

**PENGEMBANGAN SISTEM MANAJEMEN PROYEK
PERANGKAT LUNAK MENGGUNAKAN METODE KANBAN
BERBASIS WEB STUDI KASUS CV. PRIMAVISI GLOBALINDO
SURABAYA**

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Muhammad Aliyya Ilmi
NIM: 165150200111050



PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2020

PENGESAHAN

PENGEMBANGAN SISTEM MANAJEMEN PROYEK PERANGKAT LUNAK
MENGGUNAKAN METODE KANBAN BERBASIS WEB STUDI KASUS CV. PRIMAVISI
GLOBALINDO SURABAYA

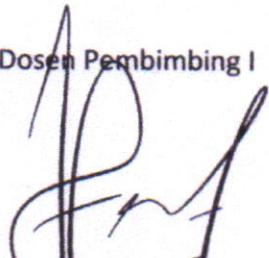
SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Muhammad Aliyya Ilmi
NIM: 165150200111050

Skripsi ini telah diuji dan dinyatakan lulus pada
4 Mei 2020
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I


Fajar Pradana, S.ST., M.Eng.
NIP. 19871121 201504 1 004

Dosen Pembimbing II


Widhy Hayuhardhika Nugraha Putra,
S.Kom., M.Kom.
NIK: 2017128704092001

Mengetahui

Ketua Jurusan Teknik Informatika




Tri Astoro Kurniawan, S.T., M.T., Ph.D.
NIP. 19710518 200312 1 001

PERNYATAAN ORISINALITAS

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

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

Malang, 12 Mei 2020



Muhammad Aliyya Ilmi

NIM: 165150200111050

PRAKATA

Puji syukur penulis kepada Allah SWT karena hanya dengan ridha-Nya skripsi ini dapat terselesaikan. Penulis menyampaikan rasa terima kasih penulis kepada berbagai pihak yang telah membantu penyelesaian skripsi ini. Karena itu penulis ingin berterima kasih pada:

1. Fajar Pradana, S.ST., M.Eng. dan Widhy Hayuhardhika Nugraha Putra, S.Kom., M.Kom. selaku pembimbing I dan pembimbing II yang telah membimbing penulis hingga selesai.
2. Orang tua dan seluruh keluarga penulis, yang telah memberikan dukungan penuh selama penggerjaan hingga selesai.
3. Wayan Firdaus Mahmudy , S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
4. Tri Astoto Kurniawan , S.T, M.T, Ph.D selaku ketua jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
5. Keluarga besar Kelas E Informatika 2016, yang telah memberikan dukungan dan informasi-informasi penting dalam penelitian.
6. Sahabat BIOS dan teater Cowboy yang senantiasa memberikan dorongan, dukungan dan semangat, Riski, Ma'ruf, Rizaldi, Mas Fardhan, Mas Fabi, Mas Ubaid dan yang lainnya yang tidak bisa disebutkan satu-persatu.

Penulis sangat menyadari bahwa selama penggerjaan skripsi ini banyak kekurangan di dalamnya sehingga penulis sangat berterima kasih apabila ada kritik dan saran yang bersifat membangun. Penulis berharap skripsi ini dapat berguna untuk pembaca dan penelitian selanjutnya.

Malang, 12 Mei 2020

Muhammad Aliyya Ilmi
muhammadaliyya19@gmail.com

ABSTRAK

Muhammad Aliyya Ilmi, PENGEMBANGAN SISTEM MANAJEMEN PROYEK PERANGKAT LUNAK MENGGUNAKAN METODE KANBAN BERBASIS WEB STUDI KASUS CV. PRIMAVISI GLOBALINDO SURABAYA

Pembimbing: Fajar Pradana, S.ST., M.Eng. dan Widhy Hayuhardhika Nugraha Putra, S.Kom., M.Kom.

Manajemen proyek adalah aspek yang sangat penting bagi perusahaan untuk mengurangi risiko kegagalan proyek yang dikerjakan. Salah satu aspek dalam manajemen proyek yang penting adalah masalah pembagian dan penjadwalan tugas pelaksana proyek. Mengembangkan proyek perangkat lunak atau sistem informasi bukanlah hal yang mudah, tercatat pada tahun 2018 dari 110 proyek sistem informasi yang dikerjakan di sejumlah kota di Indonesia, 27% diantaranya selesai dengan anggaran yang sesuai, tidak melebihi batas waktu dan telah dinilai oleh pengguna, 55% mengalami masalah, dan 18% sisanya dibatalkan. Salah satu penyebab tidak selesaikan proyek adalah buruknya pembagian dan penjadwalan proyek tersebut. Untuk mengatasi buruknya pembagian dan penjadwalan tugas pelaksana proyek adalah menggunakan metode Kanban. Penelitian ini bertujuan untuk mengembangkan sistem penjadwalan proyek yang menerapkan metode Kanban dengan menggunakan SDLC Waterfall, framework Codeigniter, template SB-Admin-2 dari Bootstrap, dan juga menggunakan Javascript. Tahapan dalam pengembangan ini yaitu analisis kebutuhan, perancangan, implementasi, dan pengujian. Hasil pengujian yang dilakukan terhadap sistem didapatkan hasil validitas 100% dari 3 pengujian unit yaitu whitebox testing pada method tambah() pada controller Project, Board dan Task. Untuk pengujian validasi, dilakukan blackbox testing terhadap 49 fungsional dan didapat nilai validasi sebesar 100%. Nilai pengujian usabilitas sebesar 76. Sehingga dengan hasil pengujian tersebut sistem dikategorikan dapat diterima.

Kata kunci: manajemen proyek, kanban, waterfall, codeigniter, aplikasi website

ABSTRACT

Muhammad Aliyya Ilmi, DEVELOPMENT OF WEB-BASED SOFTWARE PROJECT MANAGEMENT SYSTEM USING KANBAN METHOD CASE STUDY CV. PRIMAVISI GLOBALINDO SURABAYA

Supervisors: Fajar Pradana, S.ST., M.Eng. and Widhy Hayuhardhika Nugraha Putra, S.Kom., M.Kom.

Project management is a very important aspect for the company to reduce the risk of failure of the project being undertaken. One important aspect of project management is the problem of the division and scheduling of project tasks for project executor. Developing software projects or information systems is not easy, recorded in 2018 of 110 information system projects in a number of cities in Indonesia, 27% of which were completed on budget, on time and have been evaluated by users, 55% were problematic, and 18% the rest is canceled. One reason for not completing the project is the poor division and scheduling of the project. To overcome the poor division and scheduling of project implementation tasks is to use the Kanban method. This study aims to develop a project scheduling system that applies the Kanban method using SDLC Waterfall, Codeigniter framework, SB-Admin-2 templates from Bootstrap, and also using Javascript. The stages in this development are the analysis of needs, design, implementation, and testing. The results of testing conducted on the system obtained 100% validity results from 3 unit tests with whitebox testing on the add () method on the Project, Board and Task controller. For validation testing, blackbox testing was performed on 49 functional and obtained a validation value of 100%. The usability test value is 76. So with the test results the system is categorized as acceptable.

DAFTAR ISI

PENGESAHAN	ii
PERNYATAAN ORISINALITAS	iii
PRAKATA.....	iv
ABSTRAK.....	v
ABSTRACT	vi
DAFTAR ISI.....	vii
DAFTAR TABEL.....	xii
DAFTAR GAMBAR.....	xvii
DAFTAR LAMPIRAN	xviii
BAB 1 PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan	3
1.4 Manfaat.....	3
1.5 Batasan Masalah.....	3
1.6 Sistematika Pembahasan.....	3
BAB 2 LANDASAN KEPUSTAKAAN	5
2.1 Kajian Pustaka.....	5
2.2 Rekayasa Perangkat Lunak	7
2.3 Pengembangan Perangkat Lunak	7
2.3.1 Model Pengembangan Perangkat Lunak	8
2.3.2 Pemodelan <i>Unified Modelling Language (UML)</i>	10
2.3.3 Pendekatan <i>Object Oriented</i>	11
2.4 Proyek	15
2.4.1 Definisi Proyek.....	15
2.4.2 Proyek Perangkat Lunak.....	15
2.4.3 <i>Stakeholder</i>	16
2.5 Manajemen Proyek.....	18
2.5.1 Manajemen Proyek Perangkat Lunak	18
2.6 <i>Kanban</i>	19
2.6.1 Definisi <i>Kanban</i>	19

2.6.2 Prinsip Kerja <i>Kanban</i> dan Definisinya	20
2.6.3 Artefak <i>Kanban</i>	21
2.7 Konsep MVC.....	22
2.7.1 <i>Model</i>	22
2.7.2 <i>View</i>	23
2.7.3 <i>Controller</i>	23
2.8 Teknologi Pengembangan Sistem.....	23
2.8.1 Codeigniter <i>Framework</i>	23
2.8.2 Bootstrap <i>Framework</i>	23
2.8.3 SB Admin 2 <i>Framework</i>	23
2.8.4 Javascript.....	24
2.8.5 Google Charts.....	24
2.9 Pengujian Perangkat Lunak.....	24
2.9.1 White-box <i>Testing</i>	24
2.9.2 Black-box <i>Testing</i>	24
2.9.3 <i>Usability Testing</i>	25
BAB 3 METODOLOGI	26
3.1 Studi Literatur	27
3.2 Rekayasa Kebutuhan.....	27
3.3 Perancangan Sistem.....	28
3.4 Implementasi	28
3.5 Pengujian dan Analisis	29
3.6 Kesimpulan dan Saran	29
BAB 4 REKAYASA KEBUTUHAN.....	30
4.1 Gambaran Umum Sistem.....	30
4.2 Identifikasi Pengguna.....	30
4.3 Daftar Kebutuhan Fungsional	31
4.3.1 Teori Kode Kebutuhan	31
4.3.2 Kebutuhan Fungsional.....	32
4.3.3 Spesifikasi Kebutuhan	34
4.4 Analisis Data.....	41
4.5 Daftar Kebutuhan Non Fungsional	42

4.6 Pemodelan Kebutuhan	43
4.6.1 <i>Use Case Diagram</i>	43
4.6.2 <i>Use Case Scenario</i>	44
BAB 5 PERANCANGAN DAN IMPLEMENTASI SISTEM	67
5.1 Perancangan Sistem.....	67
5.1.1 Perancangan Arsitektur.....	67
5.1.2 Perancangan Komponen	72
5.1.3 Perancangan Data	74
5.1.4 Perancangan Antarmuka.....	75
5.2 Implementasi Sistem	78
5.2.1 Spesifikasi Sistem	78
5.2.2 Implementasi Data	79
5.2.3 Implementasi Kode Program	82
5.2.4 Implementasi Antarmuka	86
BAB 6 PENGUJIAN	88
6.1 White-box <i>Testing</i>	88
6.1.1 White-box <i>Testing Method tambah() pada Controller Project</i>	88
6.1.2 White-box <i>Testing Method tambah() pada Controller Board</i>	92
6.1.3 White-box <i>Testing Method tambah() pada Controller Task</i>	94
6.2 Black-box <i>Testing</i>	98
6.2.1 Pengujian Validasi <i>Login</i>	98
6.2.2 Pengujian Validasi <i>Lupa Password</i>	101
6.2.3 Pengujian Validasi <i>Logout</i>	102
6.2.4 Pengujian Validasi Melihat <i>Dashboard</i>	103
6.2.5 Pengujian Validasi Mencari Proyek.....	103
6.2.6 Pengujian Validasi Menambah Data Proyek	104
6.2.7 Pengujian Validasi Melihat Proyek.....	106
6.2.8 Pengujian Validasi Mengubah Data Proyek	106
6.2.9 Pengujian Validasi Menghapus Data Proyek.....	107
6.2.10 Pengujian Validasi Melihat Detail Proyek	107

6.2.11 Pengujian Validasi Menambah Tugas	107
6.2.12 Pengujian Validasi Melihat Daftar Tugas	111
6.2.13 Pengujian Validasi Mengupdate Tugas	111
6.2.14 Pengujian Validasi Menghapus Data Tugas	111
6.2.15 Pengujian Validasi Melihat Detail Tugas	112
6.2.16 Pengujian Validasi Menambah Feedback Tugas	112
6.2.17 Pengujian Validasi Melihat <i>Feedback</i> Tugas	113
6.2.18 Pengujian Validasi Menghapus <i>Feedback</i> Tugas.....	113
6.2.19 Pengujian Validasi Menambah <i>Board</i>	114
6.2.20 Pengujian Validasi Melihat <i>Kanban Board</i>	115
6.2.21 Pengujian Validasi Mengubah Limit.....	115
6.2.22 Pengujian Validasi Menghapus <i>Board</i>	116
6.2.23 Pengujian Validasi Melihat <i>Gantt Chart</i> Proyek.....	116
6.2.24 Pengujian Validasi Melihat <i>Pie Chart</i> Proyek	117
6.2.25 Pengujian Validasi Mengupload Dokumen	117
6.2.26 Pengujian Validasi Melihat Dokumen Proyek.....	119
6.2.27 Pengujian Validasi Membaca Dokumen Proyek	119
6.2.28 Pengujian Validasi Menghapus Dokumen	120
6.2.29 Pengujian Validasi <i>Download</i> Dokumen	120
6.2.30 Pengujian Validasi Melihat Tim Proyek.....	121
6.2.31 Pengujian Validasi Menambah Data <i>Client</i>	121
6.2.32 Pengujian Validasi Melihat <i>Client</i>	122
6.2.33 Pengujian Validasi Mengubah Data <i>Client</i>	122
6.2.34 Pengujian Validasi Menghapus Data <i>Client</i>	123
6.2.35 Pengujian Validasi Menambah Data PIC.....	123
6.2.36 Pengujian Validasi Melihat PIC.....	124
6.2.37 Pengujian Validasi Mengubah Data PIC.....	124
6.2.38 Pengujian Validasi Menghapus Data PIC.....	125
6.2.39 Pengujian Validasi Menambah Data Pegawai.....	125
6.2.40 Pengujian Validasi Melihat Data Pegawai.....	126
6.2.41 Pengujian Validasi Mengubah Data Pegawai.....	126
6.2.42 Pengujian Validasi Menghapus Data Pegawai	127

6.2.43 Pengujian Validasi Melihat Detail Data Pegawai	128
6.2.44 Pengujian Validasi Menambah Data Role Pegawai.....	128
6.2.45 Pengujian Validasi Melihat Data Role Pegawai.....	129
6.2.46 Pengujian Validasi Menghapus Data Role Pegawai	129
6.2.47 Pengujian Validasi Melihat Profil	129
6.2.48 Pengujian Validasi Mengubah Data Profil.....	130
6.2.49 Pengujian Validasi Mengubah Password	130
6.3 <i>Usability Testing</i>	132
6.3.1 Prosedur Pengujian <i>Usability</i>	132
6.3.2 Analisis dan Hasil Pengujian <i>Usability</i>	133
BAB 7 PENUTUP	135
7.1 Kesimpulan.....	135
7.2 Saran	135
DAFTAR REFERENSI	136
LAMPIRAN	138

DAFTAR TABEL

Tabel 2.1 Notasi <i>Use Case Diagram</i>	11
Tabel 2.2 Komponen <i>Use Case Scenario</i>	12
Tabel 2.3 Notasi <i>Sequence Diagram</i>	13
Tabel 2.4 Notasi <i>Class Diagram</i>	14
Tabel 4.1 Identifikasi Aktor	31
Tabel 4.2 Kebutuhan Fungsional Sistem	32
Tabel 4.3 Spesifikasi Kebutuhan Fungsional	34
Tabel 4.4 Kebutuhan Non Fungsional	42
Tabel 4.5 <i>Use Case Scenario Login</i>	44
Tabel 4.6 <i>Use Case Scenario Lupa Password</i>	45
Tabel 4.7 <i>Use Case Scenario Logout</i>	45
Tabel 4.8 <i>Use Case Scenario Melihat Dashboard</i>	46
Tabel 4.9 <i>Use Case Scenario Mencari Proyek</i>	46
Tabel 4.10 <i>Use Case Scenario Menambah Data Proyek</i>	46
Tabel 4.11 <i>Use Case Scenario Melihat Proyek</i>	47
Tabel 4.12 <i>Use Case Scenario Mengubah Data Proyek</i>	48
Tabel 4.13 <i>Use Case Scenario Menghapus Data Proyek</i>	48
Tabel 4.14 <i>Use Case Scenario Melihat Detail Proyek</i>	49
Tabel 4.15 <i>Use Case Scenario Menambah Tugas</i>	49
Tabel 4.16 <i>Use Case Scenario Melihat Daftar Tugas</i>	50
Tabel 4.17 <i>Use Case Scenario Mengupdate Tugas</i>	50
Tabel 4.18 <i>Use Case Scenario Menghapus Data Tugas</i>	51
Tabel 4.19 <i>Use Case Scenario Melihat Detail Tugas</i>	51
Tabel 4.20 <i>Use Case Scenario Menambah Feedback Tugas</i>	51
Tabel 4.21 <i>Use Case Scenario Melihat Feedback Tugas</i>	52
Tabel 4.22 <i>Use Case Scenario Menghapus Feedback Tugas</i>	52
Tabel 4.23 <i>Use Case Scenario Menambah Board</i>	53
Tabel 4.24 <i>Use Case Scenario Melihat Kanban Board</i>	53
Tabel 4.25 <i>Use Case Scenario Mengubah Limit</i>	54
Tabel 4.26 <i>Use Case Scenario Menghapus Board</i>	54
Tabel 4.27 <i>Use Case Scenario Melihat Gantt Chart Proyek</i>	54
Tabel 4.28 <i>Use Case Scenario Melihat Pie Chart Proyek</i>	55
Tabel 4.29 <i>Use Case Scenario Mengupload Dokumen</i>	55
Tabel 4.30 <i>Use Case Scenario Melihat Dokumen Proyek</i>	56
Tabel 4.31 <i>Use Case Scenario Membaca Dokumen Proyek</i>	56
Tabel 4.32 <i>Use Case Scenario Menghapus Dokumen</i>	57
Tabel 4.33 <i>Use Case Scenario Download Dokumen</i>	57
Tabel 4.34 <i>Use Case Scenario Melihat Tim Proyek</i>	58
Tabel 4.35 <i>Use Case Scenario Menambah Data Client</i>	58
Tabel 4.36 <i>Use Case Scenario Melihat Client</i>	59
Tabel 4.37 <i>Use Case Scenario Mengubah Data Client</i>	59
Tabel 4.38 <i>Use Case Scenario Menghapus Data Client</i>	59
Tabel 4.39 <i>Use Case Scenario Menambah Data PIC</i>	60

Tabel 4.40 Use Case Scenario Melihat PIC	60
Tabel 4.41 Use Case Scenario Mengubah Data PIC.....	61
Tabel 4.42 Use Case Scenario Menghapus Data PIC	61
Tabel 4.43 Use Case Scenario Menambah Data Pegawai	61
Tabel 4.44 Use Case Scenario Melihat Data Pegawai	62
Tabel 4.45 Use Case Scenario Mengubah Data Pegawai	62
Tabel 4.46 Use Case Scenario Menghapus Data Pegawai.....	63
Tabel 4.47 Use Case Scenario Lihat Detail Data Pegawai	63
Tabel 4.48 Use Case Scenario Menambah Data Role Pegawai	63
Tabel 4.49 Use Case Scenario Melihat Data Role Pegawai	64
Tabel 4.50 Use Case Scenario Menghapus Data Role Pegawai.....	64
Tabel 4.51 Use Case Scenario Melihat Profil.....	65
Tabel 4.52 Use Case Scenario Mengubah Data Profil	65
Tabel 4.53 Use Case Scenario Mengubah Password.....	66
Tabel 5.1 Pseudocode Algoritma tambah	72
Tabel 5.2 Pseudocode Algoritma tambah	73
Tabel 5.3 Pseudocode Algoritma tambah	74
Tabel 5.4 Spesifikasi Perangkat Keras	78
Tabel 5.5 Spesifikasi Perangkat Lunak	78
Tabel 5.6 Implementasi Rancangan <i>Physical Data Model</i>	79
Tabel 5.7 Source Code Method tambah() pada Controller Project	82
Tabel 5.8 Source Code Method tambah() pada Controller Board	84
Tabel 5.9 Source Code Method tambah() pada Controller Task	84
Tabel 6.1 Pseudocode Algoritme tambah	88
Tabel 6.2 Hasil Pengujian Unit Method tambah	90
Tabel 6.3 Pseudocode Algoritme tambah	92
Tabel 6.4 Hasil Pengujian Unit Method tambah	93
Tabel 6.5 Pseudocode Algoritme tambah	94
Tabel 6.6 Hasil Pengujian Unit Method tambah	96
Tabel 6.7 Kasus uji berhasil melakukan <i>login</i>	98
Tabel 6.8 Kasus uji gagal melakukan <i>login</i> dengan tidak mengisi kolom <i>password</i> dan <i>email</i>	98
Tabel 6.9 Kasus uji gagal melakukan <i>login</i> dengan tidak mengisi kolom <i>password</i>	98
Tabel 6.10 Kasus uji gagal melakukan <i>login</i> dengan tidak mengisi kolom <i>email</i> .	99
Tabel 6.11 Kasus uji gagal melakukan <i>login</i> dengan <i>email</i> yang tidak valid.....	99
Tabel 6.12 Kasus uji gagal melakukan <i>login</i> dengan emali yang belum aktif.....	100
Tabel 6.13 Kasus uji gagal melakukan <i>login</i> dengan <i>email</i> yang tidak terdaftar dalam sistem	100
Tabel 6.14 Kasus uji gagal melakukan <i>login</i> dengan <i>password</i> yang tidak sesuai	100
Tabel 6.15 Kasus uji berhasil melakukan <i>reset password</i>	101
Tabel 6.16 Kasus uji gagal melakukan <i>reset password</i> dengan tidak mengisi kolom <i>email</i>	101

Tabel 6.17 Kasus uji gagal melakukan <i>reset password</i> dengan <i>email</i> yang tidak valid	101
Tabel 6.18 Kasus uji gagal melakukan <i>reset password</i> dengan <i>email</i> yang belum aktif.....	102
Tabel 6.19 Kasus uji gagal melakukan <i>reset password</i> dengan <i>email</i> yang tidak terdaftar	102
Tabel 6.20 Kasus uji berhasil melakukan <i>logout</i>	102
Tabel 6.21 Kasus uji berhasil melihat <i>dashboard</i>	103
Tabel 6.22 Kasus uji berhasil mencari proyek.....	103
Tabel 6.23 Kasus uji gagal mencari proyek	103
Tabel 6.24 Kasus uji menampilkan semua proyek.....	104
Tabel 6.25 Kasus uji Kasus uji berhasil menambah data proyek	104
Tabel 6.26 Kasus uji gagal menambah data proyek dengan seluruh <i>field</i> belum terisi atau terdapat <i>field</i> belum terisi	104
Tabel 6.27 Kasus uji gagal menambah data proyek dengan tanggal mulai proyek melebihi tanggal selesai proyek.....	105
Tabel 6.28 Kasus uji gagal menambah data proyek dengan tanggal mulai proyek atau tanggal selesai proyek kurang dari hari ini	105
Tabel 6.29 Kasus uji berhasil melihat proyek.....	106
Tabel 6.30 Kasus uji berhasil mengubah data proyek	106
Tabel 6.31 Kasus uji gagal mengubah data proyek.....	106
Tabel 6.32 Kasus uji berhasil menghapus data proyek.....	107
Tabel 6.33 Kasus uji berhasil melihat detail proyek.....	107
Tabel 6.34 Kasus uji berhasil menambah tugas	107
Tabel 6.35 Kasus uji gagal menambah tugas dengan seluruh <i>field</i> belum terisi atau salah satu <i>field</i> belum terisi	108
Tabel 6.36 Kasus uji gagal menambah tugas dengan <i>end date</i> tugas melebihi <i>end date</i> proyek	108
Tabel 6.37 Kasus uji gagal menambah tugas dengan <i>start date</i> tugas kurang dari <i>start date</i> proyek.....	109
Tabel 6.38 Kasus uji gagal menambah tugas dengan <i>start date</i> tugas kurang dari <i>end date</i> tugas.....	109
Tabel 6.39 Kasus uji gagal menambah tugas dengan <i>end date</i> tugas dan <i>start date</i> tugas kurang dari hari ini	110
Tabel 6.40 Kasus uji berhasil melihat daftar tugas	111
Tabel 6.41 Kasus uji berhasil mengupdate tugas.....	111
Tabel 6.42 Kasus uji berhasil menghapus data tugas	111
Tabel 6.43 Kasus uji berhasil melihat detail tugas	112
Tabel 6.44 Kasus uji berhasil menambah feedback tugas	112
Tabel 6.45 Kasus uji gagal menambah feedback tugas.....	113
Tabel 6.46 Kasus uji berhasil melihat feedback tugas	113
Tabel 6.47 Kasus uji berhasil menghapus feedback tugas.....	113
Tabel 6.48 Kasus uji berhasil menambah <i>board</i>	114
Tabel 6.49 Kasus uji gagal menambah <i>board</i>	114
Tabel 6.50 Kasus uji berhasil melihat <i>kanban board</i>	115

Tabel 6.51 Kasus uji berhasil mengubah limit.....	115
Tabel 6.52 Kasus uji berhasil menghapus <i>board</i>	116
Tabel 6.53 Kasus uji berhasil melihat <i>gantt chart</i> proyek	116
Tabel 6.54 Kasus uji berhasil melihat <i>pie chart</i> proyek	117
Tabel 6.55 Kasus uji berhasil mengupload dokumen	117
Tabel 6.56 Kasus uji gagal mengupload dokumen dengan format berkas tidak sesuai.....	117
Tabel 6.57 Kasus uji gagal mengupload dokumen dengan <i>field</i> berkas tidak diisi	118
Tabel 6.58 Kasus uji gagal mengupload dokumen dengan <i>ukuran</i> berkas melebihi batas.....	118
Tabel 6.59 Kasus uji berhasil melihat dokumen proyek	119
Tabel 6.60 Kasus uji berhasil membaca dokumen proyek.....	119
Tabel 6.61 Kasus uji berhasil menghapus dokumen	120
Tabel 6.62 Kasus uji berhasil download dokumen.....	120
Tabel 6.63 Kasus uji berhasil melihat tim proyek	121
Tabel 6.64 Kasus uji berhasil menambah data <i>client</i>	121
Tabel 6.65 Kasus uji gagal menambah data <i>client</i>	121
Tabel 6.66 Kasus uji berhasil melihat <i>client</i>	122
Tabel 6.67 Kasus uji berhasil mengubah data <i>client</i>	122
Tabel 6.68 Kasus uji gagal mengubah data <i>client</i>	122
Tabel 6.69 Kasus uji berhasil menghapus data <i>client</i>	123
Tabel 6.70 Kasus uji berhasil menambah data <i>PIC</i>	123
Tabel 6.71 Kasus uji gagal menambah data <i>client</i>	123
Tabel 6.72 Kasus uji berhasil melihat <i>PIC</i>	124
Tabel 6.73 Kasus uji berhasil mengubah data <i>PIC</i>	124
Tabel 6.74 Kasus uji gagal mengubah data <i>PIC</i>	125
Tabel 6.75 Kasus uji berhasil menghapus data <i>PIC</i>	125
Tabel 6.76 Kasus uji berhasil menambah data pegawai.....	125
Tabel 6.77 Kasus uji gagal menambah data pegawai	126
Tabel 6.78 Kasus uji berhasil melihat data pegawai	126
Tabel 6.79 Kasus uji berhasil mengubah data pegawai	126
Tabel 6.80 Kasus uji gagal mengubah data pegawai	127
Tabel 6.81 Kasus uji berhasil menghapus data pegawai.....	127
Tabel 6.82 Kasus uji berhasil melihat detail data pegawai	128
Tabel 6.83 Kasus uji berhasil menambah data role pegawai.....	128
Tabel 6.84 Kasus uji gagal menambah data <i>pegawai</i>	128
Tabel 6.85 Kasus uji berhasil melihat data role pegawai	129
Tabel 6.86 Kasus uji berhasil menghapus data role pegawai	129
Tabel 6.87 Kasus uji berhasil melihat profil	129
Tabel 6.88 Kasus uji berhasil mengubah data profil	130
Tabel 6.89 Kasus uji gagal mengubah data profil	130
Tabel 6.90 Kasus uji berhasil mengubah password	130
Tabel 6.91 Kasus uji gagal mengubah password.....	131

