Data Lead pada sistem secara manual
3.	F_SIL PIA_03, F_SIL PBP_03	<i>Input Data Lead Otomatis</i>	Penyelia dan PBP dapat mengisi data debitur untuk masuk ke Data Lead pada sistem secara otomatis dari format excel/.xlsx

No.	Kode Fungsi	Nama Fungsi	Deskripsi
4.	F_SIL SCO_03	Solicit	SCO dapat mengganti status debitur menjadi <i>Solicit</i>
5.	F_SIL SCO_04	Change <i>Solicit</i>	SCO dapat membatalkan status <i>Solicit</i> pada debitur
6.	F_SIL SCO_05	<i>Be Prospect</i>	SCO dapat melakukan <i>request</i> kepada Penyelia untuk mengganti status debitur menjadi <i>Prospect</i>
7.	F_SIL PIA_04	Prospect	Penyelia dapat mengubah status debitur menjadi <i>Prospect</i> atas <i>request</i> dari SCO
8.	F_SIL SCO_06	<i>Be Pipeline</i>	SCO dapat mengisi data untuk proses <i>Pipeline</i>
9.	F_SIL PBP_04	Approval Pipeline	PBP dapat mengubah status debitur menjadi <i>Pipeline</i> atas permintaan dari SCO
10.	F_SIL PBP_05	<i>Recheck</i>	PBP dapat melakukan permintaan kepada SCO untuk melakukan revisi data
11.	F_SIL PIA_05, F_SIL PBP_06	Refuse Data Lead	Penyelia dan PBP dapat membatalkan permohonan <i>lending</i> debitur
12.	F_SIL SCO_07	Melihat Refuse Data Lead	SCO dapat melihat daftar permohonan <i>lending</i> debitur yang dibatalkan
13.	F_SIL SCO_08	<i>Approval</i> Notification	SCO akan mendapatkan notifikasi saat proses disetujui
14.	F_SIL PIA_06, F_SIL PBP_07	<i>Request</i> Notification	Penyelia dan PBP mendapatkan notifikasi saat ada permintaan persetujuan proses

5.4 Analisis Kebutuhan Non Fungsional

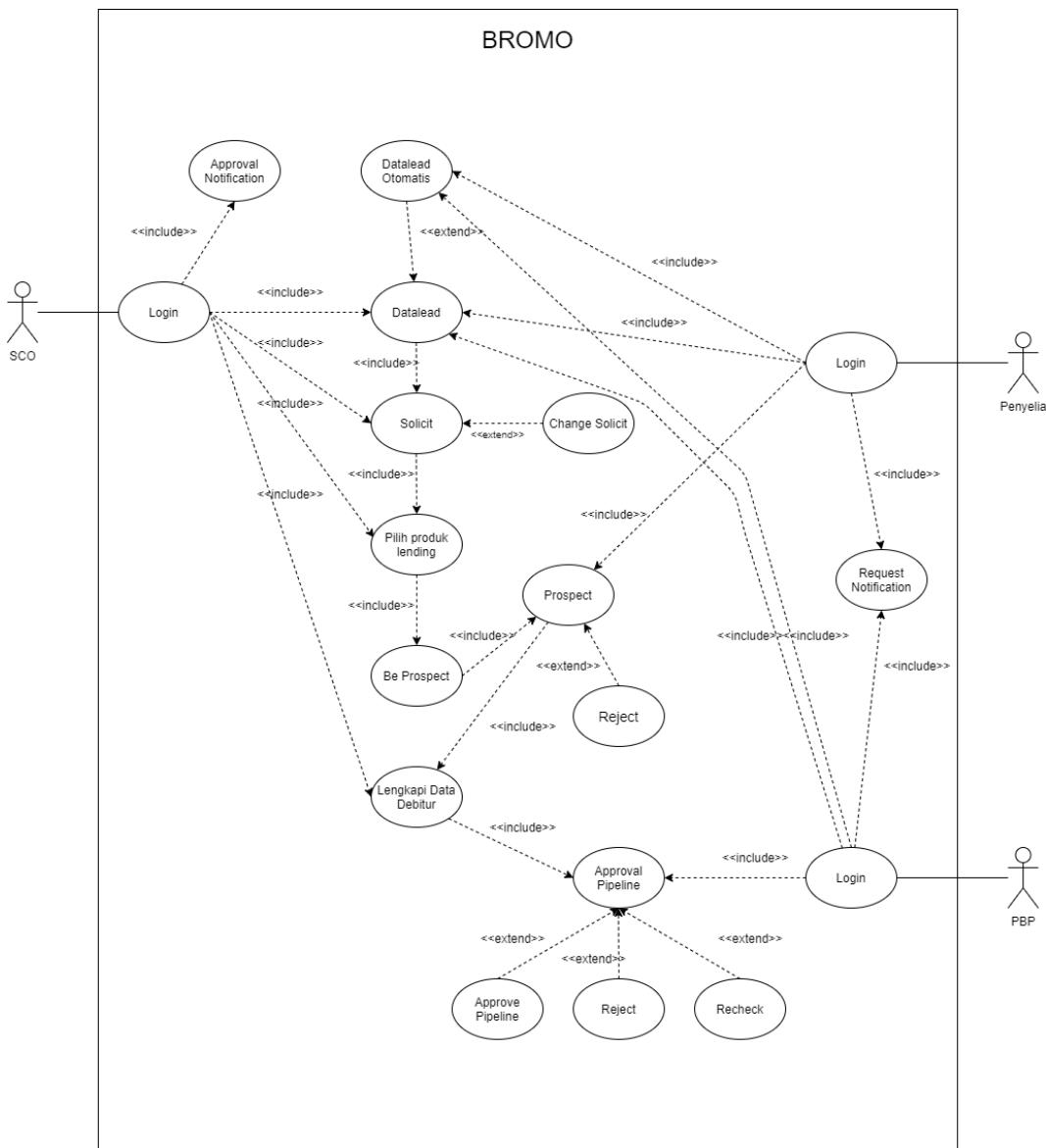
Kebutuhan Non Fungsional adalah kebutuhan yang menitikberatkan pada properti perilaku yang dimiliki oleh sistem. Kebutuhan non fungsional pada sistem ini dijelaskan pada Tabel 5.3.

Tabel 5.3 Kebutuhan Non Fungsional

No.	Kode Fungsi	Nama Fungsi	Deskripsi
1.	NF_SIL_01	<i>Compatibility Browser</i>	Sistem dapat berjalan pada aplikasi <i>web browser</i> yang berbeda
2.	NF_SIL_02	<i>Responsive Web Design</i>	Sistem memiliki desain antarmuka pengguna yang responsif, baik ketika menggunakan aplikasi <i>desktop browser</i> maupun <i>tablet browser</i>

5.5 Use Case Diagram

Use case diagram merupakan gambaran scenario dari interaksi antara pengguna dengan sistem. *Use case diagram* menggambarkan hubungan antara aktor dan kegiatan yang dapat dilakukannya terhadap sistem (Booch, 1999). Diagram Use Case dari Sistem Informasi *Lending BROMO* digambarkan seperti pada Gambar 5.2.



Gambar 5.2 Use Case Diagram Sistem Informasi Lending BROMO

5.6 Skenario Use Case

Pada bagian ini terdapat tabel-tabel yang menjelaskan skenario dari setiap kebutuhan yang telah dipaparkan pada analisis kebutuhan fungsional dan non fungsional.

Tabel 5.4 Skenario Use Case Login

Nama Use Case	<i>Login</i>
Kode	F_SIL SCO_01, F_SIL PIA_01, F_SIL PBP_01

Kebutuhan	
Aktor	SCO, Penyelia, PBP
Tujuan	Aktor masuk ke sistem menggunakan <i>username</i> dan <i>password</i> yang telah disediakan oleh pihak BNI Wilayah Malang.
Precondition	Aktor telah berada di halaman <i>login</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>login</i> 2. Sistem menampilkan form <i>login</i> berupa <i>username</i> dan <i>password</i> 3. Aktor memasukkan <i>username</i> dan <i>password</i> 4. Aktor menekan tombol ‘Login’ 5. Sistem mengecek <i>username</i> dan <i>password</i> yang dimasukkan oleh aktor 6. Sistem menampilkan halaman <i>dashboard</i>
Postcondition	Aktor berhasil masuk ke dalam sistem dan tampil halaman beranda
Alternative Flow	<ol style="list-style-type: none"> 1. Jika aktor memasukkan <i>username</i> atau <i>password</i> yang salah atau tidak terdaftar dalam sistem, maka sistem akan mengarahkan aktor pada halaman <i>login</i> kembali 2. Jika aktor hanya mengisi salah satu form diantara <i>username</i> dan <i>password</i>, maka sistem akan memberikan peringatan pada form yang kosong

Tabel 5.5 Skenario Use Case Input Data Lead Manual

Nama Use Case	<i>Input Data Lead Manual</i>
Kode Kebutuhan	F_SIL SCO_02, F_SIL PIA_02, F_SIL PBP_02
Aktor	SCO, Penyelia, PBP
Tujuan	Aktor mengisi data debitur untuk masuk ke Data Lead pada sistem secara manual
Precondition	Aktor telah berada di halaman <i>input Data Lead</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu ‘Input Data Lead’ 4. Aktor memilih kategori peminjaman dengan opsi ‘CR’,

Main Flow	<p>kemudian memilih kantor cabang tempat debitur mengajukan pinjaman</p> <ol style="list-style-type: none"> 5. Aktor memilih jenis kelamin debitur (Bapak/Ibu) 6. Aktor memilih jenis pekerjaan debitur 7. Aktor memasukkan alamat debitur 8. Aktor memasukkan nomor KTP calon debitur 9. Aktor memasukkan kabupaten, kecamatan, desa, dan kodepos debitur 10. Aktor memasukkan nomor telepon debitur 11. Aktor menekan tombol ‘Submit’ 12. Sistem melakukan validasi terhadap form yang diisi oleh aktor
Postcondition	Aktor berhasil melakukan Data Lead pada debitur
Alternative Flow	<ol style="list-style-type: none"> 1. Jika aktor mengosongkan salah satu atau beberapa <i>form</i>, maka sistem akan menolak input dari aktor dan memberi peringatan pada <i>form</i> yang masih kosong. 2. Jika aktor memasukkan nilai yang tidak sesuai pada jenis <i>form</i>, maka sistem akan memberi peringatan pada <i>form</i> yang nilainya tidak sesuai pada jenis <i>form</i> 3. Jika data pada <i>form</i> nomor KTP telah tersedia di dalam <i>database</i>, maka sistem akan menolak input dari aktor dan memberi peringatan bahwa nomor KTP tidak bisa dipakai

Tabel 5.6 Skenario *Use Case Input Data Lead Otomatis*

Nama Use Case	<i>Input Data Lead Otomatis</i>
Kode Kebutuhan	F_SIL_PIA_03, F_SIL_PBP_03
Aktor	Penyelia, PBP
Tujuan	Aktor mengisi data debitur untuk masuk ke Data Lead pada sistem secara otomatis dari format excel/.xlsx
Precondition	Aktor telah berada di halaman <i>input Data Lead</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu ‘Input Data Lead’ 4. Aktor mengunduh <i>template file</i> excel yang telah disediakan oleh sistem

Main Flow	5. Aktor mengisi data pada <i>file excel</i> 6. Aktor mengunggah <i>file excel</i> yang telah diisi 7. Aktor menekan tombol 'Import'
Postcondition	Aktor berhasil melakukan Data Lead pada debitur
Alternative Flow	Sistem akan menolak unggahan aktor jika format <i>file</i> yang diunggah aktor bukan dari format excel/.xlsx

Tabel 5.7 Skenario Use Case Solicit

Nama Use Case	Solicit
Kode Kebutuhan	F_SIL_SCO_03
Aktor	SCO
Tujuan	Aktor mengganti status debitur menjadi Solicit
Precondition	Aktor telah berada di halaman <i>List Data Lead Konsumen</i>
Main Flow	1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu Data Lead Konsumen 4. Aktor menekan tombol 'Solicit' pada nasabah yang diinginkan 5. Sistem memberikan konfirmasi kepada aktor berupa <i>pop up</i> 6. Aktor menekan tombol 'Solicit' pada <i>pop up</i>
Postcondition	Aktor berhasil melakukan solicit pada debitur
Alternative Flow	-

Tabel 5.8 Skenario Use Case Change Solicit

Nama Use Case	Change Solicit
Kode Kebutuhan	F_SIL_SCO_04
Aktor	SCO
Tujuan	Aktor membatalkan status Solicit pada debitur
Precondition	Aktor telah berada di halaman <i>Solicit</i>

Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu <i>Solicit</i> 4. Aktor menekan tombol ‘Change <i>Solicit</i>’ pada nasabah yang diinginkan 5. Sistem memberikan konfirmasi kepada aktor berupa <i>pop up</i> 6. Aktor memberikan catatan pada debitur yang akan dibatalkan status <i>solicity</i>nya 7. Aktor menekan tombol ‘Change <i>Solicit</i>’ pada <i>pop up</i>
Postcondition	Aktor berhasil membatalkan <i>solicit</i> pada debitur
Alternative Flow	-

Tabel 5.9 Skenario Use Case *Be Prospect*

Nama Use Case	<i>Be Prospect</i>
Kode Kebutuhan	F_SIL_SCO_05
Aktor	SCO
Tujuan	Aktor melakukan <i>request</i> kepada Penyelia untuk mengganti status debitur menjadi <i>Prospect</i>
Precondition	Aktor telah berada di halaman <i>Solicit</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu <i>Solicit</i> 4. Aktor menekan tombol ‘Be <i>Prospect</i>’ pada nasabah yang diinginkan 5. Aktor akan memilih produk <i>lending</i> pada <i>pop up</i> 6. Aktor menekan tombol ‘Be <i>Prospect</i>’ pada <i>pop up</i>
Postcondition	Aktor berhasil melakukan <i>request</i> kepada Penyelia untuk mengganti status debitur menjadi <i>Prospect</i>
Alternative Flow	-

Tabel 5.10 Skenario Use Case Prospect

Nama Use Case	Prospect
Kode Kebutuhan	F_SIL_PIA_04
Aktor	Penyelia
Tujuan	Aktor mengubah status debitur menjadi Prospect atas <i>request</i> dari SCO.
Precondition	Aktor telah berada di halaman Approval Prospect
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu <i>Approval Lending</i> 4. Aktor menekan tombol ‘Action’ pada nasabah yang diinginkan 5. Sistem akan menampilkan pop up berupa <i>form approval</i> debitur. Detail <i>form</i> adalah sebagai berikut : data SCO yang memasukkan data debitur, data SCO yang mengubah status debitur menjadi solicit dan melakukan permintaan prospect, dan data debitur. 6. Aktor menekan tombol ‘Approve’ pada <i>pop up</i>
Postcondition	Aktor berhasil mengubah status debitur menjadi Prospect atas <i>request</i> dari SCO
Alternative Flow	-

Tabel 5.11 Skenario Use Case Be Pipeline

Nama Use Case	Be Pipeline
Kode Kebutuhan	F_SIL_SCO_06
Aktor	SCO
Tujuan	Aktor mengisi data untuk proses Pipeline
Precondition	Aktor telah berada di halaman All Prospect

Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu <i>All Prospect</i> 4. Aktor menekan tombol ‘Action’ pada nasabah yang diinginkan 5. Sistem akan menampilkan <i>pop up</i> berupa <i>form</i> yang berisikan detail debitur. Detail <i>form</i> adalah sebagai berikut : data SCO yang memasukkan data debitur, data debitur, dan <i>form</i> data ibu debitur 6. Aktor mengisi detail produk <i>lending</i> debitur 7. Aktor mengisi NPWP debitur 8. Aktor mengisi <i>form</i> data ibu debitur
Postcondition	Aktor berhasil melakukan permintaan kepada PBP untuk mengubah status debitur menjadi Pipeline
Alternative Flow	<ol style="list-style-type: none"> 1. Jika aktor mengosongkan salah satu atau beberapa <i>form</i>, maka sistem akan menolak input dari aktor dan memberi peringatan pada <i>form</i> yang masih kosong 2. Jika aktor memasukkan nilai yang tidak sesuai pada jenis <i>form</i>, maka sistem akan memberi peringatan pada <i>form</i> yang nilainya tidak sesuai pada jenis <i>form</i> 3. Jika data pada <i>form</i> nomor KTP telah tersedia di dalam <i>database</i>, maka sistem akan menolak <i>input</i> dari aktor dan memberi peringatan bahwa nomor KTP tidak bisa dipakai

Tabel 5.12 Skenario Use Case Approval Pipeline

Nama Use Case	Approval Pipeline
Kode Kebutuhan	F_SIL_PBP_04
Aktor	PBP
Tujuan	Aktor mengubah status debitur menjadi Pipeline atas permintaan dari SCO
Precondition	Aktor telah berada di halaman <i>Approval Pipeline</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu <i>Approval Lending CR</i> 4. Aktor menekan tombol ‘Action’ pada nasabah yang

Main Flow	diinginkan 5. Sistem akan menampilkan <i>pop up</i> berupa <i>form</i> yang berisikan detail debitur. Detail <i>form</i> adalah sebagai berikut : data SCO yang memasukkan data debitur, data SCO yang mengubah status debitur menjadi solicit dan melakukan permintaan prospect, data debitur, dan data ibu debitur 6. Aktor menekan tombol ‘Approve’ di bagian bawah <i>form</i> 7. Sistem memberikan konfirmasi kepada aktor berupa <i>pop up</i> 8. Aktor menekan tombol ‘Approve’ pada <i>pop up</i>
Postcondition	Aktor berhasil mengubah status debitur menjadi Pipeline.
Alternative Flow	-

Tabel 5.13 Skenario Use Case Recheck

Nama Use Case	<i>Recheck</i>
Kode Kebutuhan	F_SIL_PBP_05
Aktor	PBP
Tujuan	Aktor melakukan permintaan kepada SCO untuk melakukan revisi data
Precondition	Aktor telah berada di halaman <i>Approval Pipeline</i>
Main Flow	1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu <i>Approval Lending CR</i> 4. Aktor menekan tombol ‘Action’ pada nasabah yang diinginkan 5. Sistem akan menampilkan <i>pop up</i> berupa <i>form</i> yang berisikan <i>detail</i> debitur. Detail <i>form</i> adalah sebagai berikut : data SCO yang memasukkan data debitur, data SCO yang mengubah status debitur menjadi solicit dan melakukan permintaan prospect, data debitur, dan data ibu debitur 6. Aktor menekan tombol ‘Recheck’ di bagian bawah <i>form</i> 7. Sistem memberikan konfirmasi kepada aktor berupa <i>pop up</i> 8. Aktor memberikan catatan pada debitur yang datanya ingin direvisi 9. Aktor menekan tombol ‘Recheck’ pada <i>pop up</i>

Postcondition	Aktor berhasil melakukan permintaan kepada SCO untuk melakukan revisi data
Alternative Flow	-

Tabel 5.14 Skenario Use Case Refuse Data Lead

Nama Use Case	Refuse Data Lead
Kode Kebutuhan	F_SIL_PIA_05, F_SIL_PBP_06
Aktor	Penyelia, PBP
Tujuan	Aktor membatalkan permohonan <i>lending</i> debitur
Precondition	Aktor telah berada di halaman <i>Approval Lending</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu <i>Approval Lending</i> 4. Aktor menekan tombol 'Action' pada nasabah yang diinginkan 5. Sistem akan menampilkan <i>pop up</i> berupa <i>form</i> yang berisikan detail debitur 6. Aktor menekan tombol 'Reject' di bagian bawah <i>form</i> 7. Sistem memberikan konfirmasi kepada aktor berupa <i>pop up</i> 8. Aktor memberikan catatan pada debitur yang ditolak 9. Aktor menekan tombol 'Reject' pada <i>pop up</i>
Postcondition	Aktor berhasil membatalkan permohonan <i>lending</i> debitur
Alternative Flow	-

Tabel 5.15 Skenario Use Case Melihat Refuse Data Lead

Nama Use Case	Melihat Refuse Data Lead
Kode Kebutuhan	F_SIL_SCO_07
Aktor	SCO

Tujuan	Aktor melihat daftar permohonan <i>lending</i> debitur yang dibatalkan
Precondition	Aktor telah berada di halaman <i>List Refuse Data Lead</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan menu <i>sidebar</i> 3. Aktor memilih menu <i>List Refuse Data Lead</i> 4. Sistem menampilkan daftar permohonan <i>lending</i> debitur yang dibatalkan
Postcondition	Aktor berhasil melihat daftar permohonan <i>lending</i> debitur yang dibatalkan
Alternative Flow	-

Tabel 5.16 Skenario Use Case Approval Notification

Nama Use Case	<i>Approval Notification</i>
Kode Kebutuhan	F_SIL SCO_08
Aktor	SCO
Tujuan	Aktor mendapatkan notifikasi saat proses disetujui
Precondition	Aktor telah melakukan <i>login</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan notifikasi apabila terdapat proses yang disetujui
Postcondition	Aktor berhasil mendapatkan notifikasi saat proses disetujui
Alternative Flow	-

Tabel 5.17 Skenario Use Case Request Notification

Nama Use Case	<i>Request Notification</i>
Kode Kebutuhan	F_SIL_PIA_06, F_SIL_PBP_07

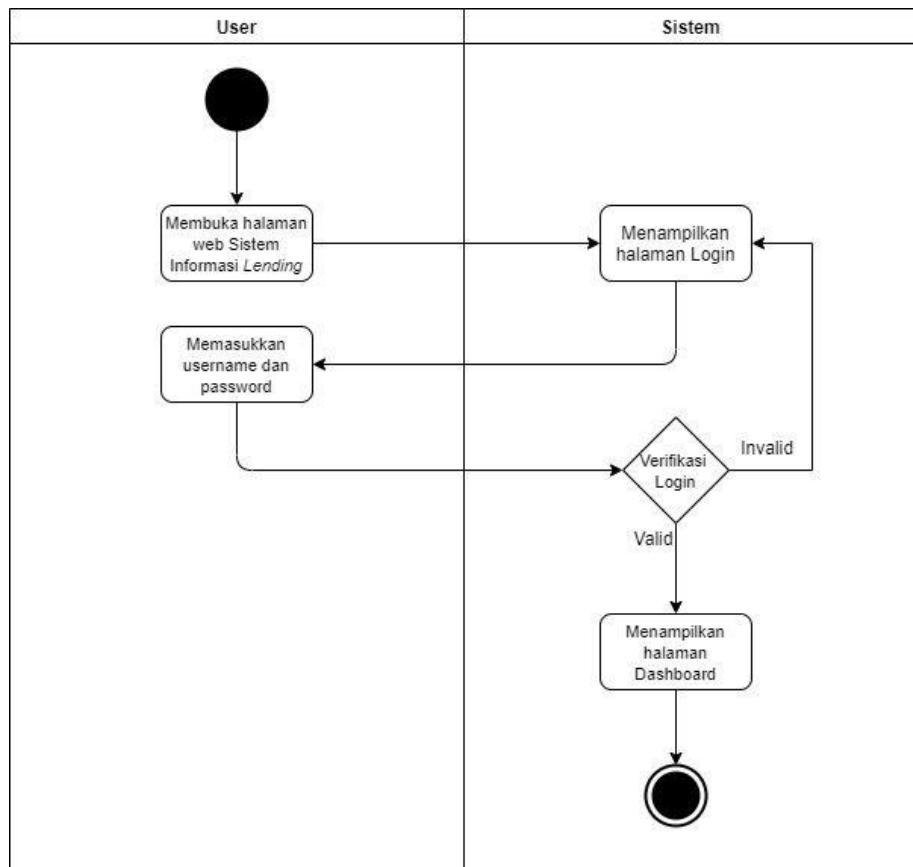
Aktor	Penyelia, PBP
Tujuan	Aktor mendapatkan notifikasi saat ada permintaan persetujuan proses
Precondition	Aktor telah melakukan <i>login</i>
Main Flow	<ol style="list-style-type: none"> 1. Sistem menampilkan halaman <i>dashboard</i> 2. Sistem menampilkan notifikasi saat ada permintaan persetujuan proses
Postcondition	Aktor berhasil mendapatkan notifikasi saat ada permintaan persetujuan proses
Alternative Flow	-

5.7 Activity Diagram

Activity Diagram merupakan *diagram* yang menggambarkan aktivitas yang terjadi dari awal hingga akhir pada sebuah sistem dan langkah-langkah dalam proses kerja sistem dari awal hingga akhir. Berikut adalah *activity diagram* dari Sistem Informasi *Lending* BROMO.

5.7.1 Activity Diagram Login

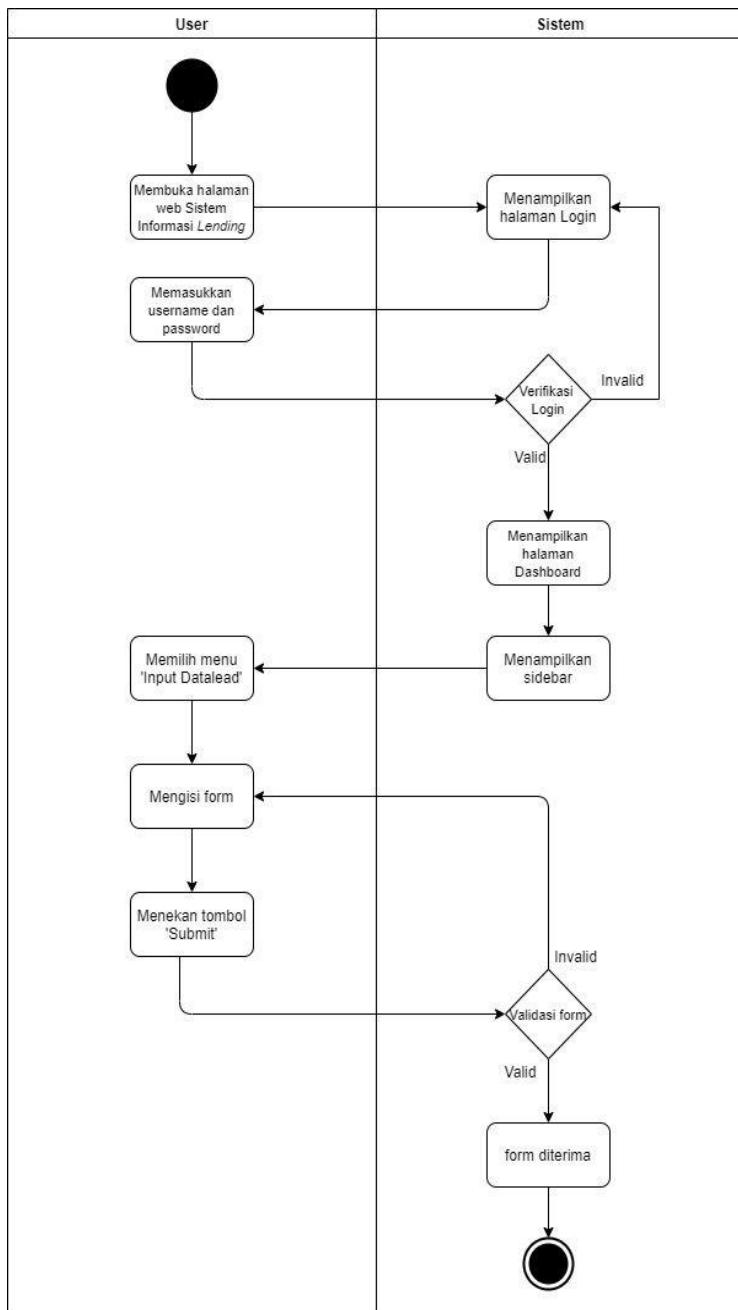
Berikut adalah Activity Diagram Login yang akan dijelaskan pada gambar 5.3. *User* melakukan *login* dengan menggunakan *username* dan *password* yang telah disediakan oleh pihak BNI Wilayah Malang. Jika *username* dan *password* sesuai, *user* akan menemui halaman *dashboard*. Jika tidak sesuai, *user* akan kembali menemui halaman *login*.



Gambar 5.3 Activity Diagram Login

5.7.2 Activity Diagram Input Data Lead Manual

Berikut adalah *Activity Diagram Input Data Lead Manual* yang akan dijelaskan pada gambar 5.4. *User* terlebih dahulu melakukan *login*, kemudian memilih menu “Input Data Lead” yang akan menampilkan halaman *form input Data Lead*. *User* mengisi data debitur dan menekan tombol submit pada *form*. Sistem akan memvalidasi data debitur yang menyebabkan *form* diterima atau tidak. Jika *form* tidak diterima, *user* akan diminta untuk mengulang pengisian *form*.

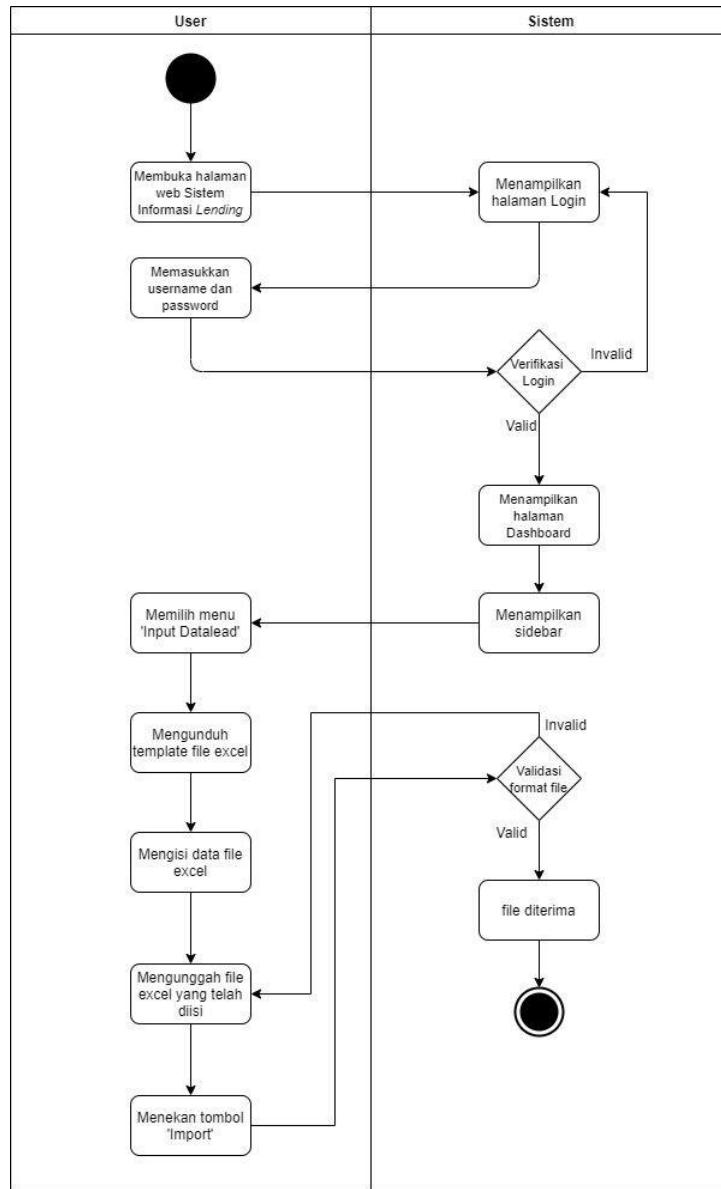


Gambar 5.4 Activity Diagram Input Data Lead Manual

5.7.3 Activity Diagram Input Data Lead Otomatis

Berikut adalah *Activity Diagram Input Data Lead Otomatis* yang akan dijelaskan pada gambar 5.5. *User* terlebih dahulu melakukan *login*, kemudian memilih menu “*Input Data Lead*” yang akan menampilkan halaman *form input Data Lead*. *User* mengunduh *template file* excel kemudian mengisinya dengan data debitur dan diupload ke sistem dengan menekan tombol “*import*”. Sistem akan memvalidasi *file* excel yang diunggah yang menyebabkan dokumen diterima

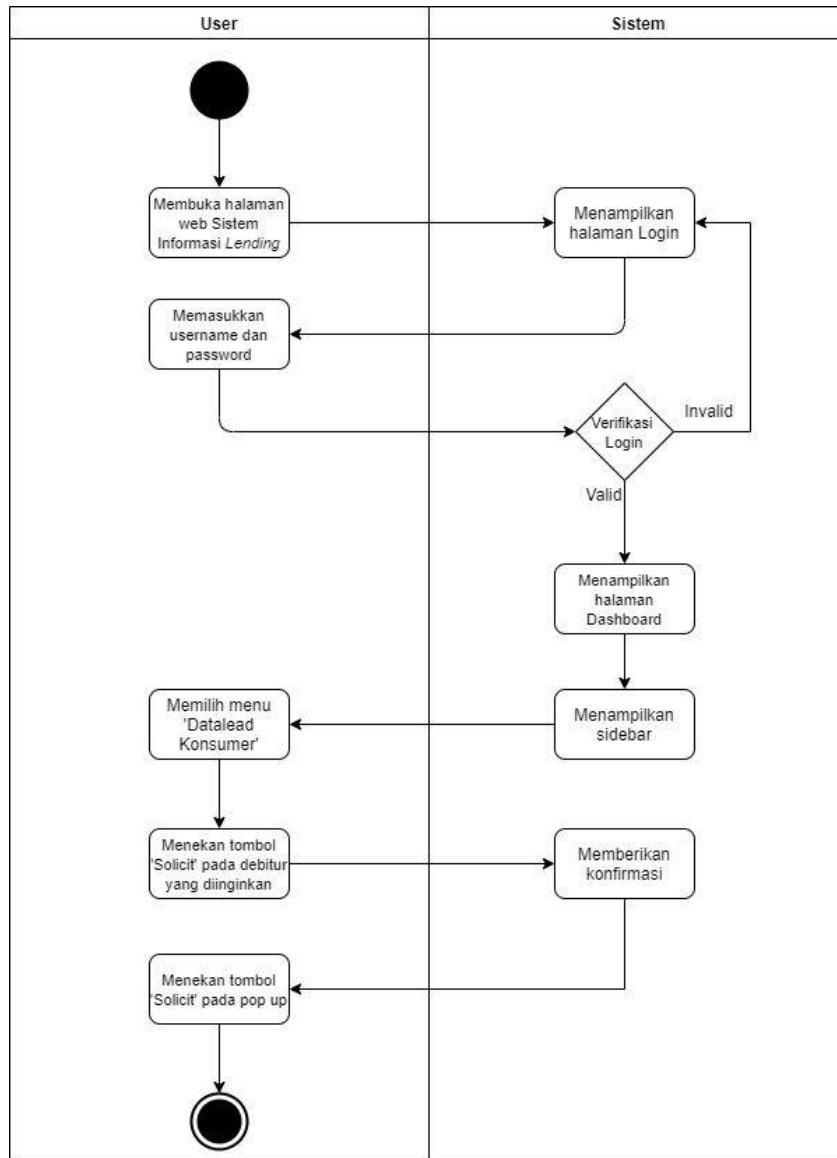
atau tidak. Jika dokumen tidak diterima, *user* akan diminta untuk mengunggah dokumen kembali.



Gambar 5.5 Activity Diagram Input Data Lead Otomatis

5.7.4 Activity Diagram Solicit

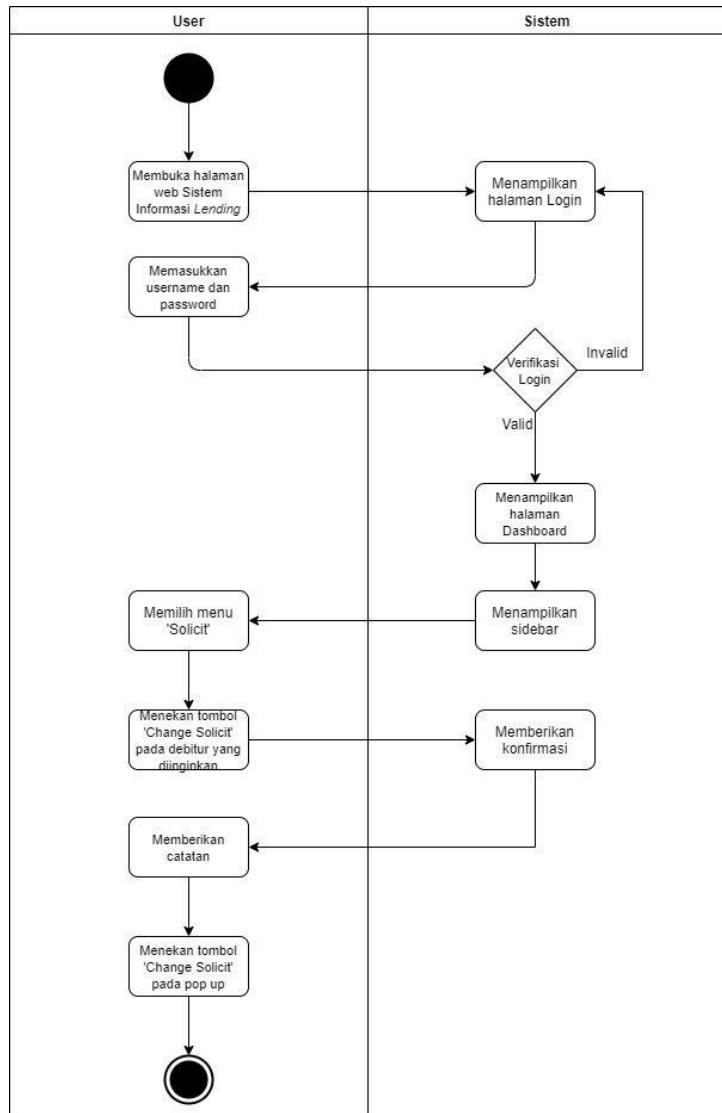
Berikut adalah *Activity Diagram* *Solicit* yang akan dijelaskan pada gambar 5.6. *User* terlebih dahulu melakukan login, kemudian memilih menu “Data Lead Konsumen” yang akan menampilkan halaman *List Data Lead*. *User* menekan tombol “Solicit” pada debitur yang diinginkan dan akan memunculkan pesan konfirmasi untuk melakukan *Solicit*.



Gambar 5.6 Activity Diagram Solicit

5.7.5 Activity Diagram Change Solicit

Berikut adalah *Activity Diagram Change Solicit* yang akan dijelaskan pada gambar 5.7. *User* terlebih dahulu melakukan *login*, kemudian memilih menu "Solicit" yang akan menampilkan halaman *All Solicit*. *User* menekan tombol "Change Solicit" pada debitur yang diinginkan dan memberikan *note* alasan debitur dibatalkan.

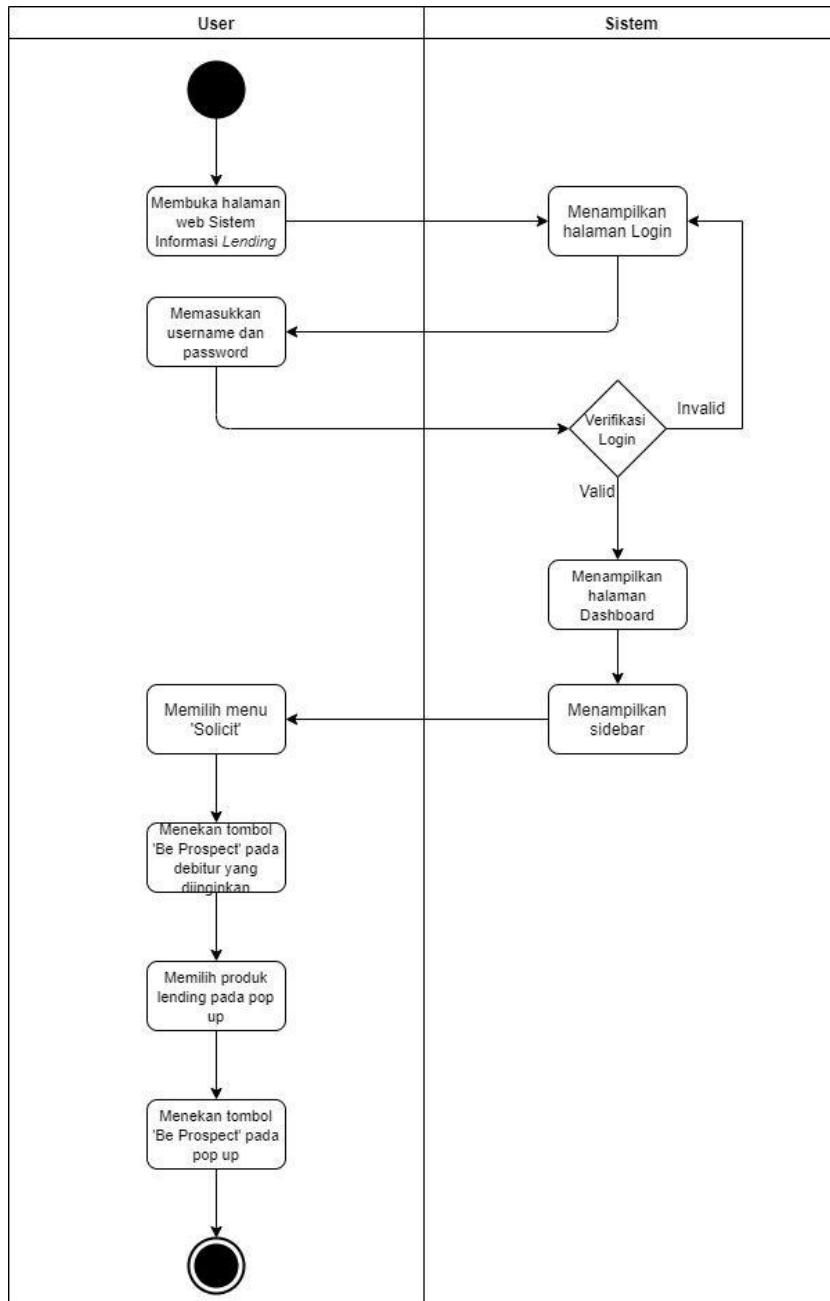


Gambar 5.7 Activity Diagram Change Solicit

5.7.6 Activity Diagram Be Prospect

Berikut adalah *Activity Diagram Be Prospect* yang akan dijelaskan pada gambar 5.8. *User* terlebih dahulu melakukan *login*, kemudian

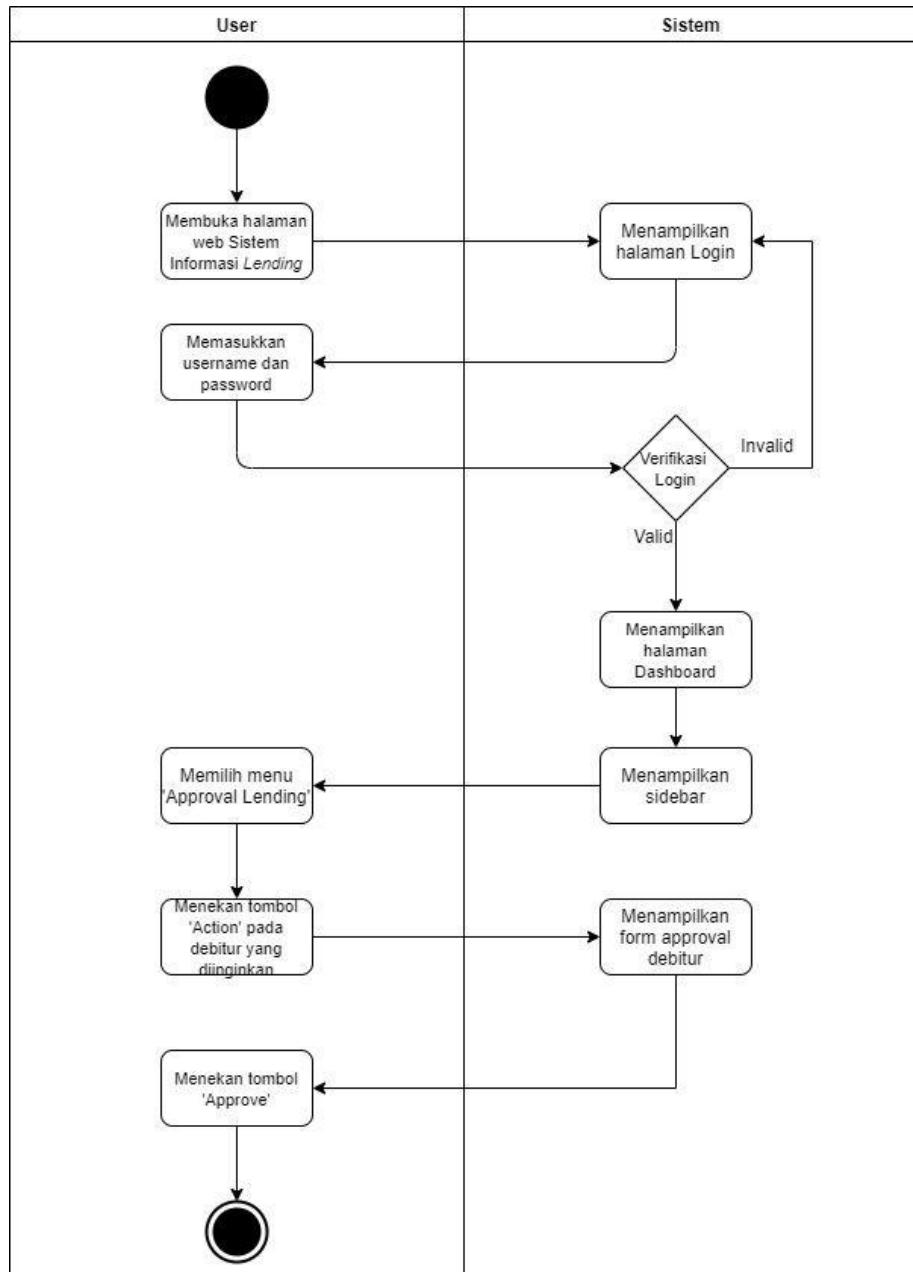
“*Solicit*” yang akan menampilkan halaman *All* *Solicit*. *User* menekan tombol “*Be Prospect*” pada debitur yang diinginkan dan memilih produk.



Gambar 5.8 Activity Diagram Be Prospect

5.7.7 Activity Diagram Approval Prospect

Berikut adalah *Activity Diagram Approval Prospect* yang akan dijelaskan pada gambar 5.9. *User* terlebih dahulu melakukan *login*, kemudian “Approval Lending” yang akan menampilkan halaman *Approval Prospect*. *User* menekan tombol “Action” pada pilihan debitur yang akan menampilkan *form approval*, kemudian menekan tombol “Approve”.

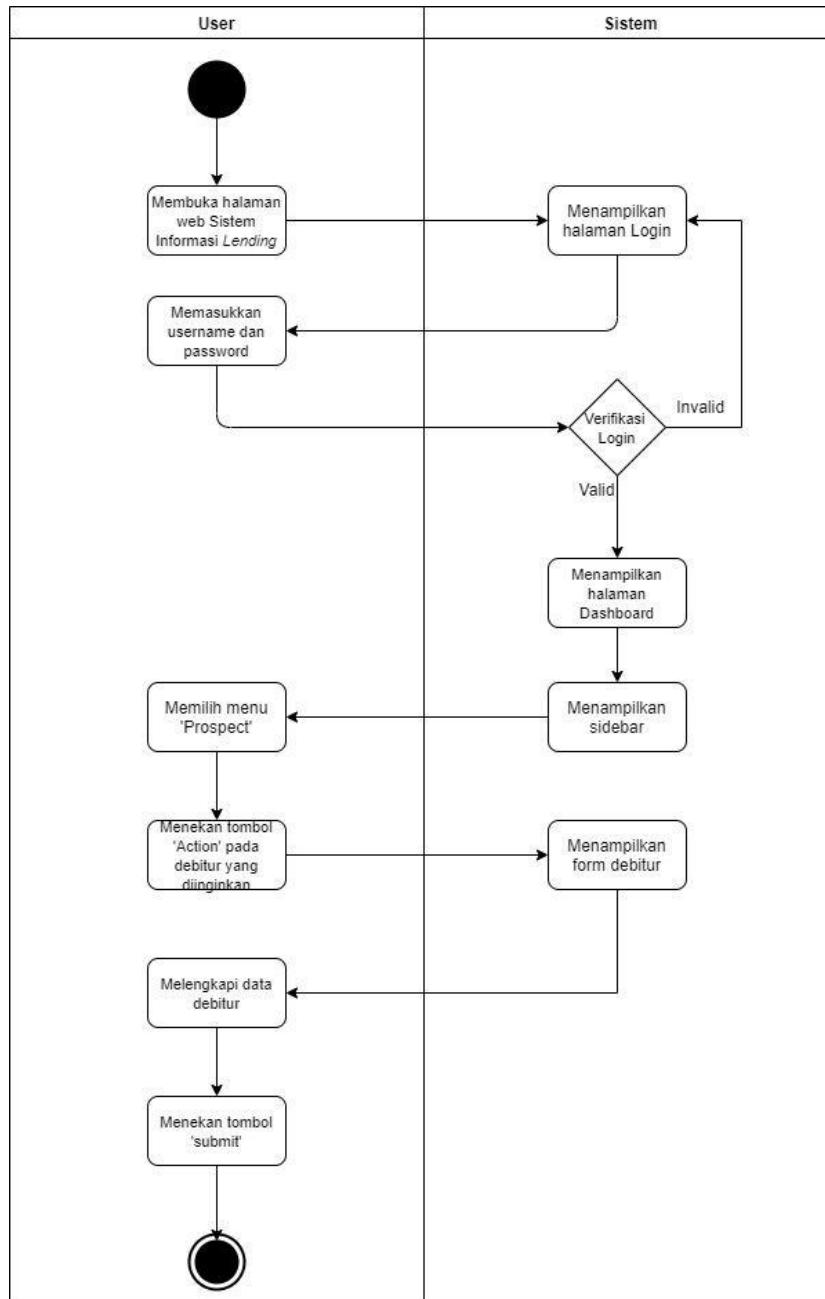


Gambar 5.9 Activity Diagram Approval Prospect

5.7.8 Activity Diagram Be Pipeline

Berikut adalah *Activity Diagram Be Pipeline* yang akan dijelaskan pada gambar 5.10. *User* terlebih dahulu melakukan *login*, kemudian

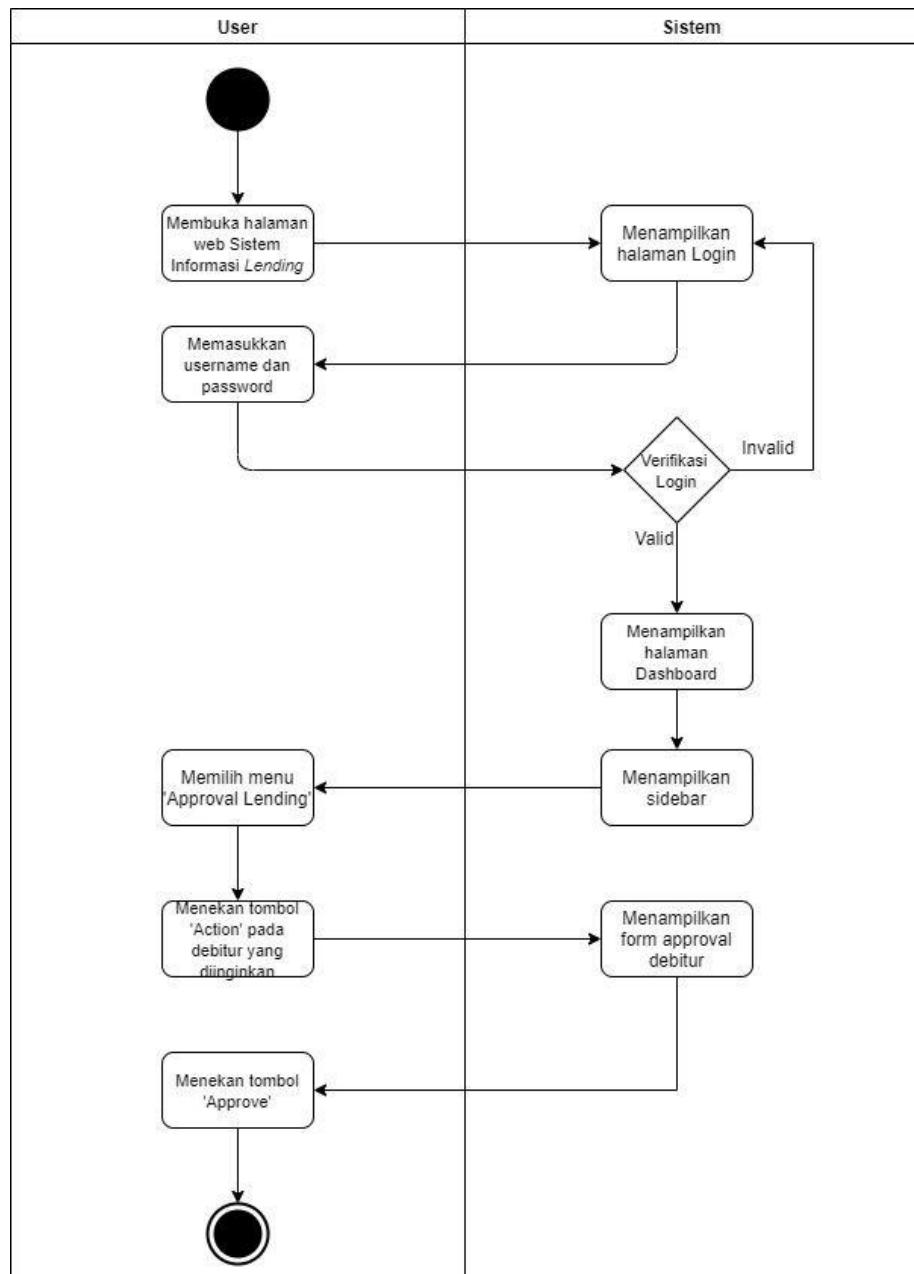
“Prospect” yang akan menampilkan halaman *All Prospect*. *User* menekan tombol “Action” pada pilihan debitur yang akan menampilkan *form* untuk melengkapi data debitur.



Gambar 5.10 Activity Diagram Be Pipeline

5.7.9 Activity Diagram Approval Pipeline

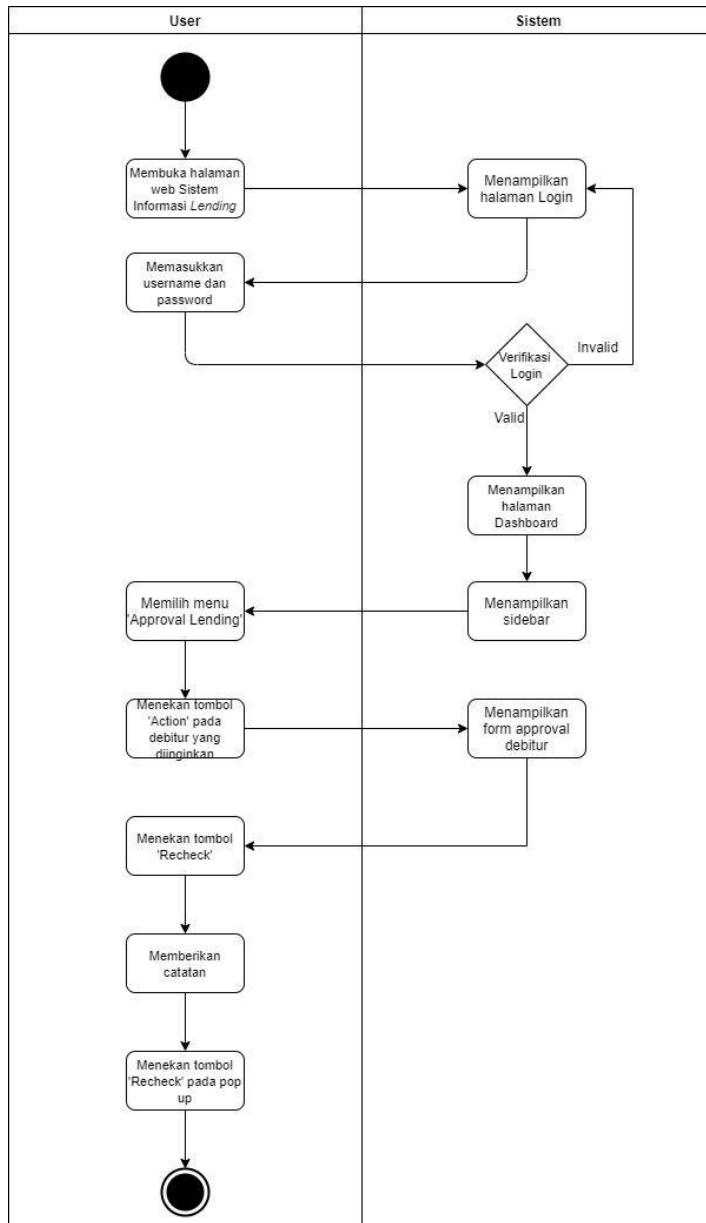
Berikut adalah *Activity Diagram Approval Pipeline* yang akan dijelaskan pada gambar 5.11. *User* terlebih dahulu melakukan *login*, kemudian memilih menu “Approval Lending” yang akan menampilkan halaman *Approval Pipeline*. *User* menekan tombol “Action” pada pilihan debitur yang akan menampilkan *form approval*, kemudian menekan tombol “Approve”.



Gambar 5.11 Activity Diagram Approval Pipeline

5.7.10 Recheck

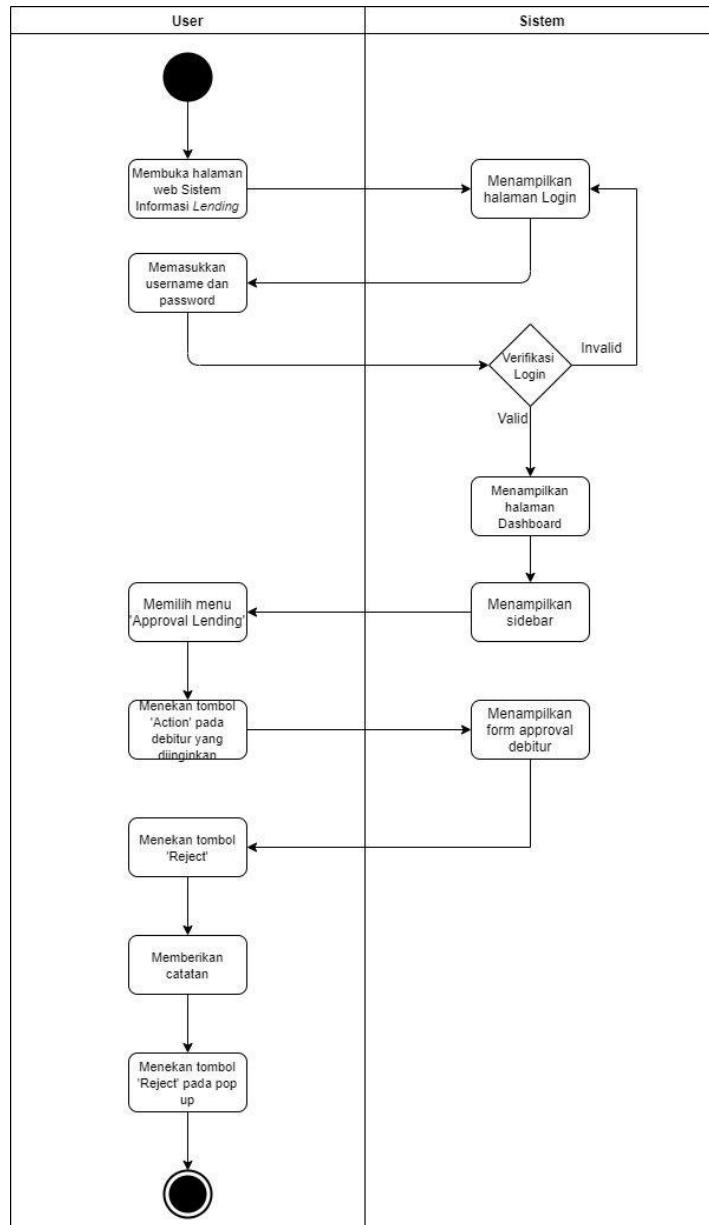
Activity Diagram *Recheck* yang akan dijelaskan pada gambar 5.12. *User* terlebih dahulu melakukan *login*, kemudian memilih menu “Approval Lending” yang akan menampilkan halaman *Approval Pipeline*. *User* menekan tombol “Action” pada pilihan debitur yang akan menampilkan *form approval*, kemudian menekan tombol “Re-Check” dan memberikan *note* pengecekan ulang data debitur.