Tabel 6.92 Kasus uji gagal mengubah password dengan inputan password saat ini tidak sesuai.....	131
Tabel 6.93 Kasus uji gagal mengubah password dengan inputan password baru tidak sesuai dengan inputan <i>field repeat password</i> dan sebaliknya	131
Tabel 6.94 Kasus uji gagal mengubah password dengan inputan password baru kurang dari enam karakter.....	132
Tabel 6.95 Daftar Pertanyaan Pengujian <i>Usability</i> menggunakan Metode SUS	133
Tabel 6.96 Hasil Kuisioner Responden terhadap Pertanyaan Metode SUS.....	133
Tabel 6.97 Hasil Perhitungan Skor Akhir Penilaian Kuisioner	134

DAFTAR GAMBAR

Gambar 2.1 Contoh penerapan <i>Kanban</i>	5
Gambar 2.2 Deskripsi <i>Kanban</i> dari Toyota.....	19
Gambar 2.3 Penerapan <i>Kanban</i>	21
Gambar 3.1 Tahapan Metodologi Penelitian	26
Gambar 4.2 Aturan Penomoran.....	32
Gambar 4.3 Analisis data sistem manajemen proyek.....	41
Gambar 4.4 <i>Use Case Diagram</i> Sistem Manajemen Proyek	43
Gambar 5.1 <i>Sequence Diagram</i> Menambah Data Proyek	68
Gambar 5.2 <i>Sequence Diagram</i> Menambah Data <i>Board</i>	69
Gambar 5.3 <i>Sequence Diagram</i> Menambah Data Tugas	70
Gambar 5.4 <i>Class Diagram</i> Sistem Manajemen Proyek	71
Gambar 5.5 <i>Physical Data Model</i> Sistem Manajemen Proyek.....	75
Gambar 5.6 Gambar Rancangan Menambah Data Proyek.....	76
Gambar 5.7 Gambar Rancangan Menambah Data <i>Board</i>	77
Gambar 5.8 Gambar Rancangan Menambah Data Tugas.....	77
Gambar 5.9 Gambar Implementasi Antarmuka Menambah Data Proyek	86
Gambar 5.10 Implementasi Antarmuka Menambah Data <i>Board</i>	87
Gambar 5.11 Implementasi Antarmuka Menambah Data Tugas	87
Gambar 6.1 <i>Flow Graph Method</i> <i>tambah()</i> pada kelas <i>Project</i>	89
Gambar 6.2 <i>Flow Graph Method</i> <i>tambah()</i> pada kelas <i>Board</i>	92
Gambar 6.3 <i>Flow Graph Method</i> <i>tambah()</i> pada kelas <i>Task</i>	95

DAFTAR LAMPIRAN

Lampiran 1 Kasus Penjadwalan.....	138
Lampiran 2 Lampiran Kasus Penjadwalan (lanjutan)	139
Lampiran 3 Lampiran Kasus Penjadwalan (lanjutan)	140
Lampiran 4 Lampiran Kasus Penjadwalan (lanjutan)	141
Lampiran 5 Lampiran Kasus Penjadwalan (lanjutan)	142

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Manajemen adalah sebuah kegiatan penataan tata ruang yang berhubungan erat pada tiap skala kehidupan, mulai dari hal terkecil yaitu individu, maupun pada organisasi kompleks yang berhubungan dengan kehidupan orang banyak, tidak terkecuali suatu perusahaan. Suatu perusahaan tentunya mengharapkan semua proyek yang ditangani dapat dikelola dengan sangat baik. Agar dapat mengelola proyek dengan baik tentunya dibutuhkan sebuah manajemen proyek yang baik pula. Hal ini berarti menyangkut perencanaan yang unggul, teknik organisasi yang matang, serta mengarahkan dan mengendalikan kekayaan perusahaan seperti sumber daya yang ada untuk mencapai tujuan yang telah disepakati pada kurun waktu yang telah diberikan (Budi Santosa, 2009).

Dalam lingkungan bisnis berbasis teknologi yang dinamis saat ini, perusahaan yang bergerak di bidang perangkat lunak dituntut untuk mampu menjawab tantangan akan kebutuhan *client* yang tidak menentu. Kebutuhan yang tidak menentu pasti memiliki pengaruh terhadap perencanaan yang telah disusun, perubahan kebutuhan ditengah berjalannya proyek pasti memiliki dampak terhadap perencanaan proyek, hal ini tentunya dapat mengakibatkan proses berjalannya proyek terhambat apabila sering terjadi perubahan yang seharusnya dapat diminimalisir dalam proses pelaksanaan proyek. Perubahan proyek dapat diminimalisir dengan melakukan perencanaan proyek yang akurat, selain itu perubahan yang terjadi dapat ditangani dengan penggunaan metode yang sesuai dengan keadaan yang telah ditelaah sebelumnya.

Pengelolaan sebuah proyek perangkat lunak merupakan hal yang sulit dan sifatnya yang terus menerus mengalami perubahan dan perkembangan hampir setiap saat menuntut pengelolaan dan manajemen untuk selalu berkembang, dan pengelolaan proyek ini makin dipersulit karena proses ini adalah suatu usaha membuat sesuatu yang tidak tampak menjadi tampak, serta memiliki karakteristik yang unik seperti *invisibility*, *complexity*, dan *flexibility* dimana karakter inilah yang membedakan antara proyek perangkat lunak dengan yang lain (Hughes & Cotterell, 1999). Sulitnya mengelola proyek perangkat lunak sering kali mengakibatkan suatu proyek mengalami kegagalan. Kasus kegagalan proyek perangkat lunak di Indonesia sendiri tidak dapat dibilang sedikit, dari 110 proyek sistem informasi di berbagai kota di Indonesia 27% diantaranya diselesaikan sesuai dengan tupoksi yang telah direncanakan sebelumnya, 55% memiliki masalah dan 18% sisanya dibatalkan (Rudi Dwi Apriyanto dan Hanson Prihantoro Putro, 2018). Proyek perangkat lunak sendiri dapat dikatakan mengalami kegagalan apabila mengalami hal-hal berikut: anggaran yang digunakan melebihi dari yang telah direncanakan, proyek selesai melebihi jadwal yang telah direncanakan, dan proyek menyimpang dari tujuan yang telah direncanakan (Project Management Institute, 2013). Tentunya memerlukan suatu usaha yang besar agar langkah – langkah manajemen proyek dapat

terlaksana dan kegagalan proyek dapat terhindarkan, contohnya adalah dengan penggunaan suatu alat, metode atau sistem manajemen proyek.

Suatu sistem manajemen proyek pasti menerapkan suatu metode manajemen proyek untuk digunakan dan divisualisasikan didalam sistem tersebut. Sebuah contoh baik yang dapat dicontoh adalah manajemen proyek perangkat lunak yang disebut dengan metode *Kanban*. *Kanban* dalam bahasa Jepang memiliki arti sebagai kartu atau papan penanda, *Kanban* adalah sebuah sistem penjadwalan kerja yang memaksimalkan produktifitas tim dengan mengurangi waktu menganggur, waktu menganggur dapat terjadi pada setiap proses, alur kerja atau prosedur dan biasanya dapat dilacak bedasarkan proses itu sendiri. *Kanban* berfokus pada pencegahan produksi berlebihan, pencegahan kegiatan yang tidak perlu, pencegahan proses berlebihan, dan waktu menunggu antara satu tugas dengan tugas lainnya (Raut, Laukik & Wakode, Rajat & Talmale, Pravin, 2015). Berakar dari *lean manufacturing Kanban* telah diterapkan di berbagai jenis industri diantaranya adalah aeronautika, kesehatan, sumberdaya manusia dan juga dalam penembangan perangkat lunak (Ahmad, Muhammad Ovais & Dennehy, Denis & Conboy, Kieran & Oivo, Markku, 2017). Dalam praktiknya, metode *Kanban* membagi suatu proyek menjadi tugas-tugas yang harus melewati beberapa tahap yang selanjutnya tahap-tahap ini disebut sebagai *Kanban board* sebelum tugas dapat dinyatakan selesai, biasanya *Kanban* memiliki tiga buah board yakni *to-do*, *in-progress* dan *done* atau juga dapat disesuaikan dengan kebutuhan perusahaan. *Kanban* memudahkan manajemen proyek karena dapat dengan mudah untuk mendeteksi bagian mana yang membuat suatu proyek terhambat, apabila suatu tugas *stuck* pada tahap *in-progress* maka tugas tersebut sedang mengalami masalah.

Studi kasus di perusahaan CV. Primavisi Globalindo yang sedang berkembang sangat memerlukan sistem informasi manajemen proyek yang mudah untuk digunakan dan handal. Memperhatikan karakteristik proses bisnis yang ada di perusahaan serta beberapa hasil penelitian sebelumnya, sangat penting untuk dilakukan pengembangan sistem informasi manajemen proyek yang sesuai. Metode *Kanban* sangat cocok dan dapat menjawab permasalahan yang dialami oleh perusahaan. Maka penelitian “Pengembangan Sistem Manajemen Proyek Menggunakan Metode *Kanban* Berbasis Web Studi Kasus Primavisi Globalindo Surabaya”, dengan tujuan utama untuk memecahkan permasalahan perusahaan terkait kebutuhan adanya sistem manajemen proyek dan pemecahan masalah tersebut akan dijabarkan dalam sub-sub permasalahan terkait bagaimana tahapan membangun sistem manajemen proyek tersebut dan apakah sistem yang dibangun dapat menyelesaikan masalah perusahaan.

1.2 Rumusan Masalah

Berdasarkan pendahuluan yang telah dijabarkan sebelumnya, dapat dituliskan sebuah rumusan sebagai berikut:

1. Bagaimana hasil analisis kebutuhan sistem informasi manajemen proyek yang akan digunakan di CV. Primavisi Globalindo?

2. Bagaimana hasil perancangan sistem informasi manajemen proyek yang akan digunakan di CV. Primavisi Globalindo?
3. Bagaimana hasil implementasi sistem informasi manajemen proyek yang akan digunakan di CV. Primavisi Globalindo?
4. Bagaimana hasil pengujian sistem informasi manajemen proyek yang akan digunakan di CV. Primavisi Globalindo?

1.3 Tujuan

Penelitian ini bertujuan untuk:

1. Mengetahui hasil analisis kebutuhan sistem informasi manajemen proyek yang akan digunakan di CV. Primavisi Globalindo.
2. Mengetahui hasil perancangan sistem informasi manajemen proyek yang akan digunakan di CV. Primavisi Globalindo.
3. Mengetahui hasil implementasi sistem informasi manajemen proyek yang akan digunakan di CV. Primavisi Globalindo.
4. Mengetahui hasil pengujian sistem informasi manajemen proyek yang akan digunakan di CV. Primavisi Globalindo.

1.4 Manfaat

1. Bagi CV. Primavisi Globalindo

Diharapkan dari penelitian ini dapat menghasilkan suatu sistem yang dapat memudahkan perusahaan dalam melakukan manajemen proyek yang ditangani oleh perusahaan.

2. Bagi penulis

Dengan disusun dan dilaksanakan penelitian ini diharapkan dapat menambah wawasan dan pengalaman penelitian serta dapat menjadi media untuk penulis mengimplementasikan ilmu yang telah didapat selama menjadi mahasiswa Fakultas Ilmu Komputer.

3. Bagi Pembaca atau Pihak Lain

Diharapkan dapat menjadi rujukan dan inspirasi bagi pembaca atau pihak lain dalam melakukan penelitian yang berkaitan dengan topik ini.

1.5 Batasan Masalah

Batasan ada pada manajemen penjadwalan tugas, manajemen pembagian tugas dan manajemen sumber daya manusia suatu proyek perangkat lunak. Sistem yang dikembangkan diimplementasikan dalam bentuk *website* dan bahasa pemrograman yang digunakan dalam implementasi perangkat lunak ini adalah HTML, PHP, CSS, JavaScript dengan *framework* CodeIgniter.

1.6 Sistematika Pembahasan

Sebuah sistematika untuk pembahasan penelitian ini adalah sebagaimana berikut :

1. BAB I Pendahuluan

Bab I berisi Latar belakang dilaksanakannya penelitian ini sehingga mengetahui apa tujuan dan batasan daripada masalah yang dikerjakan pada penelitian.

2. BAB II Landasan Kepustakaan

Bab II berisi penjabaran akan metode, teori dan konsep yang akan digunakan pada penelitian ini.

3. BAB III Metodologi

Bab III berisi Langkah-langkah penggerjaan penelitian, dimana hal tersebut menyangkup setiap teknik dan langkah-langkah yang akan dikerjakan, dan menjelaskan mengapa digunakan metode *Kanban* sebagai metode penelitian.

4. BAB IV Rekayasa Kebutuhan

Bab IV berisi kebutuhan dan hipotesa hasil rancangan yang digunakan sebagai acuan dasar pada perancangan sistem menggunakan *use case* beserta skenarionya.

5. BAB V Perancangan dan Implementasi

Bab V mengurai rancangan perangkat, yakni menyusun perancangan yang berdasar dari rekayasa kebutuhan yang telah dilakukan serta mengimplementasikan perancangan hingga menjadi sebuah aplikasi. Perancangan akan dilakukan menggunakan UML *diagram*, meliputi penggunaan *use case*, *sequence* dan *class diagram*, dimana keluaran hasil perancangan diimplementasikan menggunakan pemrograman HTML, CSS dan PHP.

6. BAB VI Pengujian dan Analisis

Bab VI mengkaji metode pengujian rancangan yang telah dibuat serta analisis hasil perancangan sesuai dengan masalah yang telah dikemukaan sebelumnya.

7. BAB VII Kesimpulan dan Saran

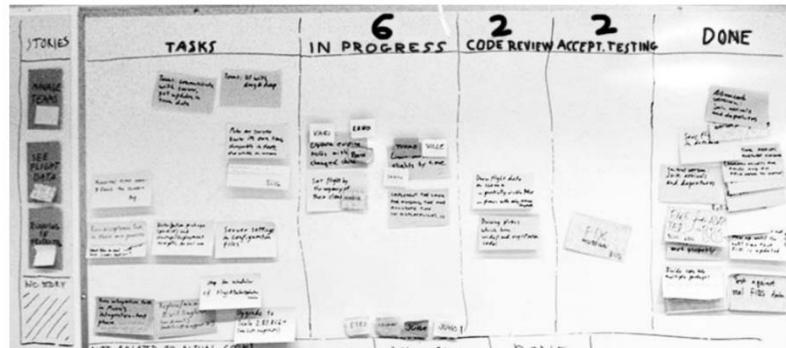
Bab VII memuat kesimpulan terhadap skripsi berdasarkan apa yang diperoleh berdasar kepada rumusan masalah yang telah dikemukaan pada awal penelitian.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini penulis membahas landasan pustaka berupa penelitian terdahulu serta dasar teori yang digunakan untuk menyusun dan melaksanakan penelitian yang diusulkan. Landasan pustaka yang dijadikan sebagai acuan adalah penelitian dan buku yang berhubungan dengan sistem manajemen proyek, proyek, metode manajemen proyek dan proses pengembangan perangkat lunak.

2.1 Kajian Pustaka

Dalam melakukan penelitian ini, peneliti mengacu pada penelitian yang telah dilakukan sebelumnya untuk menjadi tinjauan sistem. Penelitian pertama adalah penelitian yang ditulis oleh Marko Ikonen, Elena Pirinen, Fabian Fagerholm , Petri Kettunen dan Pekka Abrahamsson yang menguji dampak pada metode *Kanban* dalam sebuah proyek perangkat lunak, dalam penelitian ini peneliti fokus untuk menyelidiki dampak penerapan *Kanban* pada proyek perangkat lunak dengan studi kasus empiris. Peneliti menyajikan contoh penerapan *Kanban* dalam proyek perangkat lunak, contoh penerapan *Kanban* pada Gambar 2.1 menggambarkan sebuah papan kosong yang kemudian dibagi menjadi lima bagian, yaitu *task*, *in progress*, *code review*, *accept & testing* dan *done*, dimana tiap-tiap bagian ini menyatakan tahap yang harus dilewati suatu tugas. Dapat diamati bahwa tiap bagian/tahap berisi satu atau lebih catatan yang berisi tugas yang sedang berada pada tahap tersebut, selain itu pada tiap tahap terdapat angka yang dituliskan diatasnya yang menyatakan batas maksimal tugas yang boleh berada pada tahap tersebut.



Gambar 2.1 Contoh penerapan *Kanban*

Ikonen, Marko & Pirinen, Elena & Fagerholm, Fabian & Kettunen, Petri & Abrahamsson, Pekka. (2011)

Hasil dari penelitian pertama menyatakan bahwa model proses *Kanban* menunjukkan manfaat yang cukup besar terhadap proyek yang berlangsung, diantaranya adalah adanya peningkatan motivasi dan kontrol atas kegiatan proyek, peningkatan tersebut terjadi akibat keuntungan penggunaan *Kanban* yaitu dapat memvisualisasikan keseluruhan perkembangan proyek dalam satu waktu dan juga terdapat batas jumlah tugas yang boleh menempati satu *Kanban*

board, dengan adanya batasan jumlah ini maka tim harus meningkatkan kinerjanya agar tidak terjadi penumpukan tugas dan aliran tugas dapat berjalan sebagaimana mestinya sehingga proyek yang dilaksanakan dapat selesai sesuai dengan jadwal yang sudah disepakati sebelumnya. Penelitian ini menginspirasi peneliti untuk mengembangkan sistem yang mengimplementasikan metode Kanban dengan visualisasi digital. Penelitian pertama yang menyelidiki penggunaan Kanban secara manual menginspirasi peneliti untuk mengembangkan sistem yang mengimplementasikan metode Kanban dengan visualisasi digital.

Penelitian lainnya yang ditulis oleh Erika Corona dan Filippo Eros Pani yang mengulas tentang pendekatan *Lean-Kanban* pada perencanaan dan pengembangan perangkat lunak menjelaskan pada penelitian ini menyatakan bahwa akhir-akhir ini metode *Agile* telah banyak diadopsi dalam pengembangan perangkat lunak, metode *Agile* yang paling cepat berkembang adalah dengan pendekatan *Lean* menggunakan *Kanban* untuk implementasi praktiknya. Namun walaupun terdapat peningkatan minat penggunaan *Kanban*, masih belum ada definisi standar dan aturan spesifik tentang penerapan *Kanban*. Penelitian ini bertujuan untuk melakukan ulasan terhadap pendekatan *Kanban* untuk pengembangan perangkat lunak dan menjawab permasalahan standarisasi penerapan *Kanban*. Untuk menjawab permasalahan tersebut peneliti menggunakan metode Kitchenham dan merumuskan lima pertanyaan penelitian yang digunakan sebagai instrumen dalam penelitian tersebut, lima pertanyaan tersebut adalah: 1. apa karakteristik utama pada papan yang digunakan pada *kanban*?; 2. Aktivitas utama apa yang mendefinisikan suatu proses penggerjaan dan apa batas khusus pada WIP dalam sebuah tim pengembang perangkat lunak?; 3. Informasi apa yang biasanya diperlihatkan pada kartu yang merepresentasikan sebuah unit kerja?; 4. Diagram apa yang digunakan untuk proses manajemen secara kuantitatif?; 5. Alat otomatisasi apa yang dapat digunakan untuk papan kanban? Dan apa karakteristik utamanya?; Hasil dari penelitian ini adalah pemaparan tentang bagaimana sebenarnya pendekatan *Kanban* disajikan dan digunakan, penulis berharap dengan dilakukannya penelitian tersebut dapat memberikan persepsi dan pemahaman terhadap bagaimana pendekatan *Kanban* diterapkan kepada pembaca dan dapat digunakan sebagai pertimbangan bagi pengembang sebelum memutuskan untuk mengadopsi *Kanban*. Penelitian ini mendorong peneliti untuk menggunakannya sebagai metode kontrol yang diimplementasikan dalam sistem manajemen proyek dan juga sebagai panduan penerapan Kanban dalam sistem.

Penelitian ketiga yang dilakukan oleh Fahat, 2018 membahas tentang pengembangan sistem manajemen proyek dimana penelitian terfokus pada pengembangan sistem manajemen proyek perangkat lunak yang mengadopsi metode Scrum. Didalam penelitian tersebut penulis memaparkan teori-teori tentang metode Scrum yang digunakan sebagai dasar atau acuan pengembangan sistem manajemen proyek agar dapat menyampaikan fitur-fitur yang sesuai dengan siklus manajemen proyek dengan metode Scrum.

Berbeda dengan penelitian ketiga, dimana penelitian terfokus pada pengembangan aplikasi manajemen proyek yang mengadopsi Scrum, fokus penelitian yang sekarang adalah pada pengembangan aplikasi yang menyampaikan penerapan metode manajemen proyek *Kanban* secara digital dan diterapkan menggunakan framework Codeigniter.

2.2 Rekayasa Perangkat Lunak

Sebuah disiplin ilmu yang mempelajari setiap hal yang berkaitan pada produksi pembuatan perangkat lunak yang memiliki empat aktifitas utama yaitu spesifikasi, dimana hal ini menyangkut tentang tujuan apa perangkat lunak dibangun. Pengembangan, menyangkut setiap proses yang dilaksanakan dalam proses penggerjaan perangkat lunak. Validasi, untuk mengetahui apakah perangkat yang telah dibangun sesuai dengan spesifikasi yang telah diberikan, dan evolusi perangkat lunak yang membahas tentang perubahan yang akan terjadi pada perangkat lunak yang telah diluncurkan sebelumnya (Sommerville, 2011). Terdapat dua jenis produk perangkat lunak diantaranya adalah *generic product* dan *customized product*. *Generic product* adalah suatu produk yang diproduksi dan dipasarkan untuk semua pengguna / dipasarkan secara terbuka, contoh dari produk ini adalah aplikasi minimarket, ataupun suatu sistem akuntansi. *Customized product* adalah suatu produk yang tidak dijual atau dipasarkan secara umum, biasanya produk ini diminta oleh pihak tertentu kepada pengembang tertentu dan digunakan untuk tujuan tertentu (Sommerville, 2011).

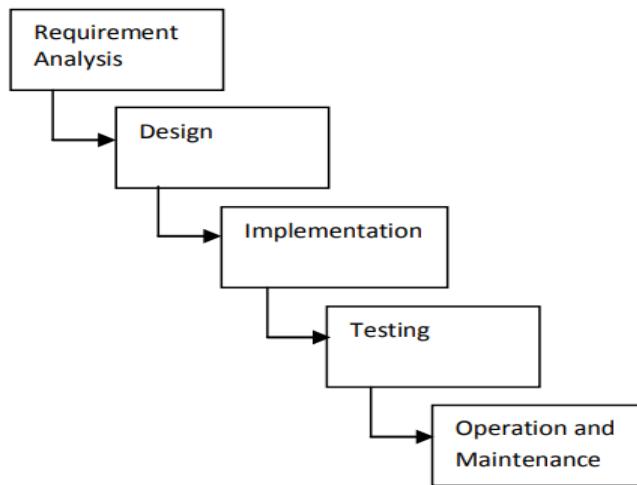
2.3 Pengembangan Perangkat Lunak

Pengembangan perangkat merupakan suatu langkah/tahapan yang dikerjakan untuk mengembangkan perangkat lunak. Tahapan pengembangan perangkat lunak biasanya difokuskan pada melakukan perancangan konstruksi data, implementasi fungsi, implementasi prosedur, melakukan analisis rancangan, dan bagaimana prosedur pengujian dibuat dan dilaksanakan (Pressman, 2010). Pada dasarnya terdapat beberapa tahap yang harus ada dan dilakukan untuk melakukan pengembangan perangkat lunak, yaitu rekayasa kebutuhan, perancangan, implementasi, dan pengujian. Proses analisis kebutuhan yang diperlukan dalam penelitian untuk mengerti fungsi/fitur yang dibutuhkan merupakan proses Rekayasa Kebutuhan. Setelah kebutuhan didapatkan kemudian dilakukan pemodelan kebutuhan yang sudah didapatkan. Perancangan adalah proses mendefinisikan dengan lebih rinci bagaimana kebutuhan yang didapat akan diimplementasikan menjadi kode program. Implementasi adalah proses menerjemahkan model/perancangan kedalam sebuah program komputer. Kemudian setelah implementasi berhasil dilakukan dan sesuai dengan kebutuhan sebelumnya, dilaksanakanlah sebuah Pengujian untuk memastikan bahwa setiap fungsi dalam program dapat bekerja sesuai dengan yang diharapkan.

2.3.1 Model Pengembangan Perangkat Lunak

Model Pengembangan Perangkat Lunak adalah sebuah proses sistematis yang dilaksanakan untuk mempermudah pengembangan sebuah perangkat lunak. Model-model ini biasa disebut dengan *Software Development Life Cycle* (SDLC) (Pressman, 2010). Siklus ini memiliki berbagai tahapan pengembangan dimana tiap tahapan memiliki tujuan dan maksud yang berbeda-beda, sehingga memudahkan pengembang untuk mengatur pelaksanaan perangkat lunak dengan lebih baik lagi. Terdapat beberapa model SDLC diantaranya adalah *waterfall*, *prototyping*, dan *RAD*.

2.3.1.1 Waterfall Model



Gambar 2.1 Tahapan Waterfall Model

Sumber: Adenowo & Adenowo (2013)

Dengan menggunakan model ini, pengembangan perangkat lunak akan dilaksanakan secara *sequential* atau berurutan. Metode ini disebut *Waterfall* karena fase-fase metode ini dilakukan secara berurutan dari atas ke bawah. Hasil dari setiap fase akan digunakan sebagai dasar dari pelaksanaan fase selanjutnya (Sommerville, 2011). Syarat SDLC *waterfall* sendiri adalah satu tahap/fase harus selesai dan lengkap terlebih dahulu sebelum dapat melanjutkan ke fase selanjutnya. *Waterfall model* memiliki lima fase yang dilakukan secara berurutan diantaranya adalah analisis kebutuhan (*requirement analysis*), dimana pada tahap ini dilaksanakan analisa kebutuhan yang diperlukan dalam sebuah perangkat lunak, serta menentukan arah dan tujuan daripada pengembangan perangkat lunak. perancangan (*design*), menjelaskan tentang bagaimana desain perangkat lunak yang akan dikerjakan. implementasi (*implementation*), mengubah desain dan analisa kebutuhan menjadi sebuah program yang utuh. pengujian (*testing*), dilakukan untuk mengetahui kesalahan yang terjadi pada program yang telah dibuat sebelumnya. Operasi serta Pemeliharaan (*operation and maintenance*), dilaksanakan untuk mengetahui dampak penggunaan program dalam jangka yang panjang dan membenahinya apabila terjadi kesalahan pada program seiring berjalannya waktu. gambaran SDLC *waterfall*

tampak pada Gambar 2.1. Penjelasan tiap tahap/fase tersebut secara detil adalah sebagai berikut:

1. *Requirement Analysis*

Tahap ini adalah tahap pertama dalam *waterfall model*, pada tahap ini pengembang melakukan analisis kebutuhan sistem dengan cara elisitasi tertentu, dalam penggunaan *waterfall model*, tahap ini adalah tahap yang paling penting dikarenakan kebutuhan harus didefinisikan dan dianalisis secara terperinci dan akurat agar tidak terjadi banyak perubahan kebutuhan.