Gambar 5.12 Activity Diagram Recheck

5.7.11 Refuse Data Lead

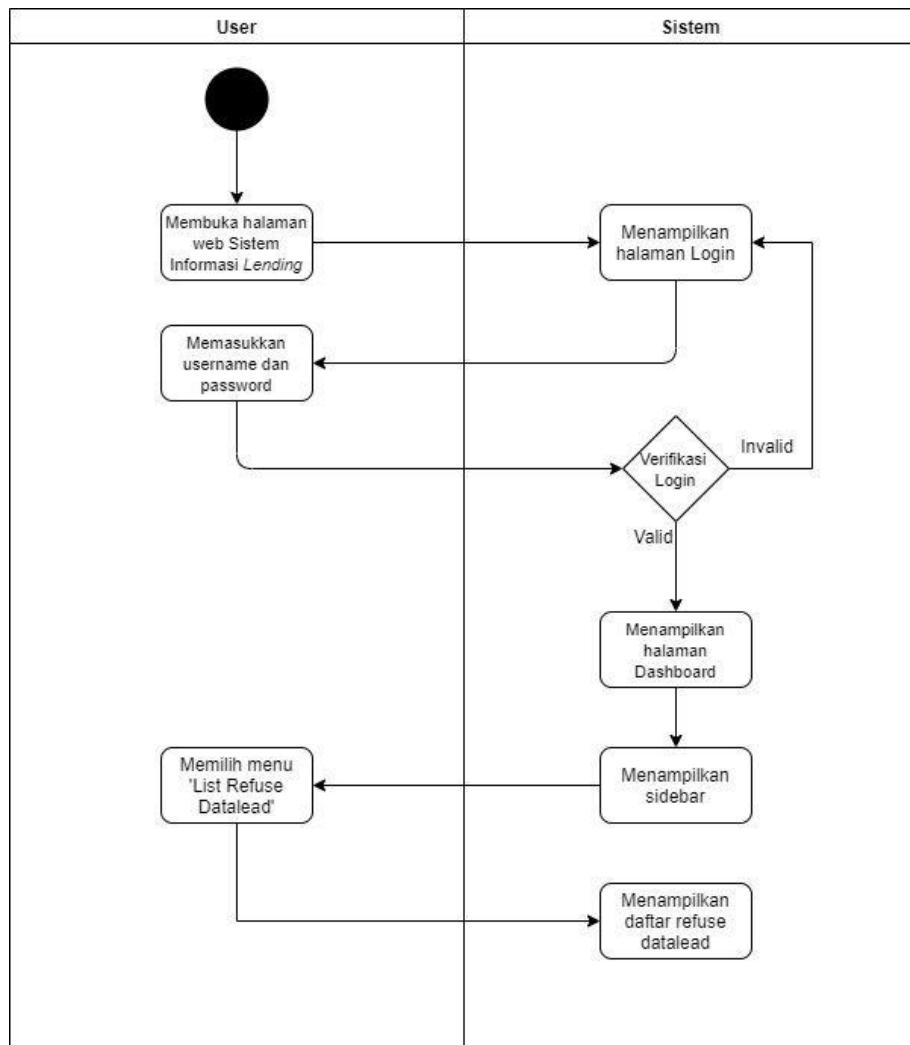
Activity Diagram Refuse Data Lead yang akan dijelaskan pada gambar 5.13. User terlebih dahulu melakukan *login*, kemudian memilih menu “Approval Lending” yang akan menampilkan halaman *Approval Prospect/Approval Pipeline*. User menekan tombol “Action” pada pilihan debitur yang akan menampilkan *form approval*, kemudian menekan tombol “Reject” dan memberikan catatan penolakan debitur.



Gambar 5.13 Activity Diagram Refuse Data Lead

5.7.12 Melihat Refuse Data Lead

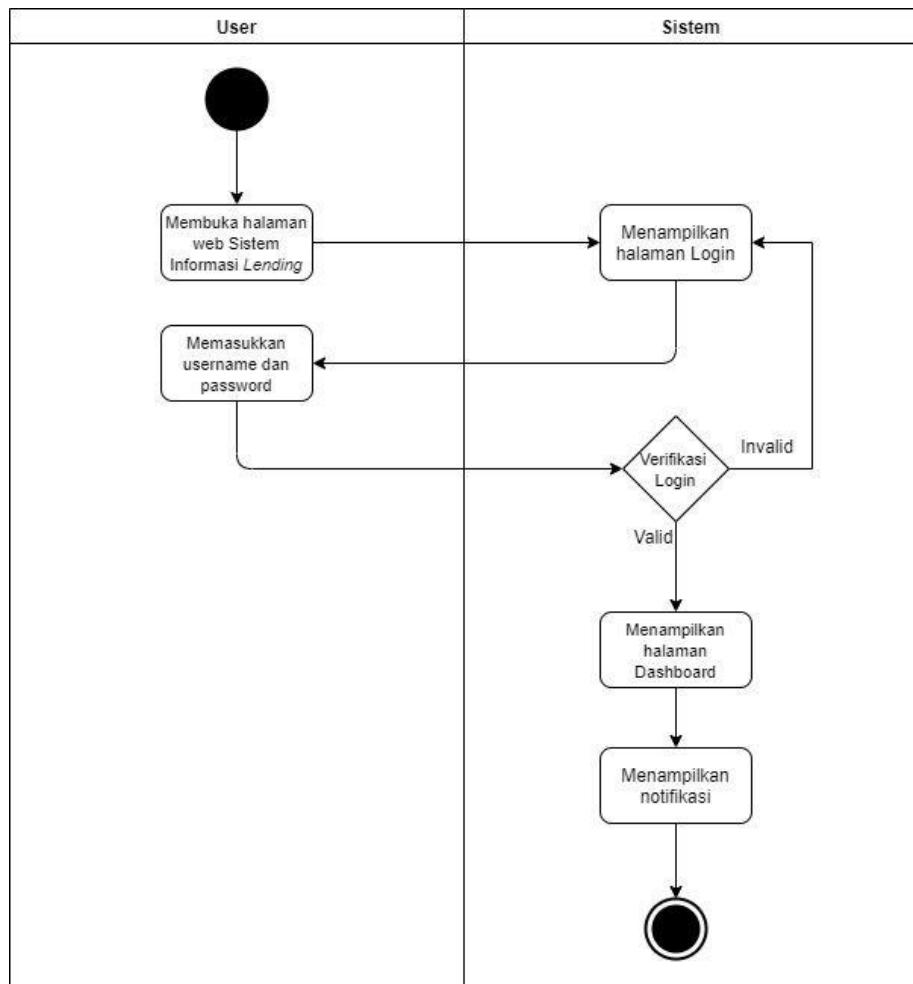
Activity Diagram Melihat Refuse Data Lead yang akan dijelaskan pada gambar 5.14. User terlebih dahulu melakukan *login*, kemudian memilih menu "List Refuse Data Lead" yang akan menampilkan data-data debitur yang telah ditolak.



Gambar 5.14 Activity Diagram Melihat Refuse Data Lead

5.7.13 Activity Diagram Notification

Berikut adalah *Activity Diagram* Melihat Refuse Data Lead yang akan dijelaskan pada gambar 5.15. *User* terlebih dahulu melakukan *login*, kemudian *user* dapat melihat notifikasi yang muncul.

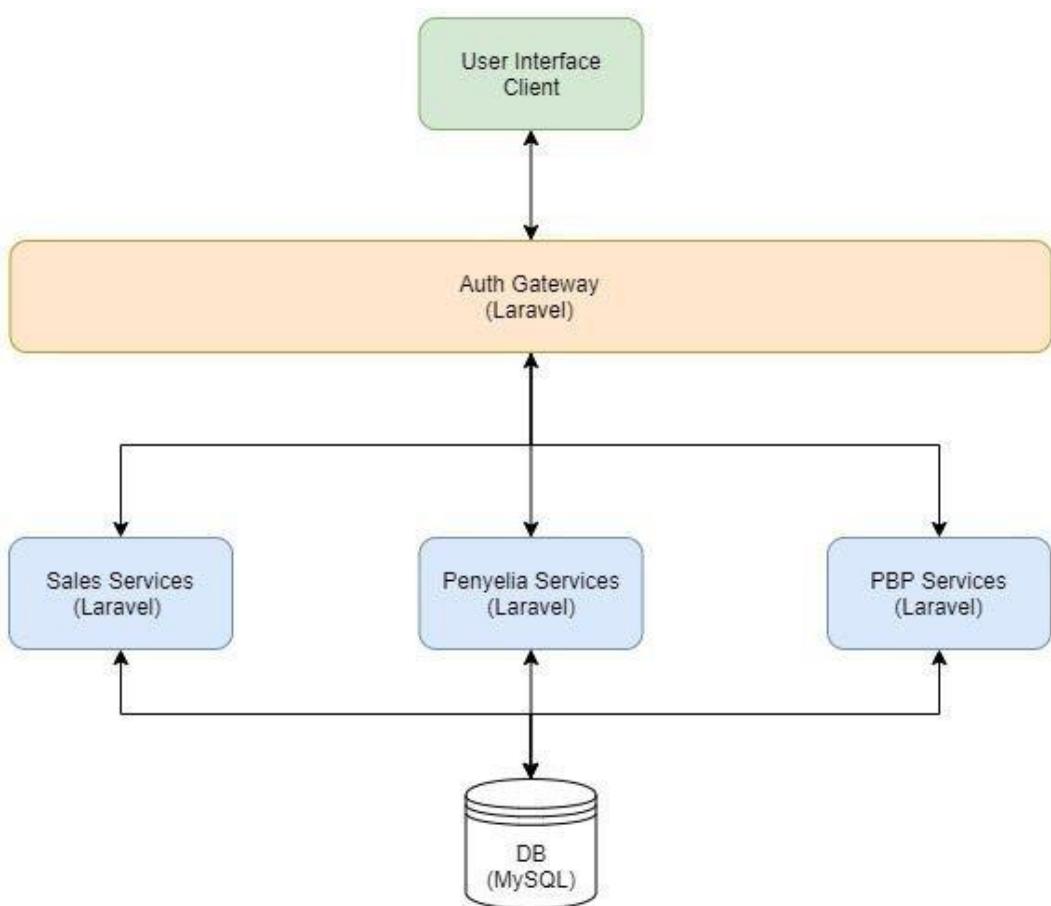


Gambar 5.15 Activity Diagram Notification

BAB 6 PERANCANGAN

6.1 Arsitektur Sistem

Arsitektur Sistem Informasi *Lending* BROMO terdapat *browser* sebagai *client*, Laravel sebagai *server*, dan MySQL sebagai *database*. Arsitektur Sistem Informasi *Lending* BROMO ini digambarkan pada gambar 6.1. Dimana proses otentifikasi pada sistem dilakukan menggunakan *server* Laravel untuk menentukan peran yang dimiliki oleh akun pengguna. Setelah peran diketahui, *client* diarahkan ke *service* spesifik berdasarkan peran yang telah ditetapkan. Tiap *service* di-deploy pada *Laravel server* yang berbeda-beda, tetapi memiliki *database* MySQL yang terpusat.



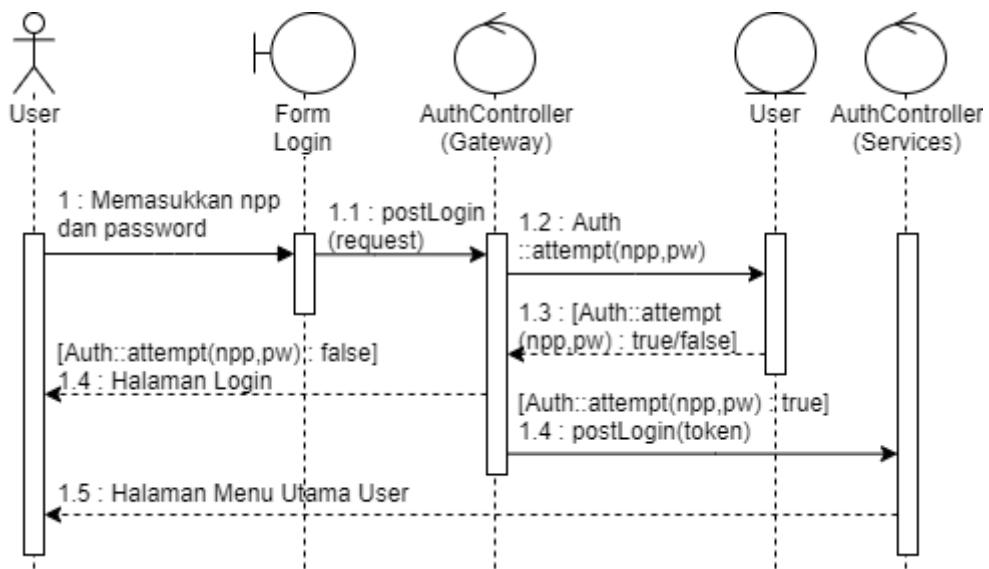
Gambar 6.1 Arsitektur Sistem Informasi *Lending* BROMO

6.2 Pemodelan Sequence Diagram

Pemodelan *Sequence Diagram* ditujukan untuk memodelkan interaksi dan skenario yang dijalankan dalam sistem. Interaksi dalam Sistem Informasi *Lending* BROMO dibagi berdasarkan *staging data*, yaitu Data Lead, Solicit, Prospect, dan Pipeline.

6.2.1 Sequence Diagram Login

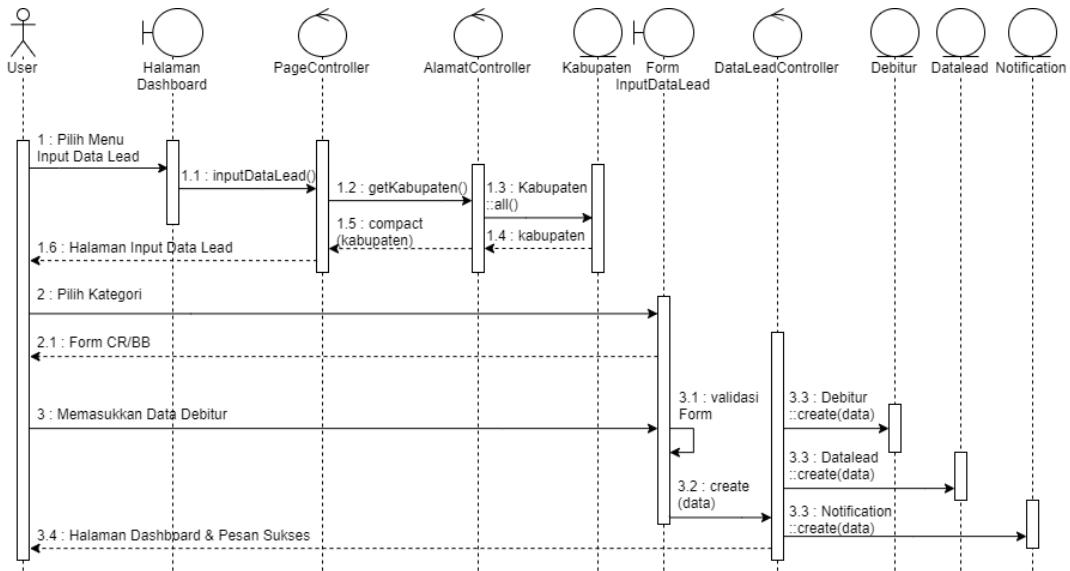
Berikut adalah *Sequence Diagram Login* oleh *User* yang dijelaskan pada gambar 6.2. Pertama-tama *User* yang terdiri dari SCO, Penyelia, maupun PBP memasuki halaman *Login*, kemudian mengisikan npp dan password lalu menekan tombol “Login”. Sistem akan mengecek *credentials user* ke *database User* dan mengembalikan halaman *user* sesuai dengan kewenangannya. Jika *credentials* tidak ditemukan, *user* akan kembali menemui halaman *Login*. *User* dapat menekan tombol “Lupa Password” dan memasukkan npp di mana sistem akan mengirimkan password baru *via e-mail* yang ter registrasi pada sistem.



Gambar 6.2 Sequence Diagram Login

6.2.2 Sequence Diagram Input Data Lead

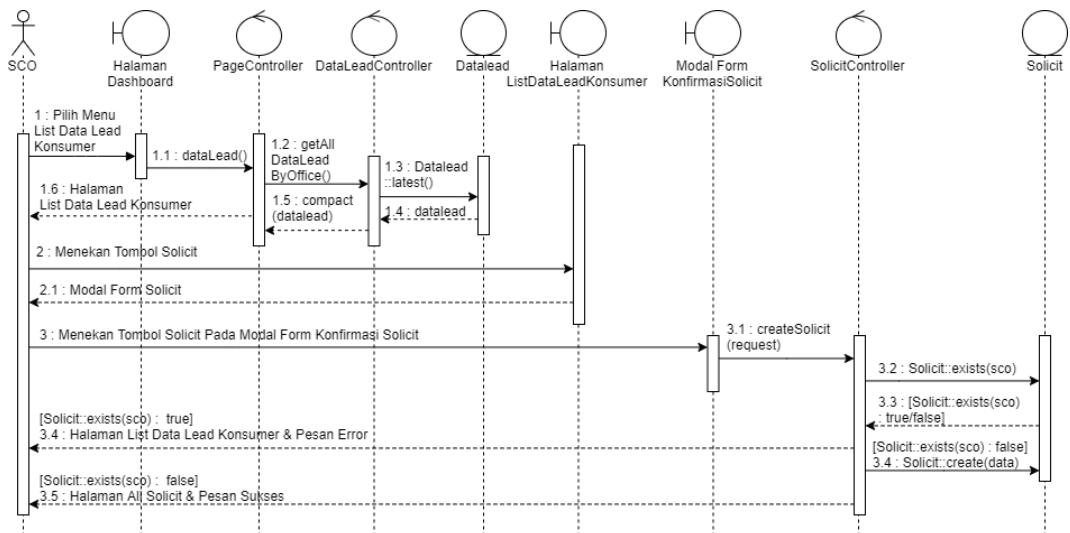
Berikut adalah *Sequence Diagram Input Data Lead* oleh *User* yang dijelaskan pada gambar 6.3. Pertama-tama *User* yang terdiri dari SCO, Penyelia, maupun PBP memasuki halaman *Input Data Lead*, kemudian memilih proses *input* dengan cara manual atau otomatis, namun proses *input* secara otomatis hanya dapat dilakukan oleh Penyelia atau PBP. Dalam proses *input* manual, *User* akan memasukkan data debitur, sementara dalam proses otomatis, *User* akan mengupload dokumen data debitur berekstensi excel/.xlsx. Setelah *input* secara manual atau otomatis selesai, data debitur akan disimpan dalam *database Data Lead* dan *database Debitur*.



Gambar 6.3 Sequence Diagram Input Data Lead

6.2.3 Sequence Diagram Solicit

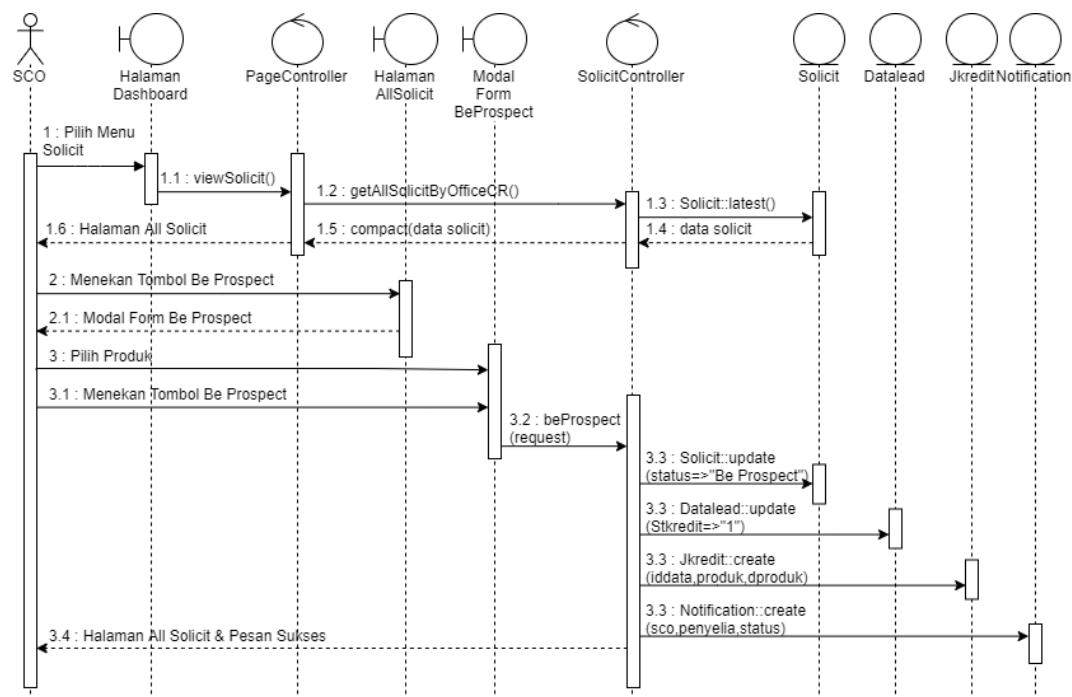
Berikut adalah *Sequence Diagram* *Solicit* oleh SCO yang dijelaskan pada gambar 6.4. Pertama-tama SCO memasuki halaman *List Data Lead* dan memilih data yang akan dilakukan proses *Solicit* dengan menekan tombol buku yang memunculkan sebuah *pop up*. Pada *pop up* tersebut SCO menekan tombol “*Solicit*” yang akan menjadikan *Solicit*. Jika sebelumnya SCO pernah mengambil data tersebut, sistem akan menampilkan pesan *error*. Jika belum, data akan disimpan ke dalam *database* *Solicit* dan dilakukan *update* data pada *database* Data Lead dengan mengubah “*StKredit*” menjadi “1”.



Gambar 6.4 Sequence Diagram Solicit

6.2.4 Sequence Diagram Be Prospect

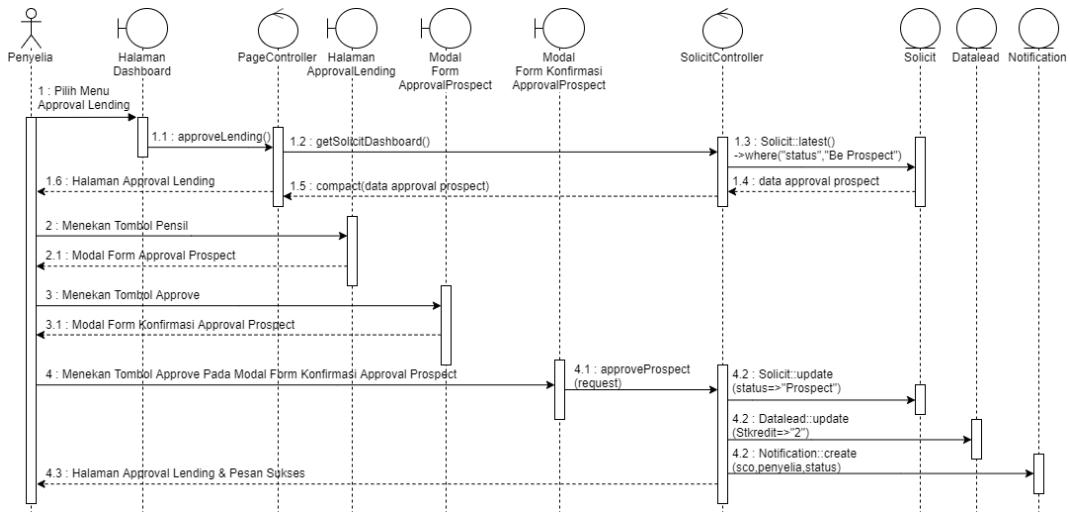
Berikut adalah *Sequence Diagram Be Prospect* oleh SCO yang dijelaskan pada gambar 6.5. SCO memasuki halaman All Solicit kemudian memilih data Solicit yang diinginkan, lalu menekan tombol “Change Solicit” atau “Be Prospect” pada data tersebut. Tombol “Change Solicit” akan mengembalikan data menjadi Data Lead dengan *note*, sementara tombol “Be Prospect” akan meminta SCO untuk memilih produk yang diinginkan, dan melakukan update data pada *database* Solicit.



Gambar 6.5 Sequence Diagram Be Prospect

6.2.5 Sequence Diagram Approval Prospect

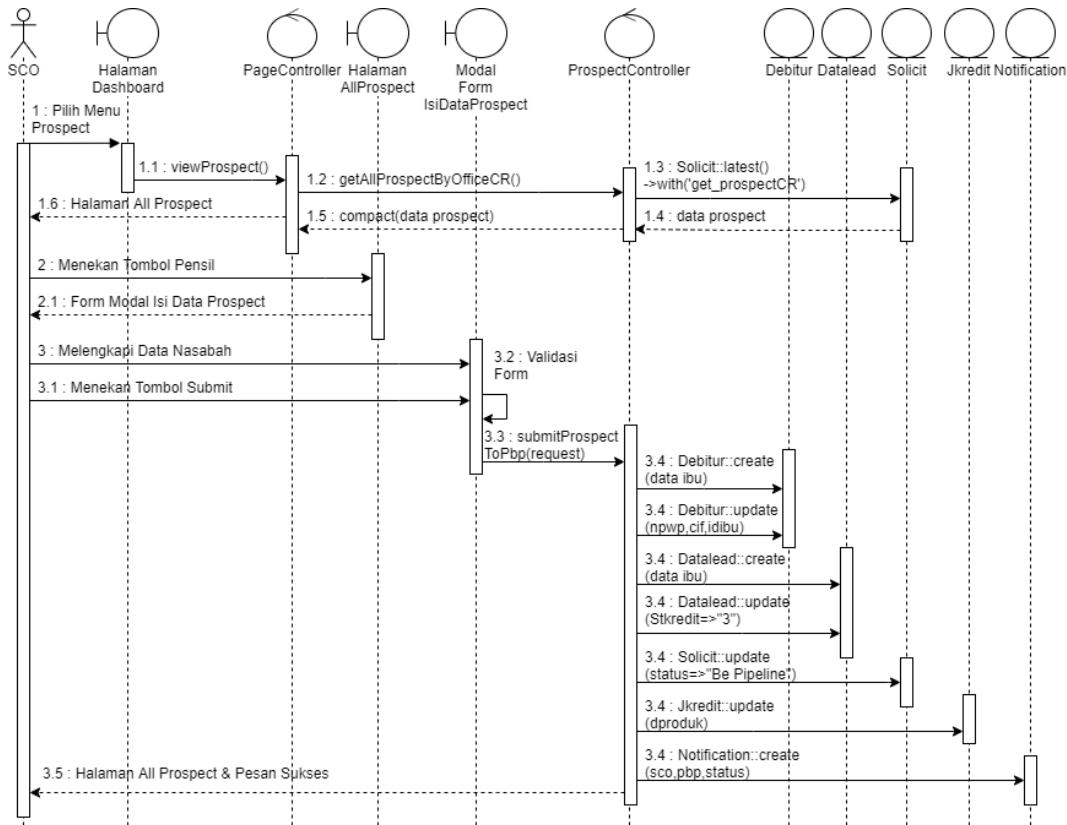
Berikut adalah *Sequence Diagram Approval Prospect* oleh SCO yang dijelaskan pada gambar 6.6. Penyelia memasuki halaman Approval Penyelia kemudian memilih data *pending approval* Prospect yang diinginkan, lalu menekan tombol *approve* atau *reject* pada data tersebut. Jika menekan tombol *approve*, akan dilakukan *insert* pada *database* JKredit dan update pada *database* Data Lead dengan mengubah “StKredit” menjadi “3”. Sementara jika menekan tombol *reject*, akan dilakukan *insert* pada *database* Cdatalead.



Gambar 6.6 Sequence Diagram Approval Prospect

6.2.6 Sequence Diagram Be Pipeline

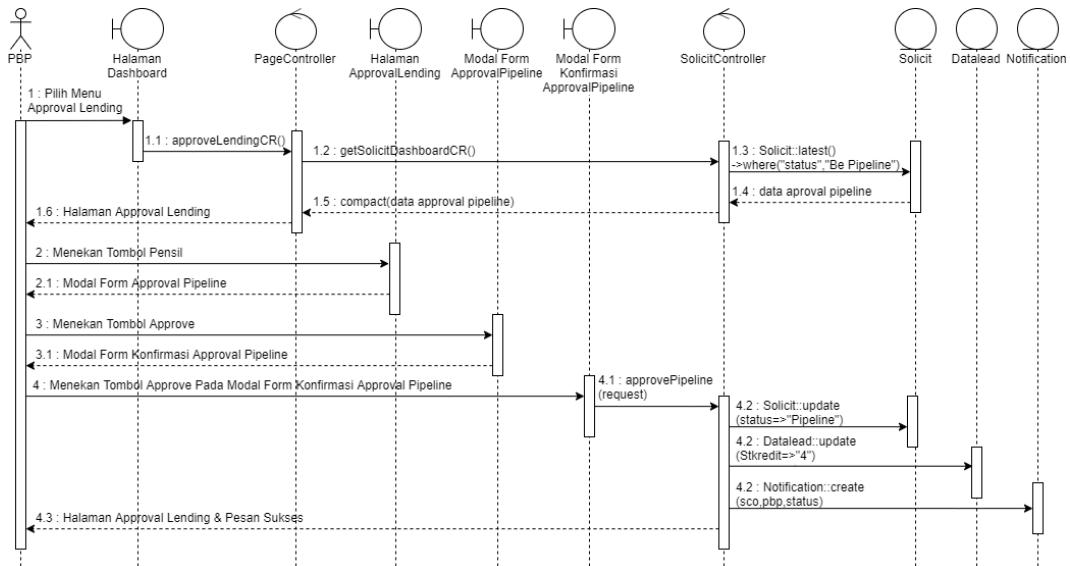
Berikut adalah *Sequence Diagram Be Pipeline* oleh SCO yang dijelaskan pada gambar 6.7. SCO memasuki halaman All Prospect kemudian memilih data prospect yang diinginkan, lalu melengkapi data debitur dan menekan tombol *submit* untuk melanjutkan proses persetujuan Pipeline menuju PBP. *Submit* akan melakukan *update* data pada *database JKredit* dan *update* data pada *database Data Lead* dengan mengubah "StKredit" menjadi "4".



Gambar 6.7 Sequence Diagram Be Pipeline

6.2.7 Sequence Diagram Approval Pipeline

Berikut adalah *Sequence Diagram Approval Pipeline* oleh PBP yang dijelaskan pada gambar 6.8. PBP memasuki halaman *Approval PBP* kemudian memilih data *pending approval* Pipeline yang diinginkan, lalu menekan tombol “Approve”, “Re-Check” atau “Reject” pada data tersebut. Jika menekan tombol “Approve”, akan dilakukan update data pada *database JKredit* dan update data pada *database Data Lead* dengan mengubah “StKredit” menjadi “5”. Jika menekan tombol “Re-Check”, maka data pada *database Data Lead* akan di-*update* dengan mengembalikan “StKredit” menjadi “1” dengan tambahan “remark”. Sementara tombol “Reject” akan melakukan *insert* pada *database Cdatalead* dengan *note*.



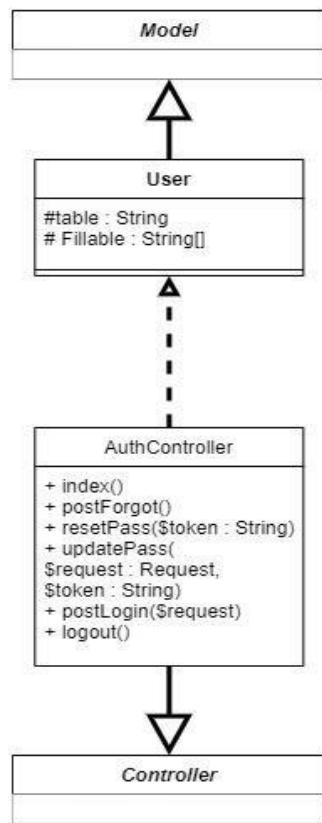
Gambar 6.8 Sequence Diagram Approval Pipeline

6.3. Pemodelan *Class Diagram*

Class diagram merupakan struktur sistem yang digambarkan melalui kelas-kelas yang mempunyai atribut, *method*, dan relasi antar kelas. Kelas-kelas yang dirancang nantinya akan diimplementasikan ke *source code* pada *Framework Laravel*. *Class Diagram* yang dibuat nantinya memperlihatkan relasi antara *Controller* dan *Model* pada *Framework Laravel*. *Class Diagram* dirancang menjadi dua jenis berdasarkan kelas abstrak yang di-*extend* yaitu kelas *Controller* dan kelas *Model* yang kelas turunannya saling berelasi.

6.3.1. Pemodelan *Class Diagram* untuk *Auth Gateway*

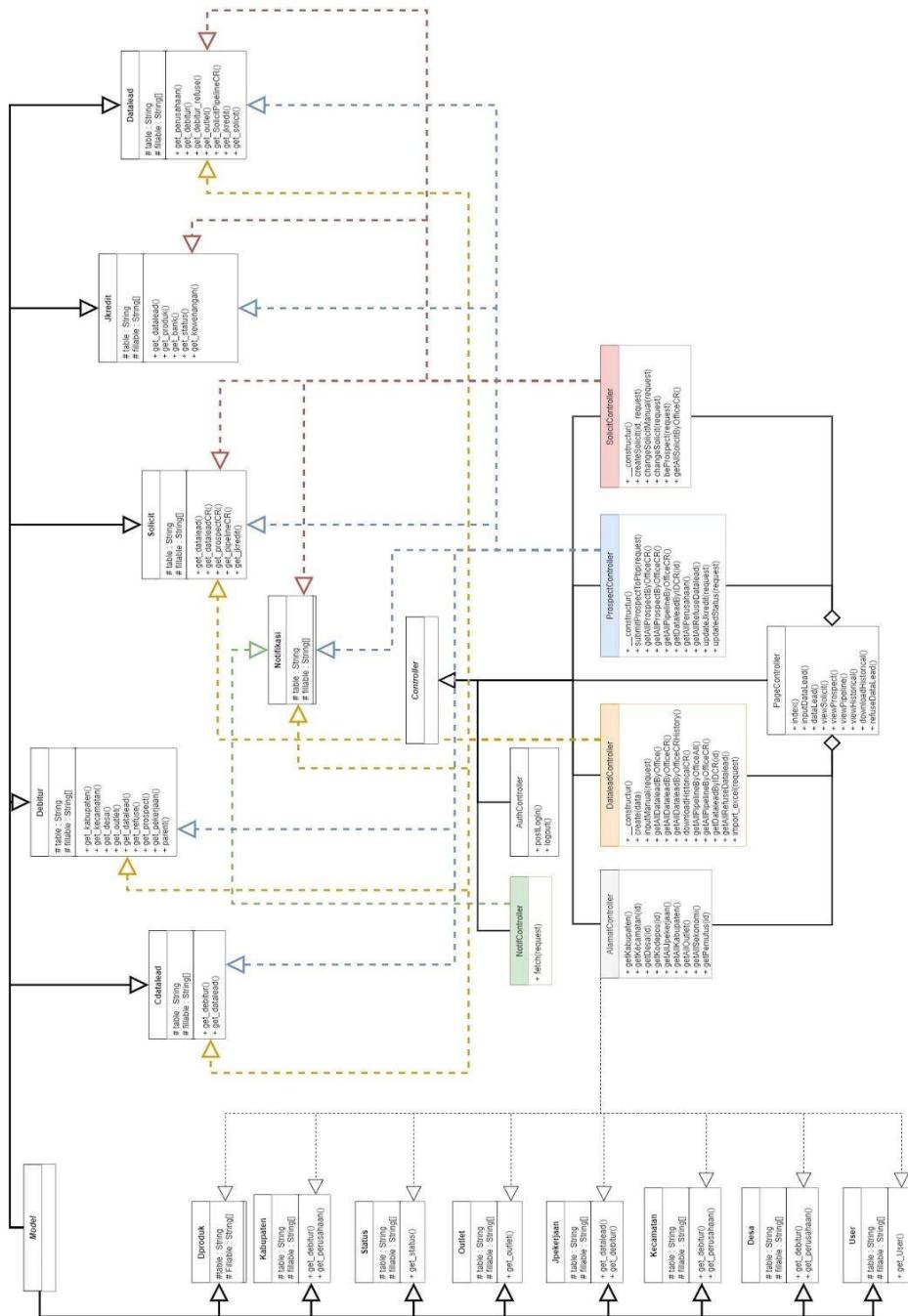
Pada gambar 6.9, terdapat 2 kelas yang dibuat yaitu *AuthController* yang merupakan turunan dari kelas *Controller* dan *User* yang merupakan turunan dari kelas *Model*.



Gambar 6.9 Perancangan Class Diagram Auth Gateway

6.3.2. Pemodelan Class Diagram untuk Sales Services

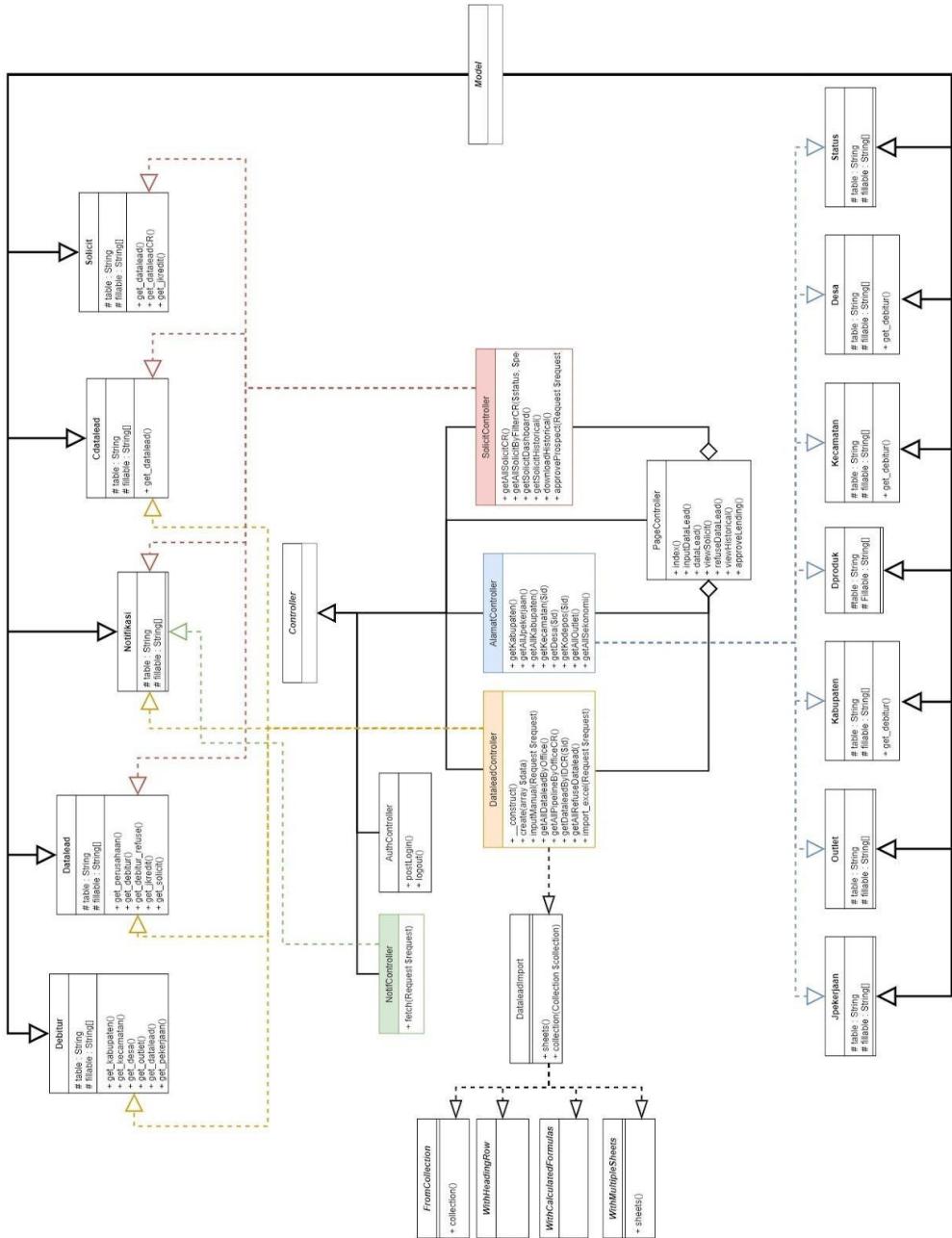
Pada gambar 6.10, terdapat 7 kelas turunan dari kelas *Controller*, yaitu NotifController, AuthController, AlamatController, DataLeadController, ProspectController, SolicitController, dan PageController. Ada 14 kelas yang merupakan turunan dari kelas *Model*, yaitu DataLead, Jkredit, Solicit, Notifikasi, Debitur, Cdatalead, Dproduk, Kabupaten, Status, Outlet, Jpekerjaan, Kecamatan, Desa, dan User.



Gambar 6.10 Perancangan *Class Diagram Sales Services*

6.3.3. Pemodelan *Class Diagram* untuk Penyelia Services

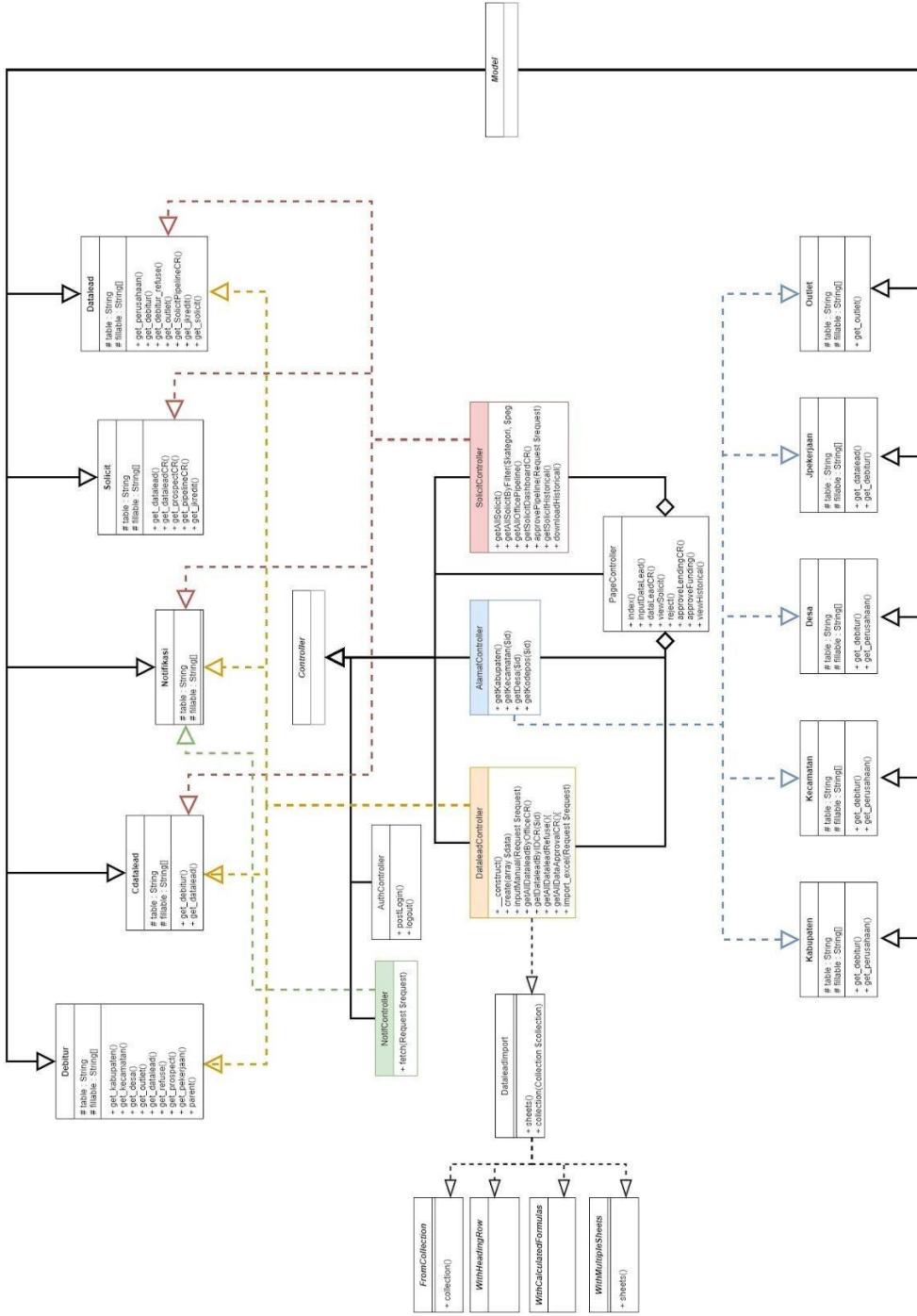
Pada gambar 6.11, terdapat 6 kelas turunan dari kelas *Controller*, yaitu NotifController, AuthController, AlamatController, DataLeadController, SolicitudController, dan PageController. Ada 12 kelas yang merupakan turunan dari kelas *Model*, yaitu Debitur, DataLead, Notifikasi, Cdatalead, Solicitud, Status, Desa, Kecamatan, Dproduk, Kabupaten, Outlet, Jpekerjaan.



Gambar 6.11 Perancangan *Class Diagram* Penyelia Services

6.3.4. Pemodelan *Class Diagram* untuk PBP Services

Pada gambar 6.12, terdapat 6 kelas turunan dari kelas *Controller*, yaitu NotifController, AuthController, AlamatController, DataLeadController, SollicitController, dan PageController. Ada 10 kelas yang merupakan turunan dari kelas *Model*, yaitu Debitur, DataLead, Notifikasi, Cdatalead, Solicit, Kabupaten, Kecamatan, Desa, Jpekerjaan, dan Outlet.



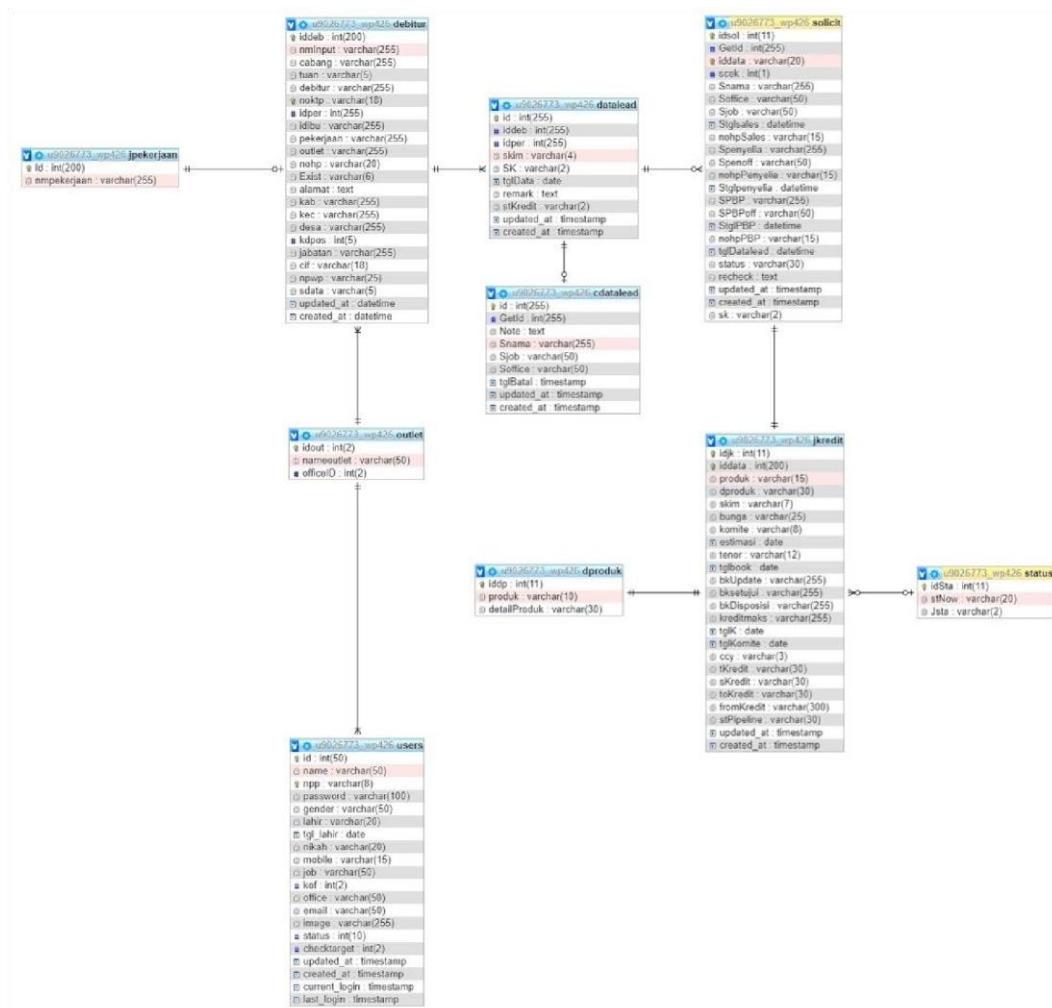
Gambar 6.12 Perancangan Class Diagram PBP Services

6.4. Pemodelan PDM (Physical Data Model)

PDM atau *Physical Data Model* adalah *diagram* yang memodelkan hubungan atau relasi antar objek data sebagai rancangan dalam membangun sebuah *database*. Ada beberapa perancangan *Physical Data Model* sesuai objek yang dibutuhkan pada Sistem Informasi *Lending* BROMO, yaitu *Physical Data Model* untuk proses *Lending*, lokasi, dan notifikasi.

6.4.1. Physical Data Model Proses Lending

Berikut adalah *Physical Data Model* proses *Lending* yang dijelaskan pada gambar 6.13. *Physical Data Model* proses *Lending* digunakan untuk membuat informasi debitur yang berkaitan dengan proses *lending* seperti status data lead, status solicit, status pipeline, dll. *Physical Data Model* proses *Lending* terdiri atas 10 tabel, yakni tabel Debitur, tabel DataLead, tabel Cdatalead, tabel Solicit, tabel Jkredit, tabel Dproduk, tabel User, tabel Status, tabel Outlet, dan tabel Jpekerjaan.



Gambar 6.13 PDM Proses Lending

6.4.1.1. Perancangan Tabel Debitur

Tabel Debitur digunakan untuk menyimpan data pribadi nasabah yang melakukan peminjaman (*lending*). Dalam tabel Debitur terdapat beberapa atribut diantaranya :

- iddeb sebagai *Primary Key* dengan tipe data *Integer*,

- nmInput dengan tipe data *Varchar*,
- cabang dengan tipe data *Varchar*,
- tuan dengan tipe data *Varchar*,
- Debitur dengan tipe data *Varchar*,
- noktp dengan tipe data *Varchar*,
- idper dengan tipe data *Integer*,
- idibu dengan tipe data *Varchar*,
- pekerjaan dengan tipe data *Varchar*,
- outlet dengan tipe data *Varchar*,
- nohp dengan tipe data *Varchar*,
- Exist dengan tipe data *Varchar*,
- alamat dengan tipe data *text*,
- kab dengan tipe data *Varchar*,
- kec dengan tipe data *Varchar*,
- desa dengan tipe data *Varchar*,
- Kdpos dengan tipe data *Integer*,
- jabatan dengan tipe data *Varchar*,
- cif dengan tipe data *Varchar*,
- npwp dengan tipe data *Varchar*,
- sdata dengan tipe data *Varchar*,
- Updated_at dengan tipe data *datetime*,
- Created_at dengan tipe data *datetime*

6.4.1.2. Perancangan Tabel Data Lead

Tabel Data Lead digunakan untuk menyimpan data inisiasi mengenai status nasabah yang digunakan untuk proses *lending*. Dalam tabel Data Lead terdapat beberapa atribut diantaranya :

- id sebagai *Primary Key* dengan tipe data *Integer*,
- iddeb dengan tipe data *Integer*,
- idper dengan tipe data *Integer*,
- skim dengan tipe data *Varchar*,
- SK dengan tipe data *Varchar*,
- tglData dengan tipe data *date*,
- remark dengan tipe data *text*,
- stKredit dengan tipe data *Varchar*,
- updated_at dengan tipe data *datetime*,
- created_at dengan tipe data *datetime*

6.4.1.3. Perancangan Tabel Cdatalead

Tabel Cdatalead digunakan untuk menyimpan data nasabah yang tidak diterima permohonan pinjamannya. Dalam tabel Cdatalead terdapat beberapa atribut diantaranya :

- id sebagai *Primary Key* dengan tipe data *Integer*,
- GetId dengan tipe data *Integer*,
- Note dengan tipe data *text*,
- Snama dengan tipe data *Varchar*,
- Sjob dengan tipe data *Varchar*,
- Soffice dengan tipe data *Varchar*,
- tglBatal dengan tipe data *datetime*,
- updated_at dengan tipe data *datetime*,
- created_at dengan tipe data *datetime*

6.4.1.4. Perancangan Tabel Solicit

Tabel *Solicit* digunakan untuk menyimpan data mengenai kapan dan siapa pengguna yang melakukan proses *Solicit* hingga *Pipeline* pada nasabah. Dalam tabel *Solicit* terdapat beberapa atribut diantaranya :

- idsol sebagai *Primary Key* dengan tipe data *Integer*,
- GetId dengan tipe data *Integer*,
- iddata dengan tipe data *Varchar*,
- scek dengan tipe data *Integer*,
- Snama dengan tipe data *Varchar*,
- Soffice dengan tipe data *Varchar*,
- Sjob dengan tipe data *Varchar*,
- Stglsales dengan tipe data *datetime*,
- nohpSales dengan tipe data *Varchar*,
- Spenyelia dengan tipe data *Varchar*,
- Spenoff dengan tipe data *Varchar*,
- nohpPenyelia dengan tipe data *Varchar*,
- Stglpenyelia dengan tipe data *datetime*,
- SPBP dengan tipe data *Varchar*,
- SPBPop dengan tipe data *Varchar*,
- StglPBP dengan tipe data *datetime*,
- nohpPBP dengan tipe data *Varchar*,
- tglData Lead dengan tipe data *datetime*,
- status dengan tipe data *Varchar*,
- recheck dengan tipe data *text*,
- updated_at dengan tipe data *datetime*,
- created_at dengan tipe data *datetime*,
- sk dengan tipe data *Varchar*

6.4.1.5. Perancangan Tabel Jkredit

Tabel *Jkredit* digunakan untuk menyimpan data tambahan nasabah yang berhasil melewati fase *Prospect* dan *Pipeline*. Dalam tabel *Jkredit* terdapat beberapa atribut diantaranya :

- idjk sebagai *Primary Key* dengan tipe data *Integer*,
- iddata dengan tipe data *Integer*,
- produk dengan tipe data *Varchar*,
- dproduk dengan tipe data *Varchar*,
- skim dengan tipe data *Varchar*,
- bunga dengan tipe data *Varchar*,
- komite dengan tipe data *Varchar*,
- estimasi dengan tipe data *datetime*,
- tenor dengan tipe data *Varchar*,
- tglbook dengan tipe data *datetime*,
- bkUpdate dengan tipe data *Varchar*,
- bksetujui dengan tipe data *Varchar*,
- bkDisposisi dengan tipe data *Varchar*,
- kreditmaks dengan tipe data *Varchar*,
- tglK dengan tipe data *datetime*,
- tglKomite dengan tipe data *datetime*,
- ccy dengan tipe data *Varchar*,
- tKredit dengan tipe data *Varchar*,
- sKredit dengan tipe data *Varchar*,
- toKredit dengan tipe data *Varchar*,
- fromKredit dengan tipe data *Varchar*,
- stPipeline dengan tipe data *Varchar*,
- updated_at dengan tipe data *datetime*,
- created_at dengan tipe data *datetime*

6.4.1.6. Perancangan Tabel Dproduk

Tabel Dproduk digunakan untuk menyimpan produk-produk peminjaman yang ditawarkan oleh pihak BNI. Dalam tabel Dproduk terdapat beberapa atribut diantaranya :

- iddp sebagai *Primary Key* dengan tipe data *Integer*,
- produk dengan tipe data *Varchar*,
- detailProduk dengan tipe data *Varchar*

6.4.1.7. Perancangan Tabel User

Tabel User digunakan untuk menyimpan data pengguna Sistem Informasi *Lending* BROMO. Dalam tabel User terdapat beberapa atribut diantaranya :

- id sebagai *Primary Key* dengan tipe data *Integer*,
- name dengan tipe data *Varchar*
- npp dengan tipe data *Varchar*
- password dengan tipe data *Varchar*
- gender dengan tipe data *Varchar*
- lahir dengan tipe data *Varchar*

- tgl_lahir dengan tipe data *date*,
- nikah dengan tipe data *Varchar*
- mobile dengan tipe data *Varchar*
- job dengan tipe data *Varchar*
- kof dengan tipe data *Integer*,
- office dengan tipe data *Varchar*
- email dengan tipe data *Varchar*
- image dengan tipe data *Varchar*
- status dengan tipe data *Integer*,
- checktarget dengan tipe data *Integer*,
- updated_at dengan tipe data *datetime*,
- created_at dengan tipe data *datetime*,
- Current_login dengan tipe data *datetime*,
- last_login dengan tipe data *datetime*

6.4.1.8. Perancangan Tabel Status

Tabel User digunakan untuk menyimpan data pengguna Sistem Informasi *Lending BROMO*. Dalam tabel Status terdapat beberapa atribut diantaranya :

- idSta sebagai *Primary Key* dengan tipe data *Integer*,
- stNow dengan tipe data *Varchar*,
- Jsta dengan tipe data *Varchar*,

6.4.1.9. Perancangan Tabel Outlet

Tabel User digunakan untuk menyimpan data pengguna Sistem Informasi *Lending BROMO*. Dalam tabel Outlet terdapat beberapa atribut diantaranya :

- idout sebagai *Primary Key* dengan tipe data *Integer*,
- nameoutlet dengan tipe data *Varchar*,
- officelD dengan tipe data *Integer*

6.4.1.10. Perancangan Tabel Jpekerjaan

Tabel jpekerjaan digunakan untuk menyimpan data tambahan nasabah berupa jenis pekerjaan dari nasabah. Dalam tabel Debitur terdapat beberapa atribut diantaranya :

- id sebagai *Primary Key* dengan tipe data *Integer*,
- nmpekerjaan dengan tipe data *Varchar*,

6.4.2 Physical Data Model Lokasi

Berikut adalah *Physical Data Model* Lokasi yang dijelaskan pada gambar 6.14. *Physical Data Model* Lokasi digunakan untuk membuat informasi debitur yang berkaitan dengan lokasi seperti tempat tinggal, tempat lahir, tempat tinggal ibu, tempat lahir ibu, dll. *Physical Data Model* Lokasi terdiri atas 3 tabel, yakni tabel Kabupaten, tabel Kecamatan, dan tabel Desa.



Gambar 6.14 PDM Lokasi

6.4.2.1. Perancangan Tabel Kabupaten

Tabel Kabupaten digunakan untuk menyimpan data wilayah kabupaten yang menjadi tanggung jawab pihak BNI Kantor Wilayah Malang. Dalam tabel Kabupaten terdapat beberapa atribut diantaranya :

- idkab sebagai *Primary Key* dengan tipe data *Integer*,
- kab dengan tipe data *Varchar*

6.4.2.2. Perancangan Tabel Kecamatan

Tabel Kecamatan digunakan untuk menyimpan data Kecamatan yang menjadi tanggung jawab pihak BNI Kantor Wilayah Malang. Dalam tabel Kecamatan terdapat beberapa atribut diantaranya :

- idkec sebagai *Primary Key* dengan tipe data *Integer*,
- idkab dengan tipe data *Integer*,
- kec dengan tipe data *Varchar*

6.4.2.3. Perancangan Tabel Desa

Tabel Desa digunakan untuk menyimpan data desa yang menjadi tanggung jawab pihak BNI Kantor Wilayah Malang. Dalam tabel Desa terdapat beberapa atribut diantaranya :

- iddesa sebagai *Primary Key* dengan tipe data *Integer*,
- idkec dengan tipe data *Integer*,
- desa dengan tipe data *Varchar*,
- knpos dengan tipe data *Integer*

6.4.3. Physical Data Model Notifikasi

Berikut adalah *Physical Data Model* Notifikasi yang dijelaskan pada gambar 6.15. *Physical Data Model* Notifikasi digunakan untuk membuat notifikasi yang dapat diterima oleh *user* dalam proses *lending*. *Physical Data Model* Notifikasi terdiri atas tabel *notification*.