2. *Design*

Tahap perancangan atau *design* adalah tahap dimana pengembang melakukan perancangan berdasarkan kebutuhan yang telah didefinisikan dan dispesifikasikan pada tahap sebelumnya. Tahap perancangan meliputi perancangan perangkat lunak yang akan dibangun dan juga penentuan persyaratan perangkat keras yang digunakan.

3. *Implementation*

Tahap implementasi atau *implementation* adalah tahap dimana perancangan yang dibuat diimplementasikan menjadi kode program. Hasil dari proses implementasi adalah suatu komponen perangkat lunak yang siap untuk diuji hingga perangkat lunak utuh yang siap dipakai.

4. *Testing*

Tahap *testing* atau pengujian adalah tahap yang dilakukan untuk memastikan apakah sistem yang dibuat telah memenuhi kebutuhan klien, apakah sistem telah terhindar dari *error* ataupun *bugs* dan apakah sistem telah dikembangkan sesuai dengan desain yang telah dirancang.

5. *Operation and Maintenance*

Tahap ini merupakan tahap terlama dalam siklus pengembangan menggunakan metode *waterfall*. Pada tahap ini pengembang akan melakukan instalasi sistem pada lingkungan kerja klien dan juga melakukan pemeliharaan sistem tanpa ada batas waktu. Pemeliharaan sistem disini meliputi perbaikan, pengecekan kesalahan dan juga pembaruan apabila sistem mengalami pembaruan. Namun dalam penelitian kali ini tahap ini tidak dilakukan.

Model *waterfall* digunakan pada penelitian ini karena kebutuhan untuk membangun sistem telah diketahui mengacu pada kajian pustaka yang disajikan dan perubahan kebutuhan selama pengembangan perangkat lunak dirasa tidak banyak sehingga peneliti dapat fokus melakukan tahapan demi tahapan tanpa adanya pengulangan tahapan, dengan memperhatikan fakta tersebut, peneliti memutuskan untuk menerapkan metode *waterfall* pada penelitian ini. perlu diketahui bahwa pada penelitian ini fase *operation and maintenance* tidak dilakukan, hal ini dikarenakan peran peneliti hanya sebagai pengembang sistem dan setelah sistem diserahkan maka selanjutnya adalah diluar kuasa peneliti.

2.3.2 Pemodelan *Unified Modelling Language* (UML)

Merupakan sebuah bahasa yang menggunakan dan berdasar pada grafik/gambar untuk menggambarkan, menspesifikasi, membangun, dan mendokumentasikan pengembangan sebuah sistem yang dikembangkan dengan pendekatan berbasis objek (Rumbaugh, et al., 1999). *Building Block*, *Placing building Block* dan mekanisme umum UML merupakan elemen-elemen penting untuk dapat mengetahui dan mempelajari UML.

Bulding block memiliki tiga komponen utama, diantaranya adalah *Thing*(Benda), *Relationship* (Hubungan) dan *Diagram* (Bagan) (Rumbaugh, et al. 1999). Benda adalah komponen yang paling mendasar dalam definisi suatu model UML, benda juga komponen yang tidak banyak mengalami perubahan (statis) dalam sebuah model selain itu benda juga dapat digunakan untuk menjelaskan komponen lain dari suatu model. *Thing* (Benda) dapat berupa *interface*, *class*, *use case*, *collaboration*, dan *nodes*. *Relationship* (hubungan) adalah suatu notasi yang menyatakan hubungan antar benda, terdapat empat jenis hubungan dalam UML diantaranya adalah: *Dependency*, *Association*, *Generalization*, dan *Realization*. *Diagram* (bagan) adalah penggambaran permasalahan ataupun solusi menggunakan model, menurut Sommerville dalam bukunya terdapat lima jenis diagram yang penting untuk merepresentasikan suatu sistem diantaranya adalah:

1. *Activity diagram*
2. *Use-case diagram*
3. *Sequence diagram*
4. *Class diagram*
5. *State diagram*

Diagaram pemodelan yang digunakan dalam penelitian ini adalah *use-case diagram*, *sequence diagram*, dan *class diagram*. Penggambaran tentang apa saja aktifitas yang dilakukan dalam tiap fungsi, dan untuk siapa saja fungsi itu bekerja adalah tugas dari sebuah *Use-case diagram*, dan diagram ini terfokus pada apa saja yang dapat dilakukan oleh sistem bukan bagaimana alur suatu aktifitas berjalan didalam sistem, diagram ini menjadi suatu bagan yang kemudian digunakan sebagai dasar dalam merancang *scenario*. *Scenario* mengambarkan bagaimana sebuah fungsi atau sistem mengerjakan suatu pekerjaannya secara beruntutan ketika suatu user menjalankan suatu fungsi tertentu. *Class diagram* menyajikan gambaran sistem secara menyeluruh, yakni tentang *class-class* apa saja yang terlibat dan hubungan antar *class* yang ada. *Class diagram* bersifat statis karena hanya menggambarkan hubungan yang terjadi antar *class* dan bukan apa yang terjadi ketika antar *class* memiliki hubungan. *Diagram class* dan *object* adalah salah satu penggambaran sistem yang bersifat statis, selain penggambaran sistem yang bersifat statis juga terdapat penggambaran sistem yang bersifat dinamis yaitu dengan menggunakan Diagram Interaksi (*Interaction Diagram*). *Sequence diagram* adalah salah satu contoh diagram interaksi, dimana *sequence diagram* menjelaskan bagaimana operasi suatu fungsi berjalan didalam suatu sistem yakni meliputi penjelasan tentang pesan (*message*) apa yang

dikirimkan dalam suatu interaksi dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu, tiap-tiap obyek yang terlibat dalam suatu operasi diurutkan dari kiri ke kanan berdasarkan pada kapan operasi tersebut terjadi kedalam sebuah pesan yang terurut.

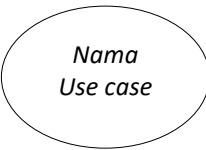
2.3.3 Pendekatan *Object Oriented*

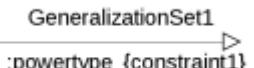
OO (*Object Oriented*) adalah suatu metode atau pendekatan yang berorientasi kepada objek dimana suatu sistem dipandang sebagai kumpulan objek yang dapat saling berinteraksi dan bekerja sama untuk menjalankan tugas tertentu. Pendekatan OO terbagi menjadi tiga, diantaranya adalah OOA (*object oriented analysis*), OOD (*object oriented design*) dan OOP (*object oriented programming*). OOA adalah tahapan analisis sistem dengan pendekatan objek, dimana dalam tahapan ini didefinisikan *use case* yang ada dalam sistem. OOD adalah proses desain yang meliputi pendefinisian semua objek yang ada dalam sistem, interaksi antar objek yang ada dalam sistem dan proses melengkapi pendefinisian objek agar lebih detail dan dapat memudahkan dalam proses implementasi. OOP adalah suatu teknik penulisan bahasa pemrograman yang dilakukan dengan pendekatan objek, OOP mendukung *object class*, *inheritance*, *reuse* dan *encapsulation* untuk menentukan apa yang dilakukan pada tiap-tiap objek (Satzinger, Jackson, & Burd, 2010).

2.3.3.1 *Use Case Diagram*

Diagram ini menggambarkan setiap aktor dan tiap operasi fungsi yang dapat dilakukan pada suatu perangkat lunak. Aktor yang dilibatkan dalam sistem dapat berupa manusia atau sistem yang lain. *Use case* merepresentasikan perilaku/case yang dapat dilakukan oleh aktor untuk berinteraksi dengan sistem. Setiap *use case* harus didefinisikan dengan deskripsi teksual (Sommerville, 2011). Setiap *use case* menetapkan beberapa perilaku yang dapat dilakukan oleh satu aktor atau beberapa aktor. Ketika sebuah *use case* memiliki asosiasi dengan aktor lebih dari satu, terdapat beberapa aktor yang dapat melakukan *use case* tersebut (Omg, 2017). Notasi pada *use case diagram* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Notasi *Use Case Diagram*

Nama Notasi	Simbol	Deskripsi
Aktor	 Actor	Aktor merepresentasikan orang, atau sistem lain yang berinteraksi dengan sistem.
Use Case		<i>Use case</i> merepresentasikan perilaku yang dapat dilakukan oleh aktor. Perilaku di deskripsi kan di dalam <i>ellips</i> yang didalamnya terdapat nama <i>use case</i> tersebut.
Asosiasi	---	Asosiasi merepresentasikan hubungan antara aktor dan <i>use case</i> .

Nama Notasi	Simbol	Deskripsi
Generalisasi		Generalisasi merepresentasikan hubungan antara klasifikasi yang lebih umum dan klasifikasi yang lebih spesifik. Setiap turunan dari klasifikasi spesifik juga merupakan turunan dari klasifikasi umum. Golongan spesifik mewarisi fitur dari klasifikasi yang lebih umum. generalisasi dimiliki oleh golongan spesifik.

(Sumber : Omg, 2017)

2.3.3.2 Use Case Scenario

Use case scenario adalah suatu deskripsi yang memberikan penjelasan bagaimana suatu fungsi didalam sistem akan digunakan. *Scenario* menjelaskan setiap interaksi antara aktor dan sistem dengan lebih terperinci (Pressman, 2010). Seseorang biasanya merasa lebih mudah memahami hal yang berhubungan dengan contoh kehidupan nyata daripada memahami deskripsi yang abstrak. Maka interaksi antara aktor dan perangkat lunak dapat dipahami dan dikritik. Beberapa macam bentuk skenario dibuat, bentuk scenario ini digunakan untuk memberikan informasi dengan tingkatan detail sistem yang berbeda (Sommerville, 2011). *Scenario* memiliki beberapa komponen yang dapat dilihat pada Tabel 2.2.

Tabel 2.2 Komponen Use Case Scenario

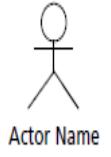
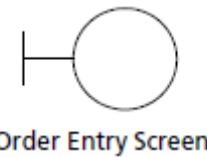
No	Nama Komponen	Keterangan
1	Nama kebutuhan	Nama kebutuhan dari <i>use case scenario</i>
2	<i>Objective</i>	Tujuan dari skenario
3	Aktor	Seorang aktor yang melakukan
4	<i>Pre-Condition</i>	Kondisi awal dari sistem
5	<i>Main Flow</i>	Alur skenario jika sistem berjalan dengan normal
6	<i>Alternative Flow</i>	Alur scenario jika sistem berjalan dengan ada kesalahan
7	<i>Post-Condition</i>	Kondisi setelah skenario telah selesai berjalan

(Sumber : Pressman, 2010)

2.3.3.3 Sequence Diagram

Interaksi antar aktor dan objek dalam sistem dan antara objek itu sendiri digambarkan pada diagram ini. UML memiliki berbagai notasi yang dapat digunakan untuk berbagai jenis interaksi pada *sequence diagram* (Sommerville, 2011). *Sequence diagram* menunjukkan bagaimana peristiwa yang menyebabkan transisi dari objek ke objek yang mengacu pada *usecase scenario*. Diagram ini berguna dalam pembuatan perancangan yang efektif dalam pembangunan sistem (Pressman, 2010). Diagram ini memiliki beberapa notasi yang dapat dilihat pada Tabel 2.3.

Tabel 2.3 Notasi Sequence Diagram

Nama Notasi	Simbol	Deskripsi
<i>Lifeline</i>		Menyatakan kehidupan suatu objek.
Aktor		Digunakan untuk menggambarkan <i>user</i> atau pengguna.
<i>Boundary</i>		Menggambarkan suatu komponen yang mengontrol interaksi sistem dengan pengguna.
<i>Control</i>		Menggambarkan suatu komponen yang mengatur perilaku yang melibatkan <i>entity</i> dan <i>boundary</i> .
<i>Entity</i>		Menggambarkan suatu komponen yang memodelkan <i>database</i> dari sistem.

Nama Notasi	Simbol	Deskripsi
<i>Object Interaction</i>	<p>Message type call</p>	Menggambarkan interaksi antar objek yang diberikan deskripsi pada garis. Interaksi dapat berupa pengiriman data informasi.
<i>Iteration</i>		Menggambarkan iterasi perulangan. Iterasi akan terus melakukan perulangan sampai kondisi perulangan adalah benar.

(Sumber : Omg, 2017)

2.3.3.4 Class Diagram

Class diagram adalah pemodelan UML yang digunakan untuk pendokumentasian perancangan perangkat lunak. Diagram ini menggambarkan struktur dari kelas objek pada sistem dan asosiasi antar kelas (Sommerville, 2011). *Class diagram* menggambarkan hubungan antar objek dan kolaborasi yang terjadi antara kelas-kelas yang ditentukan. Elemen-elemen dalam *class diagram* diantaranya adalah kelas dan objek, atribut dan operasi yang mendefinisikan perilaku objek (Pressman, 2010). *Class diagram* memiliki beberapa notasi yang dapat dilihat pada Tabel 2.4.

Tabel 2.4 Notasi *Class Diagram*

Nama Notasi	Simbol	Deskripsi
<i>Class</i>		Tiap kelas memiliki nama, <i>attribute</i> , dan <i>operation</i> , atau <i>method</i> .
Asosiasi	<hr/>	Hubungan antar kelas dengan pengertian umum.
Generalisasi		Hubungan antara kelas yang lebih umum dan kelas yang lebih spesifik.

--	--	--

(Sumber : Omg, 2017)

2.4 Proyek

2.4.1 Definisi Proyek

Menurut Project Management Institute (PMI) definisi proyek adalah “sebuah perilaku sementara untuk membuat sebuah produk, jasa atau hasil yang bernilai unik” yang artinya proyek adalah suatu upaya atau pekerjaan yang diadakan dalam selang waktu tertentu untuk menghasilkan suatu produk, layanan jasa atau hasil yang unik. Perbedaan proyek dengan pekerjaan lain adalah sifatnya yang khusus dan tidak rutin pengadaannya. Karakteristik proyek diantaranya adalah unik, sementara, adanya konsumen/sponsor, memiliki siklus hidup/fase yang harus dilewati, dan menghasilkan sesuatu yang sesuai harapan dan dapat diserahkan.

Proyek dapat diartikan sebagai suatu pekerjaan yang memiliki tujuan tertentu, diminta oleh pihak tertentu dan dikerjakan dalam waktu tertentu pula. Proyek memiliki waktu awal dan akhir penggerjaan yang harus didefinisikan dengan jelas. Suatu proyek dapat dikatakan selesai apabila telah memenuhi tujuan yang ditentukan atau ketika proyek tidak lagi dibutuhkan. Meskipun proyek bersifat sementara namun tidak demikian dengan hasil dari suatu proyek, ada kalanya produk yang dihasilkan dari suatu proyek digunakan dalam jangka waktu yang lama.

2.4.2 Proyek Perangkat Lunak

Proyek perangkat lunak bukanlah proyek yang hanya terfokus pada pemrograman atau penulisan kode program saja, umumnya teknik manajemen proyek dalam proyek lainnya juga dapat diterapkan untuk mengelola proyek perangkat lunak namun yang membedakan proyek perangkat lunak dengan proyek lainnya adalah pada hasil atau produk yang diselesaikan dan juga karakteristik dari proyek perangkat lunak itu sendiri. Hughes & Cotterell (1999) menyebutkan bahwa proyek ini memiliki tiga karakteristik, yaitu:

1. Invisibility

Perkembangan proyek perangkat lunak bersifat *invisible* (tidak terlihat). Karakteristik ini yang menjadi pembeda proyek perangkat lunak dengan lainnya, dimana perkembangan dari suatu proyek tidak dapat dilihat wujudnya dan tidak selalu dapat dilihat progresnya.

2. Complexity

Kompleksitas dari proyek perangkat lunak terletak pada artefak-artefak atau dokumen yang dihasilkan. Proses rekayasa perangkat lunak lebih rumit daripada proses rekayasa produk lainnya.

3. Flexibility

Fleksibilitas dalam proyek perangkat lunak terletak pada perubahan perangkat lunak, meliputi perubahan kebutuhan, perubahan fungsi dan

lain sebagainya. Perubahan yang mudah dilakukan dalam proyek perangkat lunak dinilai menjadi kelebihan proyek perangkat lunak, hal ini berarti proyek perangkat lunak merupakan subjek yang memiliki kemungkinan perubahan yang tinggi.

Fleksibilitas dalam proyek perangkat lunak dapat disebut sebagai kelebihan proyek perangkat lunak, namun di sisi lain fleksibilitas proyek perangkat lunak dapat menjadi ancaman, ancaman disini adalah karena perubahan yang terjadi pasti memiliki ketidakpastian. Ketidakpastian dalam perubahan proyek perangkat lunak adalah hal yang harus diperhatikan (Munir, 2015). Ketidakpastian pada spesifikasi kebutuhan dapat mempengaruhi proses penggeraan proyek perangkat lunak karena proyek perangkat lunak sangat bergantung pada spesifikasi kebutuhan dan dengan kebutuhan perangkat lunak yang sering berubah maka proses penggeraan dapat terhambat karena jika perubahan yang terjadi mempengaruhi banyak hal yang telah dikerjakan maka harus dilakukan penggeraan ulang untuk pekerjaan tersebut. Pencegahan ketidakpastian adalah dengan membuat rencana dan prediksi hasil proyek dengan akurat.

2.4.3 Stakeholder

Individu atau organisasi yang memiliki peran aktif secara tidak langsung maupun langsung pada sebuah proyek merupakan *Stakeholder* (Project Management Institute, 2013). Secara umum stakeholder dalam proyek terbagi menjadi dua, yaitu pihak klien dan pihak pengembang. *Stakeholder* yang tercatat sidalam sistem yang dikembangkan adalah klien, *PIC/Person in Charge* (penaggung jawab pihak klien) dan tim pengembang. Tim pengembang dalam sistem ini adalah tim dari perusahaan dengan anggota yang memiliki peran sendiri-sendiri yang bertugas sebagai pelaksana proyek.

2.4.3.1 Klien

Klien merupakan individu atau perusahaan yang meminta dan memiliki suatu proyek serta memiliki kepentingan tertinggi terhadap hasil akhir proyek. Untuk itu klien memiliki wewenang untuk menyetujui ataupun menolak produk yang telah dikerjakan oleh pengembang (Project Management Institute, 2013). Selain berperan sebagai pemilik proyek, biasanya klien juga berperan sebagai penyandang dana proyek. Hasil akhir proyek berupa perangkat lunak yang siap pakai yang nantinya akan menjadi milik klien dan menjadi hak klien sepenuhnya.

2.4.3.2 PIC (*Person in Charge*)

PIC (Person in Charge) atau dalam bahasa Indonesia memiliki arti penanggung jawab dalam sistem ini adalah seorang yang tercatat dalam nota kesepakatan antara pihak klien dan pihak perusahaan pengembang. *PIC* bertugas menjadi perwakilan *client* untuk menerima laporan kemajuan proyek dan juga sebagai penghubung yang mengutarakan kebutuhan klien kepada pihak *developer* terkait sistem yang akan dikembangkan.

2.4.3.3 Pelaksana Proyek

Pelaksana proyek biasanya berbentuk sebuah tim yang telah ditentukan oleh perusahaan pengembang. Rudy Tantra (2012) menyebutkan terdapat beberapa peran keahlian yang dilibatkan menjadi pelaksana proyek, diantaranya adalah:

- 1. Manajer proyek**

Manajer proyek adalah seorang yang dipilih oleh perusahaan pengembang untuk memimpin suatu proyek. Seorang manajer proyek harus memiliki kemampuan pemecahan masalah yang baik, kemampuan manajemen yang baik, kemampuan komunikasi yang baik dan juga harus kritis, cepat dan akurat dalam menentukan keputusan.

- 2. Analis Sistem**

Analis sistem adalah seorang yang bertugas untuk menganalisa suatu sistem. Fungsi utama analis sistem adalah melakukan analisis kebutuhan sistem dengan cara wawancara, studi, pengamatan, ataupun simulasi. Setelah informasi yang dicari cukup, analis sistem akan menyerahkan hasil analisa kebutuhannya kepada desainer sistem untuk di lakukan perancangan sistem.

- 3. Desainer Sistem**

Desainer sistem adalah seorang yang bertugas untuk merancang sistem berdasarkan dokumen spesifikasi kebutuhan. Desainer sistem bertanggung jawab untuk merancang struktur sistem, komponen sistem dan tampilan sistem. Umumnya hasil dari desain sistem dituangkan dalam sebuah dokumen yang berisi desain dalam beberapa tingkatan. Dokumen desain sistem ini kemudian diserahkan kepada programmer untuk diimplementasikan menjadi kode program.

- 4. Programmer**

Tugas utama *programmer* adalah mengimplementasikan desain sistem menjadi kode program dengan bahasa pemrograman tertentu. *Programmer* bertanggung jawab untuk menyelesaikan kode program hingga suatu produk perangkat lunak yang selesai.

- 5. Tester**

Tester adalah seorang yang bertugas sebagai penguji sistem, pengujian dilakukan untuk mengetahui apakah perangkat lunak/sistem yang dibuat telah mengikuti desain yang ditentukan, selain itu *tester* juga bertugas untuk menemukan kesalahan seperti *error*, *bug* maupun kesalahan logika. Untuk memudahkan pengujian, biasanya *tester* menyusun suatu *test script* berdasarkan *use case* yang telah dibuat, sehingga *tester* dapat lebih mudah untuk menentukan alur pengujian.

- 6. Anggota lain dalam tim**

Anggota lain dalam tim meliputi instalator/*deployer*, *trainer* dan *technical support*. Sebuah tim proyek perangkat lunak harus memiliki anggota-anggota tambahan ini, diantaranya instalator adalah anggota yang bertugas untuk melakukan instalasi produk yang telah dibuat di lingkungan kerja pengguna sistem, *trainer* dibutuhkan untuk

mendampingi pengguna sistem dalam jangka waktu tertentu setelah sistem diserahkan ke pengguna sistem, *technical support* dibutuhkan untuk melakukan *maintenance* atau pemeliharaan sistem.

2.5 Manajemen Proyek

Dapat dikatakan bahwa manajemen sebuah proyek adalah sebuah proses yang termasuk didalamnya adalah perencanaan proyek, pengaturan proyek yang termasuk didalamnya adalah sumber daya, kepemimpinan yang mumpuni, dan pengendalian proyek oleh setiap anggotanya sehingga dapat mencapai sasaran yang tepat sesuai dengan analisa kebutuhan sebelumnya dan tetap sesuai dengan mutu yang di standarisasikan.

Menurut Project Management Institute (2013) terdapat sembilan bidang yang harus dikelola dengan baik dalam suatu proyek, bidang-bidang tersebut adalah waktu, kualitas, biaya, sumberdaya manusia, komunikasi, risiko, pengadaan, ruang lingkup, dan integrasi.

Bidang waktu dalam manajemen proyek meliputi perencanaan penjadwalan proyek, mengidentifikasi tugas, mengurutkan setiap tugas, memperkirakan sumberdaya setiap tugas, memperkirakan jangka waktu setiap tugas, mengembangkan jadwal proyek, dan mengendalikan jadwal yang telah dibuat. Bidang kualitas dalam perencanaan proyek adalah proses menjaga kualitas proyek agar hasil proyek dapat memenuhi bahkan melebihi ekspektasi *stakeholder*. Bidang biaya terkait dengan asumsi anggaran yang diperlukan selama proyek dilaksanakan, hal ini meliputi penyandang dana yang ada termasuk sponsor. Bidang sumberdaya manusia terfokus untuk membuat anggota tim proyek mengerti tujuan dari proyek dan mampu untuk berperan aktif selama pelaksanaan proyek. Bidang komunikasi terfokus pada menjaga agar informasi yang didapatkan oleh pengembang dan *stakeholder* tetap terkini dan akurat. Bidang risiko terfokus pada analisa risiko-risiko yang ada dan dampaknya terhadap keberlangsungan proyek. bidang pengadaan terfokus menyediakan kebutuhan-kebutuhan proyek selama dilaksanakan, tidak menutup kemungkinan suatu proyek membutuhkan SDM tambahan ataupun kebutuhan akan *software* dan *hardware* tambahan. Bidang ruang lingkup terfokus untuk memastikan apakah proyek dilaksanakan sesuai prosedur, sesuai perencanaan dan sesuai dengan jalur yang telah dibuat. Bidang integrasi terfokus untuk menghubungkan semua fase dari manajemen proyek.

2.5.1 Manajemen Proyek Perangkat Lunak

Jenis manajemen proyek yang memiliki fokus spesifik pada penanganan proyek perangkat merupakan tujuan utama pada manajemen ini. Pada umumnya siklus hidup proyek perangkat lunak sama dengan siklus hidup proyek lainnya yang terdiri atas lima fase yaitu: fase inisialisasi, fase perencanaan, fase pelaksanaan, fase penyerahan/*delivery*, dan fase akhir/penutupan. Perbedaan proyek perangkat lunak dengan proyek lainnya terletak pada pedoman/metode yang digunakan untuk menangani proyek tersebut, pada proyek perangkat lunak

terdapat beberapa metodologi yang dapat digunakan, antara lain: *agile*, *extreme*, *interactive*, *incremental*, dan *phases* (Rudy Tantra, 2012).

2.6 Kanban

2.6.1 Definisi Kanban

Kanban (bahasa Jepang) memiliki beberapa arti yaitu *card* atau *sigboard*, memiliki arti sebuah kartu atau sebuah papan penanda adalah sebuah konsep/metode untuk memvisualisasikan pekerjaan, membuat aliran pekerjaan terlihat dan mengurangi pekerjaan yang sia-sia (Corona, Erika and Filippo Eros Pani, 2013). *Kanban* dikembangkan pertama kali pada tahun 1940 oleh Taiichi Ohno untuk membantu Toyota guna memenuhi efisiensi produksi untuk produk yang spesifik berdasarkan kondisi pasar menurutnya *Kanban* adalah salah satu cara untuk mencapai JIT (*Just in Time*) atau ketepatan waktu. Pada bidang manufaktur, *Kanban* menunjukkan apa, kapan dan berapa jumlah komponen harus diproduksi. Terdapat tiga alasan utama suatu perusahaan menggunakan *Kanban* yaitu: dapat mengurangi biaya pemrosesan informasi, dapat menyediakan fakta dengan cepat dan tepat, dan mampu mengoptimalkan produksi. Toyota Production System mendefinisikan *Kanban* seperti Gambar 2.2.

TPS Kanban	Description
Physical	It is a physical card.
Limit work in progress (WIP)	Prevents overproduction.
Continuous Flow	Provides information about production needs before a line runs out of stock.
Pull	Downstream processes pull items from upstream processes.
Self-Directing	Contains all information on what to do and makes production autonomous in a non-centralised manner and without micro-management. In this way, people can see the status of work at a glance and detect bottlenecks.
Visual	Stacked or posted to visually show task status and progress.
Signal	Visual status signals the next withdrawal or production action.
Kaizen	Visual process flow informs and stimulates Kaizen.