The screenshot shows the 'notification' table structure in MySQL Workbench. The table has 13 columns:

	id : int(11)	notiuser : varchar(50)	notireciver : varchar(50)	notitype : varchar(50)	time : timestamp	ddata : varchar(200)	status : int(11)	iddata : int(255)	idper : int(255)	detailStatus : text	click : int(1)	created_at : timestamp	updated_at : timestamp
--	--------------	------------------------	---------------------------	------------------------	------------------	----------------------	------------------	-------------------	------------------	---------------------	----------------	------------------------	------------------------

Gambar 6.15 PDM Notifikasi

6.4.3.1. Perancangan Tabel Notification

Tabel *notification* digunakan untuk menyimpan data pertukaran informasi seperti pengajuan persetujuan proses maupun hasil persetujuan proses ke pengguna tertentu. Dalam tabel *notification* terdapat beberapa atribut diantaranya :

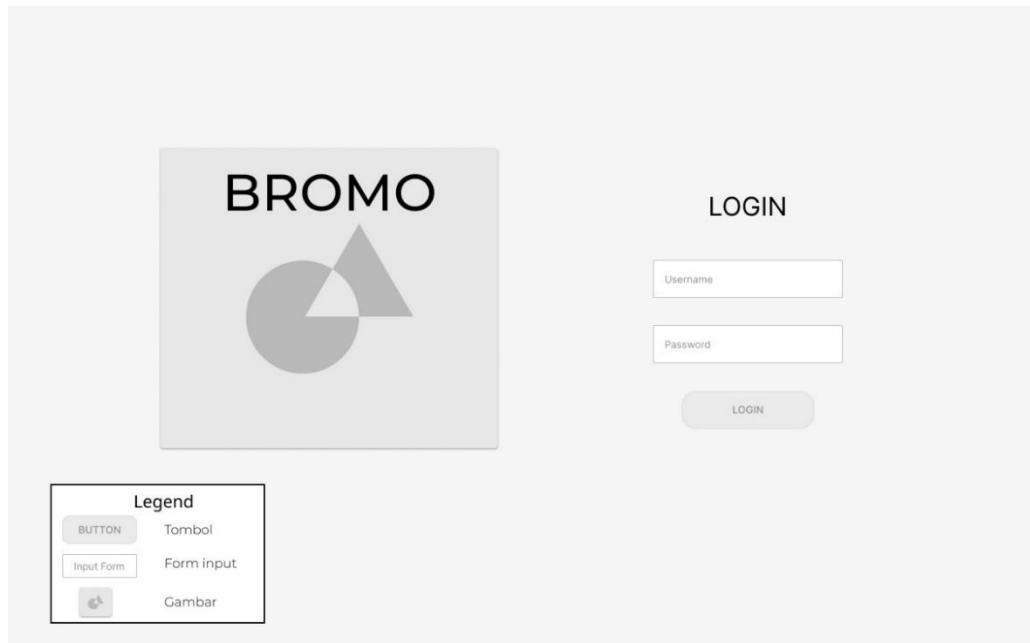
- id sebagai *Primary Key* dengan tipe data *Integer*,
- notiuser dengan tipe data *Varchar*,
- notireciver dengan tipe data *Varchar*,
- notitype dengan tipe data *Varchar*,
- time dengan tipe data *datetime*,
- ddata dengan tipe data *Varchar*,
- status dengan tipe data *Integer*,
- iddata dengan tipe data *Integer*,
- idper dengan tipe data *Integer*,
- detailStatus dengan tipe data *text*,
- click dengan tipe data *Integer*,
- created_at dengan tipe data *datetime*,
- updated_at dengan tipe data *datetime*

6.5. Perancangan User Interface

User Interface merupakan tampilan yang berhubungan langsung dengan pengguna untuk melakukan interaksi dengan sistem. Perancangan *User Interface* digunakan sebagai acuan untuk membuat tampilan dari Sistem Informasi *Lending BROMO*.

6.5.1. User Interface Login

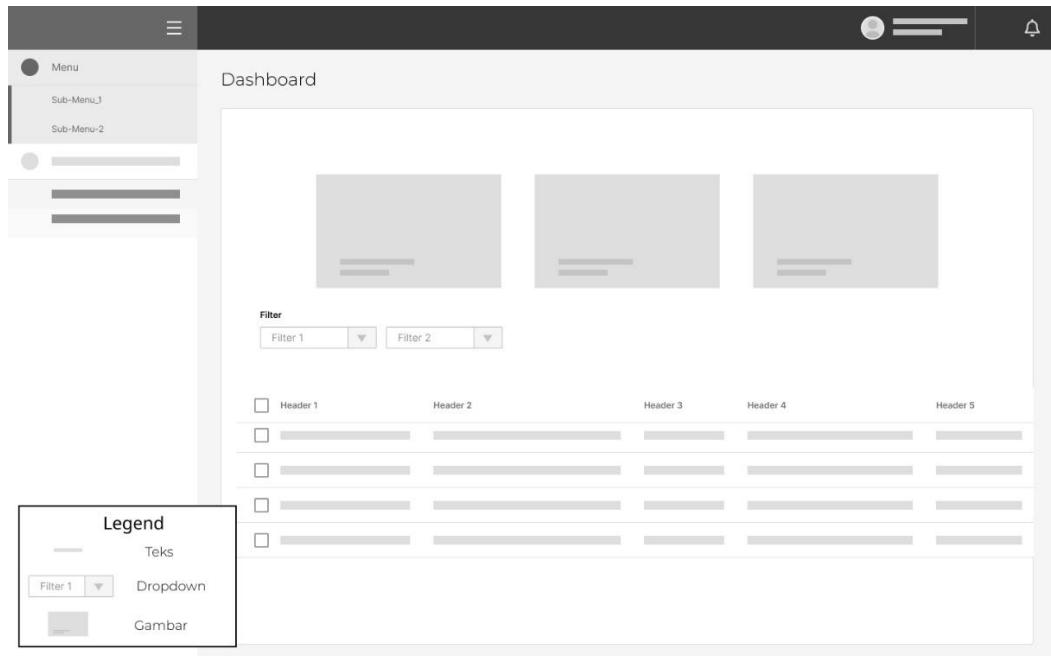
Gambar 6.16 menunjukkan rancangan dari *user interface* halaman Login. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman Login.



Gambar 6.16 Perancangan *User Interface Login*

6.5.2. User Interface Dashboard

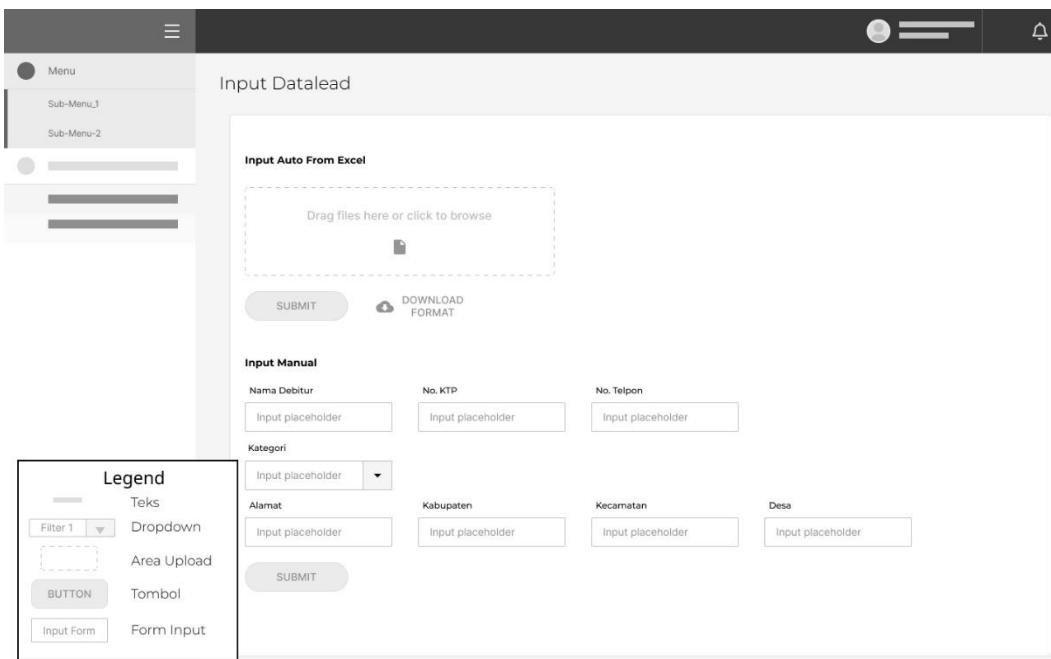
Gambar 6.17 menunjukkan rancangan dari *user interface* halaman Dashboard. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman Dashboard, Data Lead, Solicit, Prospect, dan Pipeline.



Gambar 6.17 Perancangan *User Interface* Dashboard

6.5.3. *User Interface* Input Data Lead

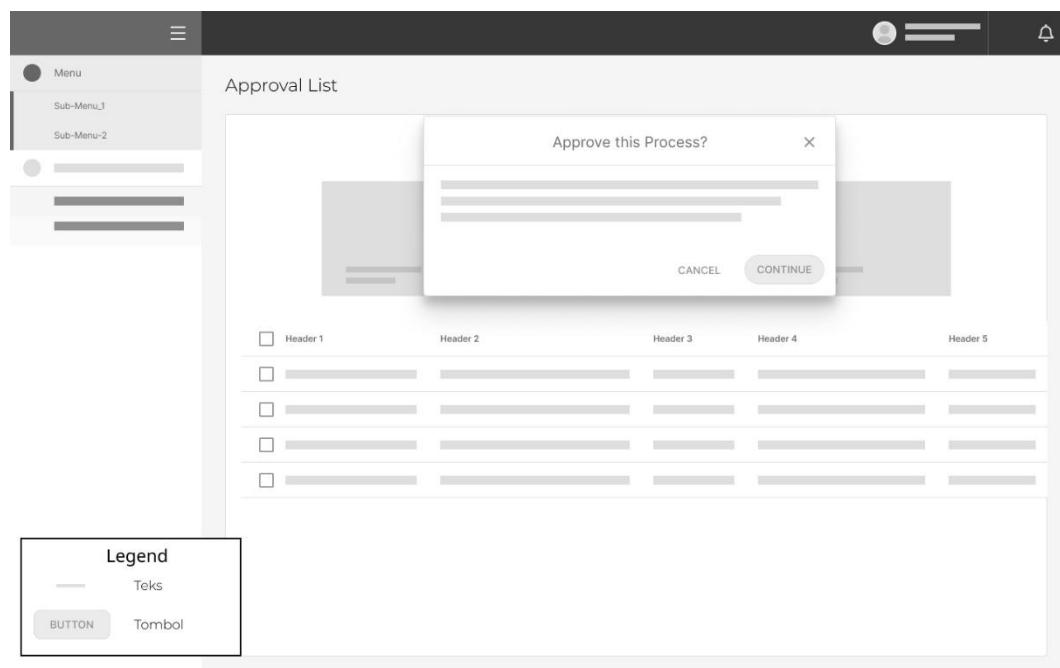
Gambar 6.18 menunjukkan rancangan dari *user interface* halaman Input Data Lead. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman form Input Data Lead, form Prospect, dan form Pipeline.



Gambar 6.18 Perancangan *User Interface* Data Lead

6.5.4. User Interface Approval

Gambar 6.19 menunjukkan rancangan dari *user interface* halaman Approval. Rancangan *user interface* ini akan menjadi acuan dalam mengimplementasikan *user interface* halaman Approval Prospect dan Approval Pipeline.



Gambar 6.19 Perancangan *User Interface Approval*

BAB 7 IMPLEMENTASI

Pada bab ini akan dilakukan proses pengembangan sistem dengan menerapkan rancangan sistem. Implementasi pada Sistem Informasi *Lending* BROMO meliputi :

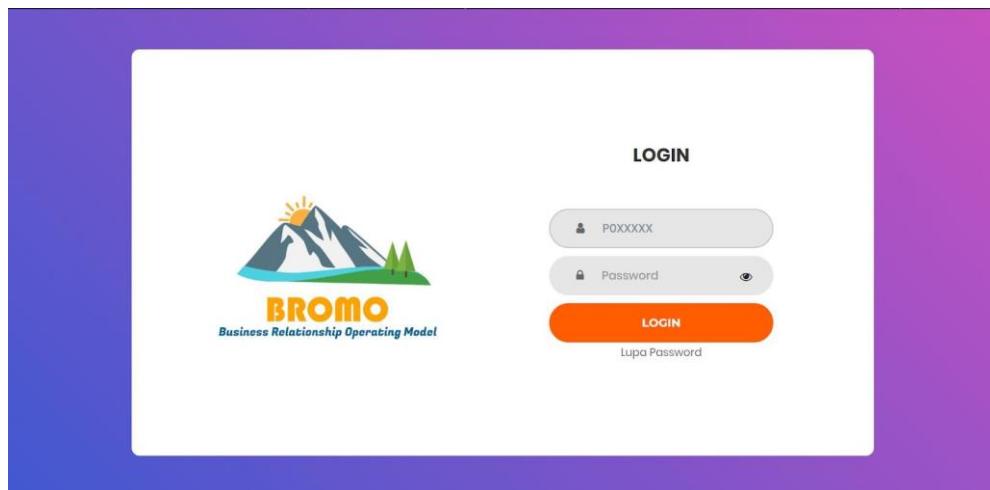
1. Implementasi *user interface*
2. Implementasi *database*
3. Implementasi kode program

7.1 Implementasi *User Interface*

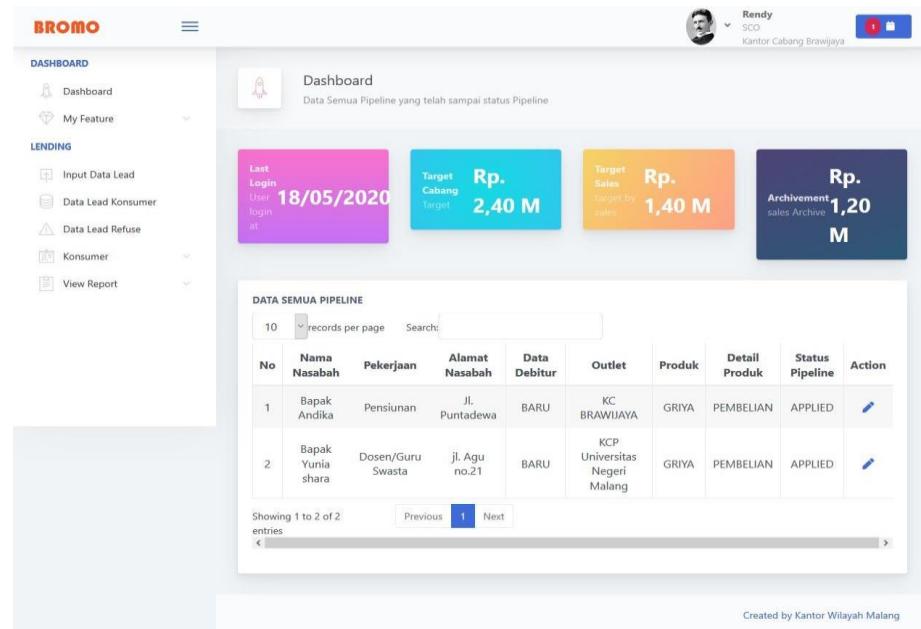
Subbab ini memaparkan hasil implementasi dari *user interface* yang mengacu pada perancangan sebelumnya.

7.1.1 *User Interface Login*

Gambar 7.1 menunjukkan implementasi dari *user interface* halaman Login yang mengacu pada perancangan *user interface* halaman Login. Kemudian pada Gambar 7.2 menunjukkan implementasi dari *user interface* halaman Dashboard yang mengacu pada perancangan *user interface* halaman Dashboard



Gambar 7.1 Halaman Login



Gambar 7.2 Dashboard

7.1.2 *User Interface Input Data Lead*

Gambar 7.3 menunjukkan implementasi dari *user interface* halaman Input Data Lead pada aktor SCO. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman Data Lead.

The screenshot shows the 'Input Data Lead' form. At the top right, there is a user profile for 'Rendy' from 'SCO Kantor Cabang Brawijaya'. The form has several sections:

- PILIH KATEGORI:** Fields for 'Kategori *' (CR) and 'Kantor *' (Pilih Outlet).
- INPUT MANUAL DEBITUR BARU:** Fields for 'Badan Usaha *' (Perseorangan), 'Nama Usaha *' (Nama Usaha), 'Nama Lengkap Nasabah *' (Bapak), and 'Nama Lengkap' (Nama Lengkap).
- Jenis Pekerjaan *:** A dropdown menu for 'Pilih Jenis Pekerjaan'.
- Alamat Nasabah *:** A dropdown menu for 'Alamat Nasabah Jalan No.xx'.
- No KTP *:** A dropdown menu for 'Nomor KTP'.
- Kabupaten / Kota *:** A dropdown menu for 'Pilih Kabupaten / Kota'.
- Kecamatan *:** A dropdown menu for 'Pilih Kecamatan'.
- Desa *:** A dropdown menu for 'Pilih Desa'.
- Kodepos *:** A dropdown menu for 'Pilih Kodepos'.
- No Telp *:** A dropdown menu for '+62 Nomer Telp'.

At the bottom left is a 'Submit' button, and at the bottom right, it says 'Created by Kantor Wilayah Malang'.

Gambar 7.3 Input Data Lead

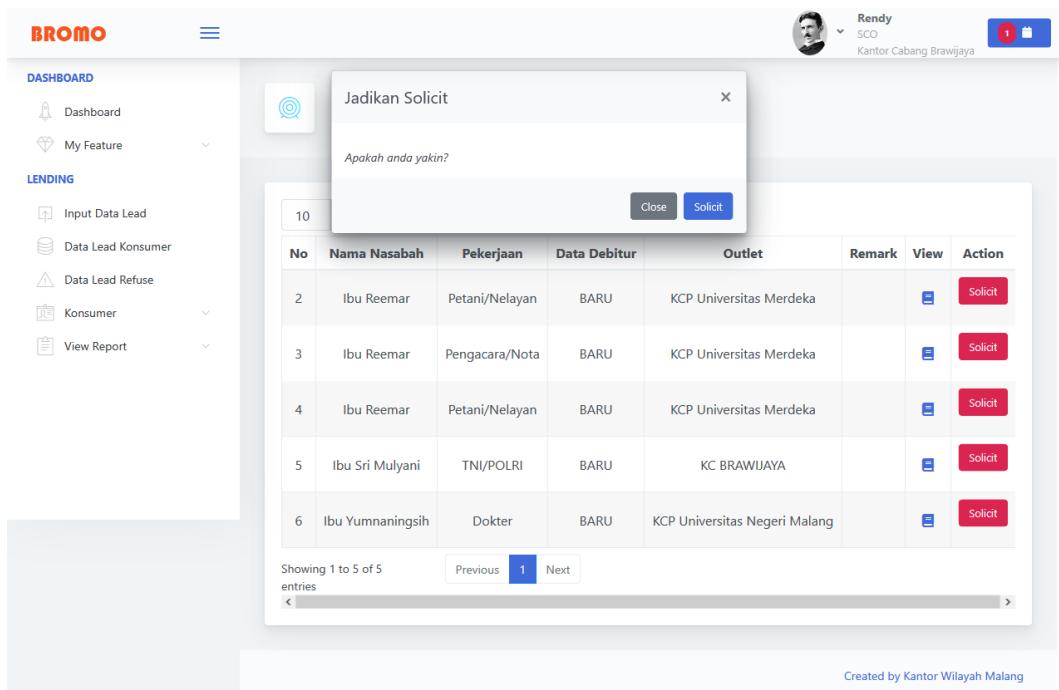
7.1.3 User Interface List Data Lead

Gambar 7.4 menunjukkan implementasi dari *user interface* halaman List Data Lead pada aktor SCO. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman Dashboard. Kemudian pada Gambar 7.5 menunjukkan *pop up* yang akan ditampilkan apabila aktor menekan tombol *Solicit*.

The screenshot shows the BROMO application interface. At the top, there is a header with the logo 'BROMO' and a user profile for 'Rendy SCO Kantor Cabang Brawijaya'. Below the header is a navigation bar with 'DASHBOARD' and 'LENDING' sections. The 'LENDING' section contains links for 'Input Data Lead', 'Data Lead Konsumen', 'Data Lead Refuse', 'Konsumen', and 'View Report'. The main content area is titled 'List Data Lead Konsumen' with a sub-instruction 'deskripsi.'. It features a table with columns: No, Nama Nasabah, Pekerjaan, Data Debitur, Outlet, Remark, View, and Action. The table contains six entries. Each entry has a 'View' button and a red 'Solicit' button. At the bottom of the table, there is a pagination bar showing 'Showing 1 to 5 of 5 entries' and buttons for 'Previous', 'Next', and a page number '1'. A footer at the bottom right says 'Created by Kantor Wilayah Malang'.

No	Nama Nasabah	Pekerjaan	Data Debitur	Outlet	Remark	View	Action
2	Ibu Reemar	Petani/Nelayan	BARU	KCP Universitas Merdeka		View	Solicit
3	Ibu Reemar	Pengacara/Nota	BARU	KCP Universitas Merdeka		View	Solicit
4	Ibu Reemar	Petani/Nelayan	BARU	KCP Universitas Merdeka		View	Solicit
5	Ibu Sri Mulyani	TNI/POLRI	BARU	KC BRAWIJAYA		View	Solicit
6	Ibu Yumananingsih	Dokter	BARU	KCP Universitas Negeri Malang		View	Solicit

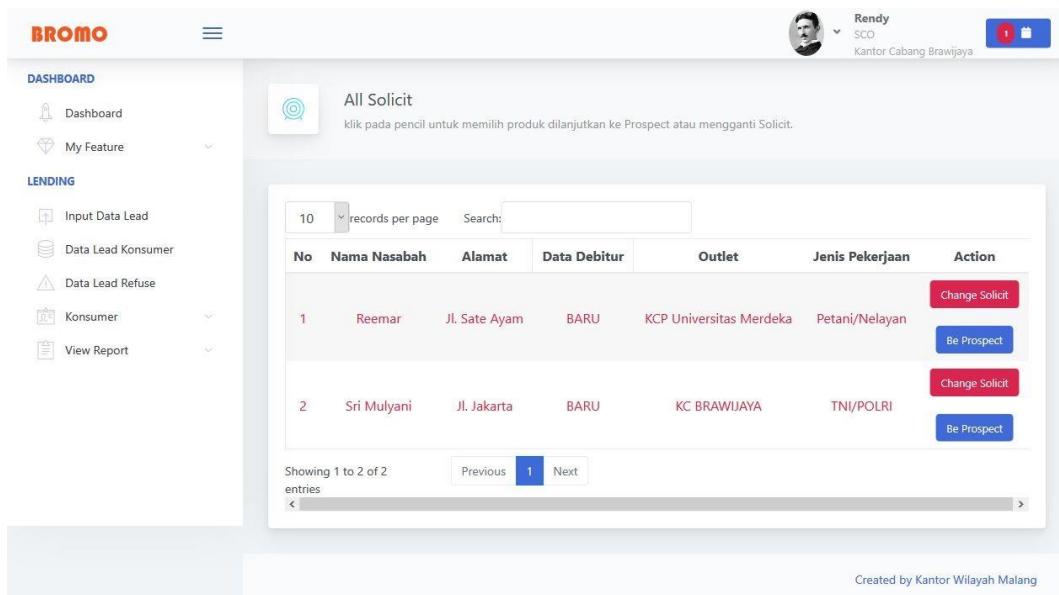
Gambar 7.4 List Data Lead



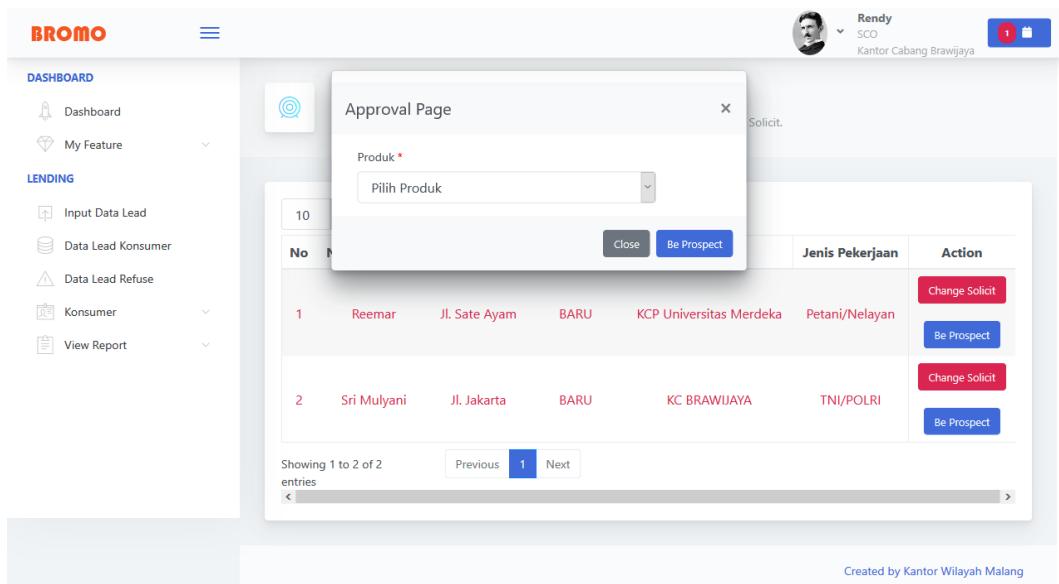
Gambar 7.5 *Pop Up* Ketika Menekan Tombol Solicitud

7.1.4 User Interface Solicitud

Gambar 7.6 menunjukkan implementasi dari *user interface* Solicitud pada aktor SCO. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman Dashboard. Pada Gambar 7.7 menunjukkan *pop up* yang akan ditampilkan apabila aktor menekan tombol Be Prospect.



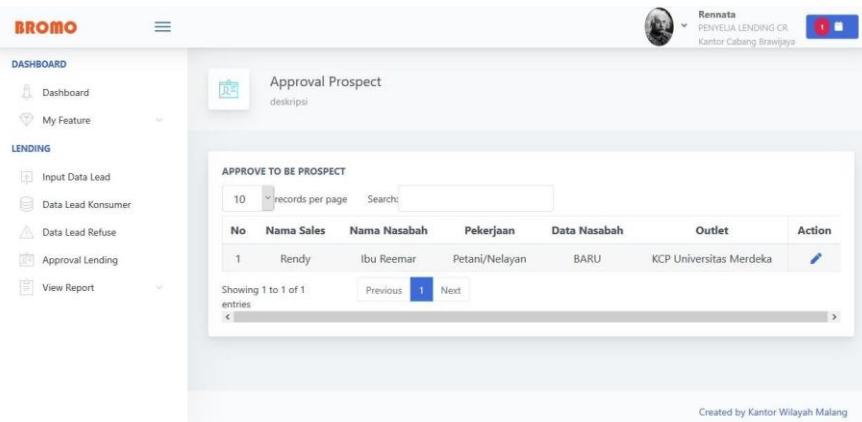
Gambar 7.6 All Solicit



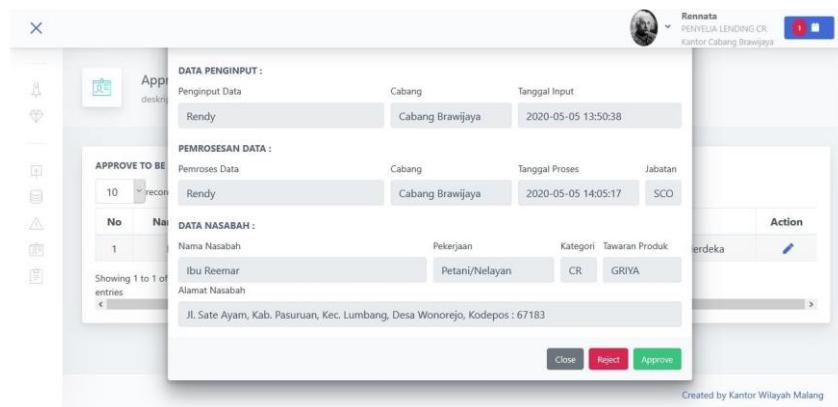
Gambar 7.7 Pop Up Ketika Menekan Button Be Prospect

7.1.5 User Interface Approval Prospect

Gambar 7.8 menunjukkan implementasi dari *user interface* Approval Prospect pada aktor Penyelia. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman Approval. Pada Gambar 7.9 menunjukkan *pop up* yang akan ditampilkan apabila aktor menekan tombol Action yang berbentuk pensil.



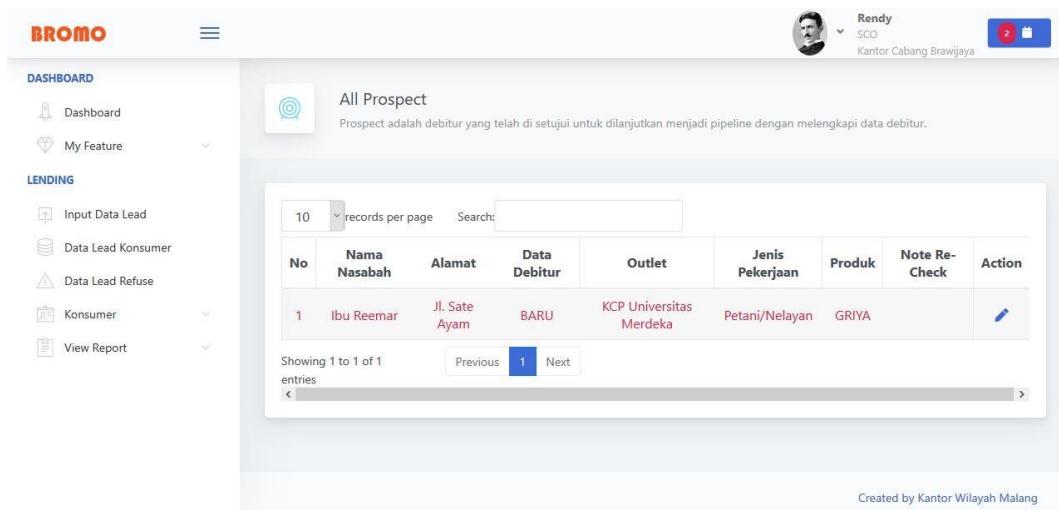
Gambar 7.8 Approval Prospect



Gambar 7.9 Pop Up Pada Halaman Approval Prospect

7.1.6 User Interface Prospect

Gambar 7.10 menunjukkan implementasi dari *user interface* Prospect pada aktor SCO. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman Dashboard. Pada Gambar 7.11 menunjukkan *pop up* yang akan ditampilkan apabila aktor menekan tombol Action yang berbentuk pensil.



Gambar 7.10 Prospect

Details for : Reemar (BARU)

DATA PENGINPUT :		Cabang	Outlet	Tanggal Data
Penginput Data	Rendy	Cabang Brawijaya	KCP Universitas Merdeka	05-05-2020
DATA NASABAH :				
Nama Nasabah	CR	Kategori Produk	Detail Produk	
Ibu Reemar	GRIYA	Pilih Detail Produk		
Alamat Nasabah	Jl. Sate Ayam, Kab. Pasuruan, Kec.Lumbang, Desa Wonorejo, Kodepos : 67183			
Pekerjaan	KTP	NPWP		
Petani/Nelayan	23123123123			
DATA IBU NASABAH :				
Tuan	Nama Lengkap Ibu			
Ibu				
Alamat Nasabah *	Jenis Pekerjaan *		No KTP	
Alamat Nasabah Jalan No.xx	Pilih Jenis Pekerjaan		Nomor KTP	
Kabupaten	Kecamatan *	Desa *	Kodepos *	No Telp *
Pilih Kabupaten	Pilih Kecamatan	Pilih Desa	Pilih Kodepos	+62 Nomer Telp
<input type="button" value="Back"/> <input type="button" value="Submit"/>				

Gambar 7.11 Pop Up Pada Halaman Prospect

7.1.7 User Interface Approval Pipeline

Gambar 7.12 menunjukkan implementasi dari *user interface* Approval Pipeline pada aktor PBP. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman Approval. Pada Gambar 7.13 menunjukkan *pop up* yang akan ditampilkan apabila aktor menekan tombol Action yang berbentuk pensil.

The screenshot shows the BROMO application interface. At the top, there is a header with the logo 'BROMO' and user information: 'IBNU ALFIAN', 'PBP', and 'Kantor Cabang Brawijaya'. Below the header, the left sidebar has sections for 'DASHBOARD' (Dashboard, My Feature) and 'LENDING' (Input Data Lead, Data Lead Produktif, Data Lead Konsumen, Data Lead Refuse, Approval Lending CR, Approval Lending BB, View Report). The main content area is titled 'Approval Pipeline' and contains a sub-section 'APPROVE TO BE PIPELINE'. It features a table with one entry:

No	Nama Sales	Nama Nasabah	Pekerjaan	Data Nasabah	Outlet	Action
1	Rendy	Ibu Reemar	Petani/Nelayan	BARU	KCP Universitas Merdeka	

Below the table, it says 'Showing 1 to 1 of 1 entries' and has navigation buttons for 'Previous', '1', and 'Next'. At the bottom right of the main content area, it says 'Created by Kantor Wilayah Malang'.

Gambar 7.12 Approval Pipeline

Form Approval Debitur Rendy (BE PIPELINE)

DATA PENGINPUT :

Penginput Data	Cabang	Tanggal Input
Rendy	Cabang Brawijaya	2020-05-05 13:50:38

PEMROSESAN DATA :

Pemroses Data	Cabang	Tanggal Proses	Jabatan
Rendy	Cabang Brawijaya	2020-05-05 14:05:17	SCO

DATA NASABAH :

Nama Nasabah	Kategori Produk	Detail Produk
Ibu Reemar	CR	GRIYA BNI KOMERSIL
Alamat Nasabah	Jl. Sate Ayam, Kab. Pasuruan, Kec. Lumbang, Desa Wonorejo, Kodepos : 67183	
Pekerjaan	No. KTP	NPWP
Petani/Nelayan	23123123123	67676756545

DATA IBU NASABAH :

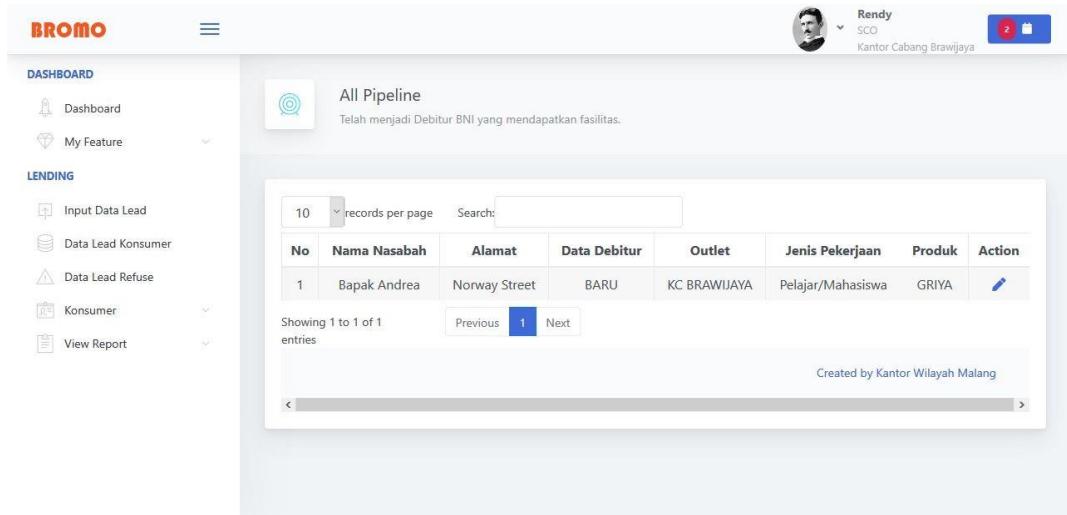
Nama Lengkap Ibu	No KTP *	
rtghhgf	234567899876	
Alamat Ibu *	dfghj. Kab. Banyuwangi, Kec. Kabat, Desa Bareng, Kodepos : 68461	
Jenis Pekerjaan *	No Telp *	Data Nasabah *
Akuntan	+62324234234	BARU

Close **Re-Checking** **Reject** **Approve**

Gambar 7.13 Pop Up Pada Halaman Approval Pipeline

7.1.8 User Interface Pipeline

Gambar 7.14 menunjukkan implementasi dari *user interface* Pipeline pada aktor SCO. Implementasi *user interface* ini mengacu pada perancangan *user interface* halaman Dashboard.



Gambar 7.14 Pipeline

7.2 Implementasi *Database*

Database Management System yang digunakan pada Sistem Informasi *Lending* BROMO adalah MySQL dan menggunakan tools PHPMYADMIN agar memudahkan proses pengembangan *database*. Gambar implementasi *database* akan ditampilkan pada Gambar 7.25.

Server: localhost:3306 » Database: u9026773_wp426

Structure SQL Search Query Export Import Operations Routines Events Trigger

Containing the word:

Table	Action	Rows	Type	Collation	Size	Overhead
admin		1	InnoDB	latin1_swedish_ci	16.0 KIB	-
agunan		0	MyISAM	utf8_general_ci	1.0 KIB	-
archive		2	MyISAM	utf8_general_ci	2.1 KIB	-
bakupdate		7	MyISAM	utf8_general_ci	2.3 KIB	-
bank		134	InnoDB	latin1_swedish_ci	16.0 KIB	-
cdatalead		3	InnoDB	latin1_swedish_ci	16.0 KIB	-
datalead		18	InnoDB	latin1_swedish_ci	16.0 KIB	-
debitur		16	MyISAM	utf8_general_ci	5.3 KIB	-
deleteduser		4	InnoDB	latin1_swedish_ci	16.0 KIB	-
desa		4,792	InnoDB	latin1_swedish_ci	240.0 KIB	-
dproduk		26	InnoDB	latin1_swedish_ci	16.0 KIB	-
feedback		0	InnoDB	latin1_swedish_ci	16.0 KIB	-
fitur		0	MyISAM	utf8_general_ci	1.0 KIB	-
funding		0	MyISAM	utf8_general_ci	1.0 KIB	-
ifunding		0	MyISAM	utf8_general_ci	1.0 KIB	-
jkredit		10	InnoDB	latin1_swedish_ci	32.0 KIB	-
Console		48	MyISAM	utf8_general_ci	2.9 KIB	-
jobs		22	MyISAM	utf8_general_ci	2.5 KIB	-
jpekerjaan		24	InnoDB	latin1_swedish_ci	16.0 KIB	-
kabupaten		30	InnoDB	latin1_swedish_ci	16.0 KIB	-
kdcabang		410	InnoDB	latin1_swedish_ci	16.0 KIB	-
kecamatan		7	MyISAM	utf8_general_ci	2.3 KIB	-
kewenangan		4	InnoDB	latin1_swedish_ci	16.0 KIB	-
ksb		29	InnoDB	latin1_swedish_ci	16.0 KIB	-
mails		49	InnoDB	latin1_swedish_ci	16.0 KIB	-
notification		16	InnoDB	latin1_swedish_ci	16.0 KIB	-
office		87	InnoDB	latin1_swedish_ci	16.0 KIB	-
outlet		1	InnoDB	latin1_swedish_ci	16.0 KIB	-
pemutus		3	MyISAM	utf8_general_ci	2.3 KIB	-
perusahaan		7,999	MyISAM	utf8_general_ci	3.6 MIB	-
saldolist		4	InnoDB	latin1_swedish_ci	16.0 KIB	-
satuhan		10	MyISAM	utf8_general_ci	2.3 KIB	-
sekonomi		13	InnoDB	latin1_swedish_ci	32.0 KIB	-
solicit		13	InnoDB	latin1_swedish_ci	16.0 KIB	-
status		16	MyISAM	utf8_general_ci	4.4 KIB	-
target		0	MyISAM	utf8_general_ci	1.0 KIB	-
tbl_form		50	InnoDB	latin1_swedish_ci	16.0 KIB	-
tbl_inventory		4	InnoDB	latin1_swedish_ci	16.0 KIB	-
tbl_invoice		0	InnoDB	latin1_swedish_ci	16.0 KIB	-
tbl_issuance		50	InnoDB	latin1_swedish_ci	32.0 KIB	-
tbl_items		1	MyISAM	utf8_general_ci	2.1 KIB	-
Tbl_Report		462	InnoDB	latin1_swedish_ci	144.0 KIB	-
users		14,359	MyISAM	utf8_general_ci	4.5 MIB	0 B
42 tables		Sum				

Gambar 7.15 Implementasi *Database* Sistem Informasi *Lending BROMO*

Database pada gambar ini adalah *database* yang telah dilakukan pengembangan lanjut oleh pihak BNI Kantor Wilayah Malang. Maka dari itu, tidak semua tabel akan dipaparkan pada laporan ini. Tabel yang akan dipaparkan pada laporan ini adalah tabel yang berhubungan dengan pengembangan yang penulis lakukan.

7.2.1 Implementasi Table Users

Tabel Users dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel Users pada Database Sistem Informasi *Lending BROMO* pada Tabel 7.1.

Tabel 7.1 Kode Program Membuat Tabel Users

users.sql	
1	CREATE TABLE `users` (
2	`id` int(50) NOT NULL,
3	`name` varchar(50) NOT NULL,
4	`npp` varchar(8) NOT NULL,
5	`password` varchar(100) NOT NULL,
6	`gender` varchar(50) DEFAULT NULL,
7	`lahir` varchar(20) DEFAULT NULL,
8	`tgl_lahir` date DEFAULT NULL,
9	`nikah` varchar(20) DEFAULT NULL,
10	`mobile` varchar(15) DEFAULT NULL,
11	`job` varchar(50) DEFAULT NULL,
12	`kof` int(2) DEFAULT NULL,
13	`office` varchar(50) DEFAULT NULL,
14	`email` varchar(50) DEFAULT NULL,
15	`image` varchar(255) DEFAULT '10.jpg',
16	`status` int(10) NOT NULL,
17	`checktarget` int(2) DEFAULT NULL,
18	`updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE current_timestamp(),
19	`created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
20	`current_login` timestamp NULL DEFAULT NULL,
21	`last_login` timestamp NULL DEFAULT NULL
22) ENGINE=InnoDB DEFAULT CHARSET=latin1;

7.2.2 Implementasi Tabel Debitur

Tabel Debitur dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel Debitur pada Database Sistem Informasi *Lending BROMO* pada Tabel 7.2.

Tabel 7.2 Kode Program Membuat Tabel Debitur

debitur.sql	
1	CREATE TABLE `debitur` (
2	`iddeb` int(200) NOT NULL,
3	`nmInput` varchar(255) NOT NULL,
4	`cabang` varchar(255) NOT NULL,
5	`tuan` varchar(5) DEFAULT NULL,
6	`debitur` varchar(255) NOT NULL,
7	`noktp` varchar(18) NOT NULL,
8	`idper` int(255) DEFAULT NULL,
9	`idibu` varchar(255) DEFAULT NULL,
10	`pekerjaan` varchar(255) DEFAULT NULL,
11	`outlet` varchar(255) DEFAULT NULL,

debitur.sql

```
12 `nohp` varchar(20) DEFAULT NULL,  
13 `Exist` varchar(6) DEFAULT NULL,  
14 `alamat` text DEFAULT NULL,  
15 `kab` varchar(255) DEFAULT NULL,  
16 `kec` varchar(255) DEFAULT NULL,  
17 `desa` varchar(255) DEFAULT NULL,  
18 `kdpos` int(5) DEFAULT NULL,  
19 `jabatan` varchar(255) DEFAULT NULL,  
20 `cif` varchar(18) DEFAULT NULL,  
21 `npwp` varchar(25) DEFAULT NULL,  
22 `sdata` varchar(5) DEFAULT NULL,  
23 `updated_at` datetime NOT NULL,  
24 `created_at` datetime NOT NULL  
25 ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

7.2.3 Implementasi Tabel Data Lead

Tabel Data Lead dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel Data Lead pada Database Sistem Informasi *Lending BROMO* pada Tabel 7.3.

Tabel 7.3 Kode Program Membuat Tabel Data lead

datalead.sql

```
1 CREATE TABLE `datalead` (  
2   `id` int(255) NOT NULL,  
3   `iddeb` int(255) NOT NULL,  
4   `idper` int(255) NOT NULL DEFAULT 0,  
5   `skim` varchar(4) DEFAULT NULL,  
6   `SK` varchar(2) NOT NULL,  
7   `tglData` date DEFAULT NULL,  
8   `remark` text DEFAULT NULL,  
9   `stKredit` varchar(2) NOT NULL,  
10  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON  
11  UPDATE current_timestamp(),  
12  `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00'  
12 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

7.2.4 Implementasi Tabel Refuse Data Lead (cdatalead)

Tabel cdatalead dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel cdatalead pada Database Sistem Informasi *Lending BROMO* pada Tabel 7.4.

Tabel 7.4 Kode Program Membuat Tabel Refuse Data lead

cdatalead.sql

```
1 CREATE TABLE `cdatalead` (
```

cdatalead.sql

```
2   `id` int(255) NOT NULL,
3   `GetId` int(255) NOT NULL,
4   `Note` text NOT NULL,
5   `Snama` varchar(255) NOT NULL,
6   `Sjob` varchar(50) NOT NULL,
7   `Soffice` varchar(50) NOT NULL,
8   `tglBatal` timestamp NOT NULL DEFAULT current_timestamp() ON
UPDATE current_timestamp(),
9   `updated_at` timestamp NOT NULL DEFAULT current_timestamp(),
10  `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00'
11 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

7.2.5 Implementasi Tabel Solicit

Tabel solicit dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel solicit pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.5.

Tabel 7.5 Kode Program Membuat Tabel Solicit

solicit.sql

```
1 CREATE TABLE `solicit` (
2   `idsol` int(11) NOT NULL,
3   `GetId` int(255) NOT NULL,
4   `iddata` varchar(20) NOT NULL,
5   `scek` int(1) NOT NULL,
6   `Snama` varchar(255) NOT NULL,
7   `Soffice` varchar(50) NOT NULL,
8   `Sjob` varchar(50) NOT NULL,
9   `Stglsales` datetime DEFAULT NULL,
10  `nohpSales` varchar(15) DEFAULT NULL,
11  `Spenyelia` varchar(255) DEFAULT NULL,
12  `Spenoff` varchar(50) DEFAULT NULL,
13  `nohpPenyelia` varchar(15) DEFAULT NULL,
14  `Stglpenyelia` datetime DEFAULT NULL,
15  `SPBP` varchar(255) DEFAULT NULL,
16  `SPBPoff` varchar(50) DEFAULT NULL,
17  `StglPBP` datetime DEFAULT NULL,
18  `nohpPBP` varchar(15) DEFAULT NULL,
19  `tglDatalead` datetime DEFAULT NULL,
20  `status` varchar(30) DEFAULT NULL,
21  `recheck` text DEFAULT NULL,
22  `updated_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
23  `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
24  `sk` varchar(2) DEFAULT NULL
25 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

7.2.6 Implementasi Tabel Jkredit

Tabel jkredit dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel jkredit pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.6.

Tabel 7.6 Kode Program Membuat Tabel JKredit

jkredit.sql	
1	CREATE TABLE `jkredit` (
2	`idjk` int(11) NOT NULL,
3	`iddata` int(200) NOT NULL,
4	`produk` varchar(15) DEFAULT NULL,
5	`dproduk` varchar(30) DEFAULT NULL,
6	`skim` varchar(7) DEFAULT NULL,
7	`bunga` varchar(25) DEFAULT NULL,
8	`komite` varchar(8) DEFAULT NULL,
9	`estimasi` date DEFAULT NULL,
10	`tenor` varchar(12) DEFAULT NULL,
11	`tglbook` date DEFAULT NULL,
12	`bkUpdate` varchar(255) DEFAULT NULL,
13	`bksetujui` varchar(255) DEFAULT NULL,
14	`bkDisposisi` varchar(255) DEFAULT NULL,
15	`kreditmaks` varchar(255) DEFAULT NULL,
16	`tglK` date DEFAULT NULL,
17	`tglKomite` date DEFAULT NULL,
18	`ccy` varchar(3) DEFAULT NULL,
19	`tKredit` varchar(30) DEFAULT NULL,
20	`sKredit` varchar(30) DEFAULT NULL,
21	`toKredit` varchar(30) DEFAULT NULL,
22	`fromKredit` varchar(300) DEFAULT NULL,
23	`stPipeline` varchar(30) DEFAULT NULL,
24	`updated_at` timestamp NOT NULL DEFAULT current_timestamp(),
25	`created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00'
26) ENGINE=InnoDB DEFAULT CHARSET=latin1;

7.2.7 Implementasi Tabel Kabupaten

Tabel kabupaten dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel kabupaten pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.7.

Tabel 7.7 Kode Program Membuat Tabel Kabupaten

kabupaten.sql	
1	CREATE TABLE `kabupaten` (
2	`idkab` int(11) NOT NULL,
3	`kab` varchar(17) CHARACTER SET utf8 DEFAULT NULL
4) ENGINE=InnoDB DEFAULT CHARSET=latin1;

7.2.8 Implementasi Tabel Kecamatan

Tabel kecamatan dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel kecamatan pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.8.

Tabel 7.8 Kode Program Membuat Tabel Kecamatan

kecamatan.sql	
1	CREATE TABLE `kecamatan` (2 `idkec` int(11) NOT NULL, 3 `idkab` int(11) DEFAULT NULL, 4 `kec` varchar(19) CHARACTER SET utf8 DEFAULT NULL 5) ENGINE=InnoDB DEFAULT CHARSET=latin1;

7.2.9 Implementasi Tabel Desa

Tabel desa dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel desa pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.9.

Tabel 7.9 Kode Program Membuat Tabel Desa

desa.sql	
1	CREATE TABLE `desa` (2 `iddesa` int(12) NOT NULL, 3 `idkec` int(12) DEFAULT NULL, 4 `desa` varchar(19) CHARACTER SET utf8 DEFAULT NULL, 5 `kdpos` int(5) DEFAULT NULL 6) ENGINE=InnoDB DEFAULT CHARSET=latin1;

7.2.10 Implementasi Tabel Outlet

Tabel outlet dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel outlet pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.10.

Tabel 7.10 Kode Program Membuat Tabel Outlet

outlet.sql	
1	CREATE TABLE `outlet` (2 `idout` int(2) NOT NULL, 3 `nameoutlet` varchar(50) NOT NULL, 4 `officeID` int(2) NOT NULL 5) ENGINE=InnoDB DEFAULT CHARSET=latin1;

7.2.11 Implementasi Tabel Dproduk

Tabel dproduk dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel dproduk pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.11.

Tabel 7.11 Kode Program Membuat Tabel Detail Produk

dproduk.sql

```
1 CREATE TABLE `dproduk` (
2   `iddp` int(11) NOT NULL,
3   `produk` varchar(10) NOT NULL,
4   `detailProduk` varchar(30) NOT NULL
5 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

7.2.12 Implementasi Tabel Jpekerjaan

Tabel jpekerjaan dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel jpekerjaan pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.12.

Tabel 7.12 Kode Program Membuat Tabel Jenis Pekerjaan

jpekerjaan.sql

```
1 CREATE TABLE `jpekerjaan` (
2   `id` int(200) NOT NULL,
3   `nmpekerjaan` varchar(255) NOT NULL
4 ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

7.2.13 Implementasi Tabel Notification

Tabel notification dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel notification pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.13.

Tabel 7.13 Kode Program Membuat Tabel Notifikasi

notification.sql

```
1 CREATE TABLE `notification` (
2   `id` int(11) NOT NULL,
3   `notiuser` varchar(50) NOT NULL,
4   `notireciver` varchar(50) NOT NULL,
```

notification.sql	
5	`notitype` varchar(50) NOT NULL,
6	`time` timestamp NOT NULL DEFAULT current_timestamp(),
7	`ddata` varchar(200) NOT NULL,
8	`status` int(11) NOT NULL DEFAULT 0,
9	`iddata` int(255) DEFAULT NULL,
10	`idper` int(255) DEFAULT NULL,
11	`detailStatus` text NOT NULL,
12	`click` int(1) DEFAULT NULL,
13	`created_at` timestamp NULL DEFAULT NULL,
14	`updated_at` timestamp NOT NULL DEFAULT current_timestamp()
15) ENGINE=InnoDB DEFAULT CHARSET=latin1;

7.2.14 Implementasi Tabel Status

Tabel status dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel status pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.14.

Tabel 7.14 Kode Program Membuat Tabel Status

status.sql	
1	CREATE TABLE `status` (
2	`idSta` int(11) NOT NULL,
3	`stNow` varchar(20) NOT NULL,
4	`Jsta` varchar(2) NOT NULL
5) ENGINE=InnoDB DEFAULT CHARSET=latin1;

7.2.15 Implementasi Tabel Kewenangan

Tabel kewenangan dibuat berdasarkan rancangan yang telah dilakukan sebelumnya. Berikut adalah hasil dari implementasi tabel kewenangan pada *Database Sistem Informasi Lending BROMO* pada Tabel 7.15.

Tabel 7.15 Kode Program Membuat Tabel Kewenangan

kewenangan.sql	
1	CREATE TABLE `kewenangan` (
2	`idkew` int(200) NOT NULL,
3	`idjk` int(200) NOT NULL,
4	`kewenangan` varchar(255) NOT NULL,
5	`nmPemutus` varchar(255) NOT NULL,
6	`pjPemutus` varchar(255) NOT NULL,
7	`updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE current_timestamp(),
8	`created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00'
10) ENGINE=MyISAM DEFAULT CHARSET=utf8;

7.3. Implementasi Kode Program

Implementasi kode program merupakan tahap penulisan kode program dengan bahasa pemrograman berdasarkan *class diagram* yang telah dirancang. Bahasa pemrograman yang digunakan adalah *PHP*. Pada bagian ini, pembahasan kode program hanya seputar fungsi utama yang mendukung berjalannya proses bisnis *lending* BNI Wilayah Malang.

7.3.1. Implementasi Login

Hasil implementasi kode program untuk proses *login* berada pada *Auth Gateway* ditunjukkan pada tabel 7.16. *Method* *postLogin* akan menerima parameter dari sebuah *form* yang dibungkus dengan objek *Request* yang memiliki key *NPP* dan *password* untuk dicek pada *database*. Jika *NPP* dan *password* tersedia pada *database*, *NPP* dan *password* akan dienkripsi. Hasil dari enkripsi disematkan pada *URL* dan diteruskan berdasarkan *role* dari *NPP*. Jika *NPP* dan *password* tidak tersedia pada *database*, akan diteruskan ke halaman *login* kembali.

Tabel 7.16 Kode Program *Method postLogin* pada *Auth Gateway*

AuthController.php	
1 public function postLogin(Request \$request) 2 { 3 \$request->validate([4 'npp' => 'required', 5 'password' => 'required', 6]); 7 \$credentials = \$request->only('npp', 'password'); 8 if (Auth::attempt(\$credentials)) { 9 \$token = Crypt::encryptString(\$request->npp . "rahasia" . \$request->password); 10 \$dateToday = date('Y/m/d H:i:s'); 11 User::where('npp', \$request->npp) 12 ->update(['last_login' => \$dateToday]); 13 switch (Auth::user()->job) { 14 case 'SCO': 15 return Redirect::to(16 "https://sales.bromo.website/login?token=\$token"); 17 case 'PENYELIA LENDING CR': 18 return Redirect::to(19 "https://penyelia.bromo.website/login?token=\$token"); 20 case 'PBP': 21 return Redirect::to(22 "https://pbp.bromo.website/login?token=\$token"); 23 } 24 return Redirect::to("login")->withSuccess('Oops! You have entered invalid credentials'); 25 }	

7.3.2. Implementasi *Input Datalead*

Hasil implementasi kode program untuk proses input datalead berada pada layanan *Sales*, *penyelia*, dan *PBP*. ditunjukkan pada tabel 7.17 dan 7.18. *Method* *inputManual* akan menerima parameter dari sebuah *form* yang dibungkus dengan objek *Request* yang memiliki key tuan, debitur, alamat, kab, kec, desa, kdpos, nohp, noktp, sk, outlet, exist, dan jepek. Selanjutnya data objek tersebut diubah menjadi *array* dan dipanggil *method create* dengan parameter *array* tersebut.

Tabel 7.17 Kode Program *Method inputManual* pada Sales, Penyelia, dan PBP

DataleadController.php	
1 public function inputManual(Request \$request) 2 { 3 if(\$request->sk=="CR") { 4 request()->validate([5 'tuan' => 'required', 6 'debitur' => 'required', 7 'alamat' => 'required', 8 'kab' => 'required', 9 'kec' => 'required', 10 'desa' => 'required', 11 'kdpos' => 'required', 12 'nohp' => 'required', 13 'noktp' => 'required unique:debitur,noktp', 14 'sk' => 'required', 15 'outlet' => 'required', 16 'exist' => 'required', 17 'jepek' => 'required' 18]); 19 } 20 \$data = \$request->all(); 21 22 \$check = \$this->create(\$data); 23 return redirect()->back()->with('message', 'Berhasil input Data Baru'); 24 25 }	

Method create yang ditunjukkan pada tabel 7.18 akan mengecek apakah data dengan key sk memiliki nilai CR. jika ya, akan disimpan pada *database* di tabel Debitur dan Datalead. Lalu akan diteruskan ke halaman sebelumnya dengan *message* “Berhasil input Data Baru”.

Tabel 7.18 Kode Program *Method create pada Sales, Penyelia, dan PBP*

<pre>DataleadController.php</pre>
<pre> 1 public function create(array \$data) 2 { 3 if (\$data['sk'] == "CR") { 4 \$debitur = Debitur::create([5 'nmInput' => ucfirst(Auth()->user()->name), 6 'cabang' => ucwords(strtolower(Auth()->user()->office)), 7 'tuan' => \$data['tuan'], 8 'debitur' => \$data['debitur'], 9 'noktp' => \$data['noktp'], 10 'pekerjaan'=> \$data['jepek'], 11 'outlet' => \$data['outlet'], 12 'nohp' => "+62" . \$data['nohp'], 13 'Exist' => \$data['exist'], 14 'alamat' => \$data['alamat'], 15 'kab' => \$data['kab'], 16 'kec' => \$data['kec'], 17 'desa' => \$data['desa'], 18 'kdpos' => \$data['kdpos'], 19 'sdata' => "INPUT", 20]); 21 22 \$datalead = Datalead::create([23 'SK' => \$data['sk'], 24 'iddeb' => \$debitur->id, 25 'stKredit' => '0', 26]); 27 28 \$notif= Notification::create([29 'notiuser'=> ucfirst(Auth()->user()->name), 30 'notireciver'=> 'ADMIN', 31 'notitype'=> 'INPUT MANUAL', 32 'time'=> date("Y-m-d")." ". date("H:i:s"), 33 'ddata'=> 'Debitur', 34 'iddata' => \$debitur->id, 35 'detailStatus' => 'Input Datalead', 36]); 37 return redirect()->back()->with('message', 'Berhasil input Data 38 Baru'); 39 } </pre>

7.3.3. Implementasi *datalead* ke *Solicit*

Hasil implementasi kode program untuk proses pengambilan data dari Datalead ke *Solicit* berada pada layanan *Sales* ditunjukkan pada tabel 7.19. *Method createSolicit* akan menerima parameter *id* dan objek *Request* yang memiliki key *tgldata*. *Method createSolicit* akan mengecek apakah *job* dari akun adalah *SCO*, lalu *id* dan *NPP* dari akun digabungkan untuk dicek pada *database* apakah akun tersebut pernah mengambil *datalead* tersebut. Jika pernah, permintaan pengambilan akan ditolak. Jika data tidak ditemukan, akan dilakukan penyimpanan ke *database* di tabel *solicit*.

Tabel 7.19 Kode Program Method *createSolicit* pada Sales

SolicitController.php	
<pre> 1 public function createSolicit(\$id, Request \$request) 2 { 3 if(Auth()->user()->job == "SCO") { 4 5 \$iddata = \$id.Auth()->user()->npp; 6 if (Solicit::where('iddata', '=', \$iddata)->exists()) { 7 return redirect()->back()->with('error', 'Anda telah 8 melakukan solicit pada debitur ini!'); 8 9 10 \$solicit = Solicit::create([11 'Snama' => ucfirst(Auth()->user()->name), 12 'Soffice' => ucwords(strtolower(Auth()->user()->office)), 13 'nohpSales' => ucfirst(Auth()->user()->mobile), 14 'GetId' => \$id, 15 'iddata' => \$iddata, 16 'scek'=> '0', 17 'Stglsales' => date("Y-m-d")." ". date("H:i:s"), 18 'Sjob' => ucfirst(Auth()->user()->job), 19 'status' => 'Solicit', 20 'tglDatalead' => date(\$request->tgldata) 21]); 22 23 return redirect('/solicit')->with('message', 'Berhasil input 24 solicit.'); 24 } 25 else { 26 return abort(403, "Akses tidak diterima!"); 26 } 27 } 28 </pre>	

7.3.4. Implementasi *Request Prospect*

Hasil implementasi kode program untuk permintaan persetujuan proses solicit ke prospect berada pada layanan *Sales* ditunjukkan pada tabel 7.20. *Method* *beProspect* akan menerima parameter dari sebuah *form* yang dibungkus dengan objek *Request* yang memiliki *key* produk, sk, solicitid dan dataleadid. Ada beberapa manipulasi data pada *database* dalam *method* ini, yaitu :

- Perubahan di tabel *solicit* untuk merubah status menjadi “Be Prospect” untuk id yang diminta.
- Perubahan di tabel *datalead* untuk merubah *stkredit* menjadi “1”.
- Perubahan di tabel *solicit* untuk merubah status menjadi “Take Over” dan *Scek* menjadi “1” untuk id yang diminta oleh akun lain.
- Pembuatan data baru pada tabel *jkredit* untuk menyimpan produk yang dipilih.
- Pembuatan data baru pada tabel *notifikasi* berupa detail permintaan.

Tabel 7.20 Kode Program *Method beProspect* pada Sales

SolicitController.php	
<pre> 1 public function beProspect(Request \$request) 2 { 3 \$office = Auth()->user()->office; 4 \$produk = \$request->produk; 5 \$solicit = Solicit::where('idsol', \$request->solicitid)- >where('Scek', "0") 6 ->update([7 'status' => "Be Prospect", 8 'sk' => \$request->sk 9]); 10 \$datalead = Datalead::where('id', \$request->dataleadid) 11 ->update([12 'Stkredit' => "1" 13]); 14 \$jkredit = Jkredit::create([15 'iddata' => \$request->dataleadid, 16 'produk' => \$produk, 17 'dproduk' => "0" 18]); 19 20 \$clearother = Solicit::where('GetId', \$request->dataleadid)- >where('status','Solicit') 21 ->update(['status' => "Take Over", 22 'Scek' => "1", 23]); 24 25 \$notif = Notification::create([26 'notiuser' => Auth()->user()->name, 27 'notireciver' => "PENYELIA LENDING ".\$request->sk." 28 ".\$office, 29 'notitype' => "Be Prospect", 30 'ddata' => "/approval", 31 'detailStatus' => Auth()->user()->name." meminta approval 32 Prospect" 33]); 34 return redirect()->back()->with('message', 'Berhasil 35 prospect.'); 36 }</pre>	

7.3.5. Implementasi Approval & Reject Prospect

Hasil implementasi kode program untuk persetujuan proses prospect ke pipeline berada pada layanan Penyelia ditunjukkan pada tabel 7.21. *Method approveProspect* akan menerima parameter dari sebuah *form* yang dibungkus dengan objek *Request* yang memiliki *key* button, solicitid, dataleadid, sk, namasales, note. Ada 2 kondisi berdasarkan nilai button, yaitu :

Jika button bernilai “approve” :

- Perubahan data di tabel solicit untuk status menjadi “Prospect” dan informasi tentang penyelia.
- Perubahan data di tabel datalead untuk stkredit menjadi “2”

- Pembuatan data baru pada tabel notifikasi berupa detail persetujuan.