Gambar 2.2 Deskripsi *Kanban* dari Toyota

(Muhammad Ovais Ahmad, 2016)

Pada Gambar 2.2 dijelaskan bahwa *Kanban* memiliki delapan karakteristik. *Physical* yakni menyatakan bentuk fisik *Kanban* yaitu sebuah kartu. *Limit work in progress (WIP)* yaitu membatasi jumlah tugas yang dikerjakan yang bertujuan untuk mencegah produksi yang berlebihan. *Continuous flow*, dimana *Kanban* dapat menyediakan informasi kebutuhan produksi secara berkelanjutan sehingga informasi akan kebutuhan produksi akan terus diperbarui sebelum perusahaan kekurangan persediaan. *Pull* yaitu menarik tugas yang tersisa setelah suatu pekerjaan selesai. *Self-directing* yang artinya *Kanban* menyediakan semua informasi tentang apa yang harus dilakukan dan dapat mengetahui status tiap

tugas sehingga apabila suatu tugas terhenti pada suatu tahap dapat dideteksi dengan mudah. *Visual, Kanban* memvisualisasikan status dan progres suatu tugas. *Signal, Kanban* dapat memvisualisasikan pekerjaan apa yang selanjutnya harus dikerjakan. *Kaizen* berarti terus mengembangkan dan menyempurnakan diri sendiri sehingga dapat membantu perusahaan untuk lebih berkembang lagi.

Dalam pengembangan perangkat lunak, *Kanban* dapat dimanfaatkan untuk memvisualisasikan berbagai fase proses pengembangan, dengan diketahuinya semua tugas dan tahap tugas tersebut maka pemantauan proyek dapat dilakukan dengan lebih mudah (Alqudah, Mashal & Razali, Rozilawati, 2017). Tidak seperti Scrum, dimana Scrum memiliki artefak-artefak yang rumit dan adanya penekanan pada pertemuan rutin tim pengembang, *Kanban* tidak memiliki artefak khusus dan juga tidak harus mengadakan pertemuan rutin anggota tim. Prinsip utama Kanban adalah visualisasi alur kerja melalui papan Kanban, membatasi pekerjaan yang sedang berjalan dengan meminimalkan jumlah fitur yang akan diterapkan, manajemen dan pengukuran aliran, membuat kebijakan yang jelas, menerapkan umpan balik, mengulang dan meningkatkan kolaborasi secara berkelanjutan. Praktik Kanban dan definisinya disajikan pada sub bab 2.6.2.

2.6.2 Prinsip Kerja *Kanban* dan Definisinya

2.6.2.1 Visualize the Workflow

Prinsip ini artinya adalah membuat seluruh tugas/pekerjaan menjadi dapat dilihat oleh seluruh anggota tim, apabila terdapat suatu tugas yang tidak terlihat maka akan berdampak pada implementasi proyek dimana tidak terlihatnya suatu tugas akan menimbulkan tidak lengkapnya suatu proyek. Keseluruhan tugas dapat terlihat dengan jelas berkat adanya *Kanban board*.

2.6.2.2 Limit Work in Progress

Prinsip ini adalah prinsip utama penerapan *Kanban*, yaitu membatasi jumlah pekerjaan yang sedang berjalan, dengan dibatasinya jumlah pekerjaan tim dapat lebih fokus untuk menyelesaikan tugas yang diterimanya. Terbukti apabila multitasking berkurang maka waktu penggerjaan tugas-tugas juga berkurang dan dengan ini waktu *deliver* produk dapat dicapai dengan lebih cepat.

2.6.2.3 Manage and Measure Flow

Prinsip ini memiliki tujuan utama untuk menyelesaikan pekerjaan yang sedang berlangsung dengan cara setiap tugas dalam tiap tahap dipantau dan dilaporkan, hal ini adalah yang disebut *measure flow*. Akibat dari monitoring yang intensif tiap tugas dapat diselesaikan tepat waktu sehingga dapat mengurangi risiko dan juga menghindari penambahan biaya akibat keterlambatan.

2.6.2.4 Make Policies Explicit

Prinsip ini menjelaskan bahwa dalam penerapan *Kanban* aturan-aturan dalam perusahaan harus dituliskan secara eksplisit sehingga seluruh anggota tim dapat memahami kebutuhan perusahaan dan juga dapat berkontribusi untuk menjaga konsistensi perusahaan.

2.6.2.5 Implement Feedback

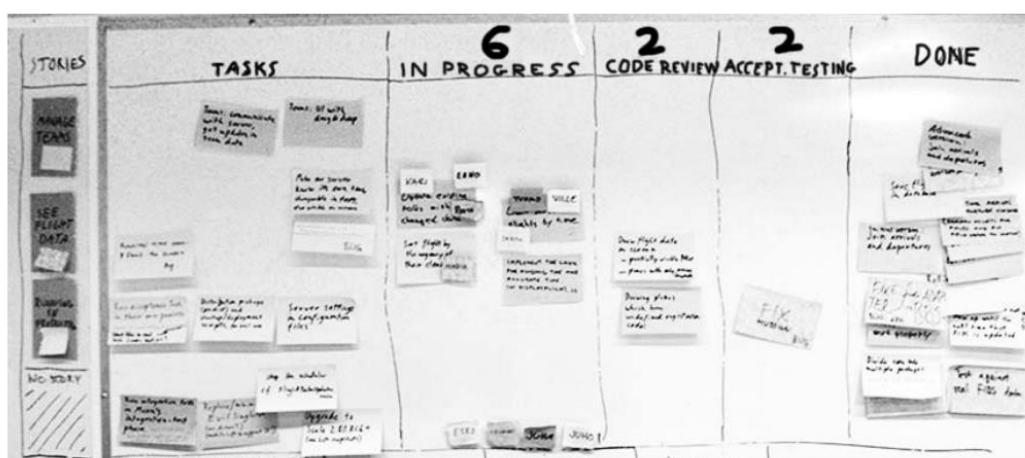
Kanban membutuhkan umpan balik untuk bekerja dan karenanya, perusahaan dapat menerapkan penggunaan *standup meeting*, *review* produk yang telah diserahkan, *review* operasi, dan *review* risiko untuk memungkinkan perbandingan antara hasil yang diharapkan dan hasil sebenarnya, selain itu umpan balik/*feedback* diperlukan untuk membuat penyesuaian yang diperlukan.

2.6.2.6 Continuous Improvement

Prinsip ini berakar pada pola pikir “*Kaizen*” yang telah disinggung sebelumnya, dimana perusahaan harus tetap berkembang dan memingkatkan kinerja dan kualitasnya secara terus menerus atau secara berkesinambungan.

2.6.3 Artefak *Kanban*

Telah disinggung sebelumnya bahwa *Kanban* tidak memiliki artefak khusus, namun dalam praktiknya tentunya *Kanban* memiliki komponen agar penerapannya dapat dilakukan, komponen tersebut antara lain adalah: 1. *Kanban board*; 2. *Kanban card*; 3. Limit/batas *work in progress* (*WIP*).



Gambar 2.3 Penerapan *Kanban*

Ikonen, Marko & Pirinen, Elena & Fagerholm, Fabian & Kettunen, Petri & Abrahamsson, Pekka. (2011)

2.6.3.1 *Kanban Board*

Kanban board merepresentasikan tahap-tahap yang harus dilewati oleh suatu task/tugas sebelum tugas tersebut dinyatakan selesai. Perhatikan Gambar 2.3, bidang yang terbagi menjadi lima bagian tersebut disebut dengan *Kanban board*.

Pada bagian paling kiri adalah *board* bernama *task*, dimana *board* ini berisi semua *task/tugas* yang telah didefinisikan oleh manajer proyek dan belum dikerjakan oleh tim proyek, di sebelah kanannya terdapat *board* bernama “*in progress*” dimana *board* ini menampung tugas yang saat ini dalam tahap pengerjaan, berlanjut hingga *board* paling kanan dengan nama “*done*”, *board* ini menampung semua *task* yang sudah selesai melewati semua tahap/*kanban board*.

2.6.3.2 Kanban Card

Kanban card merepresentasikan tugas-tugas yang harus dikerjakan oleh tim proyek. Dapat diamati pada Gambar 2.3 bahwa pada tiap-tiap *kanban board* terdapat suatu kartu yang bertuliskan suatu hal. Kartu-kartu ini bersifat dinamis dimana ketika pada ia didefinisikan ia akan menempati *board* bernama *task* dan apabila tim telah siap untuk mengerjakan *task* tersebut, maka salah kartu berisi *task* tersebut dapat dipindahkan pada *board* bernama “*in progress*” dan seterusnya sampai kartu menempati *board* “*done*”. Perlu diketahui bahwa suatu *card/tugas* dapat dipindahkan mundur, hal ini dapat terjadi semisal terjadi suatu masalah pada sebuah *task* maka manajer proyek dapat memindahkan *card* tersebut ke tahap sebelumnya.

2.6.3.3 Limit Work in Progress (WIP)

Limit Work in Progress (WIP) adalah sebuah angka yang menyatakan batas jumlah *kanban card* yang boleh menempati suatu *board*. Dapat diamati pada Gambar 2.3 bahwa diatas tiap *board* tertulis angka yang menjadi batas jumlah *card* yang boleh diletakkan di *board* tersebut.

2.7 Konsep MVC

MVC adalah sebuah konsep dalam pembuatan aplikasi yang menggunakan teknik pemisahan antara pengolah data, tampilan antarmuka dan pengontrol utama sistem. Konsep ini dapat digunakan ketika suatu perangkat lunak yang akan dibangun, terdapat banyak interaksi dengan data dan penyajian data tidak diketahui. Keuntungan dari konsep ini adalah memungkinkan perubahan data secara independen, mendukung penyajian data yang sama dengan cara yang berbeda. Sedangkan kelemahan dalam konsep ini adalah melibatkan kode tambahan ketika terdapat model data dan interaksi yang sederhana (Sommerville, 2011). Pada MVC, dibagi menjadi 3 bagian, yaitu *Model*, *View* dan *Controller*.

2.7.1 Model

Model dalam MVC memiliki peran dalam mengelola data sistem dan operasi data yang terkait (Sommerville, 2011). Di dalam *model* berisi logika pemrosesan yang digunakan untuk mengakses sumber data eksternal, konten yang digunakan untuk sistem, dan fungsi pemrosesan yang spesifik untuk sistem. *Model* dapat melakukan pengaksesan data yang disimpan di dalam *database* (Pressman, 2012).

2.7.2 View

View dalam MVC memiliki peran dalam mendefinisikan dan mengelola data yang disajikan kepada pengguna yang diperintah oleh *controller* (Sommerville, 2011). Data yang diperoleh dari *model* harus diformat dan diatur yang sesuai dengan format, kemudian ditransmisikan dari server kembali ke *browser* yang diakses oleh klien untuk ditampilkan (Pressman, 2012).

2.7.3 Controller

Controller dalam MVC memiliki peran dalam menghubungkan antara *model* dan *view*. *Controller* akan memberikan perintah kepada *model* untuk mengakses data dan akan memberikan perintah kepada *view* untuk menampilkan data dan tampilan yang dibutuhkan oleh pengguna (Sommerville, 2011).

2.8 Teknologi Pengembangan Sistem

2.8.1 Codeigniter Framework

CodeIgniter adalah sebuah kerangka kerja yang digunakan dalam pengembangan web yang menggunakan PHP dan dibangun dengan menggunakan konsep MVC. Pada *framework* ini sudah tersedia kerangka yang dapat digunakan oleh pengembang. Kelebihan dari *framework* ini adalah memungkinkan pengembang untuk mengembangkan proyek lebih cepat, karena tidak perlu menuliskan kode dari awal (CodeIgniter, 2019). Penggunaan MVC dari *framework* ini sangat membantu pada proses pengembangan perangkat lunak. Selain mempercepat dan mempermudah pengerjaan pengembangan perangkat, *framework* ini juga sangat ringan dan cepat sehingga tidak membebani user yang nantinya akan menggunakan produk yang sedang dibangun. Pembuat PHP pertama juga memuji *framework* di frOSCon (Agustus 2008) dengan mengatakan bahwa dia menyukai CodeIgniter karena “*it is faster, lighter and the least like a framework*”.

2.8.2 Bootstrap Framework

Framework Bootstrap adalah kerangka kerja yang digunakan dalam perancangan *front-end* dan bersifat terbuka untuk umum. Pada *framework* ini sudah tersedia *template* yang dapat langsung digunakan oleh pengembang dalam melakukan perancangan *front-end*. *Template* yang disediakan pada *framework* ini berbasis HTML, CSS dan JavaScript. Kelebihan dari *framework* ini adalah memudahkan dan mempercepat pengembang dalam mengembangkan *front-end* pada perangkat lunak, dengan memakai *template* yang telah disediakan dan siap dipakai (Bootstrap, 2020).

2.8.3 SB Admin 2 Framework

SB Admin 2 *Framework* adalah kerangka kerja tidak berbayar dan juga *open source* yang dikembangkan oleh Bootstrap. Dengan dasar komponen HTML, CSS, Javascript serta Bootstrap 4, *framework* SB Admin 2 dapat menghadirkan suatu

tema tampilan *dashboard* admin yang modern dan siap digunakan dalam pengembangan suatu aplikasi web. (Bootstrap, 2020)

2.8.4 Javascript

Javascript adalah suatu bahasa pemrograman berbasis teks yang dapat digunakan baik pada sisi klien maupun sisi server dan dapat digunakan untuk membuat halaman web menjadi semakin interaktif. javascript didesain oleh Brendan Eich dan juga kontributor lainnya yang tergabung dalam perusahaan Netscape Communications Corporation dan Mozilla Foundation. Javascript dapat secara dinamis melakukan perubahan pada suatu halaman web sehingga halaman web tidak menjadi suatu halama statis, perubahan dapat dilakukan baik pada konten maupun pada tampilan yang disajikan sesuai dengan keinginan pengguna (Hack Reactor, 2019)

2.8.5 Google Charts

Google Charts adalah suatu teknologi berupa *library* yang dapat digunakan untuk memvisualisasikan atau merepresentasikan data pada suatu web dalam bentuk diagram lingkaran, diagram batang, grafik gantt dan juga jenis diagram/grafik lainnya. *Library* ini dikembangkan dengan menggunakan bahasa pemrograman Javascript, cara penggunaan Google Charts adalah dengan memuat *library* Google Charts, kemudian melakukan persiapan data yang akan ditampilkan, selanjutnya adalah memilih atau membuat objek bagan pada sebuah halaman web dengan id yang kita inginkan dan terakhir adalah pembuatan `<div>` dengan id yang tadi telah kita buat. (Google Chart, 2020)

2.9 Pengujian Perangkat Lunak

Pengujian dilakukan untuk mengetahui apakah program yang elah dibangun sebelumnya sesuai dengan analisa kebutuhan yang telah disepakati sebelumnya dan apakah pada program terjadi kesalahan atau *error* yang nantinya akan dapat diperbaiki sebelum *deploy* program ke pengguna.

2.9.1 White-box Testing

White-box *testing* merupakan pengujian yang difokuskan pada internal sistem yaitu *source code* program. Tujuan dari pengujian White-box digunakan sebagai alat uji kompleksitas dari kode program. Bagi *programmer*, White-box sangat penting untuk menentukan kompleksitas dari suatu kode. Pengujian White-box juga dapat digunakan sebagai validasi apakah *source code* mengikuti desain; apakah *source code* sesuai dengan kebutuhan fungsional; dan apakah *source code* memiliki kerentanan. (M. Kumar, S. K. Singh, and R. . Dwivedi, 2015)

2.9.2 Black-box Testing

Pengujian Black-box biasanya digunakan untuk menguji pekerjaan internal aplikasi tanpa pengetahuan pemrograman. Pengujian Black-box digunakan untuk

menguji fungsional maupun input dan output suatu aplikasi. Teknik pengujian ini ditujukan kepada para penguji yang tidak memiliki pemahaman dalam pemrograman. Biasanya, pengujian Black-box diterapkan di semua level pengujian perangkat lunak seperti: *unit testing*, *integration testing*, *functional testing* dan *acceptance testing*. Fokus utama dalam pengujian Black-box adalah mengetahui input dalam sistem; luaran yang diharapkan; dan hasil nyata berdasarkan input dari program. (M. Kumar, S. K. Singh, and R. . Dwivedi, 2015)

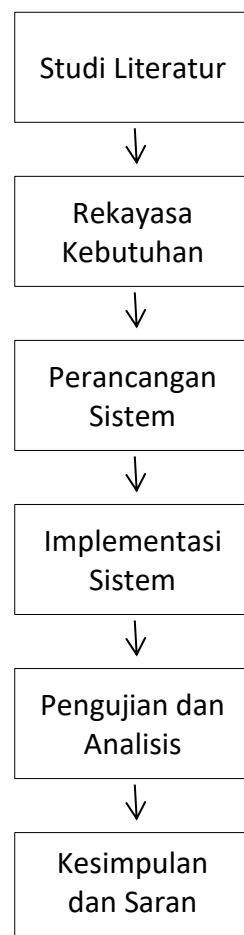
2.9.3 Usability Testing

Usability testing pada penelitian ini dilakukan untuk mengetahui seberapa efektif dan efisien sistem ketika digunakan. *Usability testing* adalah metode dimana pengguna dari suatu perangkat lunak diminta untuk melakukan tugas-tugas tertentu dalam upaya untuk mengukur kemudahan penggunaan perangkat lunak, waktu, dan persepsi pengguna tentang pengalaman dalam menggunakan perangkat lunak. *Usability* sendiri memiliki 5 komponen kualitas yaitu efisiensi, efektifitas, mudah dipelajari, mudah diingat dan memberi kepuasan kepada pengguna. (Nielsen, 2020). *Usability testing* dapat dilakukan secara resmi ataupun secara informal seperti hanya dengan menggunakan kertas mockup aplikasi atau situs web (Niranjaanamurthy, 2014).

Terdapat beberapa metode yang dapat digunakan untuk melakukan pengujian usabilitas, salah satu metode tersebut adalah metode SUS (*System Usability Scale*). Metode SUS terdiri dari 10 buah pertanyaan yang terbagi menjadi dua jenis pertanyaan. Terdapat pertanyaan dengan nomor ganjil (1, 3, 5, 7, 9) yang berisikan pertanyaan bersifat positif. Kemudian pertanyaan dengan nomor genap (2, 4, 6, 8, 10) yang berisikan pertanyaan bersifat negatif. Skala penilaian untuk setiap pertanyaan adalah rentang 1 sampai 5. Semakin tinggi penilaian yang diberikan responden maka semakin setuju responden terhadap pernyataan yang ada. Penilaian dari kedua jenis pertanyaan tersebut diberikan bobot yang berbeda. Untuk setiap pertanyaan ganjil, nilai yang diberikan oleh responden akan dikurangi 1 (satu) nilai. Sementara untuk setiap pertanyaan genap, 5 (lima) nilai akan dikurangi dengan nilai yang diberikan oleh responden. Setelah itu total skor dikalikan dengan 2.5 dan akan didapatkan skor akhir dari metode SUS. Skor akhir menggunakan metode SUS akan berada pada rentang 0 hingga 100 dengan tiga jenis kategori didalamnya yaitu *Not Acceptable* (0-50.9), *Marginal* (51-70.9), dan *Acceptable* (71-100). Alasan metode SUS ini dipilih adalah karena tidak memerlukan jumlah sampel yang besar sehingga tidak memerlukan biaya dan waktu yang besar (Brooke, 2011).

BAB 3 METODOLOGI

Pada bab metodologi ini memuat dan menjelaskan langkah-langkah yang dilakukan untuk melaksanakan penelitian “Perancangan Dan Implementasi Sistem Informasi Manajemen Proyek Menggunakan Metode *Kanban* Berbasis Web (Studi Kasus CV. Primavisi Globalindo Surabaya)”. Pertama-tama yang harus dilakukan adalah melakukan studi literatur, mencari dan memilih literatur yang sesuai merupakan hal penting yang dapat mendukung kelancaran penelitian. Kemudian melakukan rekayasa kebutuhan, dimana hasil dari tahap ini akan digunakan sebagai dasar untuk melakukan perancangan sistem. Setelah melakukan perancangan selanjutnya adalah melakukan implementasi berdasar pada rancangan yang telah dibuat sesuai dengan kebutuhan yang didapat dalam proses rekayasa kebutuhan. Pengujian dan analisa sistem adalah langkah terakhir dalam penelitian ini, langkah ini dilakukan guna menemukan kesimpulan dan saran yang dapat digunakan sebagai catatan atau evaluasi atas sistem dimana penilaian ini dapat digunakan untuk jalur pengembangan aplikasi mendatang. Adapun langkah – langkah metodologi adalah sebagai yang tercantum pada Gambar 3.1.



Gambar 3.1 Tahapan Metodologi Penelitian

3.1 Studi Literatur

Merupakan sebuah studi untuk menentukan bahwa setiap kebutuhan yang disertakan dimengerti dan dibutuhkan pada sebuah proyek pengembangan perangkat lunak. Studi literatur yang digunakan pada penelitian ini meliputi :

1. Buku dan Penelitian Terdahulu
2. Rekayasa Perangkat Lunak
3. Pengembangan Perangkat Lunak
 - a. Model Pengembangan Perangkat Lunak (*SDLC*)
 - b. Pemodelan *Unified Modelling Language (UML)*
 - c. Pendekatan *Object Oriented*
4. Proyek
 - a. Definisi Proyek
 - b. Proyek Perangkat Lunak
 - c. Stakeholder
5. Manajemen Proyek
 - a. Manajemen Proyek Perangkat Lunak
6. *Kanban*
 - a. Definisi *Kanban*
 - b. Prinsip Kerja *Kanban* dan Definisinya
 - c. Aftefak *Kanban*
7. Konsep MVC
 - a. *Model*
 - b. *View*
 - c. *Controller*
8. Teknologi Pengembangan Sistem
 - a. Codeigniter *Framework*
 - b. Bootstrap *Framework*
 - c. SB Admin 2 *Framework*
 - d. Javascript
 - e. Google Charts
9. Pengujian Perangkat Lunak
 - a. White-box *Testing*
 - b. Black-box *Testing*
 - c. Usability *Testing*

3.2 Rekayasa Kebutuhan

Dilaksanakan untuk mengerti kebutuhan yang nantinya akan dibutuhkan pada pengembangan perangkat lunak yang akan dikerjakan, dimana dilakukan hipotesa akan kebutuhan yang dibutuhkan, lalu mengeleminasi dan menambahkan kebutuhan yang sesuai dengan kebutuhan asli dari perangkat lunak. Proses rekayasa kebutuhan sistem ini adalah sebagai berikut:

1. Proses Elitisasi Kebutuhan

Penulis melakukan elitisasi kebutuhan dengan teknik observasi. Penulis melakukan observasi untuk mengetahui secara langsung keadaan dari

CV. Primavisi Globalindo dan mengetahui gambaran proses bisnis yang terjadi di lapangan. Pada tahap observasi ini penulis menggunakan *direct observation* terhadap masalah yang dihadapi guna mendapatkan data yang dibutuhkan dalam penelitian.

2. Proses Spesifikasi Kebutuhan

Setelah elistisasi kebutuhan, dilakukan sebuah proses untuk mendefinisikan sebuah kebutuhan menjadi kebutuhan yang lebih spesifik. Dalam tahap ini juga dilakukan identifikasi aktor yang akan melakukan sebuah kebutuhan yang telah di spesifikasikan pada tahap ini.

3. Proses Validasi Kebutuhan

Merinci dan mengecek apakah setiap kebutuhan yang telah didefinisikan sebelumnya, apakah setiap kebutuhan ini benar-benar dibutuhkan pada pengembangan perangkat lunak yang akan dikerjakan.

4. Proses Manajemen Kebutuhan

Proses manajemen kebutuhan ialah proses untuk melakukan pengendalian/kontrol atas kebutuhan yang telah didefinisikan serta memberikan penomoran pada setiap kebutuhan yang didapatkan, mekanisme penomoran dibahas lebih rinci pada Bab 4 Rekayasa Kebutuhan.

3.3 Perancangan Sistem

Pada tahap perancangan akan dibagi menjadi dua jenis perancangan, yaitu perancangan arsitektur dan perancangan komponen. Perancangan arsitektur akan menghasilkan *sequence diagram* dan *class diagram*. *Sequence diagram* menggambarkan interaksi antar komponen yang ada dalam sistem, interaksi antar komponen didapat dari *use case scenario* yang telah didefinisikan sebelumnya. *Class diagram* menggambarkan hubungan antara satu *class* dengan lainnya, definisi *class* didapat dari pengelompokan objek yang memiliki karakteristik yang sama. Perancangan komponen terdiri atas dua jenis perancangan, yaitu perancangan algoritma dan perancangan antarmuka. Perancangan algoritma adalah perancangan yang terfokus untuk merancang langkah kerja atau algoritma suatu fungsi, hasil dari perancangan algoritma adalah suatu *pseudocode*. Perancangan antarmuka menghasilkan *wireframe* yang dirancang sesuai spesifikasi kebutuhan. Perancangan antarmuka berisi penataan fitur-fitur yang disediakan oleh sistem untuk ditampilkan pada layar pengguna nantinya.

3.4 Implementasi

Pada tahap ini dilakukan implementasi terhadap perancangan yang telah dilakukan sebelumnya menjadi sebuah program komputer. Tahapan-tahapan dalam melakukan implementasi sistem diantaranya adalah sebagai berikut :

1. Implementasi kode program dilakukan menggunakan bahasa pemrograman PHP dengan *framework* Codeigniter 3.1.10, HTML, CSS, Javascript, dan *framework* Bootstrap.

2. Implementasi basis data dilakukan menggunakan *Database Management System* (DBMS) MySQL pada server *localhost* untuk memudahkan manipulasi dan juga penyimpanan data.
3. Implementasi akses sistem dengan menggunakan *browser* Mozilla Firefox dan Google Chrome.

3.5 Pengujian dan Analisis

Tahap ini dilaksanakan untuk mengetahui apakah program yang telah diimplementasikan telah diimplementasikan dengan baik dan tidak memiliki masalah. Sehingga ketika proyek telah selesai, tidak ditemukan kesalahan berarti pada perangkat lunak yang telah dibangun. Pengujian ini meliputi :

1. Pengujian Unit / *White-box Testing*

Pengujian ini dilakukan pada unit terkecil pada sistem, dimana mengecek setiap kelas dan objek pada sistem sehingga setiap kelas dan sistem pada perangkat berjalan dengan sebagaimana mestinya. Pengujian unit akan dilakukan pada tiga sampel uji.

2. Pengujian Validasi / *Black-box Testing*

Dilaksanakan untuk mengetahui apakah program dapat berjalan sesuai dengan fungsi yang ditetapkan, dan tidak memiliki kesalahan ketika melakukan suatu fungsi tertentu.

3. *Usability Testing*

Usability Testing dilakukan untuk mengukur seberapa mudah sistem untuk digunakan. Pengujian *usability* dilakukan dengan menggunakan metode SUS (*System Usability Scale*). Pengujian ini dilakukan dengan cara memberikan gambaran sistem berupa *mockup* atau *screenshot* beberapa halaman sistem kepada pengguna dengan tujuan pengguna dapat memperkirakan kemudahan sistem saat digunakan, kemudian pengguna diminta untuk mengisi sebuah kuesioner dan dari jawaban kuesioner tersebut yang akan dikalkulasi sebagai nilai *usability* atau kemudahan sistem.

Analisis kemudian dilakukan pada hasil dari pengujian untuk mengetahui apakah setiap fungsi, kelas, objek yang telah diimplementasikan sudah memenuhi syarat yang telah ditentukan atau tidak, dan apakah perangkat yang telah dibangun sesuai dengan kebutuhan yang telah disepakati sebelumnya.

3.6 Kesimpulan dan Saran

Setelah melakukan pengujian dan analisis terhadap sistem, kemudian dilakukan penarikan kesimpulan dan saran. Kesimpulan dapat diperoleh berdasarkan hasil pengujian dan analisis. Karena adanya kesimpulan maka dapat pula diperoleh inti dari hasil dari keseluruhan proses penelitian, selain itu dari kesimpulan dapat digali dan dievaluasi agar dapat menghasilkan suatu saran terhadap penelitian yang telah dilakukan agar dapat digunakan oleh peneliti lainnya.

BAB 4 REKAYASA KEBUTUHAN

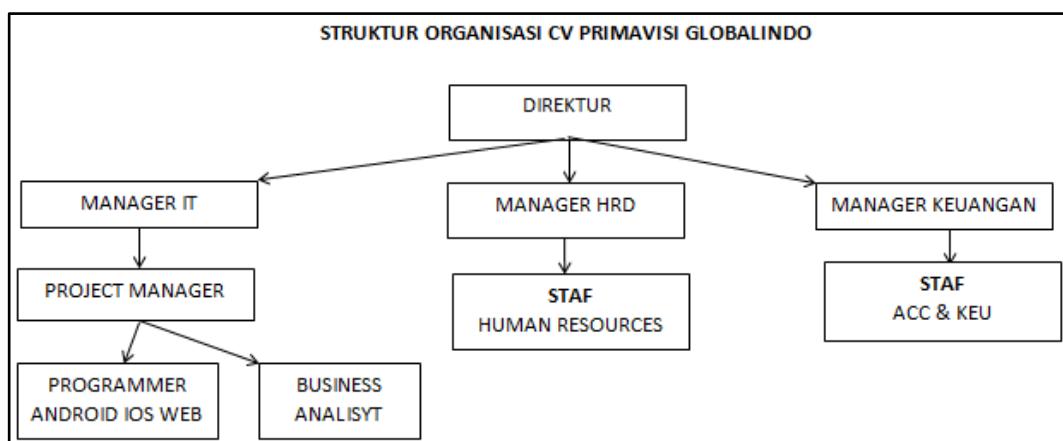
4.1 Gambaran Umum Sistem

Sistem yang dikembangkan merupakan sistem manajemen proyek perangkat lunak. Sistem ini memiliki kegunaan untuk membantu mengendalikan jalannya proyek perangkat lunak dimana sistem ini memiliki fitur untuk menambah data proyek, menghapus data proyek, mengubah data proyek, manipulasi data *client*, manipulasi data tugas di tiap proyek dan juga manipulasi data *user* yang terdaftar dalam sistem. Untuk dapat menggunakan sistem ini pengguna diharuskan *login* terlebih dahulu, pada sistem ini pengguna yang dapat *login* hanyalah pegawai yang telah dibuatkan akun oleh admin perusahaan.

Cara elisitasi yang dilakukan untuk mengembangkan sistem ini adalah dengan cara observasi. Observasi dilakukan pada sistem sebelumnya yang telah digunakan oleh perusahaan, yaitu sistem *Orange Scrum*. Berbeda dengan sistem yang sudah digunakan sebelumnya oleh perusahaan, sistem ini mengadopsi penggunaan *Kanban* untuk manajemen proyek dan juga pengembangan sistem ini dilakukan khusus untuk perusahaan sehingga berbeda dengan sistem sebelumnya yaitu pada sistem ini tidak diperlukan biaya berlangganan.

Dikarenakan sistem ini mengadopsi *Kanban*, maka aplikasi ini memiliki fitur-fitur yang mendukung dalam melakukan aktivitas sesuai dengan metode *Kanban*, seperti: membuat *board*, menghapus *board*, membuat kartu tugas, membatasi jumlah tugas di tiap *board*, memindahkan kartu tugas dari *board* satu ke *board* lainnya, dan menghapus kartu tugas. Sistem ini juga dilengkapi dengan fitur pencatatan data *client* untuk mengetahui siapa yang meminta suatu proyek dan juga siapa saja *client* yang dilayani oleh perusahaan.

4.2 Identifikasi Pengguna



Gambar 4.1 Struktur Organisasi CV. Primavisi Globalindo

Proses identifikasi aktor dilakukan berdasarkan pada struktur organisasi perusahaan. Struktur organisasi perusahaan CV. Primavisi Globalindo Surabaya dapat dilihat pada Gambar 4.1. Berdasarkan struktur organisasi tersebut

didapatkan empat pengguna dalam sistem ini, diantaranya adalah: *guest*, *admin*, *manajer proyek*, dan *pegawai*. Tabel 4.1 merupakan tabel yang memaparkan karakteristik dari tiap pengguna.

Tabel 4.1 Identifikasi Aktor

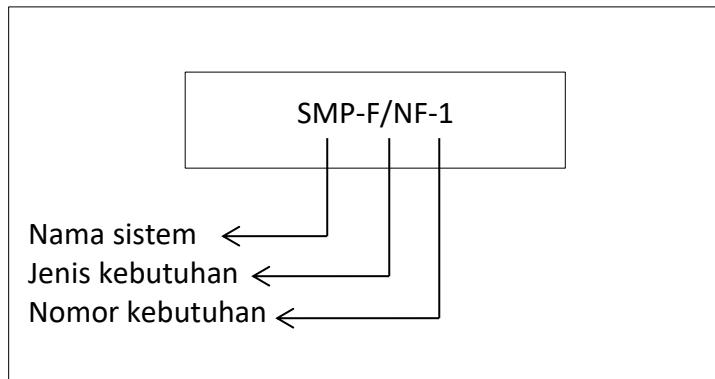
No.	Aktor	Karakteristik
1.	<i>Guest</i>	Merupakan pengguna tanpa hak akses atas fitur sistem, pengguna ini hanya dapat melakukan <i>login</i> .
2.	Pegawai	Merupakan pengguna dengan level terendah, dimana hak akses aktor ini dalam sistem hanyalah sebatas melihat data-data yang ada dan melakukan <i>update</i> tugas yang diberikan kepadanya. Aktor ini dalam struktur organisasi adalah staf human resource, staf acc & keuangan, programmer, dan analis bisnis.
3.	Manajer Proyek	Merupakan pengguna yang memiliki hak akses dibawah admin, dimana aktor ini memiliki hak untuk mengelola proyek yang ditugaskan kepadanya. Termasuk penugasan sebuah tugas kepada pegawai (programer dan analis bisnis).
4.	Admin	Merupakan pengguna dengan level tertinggi, dimana aktor ini memiliki hak untuk menggunakan semua fungsional yang disediakan oleh sistem, seperti membuat proyek, memperbarui data proyek, menghapus proyek, mencari data proyek, mengurutkan data proyek dan juga melakukan manipulasi data akun pegawai. Aktor ini dalam struktur organisasi adalah direktur perusahaan.

4.3 Daftar Kebutuhan Fungsional

Setelah proses elisitasi kebutuhan dilakukan, maka tahap selanjutnya adalah merumuskan fungsi-fungsi yang harus tersedia dalam sistem, dari proses elisitasi kebutuhan yaitu berupa observasi telah didapatkan aktor-aktor yang terlibat dalam sistem dan sejumlah kebutuhan fungsional yang harus terdapat dalam sistem yang dijelaskan dalam Tabel 4.2. Penamaan/pengkodean kebutuhan sistem ini dapat dilihat pada Gambar 4.1.

4.3.1 Teori Kode Kebutuhan

Teori kode kebutuhan digunakan untuk memberikan standarisasi aturan penomoran pada kebutuhan fungsional ataupun kebutuhan non fungsional. Aturan penomoran ini berfungsi untuk mempermudah dalam memahami spesifikasi dari daftar kebutuhan. Sebagaimana dijelaskan pada Gambar 4.1, SMP mewakili singkatan dari sistem ini yaitu Sistem Manajemen Proyek, F/NF membedakan antara kebutuhan fungsional dan non fungsional, dan nomor di belakang untuk mengetahui urutan dari kebutuhan tersebut.



Gambar 4.1 Aturan Penomoran

4.3.2 Kebutuhan Fungsional

Kebutuhan fungsional merupakan layanan yang disediakan oleh sistem beserta prosesnya, layanan tersebut dapat digunakan oleh pengguna sistem. Kebutuhan fungsional didapat dari hasil elisitasi kebutuhan yang telah dilakukan. Dari hasil elisitasi kebutuhan sistem ini memiliki 49 kebutuhan. Nomor, *use case*, spesifikasi dan kode dari kebutuhan fungisional akan ditampilkan pada Tabel 4.2.

Tabel 4.2 Kebutuhan Fungsional Sistem

No.	Use Case	Deksripsi	Kode
1	<i>Login</i>	Sistem harus dapat mengidentifikasi aktor yang login sebagai admin, manajer proyek ataupun pegawai.	SMP-F-1
2	Lupa Password	Sistem menyediakan fitur Lupa Password untuk pengguna.	SMP-F-2
3	<i>Logout</i>	Sistem harus dapat mengeluarkan pengguna dari sistem.	SMP-F-3
4	Melihat Dashboard	Sistem harus dapat menampilkan halaman <i>dahsboard</i> pada <i>user</i> .	SMP-F-4
5	Mencari Proyek	Sistem harus menyediakan fitur pencarian proyek pada halaman <i>dashboard</i> .	SMP-F-5
6	Menambah Data Proyek	Sistem harus dapat menambahkan data proyek baru kedalam sistem.	SMP-F-6
7	Melihat Proyek	Sistem harus dapat menampilkan daftar proyek yang ditangani oleh perusahaan.	SMP-F-7
8	Mengubah Data Proyek	Sistem harus dapat mengubah data proyek.	SMP-F-8
9	Menghapus Data Proyek	Sistem harus dapat menghapus data proyek.	SMP-F-9
10	Melihat Detail Proyek	Sistem harus dapat menampilkan rincian/detail dari suatu proyek berupa daftar tugas, <i>Kanban</i> , <i>gantt chart</i> , tim proyek dan dokumen proyek.	SMP-F-10
11	Menambah Tugas	Sistem harus dapat menambahkan data tugas dalam suatu proyek.	SMP-F-11
12	Melihat Daftar	Sistem harus dapat menampilkan daftar	SMP-F-12

No.	Use Case	Deksripsi	Kode
	Tugas	tugas dari suatu proyek.	
13	Mengupdate Tugas	Sistem harus dapat memperbarui data suatu tugas.	SMP-F-13
14	Menghapus Data Tugas	Sistem harus dapat menghapus data suatu tugas.	SMP-F-14
15	Melihat Detail Tugas	Sistem harus dapat menampilkan detail dari suatu tugas dalam suatu proyek.	SMP-F-15
16	Menambah Feedback Tugas	Sistem harus dapat menyimpan <i>feedback</i> terhadap suatu tugas	SMP-F-16
17	Melihat Feedback Tugas	Sistem harus dapat menampilkan <i>feedback</i> yang ada dalam suatu tugas	SMP-F-17
18	Menghapus Feedback Tugas	Sistem harus dapat menghapus <i>feedback</i> pada suatu tugas	SMP-F-18
19	Menambah Board	Sistem harus dapat menambahkan data <i>board</i> dalam suatu proyek.	SMP-F-19
20	Melihat Kanban Board	Sistem harus dapat menampilkan <i>Kanban board</i> suatu proyek beserta tugas-tugas yang menempatinya.	SMP-F-20
21	Mengubah Limit	Sistem harus dapat mengubah data <i>limit</i> tugas pada sebuah <i>board</i> .	SMP-F-21
22	Menghapus Board	Sistem harus dapat menghapus <i>board</i> dalam suatu proyek.	SMP-F-22
23	Melihat Gantt Chart Proyek	Sistem harus dapat menampilkan linimasa tugas suatu proyek dalam bentuk <i>gantt chart</i> .	SMP-F-23
24	Melihat Pie Chart Proyek	Sistem harus dapat menampilkan prosentase penyelesaian tugas suatu proyek dalam bentuk <i>pie chart</i> .	SMP-F-24
25	Mengupload Dokumen	Sistem harus dapat mengunggah dokumen dengan format yang telah ditentukan, yaitu doc, excel, dan pdf.	SMP-F-25
26	Melihat Dokumen Proyek	Sistem harus dapat menampilkan daftar dokumen yang telah diunggah kedalam sistem.	SMP-F-26
27	Membaca Dokumen Proyek	Sistem harus dapat menampilkan <i>preview</i> dari tiap dokumen yang telah diunggah kedalam sistem.	SMP-F-27
28	Menghapus Dokumen	Sistem harus dapat menghapus dokumen yang ada dalam sistem.	SMP-F-28
29	Mengunduh Dokumen	Sistem harus menyediakan fungsi <i>download</i> atau unduh untuk tiap dokumen yang telah diunggah kedalam sistem.	SMP-F-29
30	Melihat Tim Proyek	Sistem harus dapat menampilkan siapa saja yang telah dilibatkan dalam suatu proyek.	SMP-F-30
31	Menambah Data Client	Sistem harus dapat menambahkan data <i>client</i> baru.	SMP-F-31
32	Melihat Client	Sistem harus dapat menampilkan daftar <i>client</i> yang dilayani oleh perusahaan.	SMP-F-32

No.	Use Case	Deksripsi	Kode
33	Mengubah Data Client	Sistem harus dapat mengubah data suatu <i>client</i> .	SMP-F-33
34	Menghapus Data Client	Sistem harus dapat menghapus data suatu <i>client</i> .	SMP-F-34
35	Menambah Data PIC	Sistem harus dapat menambah data PIC baru kedalam sistem.	SMP-F-35
36	Melihat PIC	Sistem harus dapat menampilkan daftar <i>PIC</i> .	SMP-F-36
37	Mengubah Data PIC	Sistem harus dapat mengubah data seorang <i>PIC</i> .	SMP-F-37
38	Menghapus Data PIC	Sistem harus dapat menghapus data seorang <i>PIC</i> .	SMP-F-38
39	Menambah Data Pegawai	Sistem harus dapat menambahkan data pegawai baru.	SMP-F-39
40	Melihat Data Pegawai	Sistem harus dapat menampilkan daftar pegawai yang bekerja di perusahaan.	SMP-F-40
41	Mengubah Data Pegawai	Sistem harus dapat mengubah data pegawai.	SMP-F-41
42	Menghapus Data Pegawai	Sistem harus dapat menghapus data pegawai.	SMP-F-42
43	Melihat Detail Data Pegawai	Sistem harus dapat menampilkan rincian data seorang pegawai.	SMP-F-43
44	Menambah Data Role Pegawai	Sistem harus dapat menambahkan data <i>role</i> pegawai.	SMP-F-44
45	Melihat Data Role Pegawai	Sistem harus dapat menampilkan <i>role</i> yang ada di perusahaan dan tersimpan dalam sistem.	SMP-F-45
46	Menghapus Data Role Pegawai	Sistem harus dapat menghapus data <i>role</i> yang tersimpan dalam sistem	SMP-F-46
47	Melihat Profil	Sistem harus dapat menampilkan profil pengguna yang sedang <i>login</i> .	SMP-F-47
48	Mengubah Data Profil	Sistem harus dapat mengubah profil pengguna yang sedang <i>login</i> .	SMP-F-48
49	Mengubah Password	Sistem harus dapat mengubah <i>password</i> pengguna yang sedang <i>login</i> .	SMP-F-49

4.3.3 Spesifikasi Kebutuhan

Setiap kebutuhan yang didapatkan harus dispesifikan guna memudahkan mendapatkan gambaran sistem yang lebih terperinci, spesifikasi tiap kebutuhan dituangkan dalam Tabel 4.3.

Tabel 4.3 Spesifikasi Kebutuhan Fungsional

No.	Kode Kebutuhan	Use Case	Kode Spesifikasi	Spesifikasi Kebutuhan
1	SMP-F-1	<i>Login</i>	SMP-F-1-1	Sistem menyediakan <i>field email</i> , <i>password</i> dan juga tombol untuk melakukan <i>login</i> .

No.	Kode Kebutuhan	Use Case	Kode Spesifikasi	Spesifikasi Kebutuhan
2	SMP-F-2	Lupa <i>Password</i>	SMP-F-2-1	Sistem menyediakan tombol <i>Lupa Password</i> untuk beralih ke halaman <i>Lupa Password</i> .
			SMP-F-2-2	Sistem menyediakan <i>field email</i> dan tombol <i>Reset Password</i> .
			SMP-F-2-2	Sistem harus dapat mengirimkan pesan elektronik berisi <i>link</i> untuk nebgatur ulang <i>password</i> kepada alamat email yang dicantumkan.
3	SMP-F-3	<i>Logout</i>	SMP-F-3-1	Sistem menyediakan tombol <i>"Logout"</i> untuk <i>user</i> keluar dari sistem.
4	SMP-F-4	Melihat <i>Dashboard</i>	SMP-F-4-1	Sistem menyediakan tombol <i>dahsboard</i> untuk menampilkan halaman <i>dasboard</i> .
5	SMP-F-5	Mencari Proyek	SMP-F-5-1	Sistem menyediakan <i>field</i> pencarian dan tombol pencarian.
			SMP-F-5-2	Sistem menampilkan hasil pencarian sesuai dengan kata kunci yang dimasukkan.
6	SMP-F-6	Menambah Data Proyek	SMP-F-6-1	Sistem menyediakan tombol <i>Create</i> pada halaman daftar proyek untuk beralih ke halaman tambah proyek.
			SMP-F-6-2	Sistem menyediakan beberapa <i>field</i> untuk diisi dan sebuah tombol untuk menyimpan data.
			SMP-F-6-3	Sistem penyimpan data proyek baru.
7	SMP-F-7	Melihat Proyek	SMP-F-7-1	Sistem menyediakan tombol proyek untuk menampilkan daftar proyek perusahaan berupa sebuah tabel.
8	SMP-F-8	Mengubah Data Proyek	SMP-F-8-1	Sistem menyediakan tombol <i>update</i> pada halaman daftar proyek untuk beralih ke halaman <i>update</i> proyek.
			SMP-F-8-2	Sistem menyediakan beberapa <i>field</i> untuk diubah isinya dan sebuah tombol.
			SMP-F-8-3	Sistem penyimpan perubahan proyek.
9	SMP-F-9	Menghapus Data Proyek	SMP-F-9-1	Sistem menyediakan tombol hapus pada salah satu kolom tabel daftar proyek.
			SMP-F-9-2	Sistem menghapus data proyek dan semua data yang berhubungan dengannya.
10	SMP-F-10	Melihat	SMP-F-10-1	Sistem menampilkan daftar proyek

No.	Kode Kebutuhan	Use Case	Kode Spesifikasi	Spesifikasi Kebutuhan
		Detail Proyek		perusahaan berupa tabel.
			SMP-F-10-2	Sistem menyediakan tombol lihat <i>details</i> pada salah satu kolom tabel proyek.
			SMP-F-10-3	Sistem menampilkan halaman rincian proyek berupa <i>Kanban</i> , <i>task list</i> , <i>gantt chart</i> , tim proyek dan dokumen proyek.
11	SMP-F-11	Menambah Tugas	SMP-F-11-1	Pada halaman daftar tugas disediakan tombol “+ Tugas” untuk memunculkan <i>pop-up</i> tugas baru.
			SMP-F-11-2	Pada <i>pop-up</i> tugas baru, sistem menyediakan beberapa <i>field</i> terkait tugas yang akan ditambahkan dan juga sebuah tombol simpan.
			SMP-F-11-3	Sistem menyimpan data tugas yang baru.
12	SMP-F-12	Melihat Daftar Tugas	SMP-F-12-1	Sistem menampilkan daftar proyek perusahaan berupa tabel.
			SMP-F-12-2	Sistem menyediakan tombol lihat detail proyek pada salah satu kolom tabel proyek.
			SMP-F-12-3	Sistem menampilkan halaman daftar tugas berupa tabel berisi semua tugas proyek tersebut
13	SMP-F-13	Mengupdate Tugas	SMP-F-13-1	Pada <i>pop-up</i> lihat detail suatu tugas, sistem menyediakan tombol untuk menyimpan perubahan suatu tugas.
14	SMP-F-14	Menghapus Data Tugas	SMP-F-14-1	Sistem menyediakan tombol hapus pada salah satu kolom tabel daftar tugas.
			SMP-F-14-2	Sistem menghapus data tugas yang dikehendaki.
15	SMP-F-15	Melihat Detail Tugas	SMP-F-15-1	Sistem menyediakan tombol untuk menampilkan <i>pop-up</i> berisi detail suatu tugas.
16	SMP-F-16	Menambah <i>Feedback</i> Tugas	SMP-F-16-1	Sistem menyediakan <i>field input text</i> untuk memasukkan <i>feedback</i> suatu tugas pada <i>pop-up</i> detail tugas.
			SPM-F-16-2	Sistem menyediakan tombol untuk menyimpan <i>feedback</i> yang telah diberikan.
			SPM-F-16-3	Sistem menyimpan <i>feedback</i> kedalam <i>database</i> .
17	SMP-F-17	Melihat	SMP-F-17-1	Sistem menampilkan <i>feedback</i> suatu

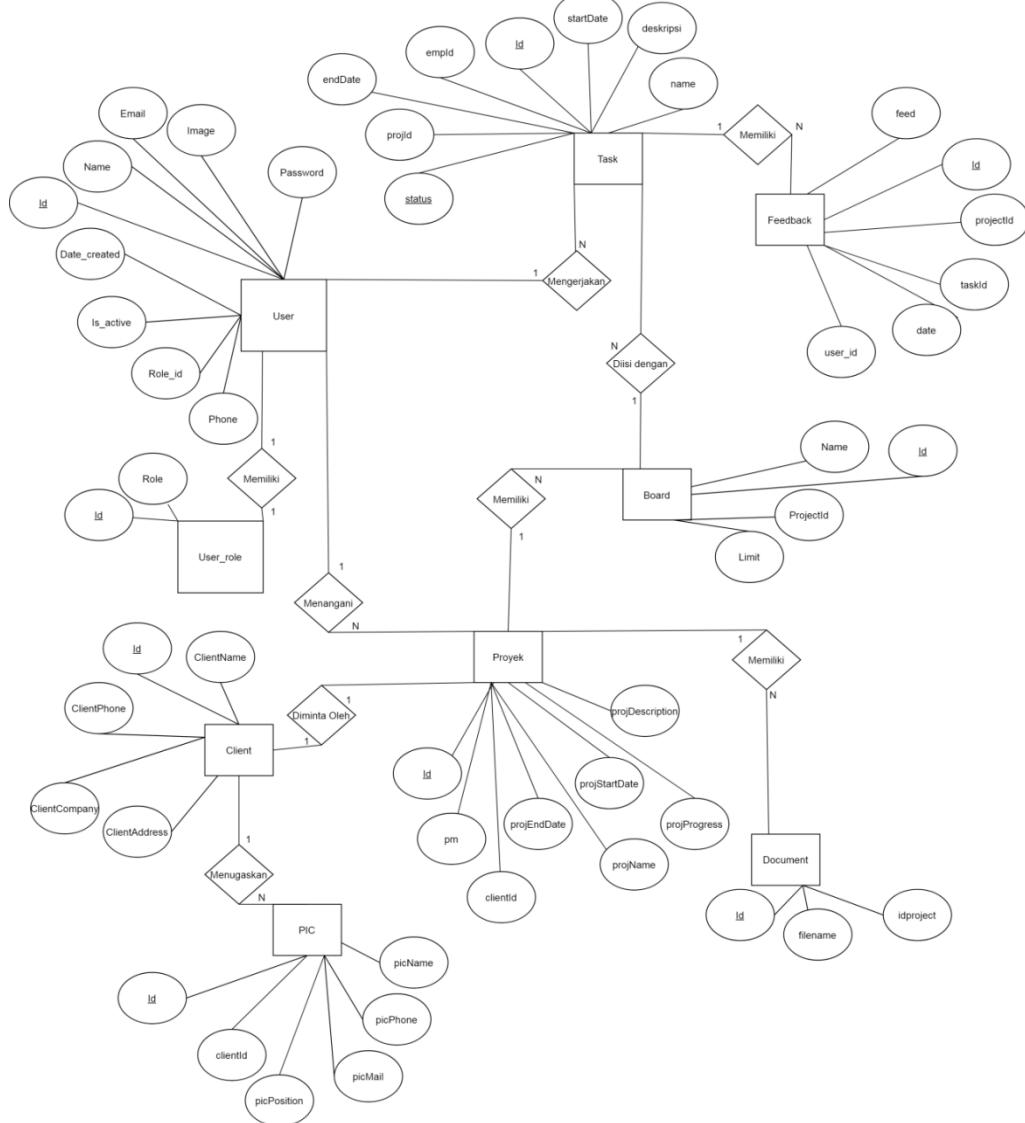
No.	Kode Kebutuhan	Use Case	Kode Spesifikasi	Spesifikasi Kebutuhan
		<i>Feedback Tugas</i>		tugas pada pop-up detil suatu tugas.
18	SMP-F-18	Menghapus <i>Feedback Tugas</i>	SPM-F-18-1	Sistem menyediakan tombol hapus untuk menghapus <i>feedback</i> suatu tugas pada <i>pop-up</i> detil tugas.
19	SMP-F-19	<i>Menambah Board</i>	SMP-F-19-1	Pada halaman <i>Kanban</i> proyek, sistem menyediakan terdapat tombol <i>new board</i> .
			SMP-F-19-2	Sistem akan menampilkan popup berisi sebuah field dan sebuah tombol.
			SMP-F-19-3	<i>Sistem</i> telah menambahkan board baru.
20	SMP-F-20	Melihat <i>Kanban Board</i>	SMP-F-20-1	Pada halaman detil proyek, sistem menyediakan tombol <i>Kanban</i> untuk beralih ke halaman <i>Kanban</i> proyek.
21	SMP-F-21	Mengubah <i>Limit</i>	SMP-F-21-1	Sistem menyediakan sebuah field dan sebuah tombol untuk mengubah limit suatu <i>board</i> .
22	SMP-F-22	Menghapus <i>Board</i>	SMP-F-22-1	Sistem menyediakan tombol untuk menghapus suatu <i>board</i> .
23	SMP-F-23	Melihat Gantt Chart Proyek	SMP-F-23-1	Pada halaman detil proyek, sistem menyediakan tombol “ <i>Grafik Proyek</i> ” untuk beralih ke halaman <i>Gantt chart</i> proyek.
24	SMP-F-24	Mlihat Pie Chart Proyek	SMP-F-24-1	Pada halaman detil proyek, sistem menyediakan tombol <i>Grafik Proyek</i> untuk beralih ke halaman <i>Pie chart</i> proyek.
25	SMP-F-25	<i>Mengupload Dokumen</i>	SMP-F-25-1	Pada halaman detil proyek, sistem menyediakan tombol <i>Dokumen</i> untuk beralih ke halaman dokumen proyek.
			SMP-F-25-2	Sistem menyediakan tombol untuk memilih berkas yang akan diunggah.
			SMP-F-25-3	Sistem menyediakan tombol untuk mengunggah berkas.
			SMP-F-25-4	Sistem menyimpan berkas yang telah dipilih.
26	SMP-F-26	Melihat Dokumen Proyek	SMP-F-26-1	Pada halaman detil proyek, sistem menyediakan tombol <i>Dokumen</i> untuk beralih ke halaman dokumen proyek.
27	SMP-F-27	Membaca Dokumen	SMP-F-27-1	Pada halaman detil proyek, sistem menyediakan tombol <i>Dokumen</i>