Jika button bernilai “reject” :

- Pembuatan data baru pada tabel cdatalead berupa detail penolakan.
- Perubahan data di tabel solicit untuk status menjadi “Reject Data” dan scek menjadi “1”
- Pembuatan data baru pada tabel notifikasi berupa detail penolakan.

Tabel 7.21 Kode Program *Method approveProspect* pada Penyelia

<pre>ProspectController.php</pre>

<pre> 1 public function approveProspect(Request \$request) 2 { 3 if (\$request->button == "approve") { 4 \$solicit = Solicit::where('idsol', \$request->solicitid) 5 ->update([6 'status' => "Prospect", 7 'Spenyelia' => ucfirst(Auth()->user()->name), 8 'Spenoff' => ucfirst(Auth()->user()->office), 9 'nchpPenyelia' => ucfirst(Auth()->user()->mobile), 10 'Stglpenyelia' => date("Y-m-d")." ".date("H:i:s") 11]); 12 \$datalead = Datalead::where('id', \$request->dataleadid) 13 ->update([14 'Stkredit' => "2" 15]); 16 17 if (\$request->sk == "CR") { 18 \$data = "/prospect"; 19 } 20 21 \$notif = Notification::create([22 'notiuser' => Auth()->user()->name, 23 'notireciver' => \$request->namasales, 24 'notitype' => "Approve Prospect", 25 'ddata' => \$data, 26 'detailStatus' => Auth()->user()->name." telah melakukan approval prospect" 27]); 28 return redirect()->back()->with('message', 'Berhasil approve.'); 29 30 } else if (\$request->button == "reject") { 31 \$cdatalead = Cdatalead::create([32 'GetId' => \$request->dataleadid, 33 'Note' => \$request->note, 34 'Snama' => ucfirst(Auth()->user()->name), 35 'Sjob' => ucfirst(Auth()->user()->job), 36 'Soffice' => ucwords(strtolower(Auth()->user()->office)) 37]); 38 \$solicit = Solicit::where('idsol', \$request->solicitid) 39 ->update([40 'status' => 'Reject Data', 41]); 42 } 43 } 44 }</pre>

```
ProspectController.php
```

```
38         'Scek' => '1'
39     ]);
40
41     $notif = Notification::create([
42         'notiuser' => Auth()>user()>name,
43         'notireciver' => $request->namasales,
44         'notitype' => "Reject Prospect",
45         'ddata' => "/refuse",
46         'detailStatus' => Auth()>user()>name." telah menolak
47         approval prospect"
48     ]);
49     return redirect()->back()->with('message', 'Berhasil
50     reject.');
51 }
```

7.3.6. Implementasi *Request Pipeline*

Hasil implementasi kode program untuk permintaan persetujuan proses prospect ke pipeline berada pada layanan *Sales* ditunjukkan pada tabel 7.22. *Method* submitProspectToPbp akan menerima parameter dari sebuah *form* yang dibungkus dengan objek *Request* yang memiliki *key* namaibu, alamat, kab, kec, desa, kdpos, nohp, noktp, idoutlet, jepek, npwp, dproduk. Ada beberapa manipulasi data pada *database* dalam *method* ini, yaitu :

- Pembuatan data baru pada tabel debitur dan datalead untuk data ibu dari debitur
- Perubahan di tabel solicit untuk merubah status menjadi “Be Pipeline” untuk iddeb yang diminta.
- Perubahan di tabel datalead untuk merubah stkredit menjadi “3”
- Perubahan di tabel jkredit untuk detail produk
- Pembuatan data baru pada tabel notifikasi berupa detail permintaan.

Tabel 7.22 Kode Program *Method submitProspectToPbp* pada Sales

```
ProspectController.php
```

```
1 public function submitProspectToPbp(Request $request)
2 {
3     $office = Auth()>user()>office;
4     $ibu = Debitur::create([
5         'nmInput' => ucfirst(Auth()>user()>name),
6         'cabang' => ucwords(strtolower(Auth()>user()>office)),
7         'tuan' => "Ibu",
8         'debitur' => $request->namaibu,
9         'alamat' => $request->alamat,
10        'kab' => $request->kab,
```

ProspectController.php

```
11     'kec' => $request->kec,
12     'desa' => $request->desa,
13     'kdpos' => $request->kdpos,
14     'nohp' => "+62".$request->nohp,
15     'noktp' => $request->noktp,
16     'sk' => 'CR',
17     'outlet' => $request->idoutlet,
18     'Exist' => "BARU",
19     'pekerjaan' => $request->jepek,
20     'sdata' => "INPUT"
21 ];
22 $ibuDatalead = Datalead::create([
23     'SK' => 'CR',
24     'iddeb' => $ibu->id,
25     'stKredit' => '0'
26 ]);
27
28 $deb = Debitur::where('iddeb', $request->debiturid)
->update([
29     'npwp' => $request->npwp,
30     'cif' => '0',
31     'idibu' => $request->noktp
32 ]);
33 $sol = Solicit::where('GetId', $request->dataleadid)-
>where('Snama', Auth()->user()->name)
->update([
34     'status' => 'Be Pipeline'
35 ]);
36 $dat = Datalead::where('id', $request->dataleadid)
->update([
37     'Stkredit' => "3"
38 ]);
39 $jkredit = Jkredit::where('idjk', $request->idjk)
->update([
40     'dproduk' => $request->dproduk
41 ]);
42
43
44 $notif = Notification::create([
45     'notiuser' => Auth()->user()->name,
46     'notireceiver' => "PBP ".$office,
47     'notitype' => "Be Pipeline",
48     'ddata' => "/approveCR",
49     'detailStatus' => Auth()->user()->name." meminta
50 approval Pipeline CR"
51     ]);
52
53     return redirect()->back()->with('message', 'Berhasil! Debitur
akan direview oleh PBP.');
    }
```

7.3.7. Implementasi *Approval, Recheck & Reject Pipeline*

Hasil implementasi kode program untuk persetujuan proses prospect ke pipeline berada pada layanan PBP ditunjukkan pada tabel 7.23. *Method approvePipeline* akan menerima parameter dari sebuah *form* yang dibungkus

dengan objek *Request* yang memiliki *key* button, solicitid, dataleadid, sk, namasales, note, remark . Ada 3 kondisi berdasarkan nilai button, yaitu :

Jika *button* bernilai “*approve*” :

- Perubahan data di tabel *solicit* untuk status menjadi “Pipeline” dan informasi tentang PBP.
- Perubahan data di tabel *datalead* untuk *stkredit* menjadi “4”
- Pembuatan data baru pada tabel notifikasi berupa detail persetujuan.

Jika *button* bernilai “*reject*” :

- Pembuatan data baru pada tabel *cdatalead* berupa detail penolakan.
- Perubahan data di tabel *solicit* untuk status menjadi “Reject Pipeline” dan *scek* menjadi “1”
- Pembuatan data baru pada tabel notifikasi berupa detail penolakan.

Jika *button* bernilai “*recheck*” :

- Pembuatan data baru pada tabel *cdatalead* berupa detail penolakan.
- Perubahan data di tabel *solicit* untuk status menjadi “Prospect”
- Perubahan data di tabel *datalead* untuk *stkredit* menjadi “2” dan diberi pesan *remark*
- Pembuatan data baru pada tabel notifikasi berupa detail untuk pengecekan ulang.

Tabel 7.23 Kode Program *Method approvePipeline* pada PBP

PipelineController.php	
1 public function approvePipeline(Request \$request) 2 { 3 if (\$request->button == "approve") { 4 \$solicit = Solicit::where('idsol', \$request->solicitid) 5 ->update([6 'status' => "Pipeline", 7 'SPBP' => ucfirst(Auth()->user()->name), 8 'SPBPOff' => ucfirst(Auth()->user()->office), 9 'nohpPBP' => ucfirst(Auth()->user()->mobile), 10 'StglPBP' => date("Y-m-d")." ".date("H:i:s") 11]); 12 \$datalead = Datalead::where('id', \$request->dataleadid) 13 ->update([14 'Stkredit' => "4" 15]); 16 if (\$request->sk == "CR") { 17 // proses CR 18 } 19 } 20 } 21 } 22 } 23 } 24 } 25 } 26 } 27 } 28 } 29 } 30 } 31 } 32 } 33 } 34 } 35 } 36 } 37 } 38 } 39 } 40 } 41 } 42 } 43 } 44 } 45 } 46 } 47 } 48 } 49 } 50 } 51 } 52 } 53 } 54 } 55 } 56 } 57 } 58 } 59 } 60 } 61 } 62 } 63 } 64 } 65 } 66 } 67 } 68 } 69 } 70 } 71 } 72 } 73 } 74 } 75 } 76 } 77 } 78 } 79 } 80 } 81 } 82 } 83 } 84 } 85 } 86 } 87 } 88 } 89 } 90 } 91 } 92 } 93 } 94 } 95 } 96 } 97 } 98 } 99 } 100 } 101 } 102 } 103 } 104 } 105 } 106 } 107 } 108 } 109 } 110 } 111 } 112 } 113 } 114 } 115 } 116 } 117 } 118 } 119 } 120 } 121 } 122 } 123 } 124 } 125 } 126 } 127 } 128 } 129 } 130 } 131 } 132 } 133 } 134 } 135 } 136 } 137 } 138 } 139 } 140 } 141 } 142 } 143 } 144 } 145 } 146 } 147 } 148 } 149 } 150 } 151 } 152 } 153 } 154 } 155 } 156 } 157 } 158 } 159 } 160 } 161 } 162 } 163 } 164 } 165 } 166 } 167 } 168 } 169 } 170 } 171 } 172 } 173 } 174 } 175 } 176 } 177 } 178 } 179 } 180 } 181 } 182 } 183 } 184 } 185 } 186 } 187 } 188 } 189 } 190 } 191 } 192 } 193 } 194 } 195 } 196 } 197 } 198 } 199 } 200 } 201 } 202 } 203 } 204 } 205 } 206 } 207 } 208 } 209 } 210 } 211 } 212 } 213 } 214 } 215 } 216 } 217 } 218 } 219 } 220 } 221 } 222 } 223 } 224 } 225 } 226 } 227 } 228 } 229 } 230 } 231 } 232 } 233 } 234 } 235 } 236 } 237 } 238 } 239 } 240 } 241 } 242 } 243 } 244 } 245 } 246 } 247 } 248 } 249 } 250 } 251 } 252 } 253 } 254 } 255 } 256 } 257 } 258 } 259 } 260 } 261 } 262 } 263 } 264 } 265 } 266 } 267 } 268 } 269 } 270 } 271 } 272 } 273 } 274 } 275 } 276 } 277 } 278 } 279 } 280 } 281 } 282 } 283 } 284 } 285 } 286 } 287 } 288 } 289 } 290 } 291 } 292 } 293 } 294 } 295 } 296 } 297 } 298 } 299 } 300 } 301 } 302 } 303 } 304 } 305 } 306 } 307 } 308 } 309 } 310 } 311 } 312 } 313 } 314 } 315 } 316 } 317 } 318 } 319 } 320 } 321 } 322 } 323 } 324 } 325 } 326 } 327 } 328 } 329 } 330 } 331 } 332 } 333 } 334 } 335 } 336 } 337 } 338 } 339 } 340 } 341 } 342 } 343 } 344 } 345 } 346 } 347 } 348 } 349 } 350 } 351 } 352 } 353 } 354 } 355 } 356 } 357 } 358 } 359 } 360 } 361 } 362 } 363 } 364 } 365 } 366 } 367 } 368 } 369 } 370 } 371 } 372 } 373 } 374 } 375 } 376 } 377 } 378 } 379 } 380 } 381 } 382 } 383 } 384 } 385 } 386 } 387 } 388 } 389 } 390 } 391 } 392 } 393 } 394 } 395 } 396 } 397 } 398 } 399 } 400 } 401 } 402 } 403 } 404 } 405 } 406 } 407 } 408 } 409 } 410 } 411 } 412 } 413 } 414 } 415 } 416 } 417 } 418 } 419 } 420 } 421 } 422 } 423 } 424 } 425 } 426 } 427 } 428 } 429 } 430 } 431 } 432 } 433 } 434 } 435 } 436 } 437 } 438 } 439 } 440 } 441 } 442 } 443 } 444 } 445 } 446 } 447 } 448 } 449 } 450 } 451 } 452 } 453 } 454 } 455 } 456 } 457 } 458 } 459 } 460 } 461 } 462 } 463 } 464 } 465 } 466 } 467 } 468 } 469 } 470 } 471 } 472 } 473 } 474 } 475 } 476 } 477 } 478 } 479 } 480 } 481 } 482 } 483 } 484 } 485 } 486 } 487 } 488 } 489 } 490 } 491 } 492 } 493 } 494 } 495 } 496 } 497 } 498 } 499 } 500 } 501 } 502 } 503 } 504 } 505 } 506 } 507 } 508 } 509 } 510 } 511 } 512 } 513 } 514 } 515 } 516 } 517 } 518 } 519 } 520 } 521 } 522 } 523 } 524 } 525 } 526 } 527 } 528 } 529 } 530 } 531 } 532 } 533 } 534 } 535 } 536 } 537 } 538 } 539 } 540 } 541 } 542 } 543 } 544 } 545 } 546 } 547 } 548 } 549 } 550 } 551 } 552 } 553 } 554 } 555 } 556 } 557 } 558 } 559 } 560 } 561 } 562 } 563 } 564 } 565 } 566 } 567 } 568 } 569 } 570 } 571 } 572 } 573 } 574 } 575 } 576 } 577 } 578 } 579 } 580 } 581 } 582 } 583 } 584 } 585 } 586 } 587 } 588 } 589 } 590 } 591 } 592 } 593 } 594 } 595 } 596 } 597 } 598 } 599 } 600 } 601 } 602 } 603 } 604 } 605 } 606 } 607 } 608 } 609 } 610 } 611 } 612 } 613 } 614 } 615 } 616 } 617 } 618 } 619 } 620 } 621 } 622 } 623 } 624 } 625 } 626 } 627 } 628 } 629 } 630 } 631 } 632 } 633 } 634 } 635 } 636 } 637 } 638 } 639 } 640 } 641 } 642 } 643 } 644 } 645 } 646 } 647 } 648 } 649 } 650 } 651 } 652 } 653 } 654 } 655 } 656 } 657 } 658 } 659 } 660 } 661 } 662 } 663 } 664 } 665 } 666 } 667 } 668 } 669 } 670 } 671 } 672 } 673 } 674 } 675 } 676 } 677 } 678 } 679 } 680 } 681 } 682 } 683 } 684 } 685 } 686 } 687 } 688 } 689 } 690 } 691 } 692 } 693 } 694 } 695 } 696 } 697 } 698 } 699 } 700 } 701 } 702 } 703 } 704 } 705 } 706 } 707 } 708 } 709 } 710 } 711 } 712 } 713 } 714 } 715 } 716 } 717 } 718 } 719 } 720 } 721 } 722 } 723 } 724 } 725 } 726 } 727 } 728 } 729 } 730 } 731 } 732 } 733 } 734 } 735 } 736 } 737 } 738 } 739 } 740 } 741 } 742 } 743 } 744 } 745 } 746 } 747 } 748 } 749 } 750 } 751 } 752 } 753 } 754 } 755 } 756 } 757 } 758 } 759 } 760 } 761 } 762 } 763 } 764 } 765 } 766 } 767 } 768 } 769 } 770 } 771 } 772 } 773 } 774 } 775 } 776 } 777 } 778 } 779 } 780 } 781 } 782 } 783 } 784 } 785 } 786 } 787 } 788 } 789 } 790 } 791 } 792 } 793 } 794 } 795 } 796 } 797 } 798 } 799 } 800 } 801 } 802 } 803 } 804 } 805 } 806 } 807 } 808 } 809 } 810 } 811 } 812 } 813 } 814 } 815 } 816 } 817 } 818 } 819 } 820 } 821 } 822 } 823 } 824 } 825 } 826 } 827 } 828 } 829 } 830 } 831 } 832 } 833 } 834 } 835 } 836 } 837 } 838 } 839 } 840 } 841 } 842 } 843 } 844 } 845 } 846 } 847 } 848 } 849 } 850 } 851 } 852 } 853 } 854 } 855 } 856 } 857 } 858 } 859 } 860 } 861 } 862 } 863 } 864 } 865 } 866 } 867 } 868 } 869 } 870 } 871 } 872 } 873 } 874 } 875 } 876 } 877 } 878 } 879 } 880 } 881 } 882 } 883 } 884 } 885 } 886 } 887 } 888 } 889 } 890 } 891 } 892 } 893 } 894 } 895 } 896 } 897 } 898 } 899 } 900 } 901 } 902 } 903 } 904 } 905 } 906 } 907 } 908 } 909 } 910 } 911 } 912 } 913 } 914 } 915 } 916 } 917 } 918 } 919 } 920 } 921 } 922 } 923 } 924 } 925 } 926 } 927 } 928 } 929 } 930 } 931 } 932 } 933 } 934 } 935 } 936 } 937 } 938 } 939 } 940 } 941 } 942 } 943 } 944 } 945 } 946 } 947 } 948 } 949 } 950 } 951 } 952 } 953 } 954 } 955 } 956 } 957 } 958 } 959 } 960 } 961 } 962 } 963 } 964 } 965 } 966 } 967 } 968 } 969 } 970 } 971 } 972 } 973 } 974 } 975 } 976 } 977 } 978 } 979 } 980 } 981 } 982 } 983 } 984 } 985 } 986 } 987 } 988 } 989 } 990 } 991 } 992 } 993 } 994 } 995 } 996 } 997 } 998 } 999 } 1000 } 1001 } 1002 } 1003 } 1004 } 1005 } 1006 } 1007 } 1008 } 1009 } 1010 } 1011 } 1012 } 1013 } 1014 } 1015 } 1016 } 1017 } 1018 } 1019 } 1020 } 1021 } 1022 } 1023 } 1024 } 1025 } 1026 } 1027 } 1028 } 1029 } 1030 } 1031 } 1032 } 1033 } 1034 } 1035 } 1036 } 1037 } 1038 } 1039 } 1040 } 1041 } 1042 } 1043 } 1044 } 1045 } 1046 } 1047 } 1048 } 1049 } 1050 } 1051 } 1052 } 1053 } 1054 } 1055 } 1056 } 1057 } 1058 } 1059 } 1060 } 1061 } 1062 } 1063 } 1064 } 1065 } 1066 } 1067 } 1068 } 1069 } 1070 } 1071 } 1072 } 1073 } 1074 } 1075 } 1076 } 1077 } 1078 } 1079 } 1080 } 1081 } 1082 } 1083 } 1084 } 1085 } 1086 } 1087 } 1088 } 1089 } 1090 } 1091 } 1092 } 1093 } 1094 } 1095 } 1096 } 1097 } 1098 } 1099 } 1100 } 1101 } 1102 } 1103 } 1104 } 1105 } 1106 } 1107 } 1108 } 1109 } 1110 } 1111 } 1112 } 1113 } 1114 }	

PipelineController.php

```
19         $data = "/pipeline";
20     }
21
22     $notif = Notification::create([
23         'notiuser' => Auth()>user()>name,
24         'notireciver' => $request->namasales,
25         'notitype' => "Approve Pipeline",
26         'ddata' => $data,
27         'detailStatus' => Auth()>user()>name." telah
melakukan approval pipeline"
28             ]);
29     return redirect()->back()->with('message', 'Berhasil
approve.');
30
31 } else if ($request->button == "reject") {
32     $cdatalead = Cdatalead::create([
33         'GetId' => $request->dataleadid,
34         'Note' => $request->note,
35         'Snama' => ucfirst(Auth()>user()>name),
36         'Sjob' => ucfirst(Auth()>user()>job),
37         'Soffice' => ucwords(strtolower(Auth()>user()-
>office))
38     ]);
39     $solicit = Solicit::where('idsol', $request->solicitid)
->update([
40         'status' => 'Reject Data',
41         'Scek' => '1'
42     ]);
43     $datalead = Datalead::where('id', $request-
>dataleadid)
->update([
44         'Stkredit' => "6"
45     ]);
46
47     $notif = Notification::create([
48         'notiuser' => Auth()>user()>name,
49         'notireciver' => $request->namasales,
50         'notitype' => "Reject Pipeline",
51         'ddata' => "/refuse",
52         'detailStatus' => Auth()>user()>name." telah menolak
approval pipeline"
53             ]);
54     return redirect()->back()->with('message', 'Berhasil
reject.');
55 }
56
57
58 else if ($request->button == "recheck") {
59     $solicit = Solicit::where('idsol', $request->solicitid)
->update([
60         'status' => "Prospect"
61     ]);
62     $datalead = Datalead::where('id', $request->dataleadid)
->update([
63         'Stkredit' => "2",
64         'remark' => $request->remark
65
66     ]);
67     $notif = Notification::create([
68         'notiuser' => Auth()>user()>name,
69         'notireciver' => $request->namasales,
70         'notitype' => "Recheck Pipeline",
71     ]);
```

```
PipelineController.php
```

```
72         'ddata' => "/prospect",
73         'detailStatus' => Auth()->user()->name." meminta
74         recheck data approval pipeline"
75     ]);
76     return redirect()->back()->with('message', 'Berhasil
77     Sending Recheck Data.');
78 }
```

7.3.8. Implementasi *Input Datalead* Otomatis

Hasil implementasi kode program untuk *input datalead* secara otomatis dari *file excel* menggunakan *library* tambahan bernama *laravel-excel* dan implementasi kode program ditunjukkan pada tabel 7.24. *Method import_excel* akan menerima parameter dari sebuah *form* yang dibungkus dengan objek.

Request yang memiliki *key file*. Dimana nantinya *file* tersebut akan di *upload* dan diproses oleh *library laravel-excel* menjadi bentuk *sql query* dan di inputkan ke *database*.

Tabel 7.24 Kode Program *Method import_excel* pada Penyelia dan PBP

```
PipelineController.php
```

```
1 public function import_excel(Request $request)
2 {
3     $this->validate($request, [
4         'file' => 'required|mimes:xlsx'
5     );
6
7     $file = $request->file('file');
8     $nama_file = rand().$file->getClientOriginalName();
9     $file->move('file_excel', $nama_file);
10    Excel::import(new DataleadImport,
11        public_path('/file_excel/' . $nama_file));
12    return redirect('/');
13 }
```

BAB 8 PENGUJIAN

Pada bab ini akan dipaparkan hasil pengujian pada Sistem Informasi *Lending BROMO* yang penulis kembangkan. Pengujian dilakukan dengan metode *Black Box Testing* untuk pengujian kebutuhan fungsional, metode *Compatibility Testing* untuk pengujian kebutuhan *Compatibility Browser*, dan metode *Responsive Web Design Testing* untuk kebutuhan *Responsive Web Design*.

8.1 Hasil Pengujian Menggunakan *Black Box Testing*

Dibawah ini terdapat beberapa tabel yang berfungsi untuk memaparkan hasil pengujian pada kebutuhan fungsional Sistem Informasi Lending Bromo menggunakan metode *Black Box Testing*.

Tabel 8.1 Pengujian Pada Proses Login

Nama	Login
Objek Uji	F_SIL SCO_01, F_SIL PIA_01, F_SIL PBP_01
Skenario Pengujian	Skenario 1: Aktor mengosongkan <i>Username</i> dan <i>Password</i> , lalu menekan tombol “Login”. Skenario 2: Aktor hanya mengisi <i>Username</i> , lalu menekan tombol “Login”. Skenario 3: Aktor hanya mengisi <i>Password</i> , lalu menekan tombol “Login”. Skenario 4: Aktor mengisi salah satu <i>field</i> dengan data yang benar dan <i>field</i> yang lain dengan data yang salah. Skenario 5: Aktor mengisi kedua <i>field</i> dengan data yang benar.
Hasil yang Diharapkan	Skenario 1-4: Sistem menolak akses <i>Login</i> dan mengembalikan <i>User</i> menuju halaman <i>Login</i> . Skenario 5: Sistem menerima akses <i>login</i> dan <i>User</i> memasuki halaman <i>Dashboard</i> pengguna.

Hasil yang Didapatkan	<p>Skenario 1-4: Sistem berhasil menolak akses <i>Login</i> dan mengembalikan aktor menuju halaman <i>Login</i>.</p> <p>Skenario 5: Sistem berhasil menerima akses <i>login</i> dan aktor memasuki halaman <i>Dashboard</i>.</p>
Kesimpulan	Diterima.

Tabel 8.2 Pengujian pada Proses Input Data Lead Manual

Nama	Input Data Lead Manual
Objek Uji	F_SIL SCO_02, F_SIL PIA_02, F_SIL PBP_02
Skenario Pengujian	<p>Skenario 1: Aktor mengosongkan salah satu field pada form pengisian Data Lead.</p> <p>Skenario 2: Aktor mengisi <i>field</i> dengan data yang tidak sesuai dengan format yang telah ditentukan.</p> <p>Skenario 3: Aktor mengisi seluruh <i>field</i> dengan format data yang benar.</p>
Hasil yang Diharapkan	<p>Skenario 1-2: Sistem akan menolak masukan data dari aktor.</p> <p>Skenario 3: Sistem akan menerima masukan data dari aktor.</p>
Hasil yang Didapatkan	<p>Skenario 1-2: Sistem berhasil menolak masukan data dari aktor.</p> <p>Skenario 3: Sistem berhasil menerima masukan data dari aktor.</p>
Kesimpulan	Diterima.

Tabel 8.3 Pengujian pada Proses Input Data Lead Otomatis

Nama	Input Data Lead Otomatis
Objek Uji	F_SIL_PIA_03, F_SIL_PBP_03
Skenario Pengujian	Skenario 1: Aktor mengunggah file dengan format selain .xls//xlsx Skenario 2: Aktor mengunggah file dengan format .xls/xlsx
Hasil yang Diharapkan	Skenario 1: Sistem akan menolak file dari aktor. Skenario 2: Sistem akan menerima file dari aktor.
Hasil yang Didapatkan	Skenario 1: Sistem berhasil menolak file dari aktor. Skenario 2: Sistem berhasil menerima file dari aktor.
Kesimpulan	Diterima.

Tabel 8.4 Pengujian pada Proses Solicit

Nama	Solicit
Objek Uji	F_SIL SCO_03
Skenario Pengujian	Aktor melakukan Solicit pada debitur yang diinginkan
Hasil yang Diharapkan	Sistem akan menyimpan data.
Hasil yang Didapatkan	Sistem berhasil menyimpan data.

Kesimpulan	Diterima.
-------------------	-----------

Tabel 8.5 Pengujian pada Proses Change Solicit

Nama	Change Solicit
Objek Uji	F_SIL_SCO_04
Skenario Pengujian	<p>Skenario 1: Aktor melakukan Change Solicit pada debitur yang diinginkan.</p> <p>Skenario 2: Aktor melakukan <i>Solicit</i> ulang pada debitur yang telah dilakukan Change Solicit.</p>
Hasil yang Diharapkan	<p>Skenario 1: Sistem akan menyimpan data dan status debitur kembali menjadi Data Lead.</p> <p>Skenario 2: Sistem akan menolak aktor untuk melakukan <i>Solicit</i> kembali karena sudah pernah melakukan proses <i>Solicit</i> dan <i>Change Solicit</i> pada debitur oleh SCO yang sama.</p>
Hasil yang Didapatkan	<p>Skenario 1: Sistem berhasil menyimpan data dan status debitur kembali menjadi Data Lead.</p> <p>Skenario 2: Sistem berhasil menolak aktor untuk melakukan <i>Solicit</i> kembali karena sudah pernah melakukan proses <i>Solicit</i> dan <i>Change Solicit</i> pada debitur oleh SCO yang sama.</p>
Kesimpulan	Diterima.

Tabel 8.6 Pengujian pada Proses Be Prospect

Nama	Change Solicit
Objek Uji	F_SIL SCO_05
Skenario Pengujian	Aktor melakukan Be Prospect pada debitur yang diinginkan.
Hasil yang Diharapkan	Sistem akan menyimpan data dan mengirim data debitur ke Penyelia.
Hasil yang Didapatkan	Sistem berhasil menyimpan data dan mengirim data debitur ke Penyelia.
Kesimpulan	Diterima.

Tabel 8.7 Pengujian pada Proses Be Prospect

Nama	Be Prospect
Objek Uji	F_SIL SCO_05
Skenario Pengujian	Aktor melakukan Be Prospect pada debitur yang diinginkan.
Hasil yang Diharapkan	Sistem akan menyimpan data dan mengirim data debitur ke Penyelia.
Hasil yang Didapatkan	Sistem berhasil menyimpan data dan mengirim data debitur ke Penyelia.
Kesimpulan	Diterima.

Tabel 8.8 Pengujian pada Proses Prospect

Nama	Prospect
Objek Uji	F_SIL_PIA_04
Skenario Pengujian	Aktor melakukan Prospect pada debitur yang diinginkan.
Hasil yang Diharapkan	Sistem akan menyimpan data dan mengirim data debitur ke SCO.
Hasil yang Didapatkan	Sistem berhasil menyimpan data dan mengirim data debitur ke SCO.
Kesimpulan	Diterima.

Tabel 8.9 Pengujian pada Proses Prospect

Nama	Prospect
Objek Uji	F_SIL_PIA_04
Skenario Pengujian	Aktor melakukan Prospect pada debitur yang diinginkan.
Hasil yang Diharapkan	Sistem akan menyimpan data masukan dari aktor, mengganti status debitur menjadi Prospect, dan mengirim data debitur ke SCO.
Hasil yang Didapatkan	Sistem berhasil menyimpan data masukan dari aktor, mengganti status debitur menjadi Prospect, dan mengirim data debitur ke SCO.
Kesimpulan	Diterima.