No.	Kode Kebutuhan	Use Case	Kode Spesifikasi	Spesifikasi Kebutuhan
		Proyek		untuk beralih ke halaman dokumen proyek.
			SMP-F-27-2	Sistem menyediakan tombol <i>preview</i> pada tiap dokumen.
			SMP-F-27-3	Sistem memunculkan pop-up berisi dokumen yang ingin dibaca.
28	SMP-F-28	Menghapus Dokumen	SMP-F-28-1	Sistem menyediakan tombol untuk menghapus dokumen yang telah diunggah.
29	SMP-F-29	Mengunduh Dokumen	SMP-F-29-1	Sistem menyediakan tombol untuk mengunduh dokumen.
30	SMP-F-30	Melihat Tim Proyek	SMP-F-30-1	Pada halaman detail proyek, sistem menyediakan tombol <i>Tim Proyek</i> untuk melihat personel yang tergabung dalam proyek.
	SMP-F-31	Menambah Data <i>Client</i>	SMP-F-31-1	Sistem menyediakan tombol <i>create</i> pada halaman daftar <i>client</i> untuk beralih ke halaman tambah <i>client</i> .
			SMP-F-31-2	Sistem menyediakan beberapa field untuk diisi dan sebuah tombol.
			SMP-F-31-3	Sistem penyimpanan data <i>client</i> baru.
32	SMP-F-32	Melihat <i>Client</i>	SMP-F-32-1	Sistem menyediakan tombol <i>client</i> untuk menampilkan daftar <i>client</i> perusahaan.
	SMP-F-33	Mengubah Data <i>Client</i>	SMP-F-33-1	Sistem menyediakan tombol <i>update</i> pada halaman daftar <i>client</i> untuk beralih ke halaman <i>update client</i> .
			SMP-F-33-2	Sistem menyediakan beberapa field untuk diubah isinya dan sebuah tombol.
			SMP-F-33-3	Sistem penyimpanan perubahan data <i>client</i> .
34	SMP-F-34	Menghapus Data <i>Client</i>	SMP-F-34-1	Sistem menyediakan tombol hapus untuk menghapus data <i>client</i> .
	SMP-F-35	Menambah Data <i>PIC</i>	SMP-F-35-1	Sistem menyediakan form dengan beberapa field <i>PIC</i> dan sebuah tombol pada halaman detail <i>client</i> untuk menambahkan data <i>PIC</i> .
			SMP-F-35-2	Sistem penyimpanan data <i>PIC</i> baru.
	SMP-F-36	Melihat <i>PIC</i>	SMP-F-36-1	Pada halaman daftar <i>client</i> , sistem menyediakan tombol “view” untuk beralih ke halaman daftar <i>PIC</i> .
			SMP-F-36-2	Sistem menampilkan daftar <i>PIC</i> yang

No.	Kode Kebutuhan	Use Case	Kode Spesifikasi	Spesifikasi Kebutuhan
				ditugaskan oleh client.
37	SMP-F-37	Mengubah Data <i>PIC</i>	SMP-F-37-1	Sistem menyediakan tombol update pada salah satu kolom tabel daftar <i>PIC</i> .
			SMP-F-37-2	Sistem mengisi field pada form tambah <i>PIC</i> dengan data yang akan diubah.
			SMP-F-37-3	Sistem menyimpan perubahan data <i>PIC</i> .
38	SMP-F-38	Menghapus Data <i>PIC</i>	SMP-F-38-1	Sistem menyediakan tombol hapus untuk menghapus data <i>PIC</i> .
39	SMP-F-39	Menambah Data Pegawai	SMP-F-39-1	Sistem menyediakan tombol <i>create</i> pada halaman daftar pegawai untuk beralih ke halaman tambah pegawai.
			SMP-F-39-2	Sistem menyediakan beberapa field untuk diisi dan sebuah tombol.
			SMP-F-39-3	Sistem penyimpanan data pegawai baru.
40	SMP-F-40	Melihat Data Pegawai	SMP-F-40-1	Sistem menyediakan tombol <i>employee</i> untuk menampilkan daftar pegawai berupa tabel.
41	SMP-F-41	Mengubah Data Pegawai	SMP-F-41-1	Sistem menyediakan tombol <i>update</i> pada halaman daftar pegawai untuk beralih ke halaman <i>update</i> pegawai.
			SMP-F-41-2	Sistem menyediakan beberapa field untuk diubah isinya dan sebuah tombol.
			SMP-F-41-3	Sistem penyimpanan perubahan data pegawai.
42	SMP-F-42	Menghapus Data Pegawai	SMP-F-42-1	Sistem menyediakan tombol hapus untuk menghapus data pegawai.
43	SMP-F-43	Melihat Detail Data Pegawai	SMP-F-43-1	Pada halaman daftar pegawai, sistem menyediakan tombol "view" untuk beralih ke halaman detail pegawai.
44	SMP-F-44	Menambah Data Role Pegawai	SMP-F-44-1	Pada halaman daftar pegawai, sistem menyediakan tombol <i>tambah role</i> .
			SMP-F-44-2	Sistem akan menampilkan pop-up berisi sebuah field dan sebuah tombol.
			SMP-F-44-3	Sistem telah menambahkan <i>role</i> baru.

No.	Kode Kebutuhan	Use Case	Kode Spesifikasi	Spesifikasi Kebutuhan
45	SMP-F-45	Melihat Data Role Pegawai	SMP-F-45-1	Pada halaman daftar pegawai, sistem menyediakan tombol <i>tambah role</i> .
			SMP-F-45-2	Sistem akan menampilkan popup berisi sebuah field dan sebuah tombol beserta <i>role</i> yang saat ini ada dalam sistem.
46	SMP-F-46	Menghapus Data Role Pegawai	SMP-F-46-1	Pada halaman daftar pegawai, sistem menyediakan tombol <i>tambah role</i> .
			SMP-F-46-2	Sistem akan menampilkan popup berisi sebuah field dan sebuah tombol beserta <i>role</i> yang saat ini ada dalam sistem.
			SMP-F-46-3	Sistem menyediakan tombol hapus pada tiap <i>role</i> yang ditampilkan.
			SMP-F-46-4	Sistem menghapus data <i>role</i> dari <i>database</i> .
47	SMP-F-47	Melihat Profil	SMP-F-47-1	Sistem menyediakan tombol <i>my profile</i> untuk menampilkan profil pengguna yang saat ini <i>login</i> .
48	SMP-F-48	Mengubah Data Profil	SMP-F-48-1	Sistem menyediakan tombol <i>edit profil</i> untuk menampilkan halaman <i>edit profil</i> .
			SMP-F-48-2	Sistem menyediakan <i>field</i> nama, gambar dan nomor telepon pegawai dan sebuah tombol simpan.
49	SMP-F-49	Mengubah Password	SMP-F-49-1	Sistem menyediakan tombol <i>change password</i> untuk menampilkan halaman ubah <i>password</i> .
			SMP-F-49-2	Sistem menyediakan <i>field</i> <i>password</i> lama, <i>password</i> baru dan <i>repeat password</i> baru serta sebuah tombol.
			SMP-F-49-3	Sistem penyimpan perubahan <i>password</i> .

4.4 Analisis Data



Gambar 4.2 Analisis data sistem manajemen proyek

Analisis data pada Gambar 4.2 menggambarkan hubungan dari data tiap entitas yang digunakan pada sistem ini. Dapat diamati terdapat sembilan entitas yaitu *user*, *user_role*, *client*, *PIC*, *proyek*, *board*, *task*, *feedback* dan *document*. Hubungan antar entitas tersebut antara lain:

1. Entitas *user* memiliki hubungan “Menangani” dengan entitas *proyek* dan memiliki kardinalitas *one-to-many*, hal ini berarti seorang *user/pegawai*, dalam kasus ini adalah pegawai dengan jabatan manajer *proyek* dapat menangani lebih dari satu *proyek* dalam satu waktu.
2. Entitas *user* memiliki hubungan “Memiliki” dengan entitas *user_role* dan memiliki kardinalitas *one-to-one*, hal ini berarti seorang *user/pegawai* memiliki tepat satu *role* atau peran dalam perusahaan.

3. Entitas *user* memiliki hubungan “Mengerjakan” dengan entitas *task* dan memiliki kardinalitas *one-to-many*, hal ini berarti seorang *user*/pegawai dapat mengerjakan lebih dari satu *task/tugas* dalam satu waktu.
4. Entitas proyek memiliki hubungan “Diminta Oleh” dengan entitas *client* dan memiliki kardinalitas *one-to-one*, hal ini berarti suatu proyek pasti diminta spesifik oleh satu pihak *client*.
5. Entitas proyek memiliki hubungan “Memiliki” dengan entitas *board* dan memiliki kardinalitas *one-to-many*, hal ini berarti suatu proyek dapat memiliki lebih dari satu *Kanban board* dalam satu waktu.
6. Entitas proyek memiliki hubungan “Memiliki” dengan entitas *document* dan memiliki kardinalitas *one-to-many*, hal ini berarti suatu proyek dapat memiliki lebih dari satu dokumen dalam satu waktu.
7. Entitas *board* memiliki hubungan “Diisi dengan” dengan entitas *task* dan memiliki kardinalitas *one-to-many*, hal ini berarti suatu *board* dapat diisi lebih dari satu *task/tugas* dalam satu waktu.
8. Entitas *task* memiliki hubungan “Memiliki” dengan entitas *feedback* dan memiliki kardinalitas *one-to-many*, hal ini berarti suatu *task* dapat memiliki lebih dari satu *feedback* dalam satu waktu.
9. Entitas *client* memiliki hubungan “Menugaskan” dengan entitas *PIC* dan memiliki kardinalitas *one-to-many*, hal ini berarti satu pihak *client* dapat menugaskan lebih dari satu *PIC* untuk terlibat dengan perusahaan pengembang selama proyek berlangsung.

4.5 Daftar Kebutuhan Non Fungsional

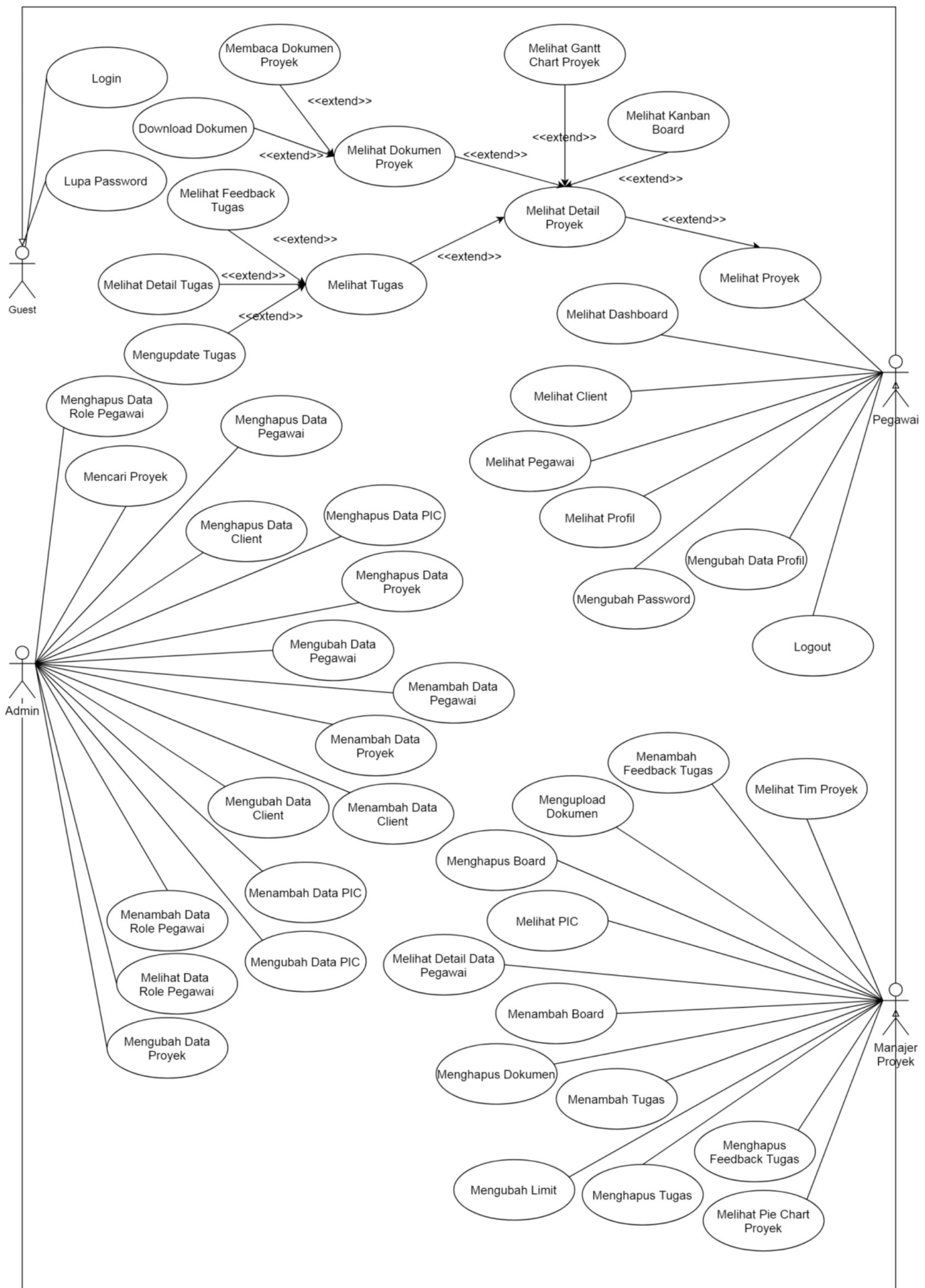
Kebutuhan non-fungsional pada sistem ini ada pada Tabel 4.4.

Tabel 4.4 Kebutuhan Non Fungsional

No.	Parameter	Deksripsi	Kode Kebutuhan
1	<i>Usability</i>	Sistem harus memiliki tampilan yang mudah digunakan oleh Admin, Manajer Proyek, dan Pegawai.	SMP-NF-1

4.6 Pemodelan Kebutuhan

4.6.1 Use Case Diagram



Gambar 4.3 Use Case Diagram Sistem Manajemen Proyek

4.6.2 Use Case Scenario

Use case scenario menjelaskan alur setiap *use case diagram* yang telah disajikan pada Gambar 4.3. *Use case scenario* dituliskan dalam bentuk tabel yang berisikan Objektif, Aktor, Prasyarat, Alur Utama, Alur Alternatif dan Kondisi Akhir. Jumlah *use case scenario* yang dituliskan berjumlah 49 *case* seperti yang telah dijelaskan pada *use case diagram*.

4.6.2.1 Login

Tabel 4.5 menjelaskan *use case scenario login* dan aktor yang dapat mengakses fungsi tersebut adalah *Guest*. Fungsi *use case login* adalah untuk memberi izin pada aktor tersebut mengakses sistem berdasarkan hak yang telah diberikan.

Tabel 4.5 Use Case Scenario Login

Obyektif	Aktor dapat masuk kedalam sistem.
Aktor	<i>Guest</i>
Prasyarat	Aktor mengakses situs sistem melalui <i>browser</i> .
Alur Utama	<ol style="list-style-type: none">1. Sistem menampilkan halaman <i>login</i>.2. Aktor memasukkan data yang diperlukan.3. Memasukkan <i>username</i>.4. Memasukkan <i>Password</i>.5. Aktor menekan tombol <i>login</i>.6. Sistem melakukan pengecekan <i>username</i> dan <i>Password</i> pengguna.
Alur Alternatif	<ol style="list-style-type: none">1. Jika pengguna tidak mengisi kolom <i>email</i> dan <i>password</i> maka sistem akan menampilkan pesan “<i>the email field is required</i>” dan pesan “<i>the password field is required</i>”.2. Jika pengguna tidak mengisi kolom <i>email</i> maka sistem akan menampilkan pesan “<i>the email field is required</i>”.3. Jika pengguna tidak mengisi kolom <i>password</i> maka sistem akan menampilkan pesan “<i>the password field is required</i>”.4. Jika pengguna mengisi kolom <i>email</i> dengan <i>email</i> yang tidak valid maka sistem akan menampilkan pesan “<i>The Email field must contain a valid email address</i>”.5. Jika <i>email</i> pengguna belum diaktifkan maka sistem akan menampilkan pesan “<i>Email is not activated!</i>”.6. Jika akun pengguna tidak terdaftar maka sistem akan menampilkan pesan “<i>Email is not registered!</i>”.7. Jika <i>Password</i> yang dimasukkan salah maka sistem akan menampilkan pesan “<i>Wrong Password!</i>”.
Kondisi Akhir	Sistem akan menampilkan halaman <i>dashBoard</i> kepada aktor, dan aktor dapat menggunakan fungsi sistem sesuai dengan <i>role</i> yang dimiliki.

4.6.2.2 Lupa Password

Tabel 4.6 menjelaskan *use case scenario lupa password* dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk melakukan *reset password* ketika pengguna lupa *password* akun miliknya.

Tabel 4.6 Use Case Scenario Lupa Password

Obyektif	Sistem dapat mereset <i>password</i> akun pengguna yang lupa akan <i>passwordnya</i> .
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna berada pada halaman <i>login</i> .
Alur Utama	<ol style="list-style-type: none">1. Aktor menekan tombol “Lupa Password ?”.2. Aktor memasukan alamat <i>email</i> sesuai akun.3. Aktor menekan tombol “Lupa Password”.4. Sistem akan mengirim notifikasi berupa link ke email pengguna untuk menuju halaman “Lupa Password”.5. Aktor mereset ulang <i>Password</i> di halaman Lupa <i>Password</i>.6. Aktor menekan tombol “Reset”.
Alur Alternatif	<ol style="list-style-type: none">1. Jika pengguna tidak mengisi kolom <i>email</i> maka sistem akan menampilkan pesan “<i>the email field is required</i>”.2. Jika pengguna mengisi kolom <i>email</i> dengan <i>email</i> yang tidak valid maka sistem akan menampilkan pesan “<i>The Email field must contain a valid email address</i>”.3. Jika <i>email</i> pengguna belum diaktifkan maka sistem akan menampilkan pesan “<i>Email is not activated!</i>”.4. Jika akun pengguna tidak terdaftar maka sistem akan menampilkan pesan “<i>Email tidak terdaftar!</i>”.
Kondisi Akhir	Sistem menampilkan halaman login dan juga pesan “ <i>Password telah diubah, silakan login kembali!</i> ”.

4.6.2.3 Logout

Tabel 4.7 menjelaskan *use case scenario logout* dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk mengeluarkan pengguna yang sedang *login* dari sistem.

Tabel 4.7 Use Case Scenario Logout

Obyektif	Pengguna dapat <i>logout</i> dari sistem
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none">1. Pengguna memilih menu ‘Logout’2. Sistem menghapus “session” pengguna dan mengembalikan ke halaman <i>login</i>
Alur Alternatif	Tidak ada
Kondisi Akhir	Aktor telah berhasil keluar dari sistem

4.6.2.4 Melihat Dashboard

Tabel 4.8 menjelaskan *use case scenario* melihat *dahsboard* dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut mengakses halaman *dashboard* sistem.

Tabel 4.8 Use Case Scenario Melihat Dashboard

Obyektif	Aktor dapat melihat halaman <i>Dashboard</i> .
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Aktor telah masuk kedalam sistem.
Alur Utama	1. Aktor memilih tombol navigasi ‘ <i>Dashboard</i> ’. 2. Sistem menampilkan halaman <i>Dashboard</i> .
Alur Alternatif	Tidak ada
Kondisi Akhir	Aktor dapat melihat <i>Dashboard</i> .

4.6.2.5 Mencari Proyek

Tabel 4.9 menjelaskan *use case scenario* mencari proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk melakukan pencarian proyek sesuai dengan kata kunci yang dimasukkan.

Tabel 4.9 Use Case Scenario Mencari Proyek

Obyektif	Admin dapat mencari data proyek
Aktor	Admin
Prasyarat	Aktor telah masuk kedalam sistem
Alur Utama	1. Aktor memilih menu <i>Dasboard</i> 2. Sistem menampilkan halaman <i>Dashboard</i> 3. Aktor memasukkan kata kunci pada <i>field</i> pencarian 4. Aktor menekan tombol “Cari” 5. Sistem menampilkan hasil pencarian
Alur Alternatif	1. Jika hasil pencarian tidak ditemukan, sistem mengosongkan <i>field</i> hasil pencarian. 2. Jika <i>field</i> pencarian tidak diisi, maka <i>field</i> hasil pencarian akan menampilkan data seluruh proyek.
Kondisi Akhir	Hasil pencarian proyek ditampilkan

4.6.2.6 Menambah Data Proyek

Tabel 4.10 menjelaskan *use case scenario* menambah data proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menambah suatu data proyek baru.

Tabel 4.10 Use Case Scenario Menambah Data Proyek

Obyektif	Admin dapat membuat <i>Project</i> baru
Aktor	Admin
Prasyarat	Admin telah masuk kedalam system

Alur Utama	<ol style="list-style-type: none"> 1. Admin memilih menu <i>Project</i> 2. Admin menekan tombol buat <i>Project</i> 3. Admin menginputkan nama proyek 4. Admin menginputkan tanggal mulai dan tanggal selesai 5. Admin menginputkan deskripsi proyek 6. Admin memilih <i>client</i> yang meminta proyek 7. Admin memilih <i>Manajer Proyek</i> yang menangani proyek ini 8. Admin menekan tombol buat
Alur Alternatif	<ol style="list-style-type: none"> 1. Jika pengguna tidak mengisi salah satu atau seluruh <i>field</i>, sistem menampilkan halaman tambah proyek dan pesan "<i>The 'nama-field' field is required</i>" pada kolom yang masih kosong. 2. Jika <i>start date</i> atau <i>end date</i> kurang dari hari ini, sistem menampilkan halaman tambah proyek dan menampilkan pesan "<i>Start / end date</i> tidak boleh kurang dari hari ini !". 3. Jika <i>start date</i> melebihi <i>end date</i>, sistem menampilkan halaman tambah proyek dan menampilkan pesan "<i>Start date</i> tidak boleh melebihi <i>end date</i> !" dan "<i>End date</i> tidak boleh kurang dari <i>start date</i> !".
Kondisi Akhir	Admin telah berhasil membuat proyek

4.6.2.7 Melihat Proyek

Tabel 4.11 menjelaskan *use case scenario* melihat proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut mengakses halaman daftar proyek.

Tabel 4.11 Use Case Scenario Melihat Proyek

Obyektif	Aktor dapat mengetahui daftar proyek yang ditangani oleh perusahaan.
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Aktor telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Aktor memilih tombol '<i>Project</i>'. 2. Sistem menampilkan daftar proyek yang ada dalam database.
Alur Alternatif	Tidak ada
Kondisi Akhir	Aktor dapat melihat daftar proyek yang tercatat dalam database.

4.6.2.8 Mengubah Data Proyek

Tabel 4.12 menjelaskan *use case scenario* mengubah data proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melakukan update pada suatu proyek.

Tabel 4.12 Use Case Scenario Mengubah Data Proyek

Obyektif	Admin dapat mengubah data proyek
Aktor	Admin
Prasyarat	Admin telah masuk ke dalam sistem
Alur Utama	<ol style="list-style-type: none">1. Admin memilih tab “Project”2. Sistem menampilkan data proyek yang ditangani oleh perusahaan3. Admin menekan tombol “Update” pada proyek yang ingin diUpdate4. Sistem menampilkan halaman detail proyek5. Admin mengubah data proyek6. Admin menekan tombol <i>Update</i>7. Sistem menyimpan perubahan dan kembali ke halaman proyek
Alur Alternatif	Sistem menampilkan halaman tambah proyek dan menampilkan pesan “The ‘nama-field’ field is required” pada kolom yang masih kosong.
Kondisi Akhir	Admin dapat mengubah proyek yang ditangani perusahaan

4.6.2.9 Menghapus Data Proyek

Tabel 4.13 menjelaskan *use case scenario* menghapus data proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menghapus data suatu proyek.

Tabel 4.13 Use Case Scenario Menghapus Data Proyek

Obyektif	Admin dapat menghapus Project
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none">1. Admin memilih menu ‘Project’2. Sistem menampilkan halaman <i>Project</i>3. Admin menekan tombol <i>Delete</i>4. Sistem menampilkan <i>pop-up</i> konfirmasi hapus proyek5. Aktor menekan “ok” atau “yes”
Alur Alternatif	Tidak ada
Kondisi Akhir	Proyek dan semua data yang berkaitan dengannya berhasil dihapus

4.6.2.10 Melihat Detail Proyek

Tabel 4.14 menjelaskan *use case scenario* melihat detail proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan juga pegawai. Fungsi *use case* ini adalah untuk melihat halaman detail suatu proyek.

Tabel 4.14 Use Case Scenario Melihat Detail Proyek

Obyektif	Pengguna dapat melihat halaman detail proyek
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none">1. Pengguna memilih menu ‘Project’2. Sistem menampilkan halaman daftar proyek3. Pengguna memilih tombol “details”4. Sistem menampilkan halaman Detail Proyek
Alur Alternatif	Tidak ada
Kondisi Akhir	Halaman detail proyek ditampilkan

4.6.2.11 Menambah Tugas

Tabel 4.15 menjelaskan *use case scenario* menambah tugas dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menambah tugas baru pada suatu proyek.

Tabel 4.15 Use Case Scenario Menambah Tugas

Obyektif	<i>Manajer Proyek</i> dapat membuat <i>Task</i> untuk <i>Pegawai</i>
Aktor	<i>Manajer Proyek</i>
Prasyarat	<i>Manajer Proyek</i> telah masuk kedalam lihat proyek
Alur Utama	<ol style="list-style-type: none">1. Aktor menekan tombol “+” pada <i>Board</i>2. Aktor mengisi nama <i>Task</i>3. Aktor mengisi deskripsi <i>Task</i>4. Aktor menginisialisasi tanggal mulai dan tanggal selesai5. Aktor memilih <i>pegawai</i> yang ditugaskan6. Aktor menekan tombol tambah
Alur Alternatif	<ol style="list-style-type: none">1. Sistem menampilkan halaman detail proyek dan pesan “Task gagal ditambahkan !”.2. Sistem menampilkan halaman detail proyek dan pesan “End date task tidak boleh melebihi end date project !”.3. Sistem menampilkan halaman detail proyek dan pesan “Start date task tidak boleh kurang dari start date project !”.4. Sistem menampilkan halaman detail proyek dan pesan “<i>Start date</i> tidak boleh melebihi <i>end date</i> !” dan “<i>End date</i> tidak boleh kurang dari <i>start date</i> !”.5. Sistem menampilkan halaman detail proyek dan pesan “Start / end date tidak boleh kurang dari hari ini !”.
Kondisi Akhir	<i>Manajer Proyek</i> telah berhasil membuat tugas. Sistem menampilkan halaman detail proyek dan pesan “Task telah dibuat! Notifikasi e-mail telah dikirim ke employee.”.

4.6.2.12 Melihat Daftar Tugas

Tabel 4.16 menjelaskan *use case scenario* melihat tugas dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut mengakses halaman *Kanban* proyek yang menampilkan seluruh tugas dari suatu proyek.

Tabel 4.16 Use Case Scenario Melihat Daftar Tugas

Obyektif	Aktor dapat mengetahui tugas yang ditugaskan kepadanya di suatu proyek, sedangkan manajer proyek dan admin dapat mengetahui semua tugas yang ada di tiap – tiap proyek.
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Aktor telah masuk ke halaman proyek.
Alur Utama	<ol style="list-style-type: none">1. Aktor menekan tombol “Details” pada salah satu data proyek.2. Sistem menampilkan daftar tugas dalam proyek tersebut.
Alur Alternatif	Tidak ada
Kondisi Akhir	Aktor dapat melihat data tugas sesuai dengan proyek yang dipilih.

4.6.2.13 Mengupdate Tugas

Tabel 4.17 menjelaskan *use case scenario* mengupdate tugas dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melakukan *update* pada suatu tugas.

Tabel 4.17 Use Case Scenario Mengupdate Tugas

Obyektif	Aktor dapat melakukan <i>update</i> tugas yang ditugaskan kepadanya.
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Aktor telah masuk ke halaman proyek
Alur Utama	<ol style="list-style-type: none">1. Aktor menekan tombol “Details” pada salah satu data proyek.2. Sistem menampilkan data tugas dalam proyek tersebut.3. Aktor menekan tombol “view” pada tugas yang ingin diupdate.4. Sistem menampilkan detail tugas yang telah dipilih.5. Pengguna mengubah status tugas tersebut dan menekan tombol “Update”.6. Sistem menyimpan perubahan yang dilakukan.
Alur Alternatif	Tidak ada
Kondisi Akhir	Pengguna berhasil mengupdate tugas sesuai dengan tugas yang dipilih.

4.6.2.14 Menghapus Data Tugas

Tabel 4.18 menjelaskan *use case scenario* menghapus tugas dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menghapus suatu tugas pada suatu proyek.

Tabel 4.18 Use Case Scenario Menghapus Data Tugas

Obyektif	<i>Manajer Proyek</i> dapat menghapus <i>Task</i> yang tidak dibutuhkan
Aktor	<i>Manajer Proyek</i>
Prasyarat	<i>Manajer Proyek</i> telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none">1. Aktor memilih menu “<i>Project</i>”2. Sistem menampilkan daftar proyek yang ditangani oleh <i>Manajer Proyek</i>3. Aktor menekan tombol “<i>Details</i>” pada <i>Project</i> yang akan dihapus <i>Task</i>nya4. Sistem menampilkan halaman kanban proyek5. Aktor menekan tombol <i>view</i> pada <i>Task</i> yang akan dihapus6. <i>Manajer Proyek</i> menekan tombol hapus <i>Task</i>
Alur Alternatif	Tidak ada
Kondisi Akhir	<i>Task</i> berhasil dihapus

4.6.2.15 Melihat Detail Tugas

Tabel 4.19 menjelaskan *use case scenario* melihat detail tugas dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melihat detail informasi dari suatu tugas.

Tabel 4.19 Use Case Scenario Melihat Detail Tugas

Obyektif	Aktor dapat mengetahui detail tugas yang ditugaskan kepadanya di suatu proyek, sedangkan manajer proyek dan admin dapat mengetahui semua detail tugas yang ada di tiap – tiap proyek.
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Aktor telah masuk ke halaman proyek.
Alur Utama	<ol style="list-style-type: none">1. Aktor menekan tombol “<i>Details</i>” pada salah satu data proyek.2. Sistem menampilkan data tugas dalam proyek tersebut.3. Aktor menekan tombol “<i>view</i>” pada tugas yang ingin dilihat detailnya.4. Sistem menampilkan detail tugas yang telah dipilih.
Alur Alternatif	Tidak ada
Kondisi Akhir	Aktor dapat melihat data detail tugas sesuai dengan tugas yang dipilih.

4.6.2.16 Menambah Feedback Tugas

Tabel 4.20 menjelaskan *use case scenario* menambah *feedback* tugas dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk menambahkan *feedback* pada suatu tugas.

Tabel 4.20 Use Case Scenario Menambah Feedback Tugas

Obyektif	Aktor dapat menambahkan <i>feedback</i> pada tugas milik <i>Pegawai</i>
Aktor	Admin, Manajer Proyek, Pegawai
Prasyarat	Aktor telah masuk di halaman detail proyek

Alur Utama	<ol style="list-style-type: none"> 1. Aktor menekan tombol “Daftar Tugas” 2. Aktor menekan tombol “view” pada salah satu tugas 3. Sistem menampilkan <i>modal dialog</i> detil tugas 4. Aktor mengisi <i>feedback</i> untuk tugas tersebut 5. Aktor menekan tombol tambah
Alur Alternatif	Sistem menampilkan halaman detail proyek dan menampilkan pesan “Feedback gagal ditambahkan!”.
Kondisi Akhir	Aktor telah berhasil menambahkan <i>feedback</i>

4.6.2.17 Melihat Feedback Tugas

Tabel 4.21 menjelaskan *use case scenario* melihat *feedback* tugas dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek, dan pegawai. Fungsi *use case* ini adalah untuk melihat *feedback* pada suatu tugas.

Tabel 4.21 Use Case Scenario Melihat Feedback Tugas

Obyektif	Aktor dapat menambahkan <i>feedback</i> pada tugas milik Pegawai
Aktor	<i>Admin, Manajer Proyek, Pegawai</i>
Prasyarat	Aktor telah masuk di halaman detil proyek
Alur Utama	<ol style="list-style-type: none"> 1. Aktor menekan tombol “Daftar Tugas” 2. Aktor menekan tombol “view” pada salah satu tugas 3. Sistem menampilkan <i>modal dialog</i> detil tugas dan juga <i>feedback</i> tugas
Alur Alternatif	Tidak ada
Kondisi Akhir	Sistem menampilkan <i>feedback</i> suatu tugas

4.6.2.18 Menghapus Feedback Tugas

Tabel 4.22 menjelaskan *use case scenario* menghapus *feedback* tugas dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk menghapus *feedback* pada suatu tugas.

Tabel 4.22 Use Case Scenario Menghapus Feedback Tugas

Obyektif	Aktor dapat menghapus <i>feedback</i> pada tugas milik Pegawai
Aktor	<i>Admin, Manajer Proyek, Pegawai</i>
Prasyarat	Aktor telah masuk di halaman detil proyek
Alur Utama	<ol style="list-style-type: none"> 1. Aktor menekan tombol “Daftar Tugas” 2. Aktor menekan tombol “view” pada salah satu tugas 3. Sistem menampilkan <i>modal dialog</i> detil tugas dan juga <i>feedback</i> tugas 4. Aktor menekan tombol “hapus” pada <i>feedback</i> yang ingin dihapus 5. Sistem menghapus data <i>feedback</i>
Alur Alternatif	Tidak ada
Kondisi Akhir	Sistem menghapus <i>feedback</i> yang dipilih

4.6.2.19 Menambah Board

Tabel 4.23 menjelaskan *use case scenario* menambah board dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menambah suatu *Kanban board* pada suatu proyek.

Tabel 4.23 Use Case Scenario Menambah Board

Obyektif	<i>Manajer Proyek</i> dapat membuat <i>Board</i> baru
Aktor	<i>Manajer Proyek</i>
Prasyarat	<i>Manajer Proyek</i> telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"><i>Manajer Proyek</i> memilih menu “Project”Sistem menampilkan daftar proyek yang ditangani oleh <i>Manajer Proyek</i><i>Manajer Proyek</i> menekan tombol “Details” pada <i>Project</i> yang akan ditambahkan <i>Board</i> nyaSistem menampilkan halaman kanban proyek<i>Manajer Proyek</i> menekan tombol <i>new Board</i>Sistem menampilkan <i>form Board</i><i>Manajer Proyek</i> memasukan nama <i>Board</i><i>Manajer Proyek</i> menekan tombol <i>add</i>
Alur Alternatif	Sistem menampilkan halaman detail proyek dan menampilkan pesan “Nama board harus diisi !”.
Kondisi Akhir	<i>Board</i> telah ditambahkan

4.6.2.20 Melihat Kanban Board

Tabel 4.24 menjelaskan *use case scenario* melihat Kanban proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan juga pegawai. Fungsi *use case* ini adalah untuk melihat halaman Kanban suatu proyek.

Tabel 4.24 Use Case Scenario Melihat Kanban Board

Obyektif	Pengguna dapat melihat halaman Kanban proyek
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none">Pengguna memilih menu ‘Project’Sistem menampilkan halaman daftar proyekPengguna memilih tombol “details”Sistem menampilkan halaman Detail ProyekAktor menekan tombol “Kanban”Sistem menampilkan halaman Kanban Proyek
Alur Alternatif	Tidak ada
Kondisi Akhir	Halaman Kanban proyek ditampilkan

4.6.2.21 Mengubah Limit

Tabel 4.25 menjelaskan *use case scenario* mengubah limit dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk mengubah limit pada suatu *Kanban board*.

Tabel 4.25 Use Case Scenario Mengubah Limit

Obyektif	Aktor dapat mengubah <i>limit</i> suatu <i>board</i>
Aktor	<i>Manajer Proyek, Admin</i>
Prasyarat	<i>Manajer Proyek</i> telah membuka halaman <i>Kanban</i> proyek
Alur Utama	<ol style="list-style-type: none">1. Aktor memasukkan limit baru pada suatu <i>board</i>2. Aktor menekan tombol check (v)
Alur Alternatif	Tidak ada
Kondisi Akhir	<i>Limit</i> berhasil diperbarui

4.6.2.22 Menghapus Board

Tabel 4.26 menjelaskan *use case scenario* menghapus *board* dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menghapus *Kanban board* pada suatu proyek.

Tabel 4.26 Use Case Scenario Menghapus Board

Obyektif	<i>Manajer Proyek</i> dapat menghapus <i>Board</i>
Aktor	<i>Manajer Proyek</i>
Prasyarat	<i>Manajer Proyek</i> telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none">1. Aktor memilih menu “Project”2. Sistem menampilkan daftar proyek yang ditangani oleh <i>Manajer Proyek</i>3. Aktor menekan tombol “Details” pada proyek yang akan dihapus <i>Board</i>nya4. Sistem menampilkan halaman detail proyek5. Aktor menekan tombol “silang” pada <i>card Board</i>
Alur Alternatif	Tidak ada
Kondisi Akhir	<i>Board</i> berhasil dihapus

4.6.2.23 Melihat Gantt Chart Proyek

Tabel 4.27 menjelaskan *use case scenario* melihat *Gantt Chart* proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan juga pegawai. Fungsi *use case* ini adalah untuk melihat halaman *Gantt Chart* suatu proyek.

Tabel 4.27 Use Case Scenario Melihat Gantt Chart Proyek

Obyektif	Pengguna dapat melihat halaman <i>Gantt Chart</i> proyek
Aktor	Pegawai, <i>Manajer Proyek, Admin</i>
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none">1. Pengguna memilih menu ‘Project’2. Sistem menampilkan halaman daftar proyek

	<ol style="list-style-type: none"> 3. Pengguna memilih tombol “details” 4. Sistem menampilkan halaman Detail Proyek 5. Aktor menekan tombol “Grafik Proyek” 6. Sistem menampilkan halaman Gantt Chart Proyek
Alur Alternatif	Tidak ada
Kondisi Akhir	Halaman Gantt Chart proyek ditampilkan

4.6.2.24 Melihat *Pie Chart* Proyek

Tabel 4.28 menjelaskan *use case scenario* melihat *Pie Chart* proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk melihat halaman *Pie Chart* suatu proyek.

Tabel 4.28 Use Case Scenario Melihat Pie Chart Proyek

Obyektif	Pengguna dapat melihat halaman Pie Chart proyek
Aktor	Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu ‘Project’ 2. Sistem menampilkan halaman daftar proyek 3. Pengguna memilih tombol “details” 4. Sistem menampilkan halaman Detail Proyek 5. Aktor menekan tombol “Grafik Proyek” 6. Sistem menampilkan halaman Pie Chart Proyek
Alur Alternatif	Tidak ada
Kondisi Akhir	Halaman Pie Chart proyek ditampilkan

4.6.2.25 Mengupload Dokumen

Tabel 4.29 menjelaskan *use case scenario* mengupload dokumen dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk mengunggah suatu dokumen kedalam sistem.

Tabel 4.29 Use Case Scenario Mengupload Dokumen

Obyektif	<i>Manajer Proyek</i> dapat melakukan <i>Upload</i> dokumen
Aktor	<i>Manajer Proyek</i>
Prasyarat	<i>Manajer Proyek</i> telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. <i>Manajer Proyek</i> memilih tab “Project” 2. Sistem menampilkan daftar proyek yang ditangani oleh <i>Manajer Proyek</i> 3. <i>Manajer Proyek</i> menekan tombol “Details” pada proyek yang akan ditambahkan dokumennya 4. Sistem menampilkan halaman kanban proyek 5. <i>Manajer Proyek</i> menekan tombol “Browse” 6. <i>Manajer Proyek</i> memilih dokumen yang akan diunggah 7. <i>Manajer Proyek</i> menekan tombol “+ Dokumen” 8. Sistem menambahkan dokumen dan kembali ke halaman kanban proyek

Alur Alternatif	<ol style="list-style-type: none"> 1. Jika berkas belum dipilih maka sistem akan menampilkan halaman Detail Proyek dan menampilkan pesan "<i>Berkas gagal diupload ! The filetype you are attempting to upload is not allowed.</i>". 2. Jika format berkas tidak sesuai, maka sistem akan menampilkan halaman Detail Proyek dan menampilkan pesan "<i>Berkas gagal diupload ! The filetype you are attempting to upload is not allowed.</i>". 3. Jika ukuran berkas melebihi batas, maka sistem akan menampilkan halaman Detail Proyek dan menampilkan pesan "<i>Berkas gagal diupload ! The uploaded file exceeds the maximum allowed size in your PHP configuration file.</i>".
Kondisi Akhir	Berkas / dokumen telah terunggah

4.6.2.26 Melihat Dokumen Proyek

Tabel 4.30 menjelaskan *use case scenario* melihat *dokumen* proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk melihat dokumen suatu proyek.

Tabel 4.30 Use Case Scenario Melihat Dokumen Proyek

Obyektif	Pengguna dapat melihat halaman Dokumen proyek
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu '<i>Project</i>' 2. Sistem menampilkan halaman daftar proyek 3. Pengguna memilih tombol "<i>details</i>" 4. Sistem menampilkan halaman Detail Proyek 5. Aktor menekan tombol "<i>Dokumen</i>" 6. Sistem menampilkan halaman Dokumen Proyek
Alur Alternatif	Tidak ada
Kondisi Akhir	Halaman Dokumen proyek ditampilkan

4.6.2.27 Membaca Dokumen Proyek

Tabel 4.31 menjelaskan *use case scenario* membaca *dokumen* proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk melakukan *preview* terhadap dokumen suatu proyek.

Tabel 4.31 Use Case Scenario Membaca Dokumen Proyek

Obyektif	Pengguna dapat membaca Dokumen proyek
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu '<i>Project</i>' 2. Sistem menampilkan halaman daftar proyek 3. Pengguna memilih tombol "<i>details</i>" 4. Sistem menampilkan halaman Detail Proyek

	5. Aktor menekan tombol “Dokumen” 6. Sistem menampilkan halaman Dokumen Proyek 7. Aktor menekan tombol “view” 8. Sistem menampilkan modal dialog berisi dokumen yang dipilih
Alur Alternatif	Tidak ada
Kondisi Akhir	Preview Dokumen ditampilkan dan siap untuk dibaca

4.6.2.28 Menghapus Dokumen

Tabel 4.32 menjelaskan *use case scenario* menghapus dokumen dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menghapus dokumen pada suatu proyek.

Tabel 4.32 Use Case Scenario Menghapus Dokumen

Obyektif	<i>Manajer Proyek</i> dapat menghapus dokumen proyek
Aktor	<i>Manajer Proyek</i>
Prasyarat	<i>Manajer Proyek</i> telah masuk kedalam sistem
Alur Utama	1. <i>Manajer Proyek</i> memilih manu “Project” 2. Sistem menampilkan daftar proyek yang ditangani oleh <i>Manajer Proyek</i> 3. <i>Manajer Proyek</i> menekan tombol “Details” pada <i>Project</i> yang akan dihapus dokumennya 4. Sistem menampilkan halaman kanban proyek 5. <i>Manajer Proyek</i> menekan tombol “silang” pada <i>card</i> dokumen
Alur Alternatif	Tidak ada
Kondisi Akhir	Dokumen proyek telah dihapus

4.6.2.29 Download Dokumen

Tabel 4.33 menjelaskan *use case scenario* download dokumen dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk mengunduh dokumen dalam suatu proyek.

Tabel 4.33 Use Case Scenario Download Dokumen

Obyektif	Aktor dapat melakukan <i>download</i> pada dokumen suatu proyek.
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Aktor telah masuk ke halaman proyek.
Alur Utama	1. Pengguna menekan tombol “Details” pada salah satu data proyek. 2. Sistem menampilkan data dokumen dalam proyek tersebut. 3. Pengguna menekan tombol “download” pada dokumen yang ingin diunduh. 4. Sistem mengunduh dokumen yang dipilih.
Alur Alternatif	Tidak ada
Kondisi Akhir	Aktor berhasil mengunduh dokumen yang dipilih

4.6.2.30 Melihat Tim Proyek

Tabel 4.34 menjelaskan *use case scenario* melihat tim proyek dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk melihat pegawai yang terlibat dalam suatu proyek.

Tabel 4.34 Use Case Scenario Melihat Tim Proyek

Obyektif	Pengguna dapat melihat halaman tim proyek
Aktor	Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none">1. Pengguna memilih menu ‘Project’2. Sistem menampilkan halaman daftar proyek3. Pengguna memilih tombol “details”4. Sistem menampilkan halaman Detail Proyek5. Aktor menekan tombol “Tim Proyek”6. Sistem menampilkan halaman tim proyek
Alur Alternatif	Tidak ada
Kondisi Akhir	Halaman tim proyek ditampilkan

4.6.2.31 Menambah Data Client

Tabel 4.35 menjelaskan *use case scenario* menambah data client dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menambah data client baru.

Tabel 4.35 Use Case Scenario Menambah Data Client

Obyektif	Admin dapat menambahkan <i>client</i>
Aktor	Admin
Prasyarat	Admin telah masuk ke dalam sistem
Alur Utama	<ol style="list-style-type: none">1. Admin memilih menu <i>client</i>2. Sistem menampilkan halaman <i>client</i>3. Admin menekan tombol tambah <i>client</i>4. Sistem menampilkan form tambah <i>client</i>5. Admin mengisi data-data pada form6. Admin menekan tombol tambah
Alur Alternatif	Jika terdapat <i>field</i> kosong, sistem akan menampilkan halaman tambah <i>client</i> dan menampilkan pesan “The ‘nama-field’ field is required” pada kolom yang masih kosong.
Kondisi Akhir	Data <i>client</i> berhasil dibuat

4.6.2.32 Melihat Client

Tabel 4.36 menjelaskan *use case scenario* melihat client dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melihat daftar *client* yang dilayani perusahaan.

Tabel 4.36 Use Case Scenario Melihat Client

Obyektif	Aktor dapat mengetahui <i>client</i> yang dilayani oleh perusahaan
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Aktor telah masuk kedalam sistem
Alur Utama	1. Aktor memilih menu ' <i>Client</i> ' 2. Sistem menampilkan daftar <i>client</i> yang ada dalam database
Alur Alternatif	Tidak ada
Kondisi Akhir	Aktor dapat melihat daftar <i>client</i> yang tercatat dalam database

4.6.2.33 Mengubah Data Client

Tabel 4.37 menjelaskan *use case scenario* mengubah data *client* dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk mengubah data *client*.

Tabel 4.37 Use Case Scenario Mengubah Data Client

Obyektif	Admin dapat melakukan perubahan pada data <i>client</i>
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	1. Admin memilih menu <i>Client</i> 2. Sistem menampilkan halaman <i>client</i> 3. Admin menekan tombol <i>Update</i> 4. Sistem menampilkan form <i>Update</i> 5. Admin mengisi data yang ingin di <i>Update</i> 6. Admin menekan tombol <i>Update</i>
Alur Alternatif	Jika terdapat <i>field</i> kosong, sistem akan menampilkan halaman tambah <i>client</i> dan menampilkan pesan " <i>The 'nama-field' field is required</i> " pada kolom yang masih kosong.
Kondisi Akhir	Data berhasil di <i>Update</i>

4.6.2.34 Menghapus Data Client

Tabel 4.38 menjelaskan *use case scenario* menghapus data *client* dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menghapus data *client*.

Tabel 4.38 Use Case Scenario Menghapus Data Client

Obyektif	Admin dapat menghapus data <i>client</i>
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem

Alur Utama	<ol style="list-style-type: none"> 1. Admin memilih menu <i>client</i> 2. Sistem menampilkan halaman <i>client</i> 3. Admin menekan tombol <i>Delete</i>
Alur Alternatif	Tidak ada
Kondisi Akhir	Data <i>client</i> telah berhasil dihapus

4.6.2.35 Menambah Data PIC

Tabel 4.39 menjelaskan *use case scenario* menambah data PIC dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menambahkan data PIC yang ditugaskan oleh *client*.

Tabel 4.39 Use Case Scenario Menambah Data PIC

Obyektif	Admin dapat menambahkan PIC
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Admin memilih menu <i>Client</i> 2. Sistem menampilkan halaman <i>client</i> 3. Admin menekan tombol “view” 4. Sistem menampilkan form pengisian data PIC 5. Aktor mengisi data-data PIC 6. Aktor menekan tombol tambah
Alur Alternatif	Jika terdapat <i>field</i> kosong, sistem akan menampilkan halaman detail <i>client</i> dan menampilkan pesan “PIC gagal ditambahkan!”.
Kondisi Akhir	PIC telah berhasil dibuat

4.6.2.36 Melihat PIC

Tabel 4.40 menjelaskan *use case scenario* melihat PIC dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melihat daftar PIC yang ditugaskan oleh *client* terkait.

Tabel 4.40 Use Case Scenario Melihat PIC

Obyektif	Pengguna dapat mengetahui PIC yang terlibat dalam proyek
Aktor	Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu ‘Client’ 2. Pengguna memilih tombol “view” pada kolom PIC
Alur Alternatif	Tidak ada
Kondisi Akhir	Aktor dapat melihat PIC

4.6.2.37 Mengubah Data PIC

Tabel 4.41 menjelaskan *use case scenario* mengubah data PIC dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk mengubah data PIC yang telah tercatat dalam sistem.

Tabel 4.41 Use Case Scenario Mengubah Data PIC

Obyektif	Admin dapat menngubah data PIC
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Admin memilih menu <i>Client</i> 2. Sistem menampilkan halaman <i>client</i> 3. Admin menekan tombol “view” 4. Admin menekan tombol <i>edit</i> pada data PIC yang ingin diedit 5. Sistem menampilkan data pada form 6. Aktor mengisi data yang diperlukan 7. Aktor menekan tombol <i>Update</i>
Alur Alternatif	Jika terdapat <i>field</i> kosong, sistem akan menampilkan halaman detail <i>client</i> dan menampilkan pesan “PIC gagal diupdate!”.
Kondisi Akhir	Data telah berhasil di <i>Update</i>

4.6.2.38 Menghapus Data PIC

Tabel 4.42 menjelaskan *use case scenario* menghapus data PIC dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menghapus data seorang PIC.

Tabel 4.42 Use Case Scenario Menghapus Data PIC

Obyektif	Admin dapat menghapus data PIC
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Admin memilih menu “Client” 2. Sistem menampilkan halaman <i>client</i> 3. Admin menekan tombol “view” 4. Sistem menampilkan detail <i>client</i> dan tabel PIC 5. Admin menekan tombol hapus pada data PIC yang ingin dihapus
Alur Alternatif	Tidak ada
Kondisi Akhir	Data PIC berhasil dihapus

4.6.2.39 Menambah Data Pegawai

Tabel 4.43 menjelaskan *use case scenario* menambah data pegawai dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menambah data pegawai apabila ada pegawai yang baru bekerja di perusahaan.

Tabel 4.43 Use Case Scenario Menambah Data Pegawai

Obyektif	Admin dapat menambahkan Pegawai
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Admin memilih menu <i>Pegawai</i>

	<ol style="list-style-type: none"> 2. Sistem menampilkan halaman <i>Pegawai</i> 3. Admin menekan tombol Tambah Akun <i>Pegawai</i> 4. Sistem menampilkan form tambah data <i>Pegawai</i> 5. Admin mengisi data-data <i>Pegawai</i> 6. Aktor menekan tombol tambah
Alur Alternatif	Jika terdapat <i>field</i> kosong, sistem akan menampilkan halaman tambah <i>Employee</i> dan menampilkan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.
Kondisi Akhir	<i>Pegawai</i> berhasil ditambah

4.6.2.40 Melihat Data Pegawai

Tabel 4.44 menjelaskan *use case scenario* lihat pegawai dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melihat daftar pegawai yang bekerja di perusahaan.

Tabel 4.44 Use Case Scenario Melihat Data Pegawai

Obyektif	Pengguna dapat mengetahui daftar <i>pegawai</i> yang bekerja di perusahaan
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu ‘<i>Pegawai</i>’ 2. Sistem menampilkan daftar <i>pegawai</i> yang ada dalam database
Alur Alternatif	Tidak ada
Kondisi Akhir	Pengguna dapat melihat daftar <i>pegawai</i> yang tercatat dalam database

4.6.2.41 Mengubah Data Pegawai

Tabel 4.45 menjelaskan *use case scenario* mengubah data pegawai dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk mengubah data pegawai yang bekerja di perusahaan.

Tabel 4.45 Use Case Scenario Mengubah Data Pegawai

Obyektif	Admin dapat mengubah data Pegawai
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Admin memilih menu <i>Pegawai</i> 2. Sistem menampilkan halaman <i>Pegawai</i> 3. Aktor menekan tombol <i>Update</i> 4. Sistem menampilkan form <i>Update</i> 5. Aktor mengisi data yang ingin di <i>Update</i> 6. Aktor menekan tombol <i>Update</i>
Alur Alternatif	Jika terdapat <i>field</i> kosong, sistem akan menampilkan halaman <i>Update Employee</i> dan menampilkan pesan “ <i>The ‘nama-field’</i>

	<i>field is required” pada kolom yang masih kosong.</i>
Kondisi Akhir	Data telah berhasil di <i>Update</i>

4.6.2.42 Menghapus Data Pegawai

Tabel 4.46 menjelaskan *use case scenario* menghapus data pegawai dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menghapus data seorang pegawai.

Tabel 4.46 Use Case Scenario Menghapus Data Pegawai

Obyektif	Admin dapat menghapus data Pegawai
Aktor	Admin
Prasyarat	Aktor telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Aktor memilih menu <i>Pegawai</i> 2. Sistem menampilkan halaman <i>Pegawai</i> 3. Aktor menekan tombol <i>Delete</i>
Alur Alternatif	Tidak ada
Kondisi Akhir	Data telah berhasil di hapus

4.6.2.43 Melihat Detail Data Pegawai

Tabel 4.47 menjelaskan *use case scenario* lihat detail data pegawai dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, dan manajer proyek. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melihat detail data pegawai yang bekerja di perusahaan.

Tabel 4.47 Use Case Scenario Lihat Detail Data Pegawai

Obyektif	Pengguna dapat mengetahui detail data dari masing-masing pegawai
Aktor	<ol style="list-style-type: none"> 1. <i>Manajer Proyek</i> 2. <i>Admin</i>
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu ‘<i>Pegawai</i>’ 2. Pengguna memilih tombol “<i>Detail</i>” pada kolom “<i>Action</i>”
Alur Alternatif	Tidak ada
Kondisi Akhir	Pengguna dapat melihat detail data pegawai

4.6.2.44 Menambah Data Role Pegawai

Tabel 4.48 menjelaskan *use case scenario* menambah data *role* pegawai dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menambah *role* atau jabatan dalam perusahaan.

Tabel 4.48 Use Case Scenario Menambah Data Role Pegawai

Obyektif	Admin dapat membuat <i>role</i> baru
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem

Alur Utama	<ol style="list-style-type: none"> 1. <i>Admin</i> memilih menu “Employee” 2. Sistem menampilkan employee yang tergabung di perusahaan 3. <i>Admin</i> menekan tombol “+ Role” 4. Sistem menampilkan form Role 5. <i>Admin</i> memasukan nama Role 6. <i>Admin</i> menekan tombol add
Alur Alternatif	Jika <i>field</i> nama belum terisi, sistem menampilkan halaman Employee dan menampilkan pesan “Nama role harus diisi!”.
Kondisi Akhir	<i>Role</i> telah ditambahkan

4.6.2.45 Melihat Data Role Pegawai

Tabel 4.49 menjelaskan *use case scenario* melihat data *role* pegawai dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melihat *role* atau jabatan yang ada dalam perusahaan.