Tabel 8.10 Pengujian pada Proses Be Pipeline

Nama	Be Pipeline
Objek Uji	F_SIL SCO_06
Skenario Pengujian	<p>Aktor memilih debitur yang akan d Skenario 1: Aktor mengosongkan salah satu field pada form.</p> <p>Skenario 2: Aktor mengisi <i>field</i> dengan data yang tidak sesuai dengan format yang telah ditentukan.</p> <p>Skenario 3: Aktor mengisi seluruh <i>field</i> dengan format data yang benar.</p>
Hasil yang Diharapkan	<p>Skenario 1-2: Sistem akan menolak masukan data dari aktor.</p> <p>Skenario 3: Sistem akan menerima masukan data dari aktor.</p>
Hasil yang Didapatkan	<p>Skenario 1-2: Sistem berhasil menolak masukan data dari aktor.</p> <p>Skenario 3: Sistem berhasil menerima masukan data dari aktor.</p>
Kesimpulan	Diterima.

Tabel 8.11 Pengujian pada Proses Approval Pipeline

Nama	Approval Pipeline
Objek Uji	F_SIL_PBP_04
Skenario Pengujian	Aktor melakukan Pipeline pada debitur yang diinginkan.

Hasil yang Diharapkan	Sistem akan menyimpan data masukan dari aktor, mengganti status debitur menjadi Pipeline, dan mengirim data debitur ke SCO.
Hasil yang Didapatkan	Sistem berhasil menyimpan data masukan dari aktor, mengganti status debitur menjadi Pipeline, dan mengirim data debitur ke SCO.
Kesimpulan	Diterima.

Tabel 8.12 Pengujian pada Proses Recheck

Nama	Recheck
Objek Uji	F_SIL_PBP_05
Skenario Pengujian	Aktor melakukan permintaan untuk melakukan revisi data debitur kepada SCO. Kemudian aktor menuliskan catatan kepada debitur yang dipilih.
Hasil yang Diharapkan	Sistem akan menyimpan data masukan dari aktor, dan mengirim data debitur ke SCO.
Hasil yang Didapatkan	Sistem berhasil menyimpan data masukan dari aktor, mengganti status debitur menjadi Prospect, dan mengirim data debitur ke SCO.
Kesimpulan	Diterima.

Tabel 8.13 Pengujian pada Proses Reject (Prospect)

Nama	Reject
Objek Uji	F_SIL_PIA_05

Skenario Pengujian	Aktor membatalkan permohonan <i>lending</i> debitur pada tahap Prospect. Kemudian aktor menuliskan catatan kepada debitur yang dipilih.
Hasil yang Diharapkan	Sistem akan menyimpan data masukan dari aktor dan permohonan <i>lending</i> debitur telah dibatalkan.
Hasil yang Didapatkan	Sistem berhasil menyimpan data masukan dari aktor dan permohonan <i>lending</i> debitur telah dibatalkan.
Kesimpulan	Diterima.

Tabel 8.14 Pengujian pada Proses Reject (Pipeline)

Nama	Reject
Objek Uji	F_SIL_PBP_06
Skenario Pengujian	Aktor membatalkan permohonan <i>lending</i> debitur pada tahap Pipeline. Kemudian aktor menuliskan catatan kepada debitur yang dipilih.
Hasil yang Diharapkan	Sistem akan menyimpan data masukan dari aktor dan permohonan <i>lending</i> debitur telah dibatalkan.
Hasil yang Didapatkan	Sistem berhasil menyimpan data masukan dari aktor dan permohonan <i>lending</i> debitur telah dibatalkan.
Kesimpulan	Diterima.

Tabel 8.15 Pengujian pada Proses Melihat Refuse Data Lead

Nama	Melihat <i>Refuse Data Lead</i>
Objek Uji	F_SIL SCO_07
Skenario Pengujian	Aktor membuka halaman <i>refuse Data Lead</i> pada <i>side menu</i> .
Hasil yang Diharapkan	Sistem akan menampilkan daftar permohonan <i>lending debitur</i> yang dibatalkan.
Hasil yang Didapatkan	Sistem berhasil menampilkan daftar permohonan <i>lending debitur</i> yang dibatalkan.
Kesimpulan	Diterima.

Tabel 8.16 Pengujian pada Proses Approval Notification

Nama	<i>Approval Notification</i>
Objek Uji	F_SIL SCO_08
Skenario Pengujian	Jika Penyelia atau PBP merespon debitur yang telah <i>request</i> oleh aktor, maka sistem akan memberikan notifikasi kepada aktor.
Hasil yang Diharapkan	Sistem akan memberikan notifikasi kepada aktor
Hasil yang Didapatkan	Sistem berhasil memberikan notifikasi kepada aktor.
Kesimpulan	Diterima.

Tabel 8.17 Pengujian pada Proses *Request Notification*

Nama	<i>Request Notification</i>
Objek Uji	F_SIL_PIA_06, F_SIL_PBP_07
Skenario Pengujian	Jika SCO melakukan <i>request</i> debitur untuk diproses oleh aktor, maka sistem akan memberikan notifikasi kepada aktor.
Hasil yang Diharapkan	Sistem akan memberikan notifikasi kepada aktor
Hasil yang Didapatkan	Sistem berhasil memberikan notifikasi kepada aktor.
Kesimpulan	Diterima.

8.2 Hasil Pengujian Menggunakan *Compatibility Testing*

Compatibility Testing dilakukan dengan menggunakan perangkat lunak SortSite. Pengujian ini bertujuan untuk memastikan bahwa Sistem Informasi *Lending* dapat berjalan di beberapa *browser* yang berbeda dengan baik. *Browser* yang digunakan ditunjukkan pada Tabel 8.18.

Tabel 8.18 Browser untuk Compatibility Testing

No.	Nama <i>Browser</i>	Versi <i>Browser</i>
1	Internet Explorer	11
2	Edge	14
3	Firefox	76
4	Safari	13
5	Opera	68

No.	Nama Browser	Versi Browser
6	Chrome	81
7	iOS	<= 11, 12, dan 13
8	Android	<= 3 dan 4*

Compatibility Testing menggunakan perangkat lunak SortSite membagi masalah menjadi 3 jenis, yaitu *critical issues*, *major issues* dan *minor issues*. Hasil pengujian kompatibilitas ditunjukkan pada Gambar 8.1. Dari pengujian yang dilakukan dapat diketahui bahwa tidak terdapat *critical issues*. Terdapat sembilan kesalahan mayor dan delapan kesalahan minor pada sistem. Adanya kesalahan tersebut dikarenakan pada peramban tertentu tidak mendukung penggunaan tampilan sistem seperti format CSS, HTML, dan bootstrap. Hasil dari *compatibility testing* menyatakan sistem mampu berjalan dengan baik sesuai dengan prosedur uji yang sudah ditentukan pada berbagai jenis peramban.



Gambar 8.1 Hasil *Compatibility Testing*

8.3 Hasil Pengujian Menggunakan *Responsive Web Design (RWD) Testing*

Responsive Web Design Testing dilakukan dengan menggunakan Google Chrome DevTools. Pengujian ini bertujuan untuk memastikan bahwa Sistem Informasi *Lending* dapat memberikan *user interface* yang responsif terhadap perangkat yang digunakan. Perangkat yang digunakan ditunjukkan pada Tabel 8.19.

Tabel 8.19 Perangkat untuk *Responsive Web Design Testing*

No.	Jenis Perangkat	Resolusi
1	PC	1920x1080px

2	Tablet	768x1024px
---	--------	------------

Responsive Web Design Testing dilakukan dengan menguji tampilan yang tampak pada Google Chrome DevTools. Dari pengujian yang dilakukan dapat diketahui bahwa tidak terdapat *critical issues*. Hal tersebut menunjukkan bahwa Sistem Informasi *Lending* dapat berjalan dengan baik di semua resolusi yang diuji. Hasil pengujian *responsive web design* ditunjukkan pada Gambar Tabel 8.20 dan 8.21.

Tabel 8.20 Hasil Responsive Web Design Testing pada PC

No.	Halaman	Landscape	Portrait
1	<i>Dashboard</i>	✓	✓
2	<i>Input Data Lead</i>	✓	✓
3	<i>List Data Lead</i>	✓	✓
4	<i>List Refuse Data Lead</i>	✓	✓
5	<i>All Solicit</i>	✓	✓
6	<i>All Prospect</i>	✓	✓
7	<i>All Pipeline</i>	✓	✓
8	<i>Approval Prospect</i>	✓	✓
9	<i>Approval Pipeline</i>	✓	✓

Tabel 8.21 Hasil *Responsive Web Design Testing* pada Tablet

No.	Halaman	Landscape	Portrait
1	<i>Dashboard</i>	✓	✓

No.	Halaman	Landscape	Portrait
2	<i>Input Data Lead</i>	✓	✓
3	<i>List Data Lead</i>	✓	✓
4	<i>List Refuse Data Lead</i>	✓	✓
5	<i>All Solicit</i>	✓	✓
6	<i>All Prospect</i>	✓	✓
7	<i>All Pipeline</i>	✓	✓
8	<i>Approval Prospect</i>	✓	✓
9	<i>Approval Pipeline</i>	✓	✓

BAB 9 PENUTUP

9.1 Kesimpulan

Proses pengembangan Sistem Informasi *Lending* BROMO telah dilalui dengan melakukan analisis kebutuhan dan perancangan sistem, kemudian dilanjutkan dengan melakukan implementasi dan pengujian sistem. Berdasarkan kegiatan tersebut, dapat diambil kesimpulan sebagai berikut :

1. Pada proses analisis kebutuhan didapatkan dua kebutuhan sistem yang terdiri dari kebutuhan fungsional dan kebutuhan non fungsional. Kebutuhan fungsional adalah kebutuhan pokok dari sistem. Kebutuhan fungsional dibagi berdasarkan *role* atau *user* dalam sistem yang terdiri dari SCO, Penyelia, dan PBP yang masing-masing memiliki sebanyak 7 fungsi dan dijabarkan lebih detail dalam *Use Case Diagram* dan *Activity Diagram*. Kebutuhan non fungsional adalah kebutuhan yang diperlukan oleh sistem itu sendiri atau bisa dikatakan sebagai batasan layanan dari sistem dan terdiri dari 3 fungsi.
2. Pada proses perancangan sistem dilakukan tiga buah perancangan yang terdiri dari Pemodelan *Sequence Diagram*, Pemodelan PDM (*Physical Data Model*), dan Perancangan UI (*User Interface*). Pemodelan *Sequence Diagram* bertujuan untuk menjelaskan hubungan dan interaksi antar objek dengan *user* dalam sistem sehingga mempermudah *user* dalam memahami cara sistem bekerja. *Sequence Diagram* terdiri dari 6 urutan yaitu Input Data Lead, Solicit, Be Prospect, Approval Prospect, Be Pipeline, dan Approval Pipeline. Pemodelan PDM bertujuan untuk menjelaskan hubungan antar objek data dalam database yang meliputi 11 tabel proses *lending*, 3 tabel lokasi, dan 1 tabel notifikasi. Perancangan UI ditujukan sebagai acuan dalam membuat tampilan dari Sistem Informasi *Lending* BROMO yang terdiri dari login, dashboard, input Data Lead, dan approval.
3. Pada proses implementasi diterapkan sistem yang telah dirancang, yaitu implementasi *user interface*, implementasi *database*, dan implementasi kode program. Implementasi *user interface* dilakukan dengan memperhatikan kebutuhan user dalam sistem yang berbeda-beda antara lain SCO, Penyelia, dan PBP. Implementasi *database* dilakukan dengan menggunakan MySQL dan phpMyAdmin yang terdiri atas 16 tabel yang dibutuhkan dalam sistem. Implementasi kode program dilakukan dengan menggunakan bahasa pemrograman PHP yang terdiri atas fungsi utama antara lain *method postLogin*, *method inputManual*, *method create*, *method createSolicit*, *method beProspect*, *method approveProspect*, *method submitProspectToPbp*, dan *method approvePipeline*.

4. Pada proses pengujian dilakukan metode *Black Box Testing* yang menguji validitas dari Sistem Informasi *Lending* BROMO dengan spesifikasi yang telah ditentukan tanpa memperhatikan internal sistem. Hasil dari pengujian *Black Box Testing* didapatkan bahwa setiap fungsionalitas sistem telah terpenuhi, yakni Proses *Login*, Proses *Input Data Lead*, Proses *Solicit*, Proses *Change Solicit*, Proses *Be Prospect*, Proses *Reject Prospect*, Proses *Approve Prospect*, Proses *Be Pipeline*, Proses *Re-Check Pipeline*, Proses *Reject Pipeline*, dan Proses *Approve Pipeline*. Sedangkan untuk hasil pengujian *Black Box Testing* dibantu dengan *tools SortSite* dan *Google Chrome DevTools* didapatkan bahwa sistem telah memenuhi kriteria *compatibility* dan *responsive web design* (RWD).

9.2 Saran

Berdasarkan perancangan Sistem Informasi *Lending* berbasis *web* yang telah dilakukan bersama dengan pihak BNI Kantor Wilayah Malang, peneliti ingin memberikan saran yang dapat digunakan sebagai acuan penelitian selanjutnya :

1. Hasil perancangan dan implementasi Sistem Informasi *Lending* berbasis *web* diharapkan kedepannya dapat digunakan oleh pihak BNI Kantor Wilayah Malang untuk dilakukan pengembangan lebih lanjut.
2. Untuk pihak BNI Kantor Wilayah Malang agar lebih mematuhi model pengembangan sistem yang telah ditentukan sehingga sistem dapat lebih siap untuk digunakan.

DAFTAR PUSTAKA

- Ali Ridho Barakbah, Tita Karlita, Ahmad Syauqi Ahsan. 2013. Politeknik Elektronika Negeri Surabaya. *Logika dan Algoritma*, pp.19-23. Surabaya: Program Studi Teknik Informatika Politeknik Elektronika Negeri Surabaya
- Aminudin. 2015. *Cara Efektif Belajar Framework Laravel*. Yogyakarta: CV. LOKOMEDIA .
- Arief M Rudianto. 2011. *Pemrograman Web Dinamis menggunakan PHP dan MySQL*. Yogyakarta: CV ANDI OFFSET.
- Bentley, W., Whitten, J., & Bentley, L. 2007. *Systems Analysis & Design Methods* (9nd ed.). New York: Whitten dan Bentley.
- Binarso, Yudi Arsi, Eko Adi Sarwoko, Nurdin Bahtiar. 2012. *Pembangunan Sistem Informasi Alumni Berbasis Web Pada Program Studi Teknik Informatika Universitas Diponegoro*. Journal of Informatics and Technology, Vol 1, No 1, pp. 76. Tersedia melalui: Perpustakaan Universitas Diponegoro <<http://ejournal-s1.undip.ac.id/index.php/joint>> [Diakses 22 April 2020]
- Eugene. 2019. "Responsive Web Design Testing: Tips, Tools and Checklist". Qawerk, [www.qawerk.com/blogs/responsive-web-design-testing-tips-tools-and-ch ecklist](http://www.qawerk.com/blogs/responsive-web-design-testing-tips-tools-and-checklist). Diakses pada 11 Juni. 2020
- Google Developers. 2017. "Chrome DevTools". <https://developers.google.com/web/tools/chrome-devtools/>. Diakses pada 10 Juni. 2020
- Guru99. 2013. "What is PHP? Write your first PHP Program". <https://www.guru99.com/what-is-php-first-php-program.html>. Diakses pada 4 Mei. 2020
- Gustino, Dimas. 2018. "WHEN TO USE WATERFALL?". <https://sis.binus.ac.id/2018/04/26/when-to-use-waterfall/>. Diakses pada 28 Mei. 2020
- Handika, I. Gede, Ayi Purbasari. 2018. *Pemanfaatan Framework Laravel Dalam Pembangunan Aplikasi E-Travel Berbasis Website*. Konferensi Nasional Sistem Informasi STMIK Atma Luhur Pangkalpinang, pp. 1332-1333. Tersedia melalui: <<http://jurnal.atmaluhur.ac.id/index.php/knsi2018/article/view/533>>.
- Hass, Anne Mette. 2014. *Guide to Advanced Software Testing, Second Edition*. Norwood: Artech House, Inc.

- Kumar, Naresh, A. S. Zadgaonkar, Abhinav Shukla. 2013. *Evolving a New Software Development Life Cycle Model SDLC-2013 with Client Satisfaction*. *International Journal of Soft Computing and Engineering*.
- Nugroho, Adi. 2004. *Analisis dan Perancangan Informasi dengan Metodologi Berorientasi Objek*. Bandung: Informatika.
- Nugroho, Adi. 2006. *E-commerce Memahami Perdagangan Modern Di Dunia Maya*. Bandung: Informatika.
- Nugroho, Adi. 2010. *Rekayasa Perangkat Lunak Berbasis Objek dengan Metode USDP*. Yogyakarta: Penerbit Andi.
- O'Brien, J. A., & Marakas, G. M. 2007. *Introduction to information systems*. Boston: McGraw-Hill/Irwin.
- PowerMapper. 2020. SortSite (5.22.1766). [Program Komputer]. Tersedia di: <<http://www.powermapper.com/products/sortsite/>> [Diakses 10 Juni 2020].
- Prokofyeva, N., Victoria Boltunova. 2016. *Analysis and Practical Application of PHP Frameworks in Development of Web Information Systems*. Procedia Computer Science, Vol 104, pp. 51–52. Tersedia melalui: www.sciencedirect.com
- Rosa, & Shalahuddin. 2015. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Bandung: Informatika
- Rosmala, D., Muhammad Ichwan, M. Irzan Gandalisha. 2011. *Komparasi Framework Mvc(Codeigniter, Dan Cakephp) Pada Aplikasi Berbasis Web*. Jurnal Informatika, pp. 24
- Sagala, A., & Entik, I. 2014. Sistem Multimedia Telkom University. *Perancangan Aplikasi Berbasis Web Interaktif Haloapp Berbasis Android dan Ios*, pp. 20-22.
- Sommerville, Ian. 2011. *Software Engineering. 9th Edition*. Boston: Pearson Education, Inc.
- Sukamto, R., & Shalahuddin, M. 2013. *Rekayasa Perangkat Lunak*. Bandung: Informatika.
- Sukamto, Rosa Ariani. 2009. *Langkah-langkah Pengujian Perangkat dan Evaluasi Piranti Lunak*. Bandung: Informatika.

Zakir, Ahmad. 2016. *Rancang Bangun Responsive Web Layout Dengan Menggunakan Bootstrap Framework*. InfoTekJar (Jurnal Nasional Informatika Dan Teknologi Jaringan), Vol 1, No 1, pp 7. Tersedia melalui: <<https://jurnal.uisu.ac.id/index.php/infotekjar/article/view/31>>

DAFTAR LAMPIRAN

Lampiran A : Foto Kegiatan



Gambar Lampiran 1 Foto kegiatan bersama BNI Wilayah Malang



Gambar Lampiran 2 Foto kegiatan bersama BNI Wilayah Malang

Lampiran B : Daftar Kehadiran

PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

DAFTAR HADIR						
Nama	: Muhammad Kevin Sandryan					
Jabatan	: Pegawai Magang					
Unit Organisasi	: PDB 2					
No	Hari	Tanggal	Datang		Pulang	
			Jam	Paraf	Jam	Paraf
1	Selasa	28/01/2020	08.00	✓	12.00	✓
2	Rabu	29/01/2020	10.00	✓	12.00	✓
3	Kamis	30/01/2020	16.00	✓	17.00	✓
4	Jumat	31/01/2020	07.30	✓	17.00	✓
5	Senin	03/02/2020	16.00	✓	17.00	✓
6	Selasa	04/02/2020	08.00	✓	12.00	✓
7	Selasa	04/02/2020	16.00	✓	17.00	✓
8	Rabu	05/02/2020	10.00	✓	12.00	✓
9	Rabu	05/02/2020	16.00	✓	17.00	✓
10	Kamis	06/02/2020	16.00	✓	17.00	✓
11	Jumat	07/02/2020	07.30	✓	17.00	✓
12	Senin	10/02/2020	16.00	✓	17.00	✓
13	Selasa	11/02/2020	08.00	✓	12.00	✓
14	Selasa	11/02/2020	16.00	✓	17.00	✓
15	Rabu	12/02/2020	10.00	✓	12.00	✓
16	Rabu	12/02/2020	16.00	✓	17.00	✓
17	Kamis	13/02/2020	16.00	✓	17.00	✓
18	Jumat	14/02/2020	07.30	✓	17.00	✓
19	Senin	17/02/2020	16.00	✓	17.00	✓
20	Selasa	18/02/2020	08.00	✓	12.00	✓
21	Selasa	18/02/2020	16.00	✓	17.00	✓
22	Rabu	19/02/2020	10.00	✓	12.00	✓
23	Rabu	19/02/2020	16.00	✓	17.00	✓
24	Kamis	20/02/2020	16.00	✓	17.00	✓
25	Jumat	21/02/2020	07.30	✓	17.00	✓

Mengatahui,
PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG


Dapat Pengesahan / P047 349.

Gambar Lampiran 3 Daftar Kehadiran Muhammad Kevin Sandryan

PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

DAFTAR HADIR						
Nama	: Muhammad Kevin Sandryan					
Jabatan	: Pegawai Magang					
Unit Organisasi	: PDB 2					
No	Hari	Tanggal	Datang		Pulang	
			Jam	Paraf	Jam	Paraf
1	Senin	24/02/2020	16.00	✓	17.00	✓
2	Selasa	25/02/2020	08.00	✓	12.00	✓
3	Selasa	25/02/2020	16.00	✓	17.00	✓
4	Rabu	26/02/2020	10.00	✓	12.00	✓
5	Rabu	26/02/2020	16.00	✓	17.00	✓
6	Kamis	27/02/2020	16.00	✓	17.00	✓
7	Jumat	28/02/2020	07.30	✓	17.00	✓
8	Senin	02/03/2020	16.00	✓	17.00	✓
9	Selasa	03/03/2020	08.00	✓	12.00	✓
10	Selasa	03/03/2020	16.00	✓	17.00	✓
11	Rabu	04/03/2020	10.00	✓	12.00	✓
12	Rabu	04/03/2020	16.00	✓	17.00	✓
13	Kamis	05/03/2020	16.00	✓	17.00	✓
14	Jumat	06/03/2020	07.30	✓	17.00	✓
15	Senin	09/03/2020	16.00	✓	17.00	✓
16	Selasa	10/03/2020	08.00	✓	12.00	✓
17	Selasa	10/03/2020	16.00	✓	17.00	✓
18	Rabu	11/03/2020	10.00	✓	12.00	✓
19	Rabu	11/03/2020	16.00	✓	17.00	✓
20	Kamis	12/03/2020	16.00	✓	17.00	✓
21	Jumat	13/03/2020	07.30	✓	17.00	✓
22	Senin	16/03/2020	16.00	✓	17.00	✓
23	Selasa	17/03/2020	08.00	✓	12.00	✓
24	Selasa	17/03/2020	16.00	✓	17.00	✓
25	Rabu	18/03/2020	10.00	✓	12.00	✓

Mengatahui,
PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

Gambar Lampiran 4 Daftar Kehadiran Muhammad Kevin Sandryan

PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

DAFTAR HADIR								
No	Hari	Tanggal	Datang		Pulang		Paraf	Paraf
			Jam	Paraf	Jam	Paraf		
1	Selasa	28/01/2020	08.00	✓	12.00	✓		
2	Rabu	29/01/2020	10.00	✓	12.00	✓		
3	Kamis	30/01/2020	16.00	✓	17.00	✓		
4	Jumat	31/01/2020	07.30	✓	17.00	✓		
5	Senin	03/02/2020	10.00	✓	17.00	✓		
6	Selasa	04/02/2020	08.00	✓	12.00	✓		
7	Rabu	04/02/2020	16.00	✓	17.00	✓		
8	Rabu	05/02/2020	10.00	✓	12.00	✓		
9	Rabu	05/02/2020	16.00	✓	17.00	✓		
10	Kamis	06/02/2020	16.00	✓	17.00	✓		
11	Jumat	07/02/2020	07.30	✓	17.00	✓		
12	Senin	10/02/2020	16.00	✓	17.00	✓		
13	Selasa	11/02/2020	08.00	✓	12.00	✓		
14	Selasa	11/02/2020	16.00	✓	17.00	✓		
15	Rabu	12/02/2020	10.00	✓	12.00	✓		
16	Rabu	12/02/2020	16.00	✓	17.00	✓		
17	Kamis	13/02/2020	16.00	✓	17.00	✓		
18	Jumat	14/02/2020	07.30	✓	17.00	✓		
19	Senin	17/02/2020	16.00	✓	17.00	✓		
20	Selasa	18/02/2020	08.00	✓	12.00	✓		
21	Selasa	18/02/2020	16.00	✓	17.00	✓		
22	Rabu	19/02/2020	10.00	✓	12.00	✓		
23	Rabu	19/02/2020	16.00	✓	17.00	✓		
24	Kamis	20/02/2020	16.00	✓	17.00	✓		
25	Jumat	21/02/2020	07.30	✓	17.00	✓		

Mengelalui,
PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG


Dyah Pengaristi / P047349

Gambar Lampiran 5 Daftar Kehadiran Muhammad Rasyid Al Faruqi

PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

DAFTAR HADIR								
No	Hari	Tanggal	Datang		Pulang		Paraf	Paraf
			Jam	Paraf	Jam	Paraf		
1	Senin	24/02/2020	16.00	✓	17.00	✓		
2	Selasa	25/02/2020	08.00	✓	12.00	✓		
3	Selasa	25/02/2020	16.00	✓	17.00	✓		
4	Rabu	26/02/2020	10.00	✓	12.00	✓		
5	Rabu	26/02/2020	16.00	✓	17.00	✓		
6	Kamis	27/02/2020	16.00	✓	17.00	✓		
7	Jumat	28/02/2020	07.30	✓	17.00	✓		
8	Senin	02/03/2020	16.00	✓	17.00	✓		
9	Selasa	03/03/2020	08.00	✓	12.00	✓		
10	Selasa	03/03/2020	16.00	✓	17.00	✓		
11	Rabu	04/03/2020	10.00	✓	12.00	✓		
12	Rabu	04/03/2020	16.00	✓	17.00	✓		
13	Kamis	05/03/2020	16.00	✓	17.00	✓		
14	Jumat	06/03/2020	07.30	✓	17.00	✓		
15	Senin	09/03/2020	16.00	✓	17.00	✓		
16	Selasa	10/03/2020	08.00	✓	12.00	✓		
17	Selasa	10/03/2020	16.00	✓	17.00	✓		
18	Rabu	11/03/2020	10.00	✓	12.00	✓		
19	Rabu	11/03/2020	16.00	✓	17.00	✓		
20	Kamis	12/03/2020	16.00	✓	17.00	✓		
21	Jumat	13/03/2020	07.30	✓	17.00	✓		
22	Senin	16/03/2020	16.00	✓	17.00	✓		
23	Selasa	17/03/2020	08.00	✓	12.00	✓		
24	Selasa	17/03/2020	16.00	✓	17.00	✓		
25	Rabu	18/03/2020	10.00	✓	12.00	✓		

Mengelalui,
PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

Gambar Lampiran 6 Daftar Kehadiran Muhammad Rasyid Al Faruqi

PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

DAFTAR HADIR

No	Hari	Tanggal	Datang		Pulang	
			Jam	Paraf	Jam	Paraf
1	Selasa	28/01/2020	06.00	✓	12.00	✓
2	Rabu	29/01/2020	10.00	✓	12.00	✓
3	Kamis	30/01/2020	16.00	✓	17.00	✓
4	Jumat	31/01/2020	07.30	✓	17.00	✓
5	Senin	03/02/2020	16.00	✓	17.00	✓
6	Selasa	04/02/2020	08.00	✓	12.00	✓
7	Selasa	04/02/2020	16.00	✓	17.00	✓
8	Rabu	05/02/2020	10.00	✓	12.00	✓
9	Rabu	05/02/2020	16.00	✓	17.00	✓
10	Kamis	06/02/2020	16.00	✓	17.00	✓
11	Jumat	07/02/2020	07.30	✓	17.00	✓
12	Senin	10/02/2020	16.00	✓	17.00	✓
13	Selasa	11/02/2020	08.00	✓	12.00	✓
14	Selasa	11/02/2020	16.00	✓	17.00	✓
15	Rabu	12/02/2020	10.00	✓	12.00	✓
16	Rabu	12/02/2020	16.00	✓	17.00	✓
17	Kamis	13/02/2020	16.00	✓	17.00	✓
18	Jumat	14/02/2020	07.30	✓	17.00	✓
19	Senin	17/02/2020	16.00	✓	17.00	✓
20	Selasa	18/02/2020	08.00	✓	12.00	✓
21	Selasa	18/02/2020	16.00	✓	17.00	✓
22	Rabu	19/02/2020	10.00	✓	12.00	✓
23	Rabu	19/02/2020	16.00	✓	17.00	✓
24	Kamis	20/02/2020	16.00	✓	17.00	✓
25	Jumat	21/02/2020	07.30	✓	17.00	✓

Mengetahui,
PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

[Signature]
Dian Pangestuti /P047349

Gambar Lampiran 7 Daftar Kehadiran Mohammad Raska

PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

DAFTAR HADIR

No	Hari	Tanggal	Datang		Pulang	
			Jam	Paraf	Jam	Paraf
1	Senin	24/02/2020	16.00	✓	17.00	✓
2	Selasa	25/02/2020	08.00	✓	12.00	✓
3	Selasa	25/02/2020	16.00	✓	17.00	✓
4	Rabu	26/02/2020	10.00	✓	12.00	✓
5	Rabu	26/02/2020	16.00	✓	17.00	✓
6	Kamis	27/02/2020	16.00	✓	17.00	✓
7	Jumat	28/02/2020	07.30	✓	17.00	✓
8	Senin	02/03/2020	16.00	✓	17.00	✓
9	Selasa	03/03/2020	08.00	✓	12.00	✓
10	Selasa	03/03/2020	16.00	✓	17.00	✓
11	Rabu	04/03/2020	10.00	✓	12.00	✓
12	Rabu	04/03/2020	16.00	✓	17.00	✓
13	Kamis	05/03/2020	16.00	✓	17.00	✓
14	Jumat	06/03/2020	07.30	✓	17.00	✓
15	Senin	09/03/2020	16.00	✓	17.00	✓
16	Selasa	10/03/2020	08.00	✓	12.00	✓
17	Selasa	10/03/2020	16.00	✓	17.00	✓
18	Rabu	11/03/2020	10.00	✓	12.00	✓
19	Rabu	11/03/2020	16.00	✓	17.00	✓
20	Kamis	12/03/2020	16.00	✓	17.00	✓
21	Jumat	13/03/2020	07.30	✓	17.00	✓
22	Senin	16/03/2020	16.00	✓	17.00	✓
23	Selasa	17/03/2020	08.00	✓	12.00	✓
24	Selasa	17/03/2020	16.00	✓	17.00	✓
25	Rabu	18/03/2020	10.00	✓	12.00	✓

Mengetahui,
PT. BANK NEGARA INDONESIA (Persero) Tbk.
KANTOR WILAYAH MALANG

Gambar Lampiran 8 Daftar Kehadiran Mohammad Raska