Tabel 4.49 Use Case Scenario Melihat Data Role Pegawai

Obyektif	Admin dapat melihat role yang ada di perusahaan
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. <i>Admin</i> memilih menu “Employee” 2. Sistem menampilkan employee yang tergabung di perusahaan 3. <i>Admin</i> menekan tombol “+ Role” 4. Sistem menampilkan form Role dan daftar role yang ada di perusahaan
Alur Alternatif	Tidak ada
Kondisi Akhir	<i>Role</i> ditampilkan

4.6.2.46 Menghapus Data Role Pegawai

Tabel 4.50 menjelaskan *use case scenario* menghapus data *role* pegawai dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk menghapus *role* atau jabatan yang ada dalam perusahaan.

Tabel 4.50 Use Case Scenario Menghapus Data Role Pegawai

Obyektif	Admin dapat menghapus role yang ada di perusahaan
Aktor	Admin
Prasyarat	Admin telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. <i>Admin</i> memilih menu “Employee” 2. Sistem menampilkan halaman Daftar employee 3. <i>Admin</i> menekan tombol “+ Role” 4. Sistem menampilkan form Role dan daftar role yang ada di perusahaan 5. <i>Admin</i> menekan tombol “hapus” pada role yang dipilih 6. Sistem menghapus role yang dipilih

Alur Alternatif	Tidak ada
Kondisi Akhir	<i>Role</i> dihapus

4.6.2.47 Melihat Profil

Tabel 4.51 menjelaskan *use case scenario* lihat profil dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk melihat profil/akun miliknya.

Tabel 4.51 Use Case Scenario Melihat Profil

Obyektif	Pengguna dapat melihat profil diri yang tersimpan didalam sistem
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu ‘My Profile’ 2. Sistem menampilkan profil pengguna yang saat ini masuk kedalam sistem
Alur Alternatif	Tidak ada
Kondisi Akhir	Pengguna dapat melihat rincian profilnya

4.6.2.48 Mengubah Data Profil

Tabel 4.52 menjelaskan *use case scenario* mengubah data profil dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk mengubah data profil yang dimilikinya.

Tabel 4.52 Use Case Scenario Mengubah Data Profil

Obyektif	Pengguna dapat mengedit profil berupa nama,foto, dan telpon
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu ‘Edit Profil’ 2. Pengguna memasukan nama baru 3. Pengguna mengunggah foto profil baru 4. Pengguna memasukan nomor telepon baru 5. Pengguna menekan tombol “simpan” 6. Sistem menyimpan <i>Update</i> dan mengembalikan pengguna ke halaman profil
Alur Alternatif	Jika terdapat <i>field</i> belum terisi, sistem menampilkan halaman Edit Profil dan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.
Kondisi Akhir	Profil telah diperbarui oleh sistem

4.6.2.49 Mengubah Password

Tabel 4.53 menjelaskan *use case scenario* mengubah password dan aktor yang dapat mengakses fungsi tersebut adalah pengguna. Pengguna tersebut adalah admin, manajer proyek dan pegawai. Fungsi *use case* ini adalah untuk memberi izin pada aktor tersebut untuk mengubah password akun miliknya.

Tabel 4.53 Use Case Scenario Mengubah Password

Obyektif	Pengguna dapat mengubah <i>Password</i> akun miliknya
Aktor	Pegawai, Manajer Proyek, Admin
Prasyarat	Pengguna telah masuk kedalam sistem
Alur Utama	<ol style="list-style-type: none"> 1. Pengguna memilih menu ‘<i>Change Password</i>’ 2. Pengguna memasukan <i>Password</i> lama 3. Pengguna memasukan <i>Password</i> baru 4. Pengguna memasukan ulang <i>Password</i> baru 5. Pengguna menekan tombol “<i>Change Password</i>” 6. Sistem menyimpan <i>Password</i> yang baru dan mengembalikan pengguna ke halaman profil pengguna
Alur Alternatif	<ol style="list-style-type: none"> 1. Jika terdapat <i>field</i> belum terisi, sistem akan menampilkan halaman <i>Change Password</i> dan pesan “<i>The ‘nama-field’ field is required</i>” pada kolom yang masih kosong. 2. Jika inputan <i>current password</i> tidak sesuai dengan <i>password</i> pengguna saat ini, sistem akan menampilkan halaman <i>Change Password</i> dan pesan “<i>Wrong current password!</i>”. 3. Jika inputan <i>new password</i> tidak sesuai dengan <i>field repeat password</i>, sistem akan menampilkan halaman <i>Change Password</i> dan pesan “<i>The New Password field does not match the Repeat Password field.</i>” pada kolom <i>New password</i> dan pesan “<i>The Repeat Password field does not match the New Password field.</i>” pada kolom <i>Repat password</i>. 4. Jika inputan <i>new password</i> kurang dari enam karakter, sistem akan menampilkan halaman <i>Change Password</i> dan pesan “<i>The New Password field must be at least 6 character in length</i>” pada kolom <i>New password</i>.
Kondisi Akhir	<i>Password</i> telah diganti

BAB 5 PERANCANGAN DAN IMPLEMENTASI SISTEM

5.1 Perancangan Sistem

Tahap ini dilakukan berdasarkan hasil analisis kebutuhan yang sudah didapatkan. Tahap perancangan sistem memiliki empat tahapan yang masing-masing adalah perancangan arsitektur, perancangan komponen, perancangan data, dan perancangan antarmuka. Hasil dari tahap perancangan sistem kemudian akan menjadi dasar untuk implementasi sistem.

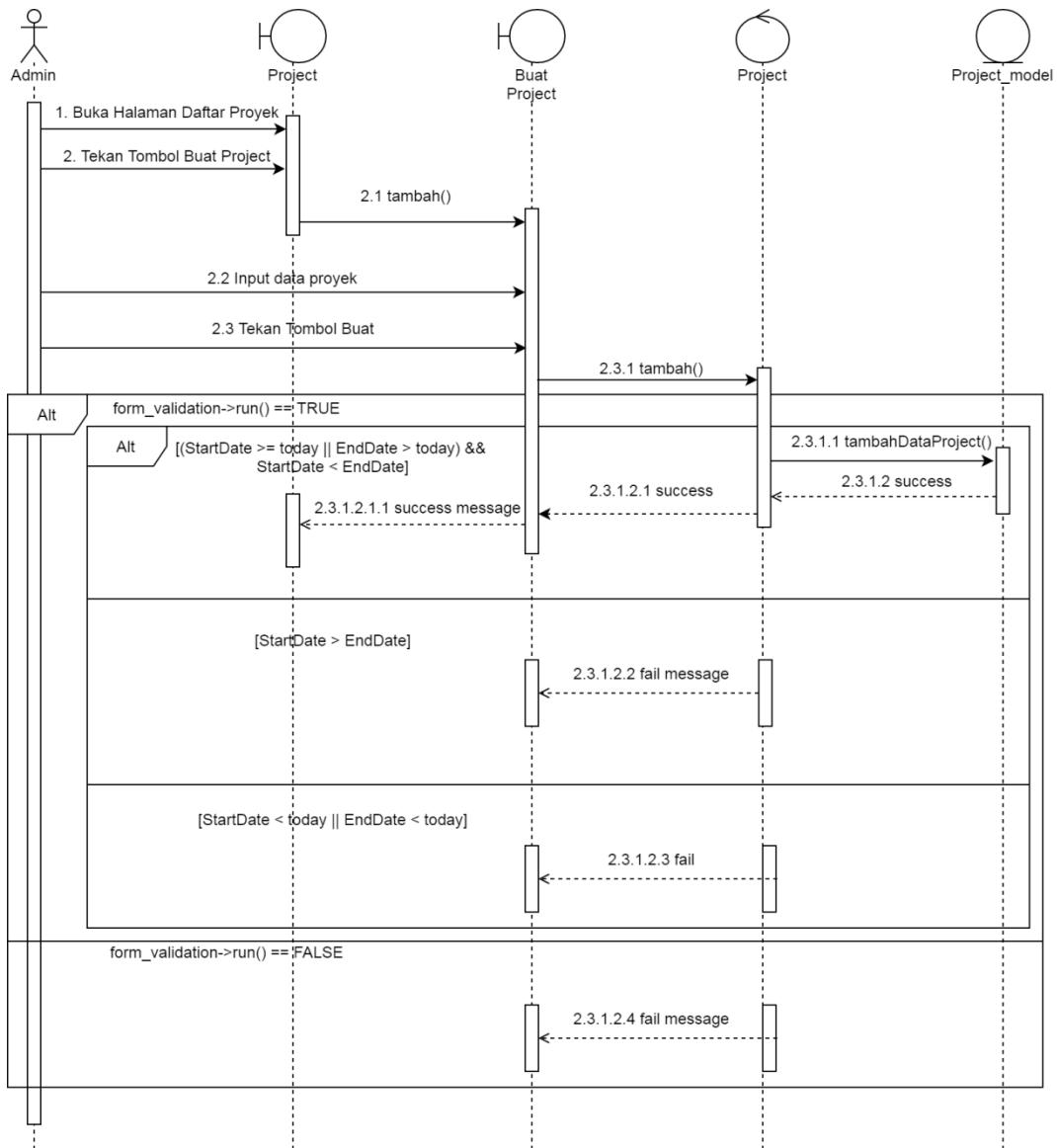
5.1.1 Perancangan Arsitektur

Perancangan arsitektur akan menjelaskan secara rinci mengenai *sequence diagram* dan *class diagram*. *Sequence diagram* menjelaskan alur pesan antar objek dalam sistem. Lebih lanjut, terdapat tiga *sequence diagram* yang akan dijelaskan dimana ketiganya mewakili proses utama dari sistem manajemen proyek menggunakan metode *Kanban*. Ketiga proses tersebut yakni menambahkan proyek, menambahkan *board*, dan menambah tugas. Selain itu, juga akan dijelaskan mengenai *class diagram* yang menggambarkan *class-class* beserta hubungannya sebagai penyusun sistem.

5.1.1.1 Sequence Diagram Menambah Data Proyek

Sequence diagram menggambarkan interaksi yang terjadi ketika suatu fitur sistem dijalankan. Model menggunakan *sequence diagram* dikerjakan mengacu pada *use case diagram* dan *use case scenario* yang telah disusun pada tahap sebelumnya. Setiap *sequence diagram* menjelaskan alur berjalanannya suatu *use case*.

Gambar 5.1 merupakan alur proses dari *use case* Menambah Data Proyek pada sistem ini. *Sequence diagram* ini terdiri dari 5 objek yaitu, 1 aktor, 2 *boundary* yaitu *Project* dan *Buat Project*, 1 *controller* yaitu *Project*, dan 1 model yaitu *Project_model*. Pada langkah pertama pengguna memilih tombol “Buat Project” pada halaman *Project* yang kemudian sistem memanggil *method tambah()* milik *controller Project*, setelah itu sistem menampilkan halaman *Buat Proyek*. Selanjutnya aktor diminta untuk mengisi data yang diperlukan berupa nama proyek, tanggal mulai dan selesai, deskripsi proyek, *client*, *project manager* dan kemudian setelah data terisi aktor menekan tombol “Buat”. Data tersebut akan diproses oleh *controller Project* dengan memanggil *method tambah()*. *Controller Project* kemudian akan memanggil *method* milik entitas *Project_model* yaitu *method tambahDataProject* untuk memasukkan data ke database, *method* ini tidak memiliki parameter karena data diproses dengan metode pengiriman “POST”. Apabila proses buat proyek berjalan lancar maka selanjutnya akan ditampilkan halaman daftar proyek untuk aktor, namun apabila gagal maka sistem akan menampilkan pesan error yang terjadi. Sebelum data masuk ke *database* data akan dicek terlebih dahulu, data yang dicek adalah apakah form telah terisi penuh, apakah *StartDate* dan *EndDate* tidak kurang dari hari ini, dan apakah *StartDate* melebihi *EndDate*.

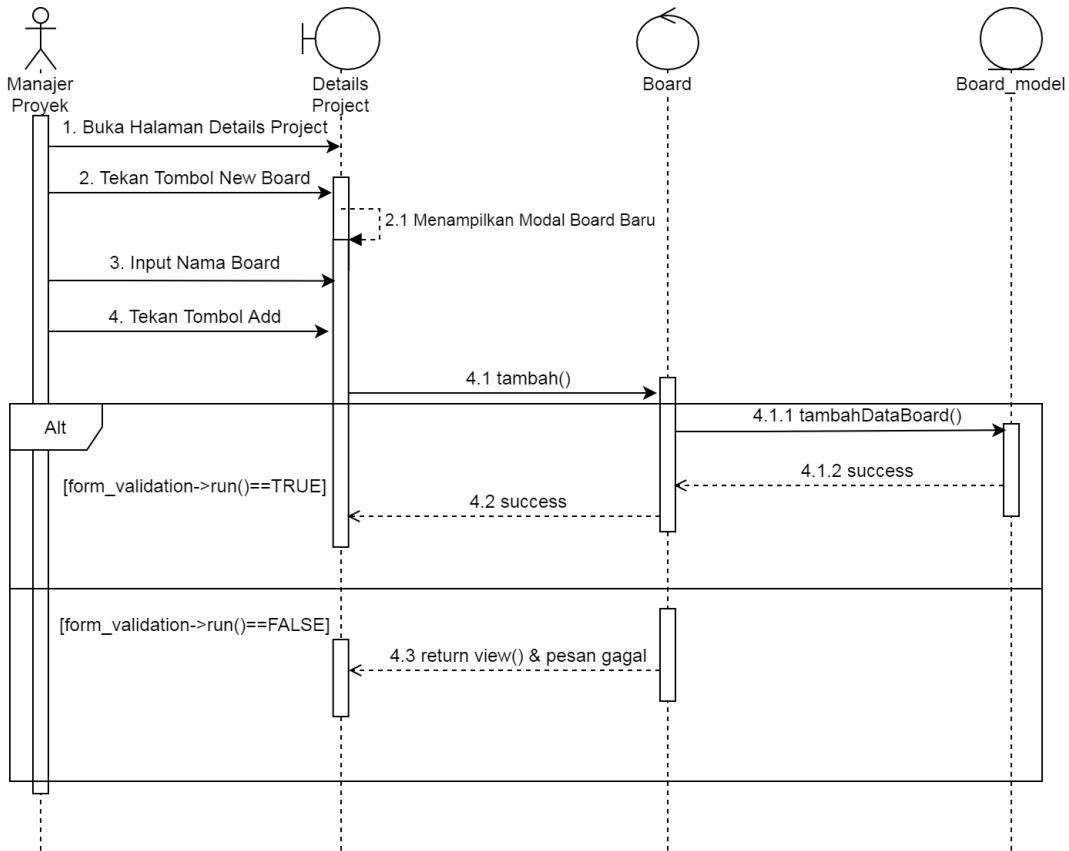


Gambar 5.1 Sequence Diagram Menambah Data Proyek

5.1.1.2 Sequence Diagram Menambah Data Board

Gambar 5.2 merupakan alur proses dari *use case* Menambah Data *Board* pada sistem ini. *Sequence diagram* ini terdiri dari 4 objek yaitu, 1 aktor, 1 *boundary* yaitu *Details Project*, 1 *controller* yaitu *Board*, dan 1 model yaitu *Board_model*. Pada langkah pertama aktor membuka halaman Details Proyek, kemudian aktor memilih tombol “New Board” yang kemudian sistem akan menampilkan sebuah modal dialog. Selanjutnya sistem akan meminta aktor untuk mengisi data yang diperlukan berupa nama *board* dan kemudian setelah data terisi aktor menekan tombol “Add”. Data tersebut akan diproses oleh *controller* *Board* dengan memanggil method *tambahDataBoard()*. *Controller* *Board* kemudian akan memanggil method milik entitas *Board_model* yaitu method *tambahDataBoard()* untuk memasukkan data ke database, method ini tidak memiliki parameter karena data diproses dengan metode pengiriman

“POST”. Apabila proses tambah *Board* berjalan maka selanjutnya akan ditampilkan halaman kanban proyek untuk aktor. Sebelum data dimasukkan ke database maka akan dilakukan pengecekan terlebih dahulu, pengecekan yang dilakukan adalah apakah form telah diisi, apabila belum maka sistem akan menampilkan pesan error yang terjadi.

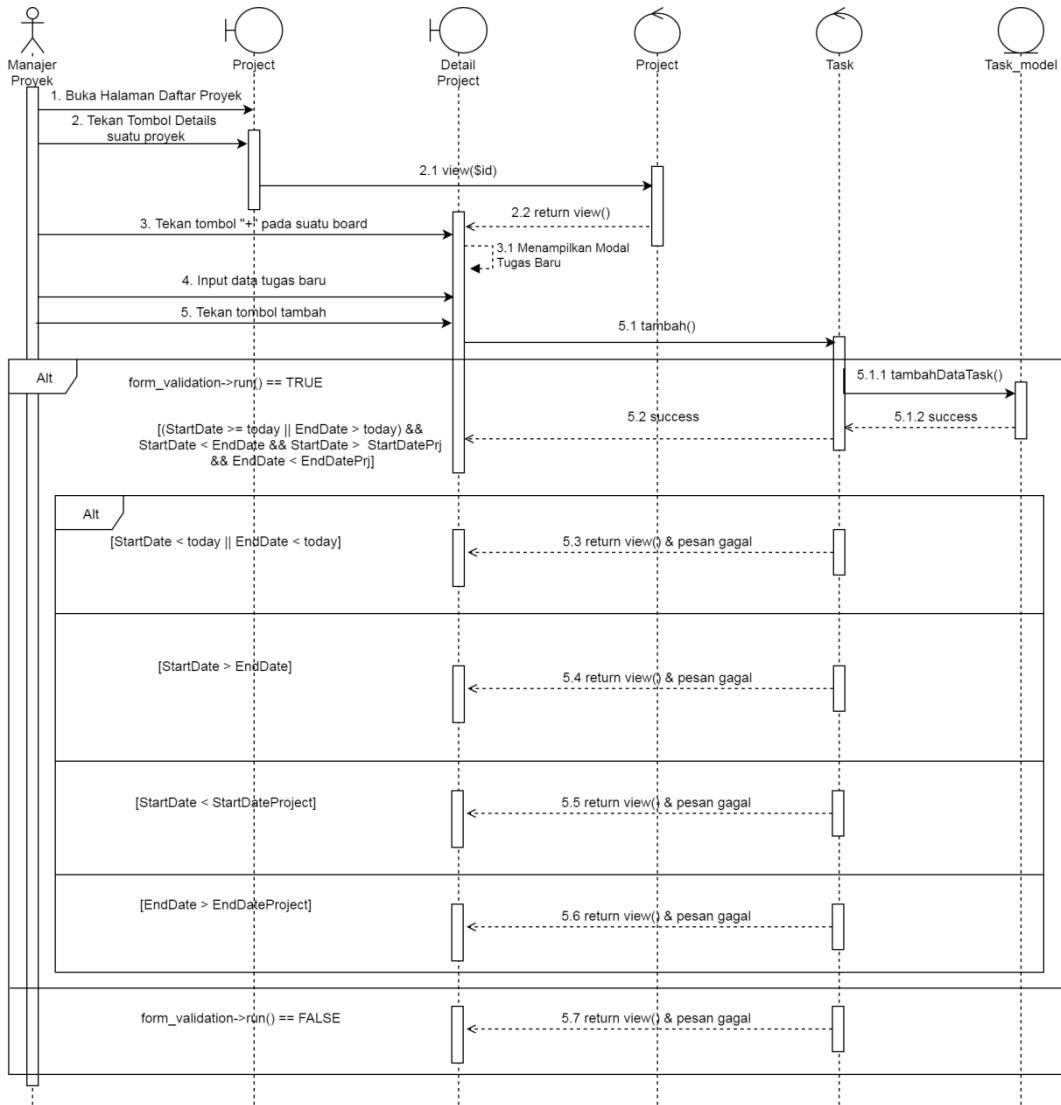


Gambar 5.2 Sequence Diagram Menambah Data Board

5.1.1.3 Sequence Diagram Menambah Data Tugas

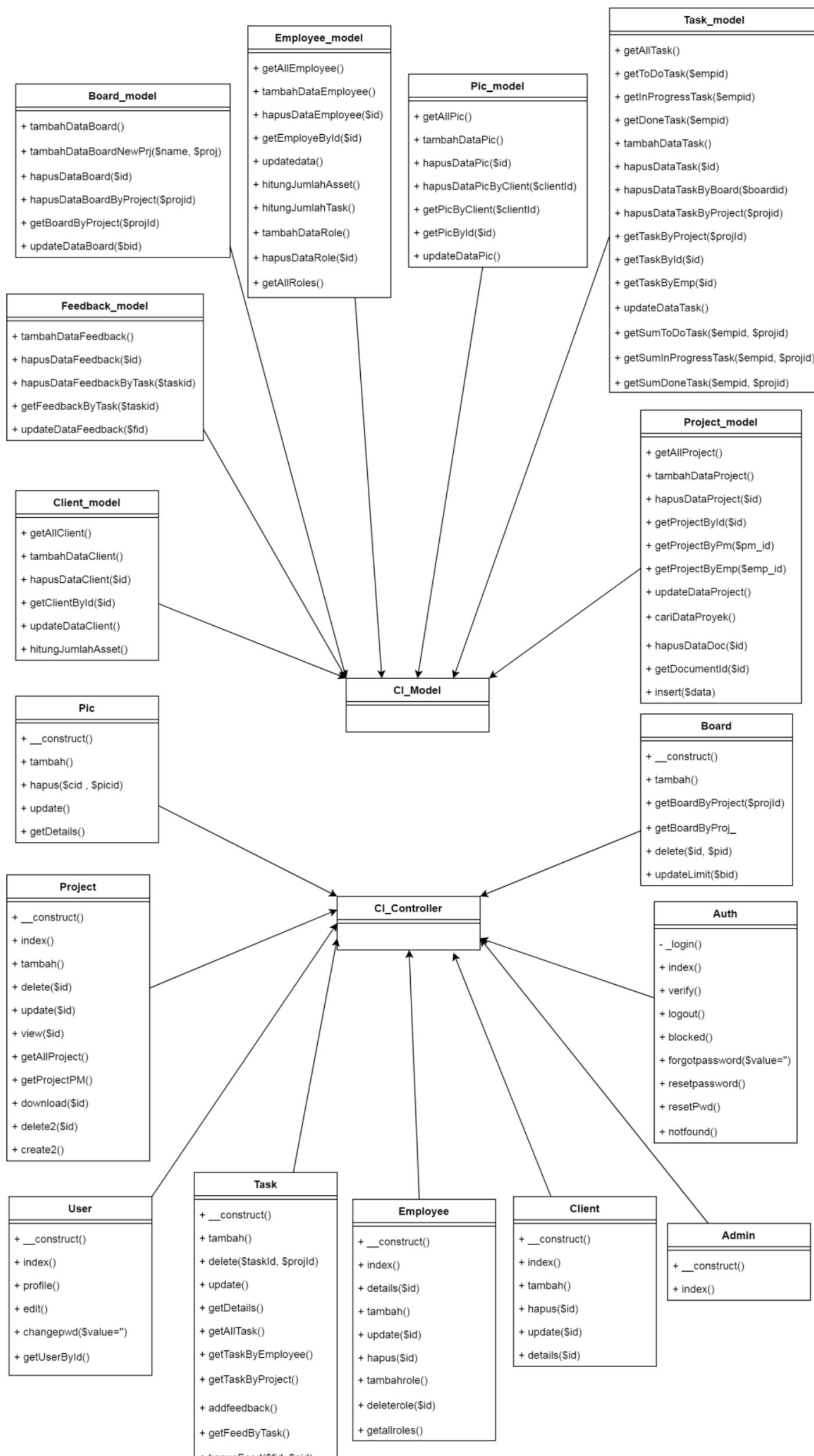
Gambar 5.3 merupakan alur proses dari *use case* Menambah Data Tugas pada sistem ini. Pada langkah pertama aktor membuka halaman Project, kemudian aktor menekan tombol “Details” pada salah satu proyek yang kemudian sistem memanggil *method* *view(\$id)* milik *controller Project*, setelah itu sistem menampilkan halaman Detail Project. Selanjutnya aktor menekan tombol “+” pada salah satu *board* proyek, maka sistem akan menampilkan sebuah modal dialog. Selanjutnya sistem akan meminta aktor untuk mengisi data yang diperlukan berupa nama tugas, tanggal mulai dan selesai, deskripsi tugas, status, penerima tugas dan kemudian setelah data terisi aktor menekan tombol “Tambah”. Data tersebut akan diproses oleh *controller Task* dengan memanggil *method* *tambah()*. *Controller Task* kemudian akan memanggil *method* milik entitas *Task_model* yaitu *method* *tambahDataTask()* untuk memasukkan data ke database, *method* ini tidak memiliki parameter karena data diproses dengan metode pengiriman “POST”. Apabila proses tambah tugas berjalan lancar maka selanjutnya akan ditampilkan halaman kanban proyek dan pesan sukses untuk

aktor. Sebelum data dimasukkan ke database maka akan dilakukan pengecekan terlebih dahulu, pengecekan yang dilakukan adalah apakah Startdate tidak kurang dari hari ini, apakah EndDate kurang dari hari ini, apakah StartDate melebihi EndDate, apakah StartDate kurang dari StartDateProject, apakah EndDate lebih dari EndDateProject dan apakah form telah diisi sepenuhnya apabila belum maka sistem akan menampilkan pesan error sesuai seleksi yang telah dilakukan.



Gambar 5.3 Sequence Diagram Menambah Data Tugas

5.1.1.4 Class Diagram



Gambar 5.4 Class Diagram Sistem Manajemen Proyek

Pada Gambar 5.4 ditunjukkan *class diagram* yang menghubungkan antar *class* yang terdapat pada sistem. Terdapat sembilan *class controller* yang merupakan turunan dari *class CI_Controller* yaitu *class Pic, Project, User, Task, Employee, Client, Admin, Auth, dan Board*. Terdapat tujuh *class model* yang merupakan turunan dari *class CI_Model* yaitu *class Client_model, Feedback_model, Board_model, Employee_model, Pic_model, Task_model, dan Project_model*.

5.1.2 Perancangan Komponen

Perancangan komponen akan memaparkan rincian komponen dari suatu *use case* hingga bagian terkecilnya dalam bentuk alur algoritma. Pada penelitian ini akan dipaparkan tiga algoritma dari *method* yang ada dalam sistem. Pemaparan tersebut dapat dilihat pada sub bab 5.1.2.1 hingga sub bab 5.1.2.3.

5.1.2.1 Perancangan Komponen *Method tambah()* pada *Controller Project*

Pada Tabel 5.1 memaparkan algoritma *method tambah* yang digunakan untuk menambahkan data proyek pada tabel *project*. Mula-mula algoritma ini melakukan pengecekan terhadap user yang mengakses method tersebut, kemudian menyimpan data ‘title’, ‘client’, ‘emp’, ‘user’ pada suatu array. Setelah data disimpan dalam array, algoritma ini akan mengecek apakah algoritma ini diakses untuk memasukkan data ke *database*, jika algoritma ini tidak digunakan untuk memasukkan data ke *database* maka algoritma akan memuat *view* untuk menambahkan proyek baru, apabila algoritma dipanggil untuk memasukkan data ke *database* maka akan dilakukan pengecekan, pengecekan pertama adalah apakah startdate dan enddate kurang dari hari ini, pengecekan yang kedua adalah apakah startdate melebihi enddate. Jika kondisi pengecekan terpenuhi maka sistem akan melakukan set *session* yang berisi pesan eror yang terjadi, dan apabila kondisi pengecekan tidak terpenuhi maka algoritma menambahkan data ke *database* dan sistem akan mengarahkan user ke halaman daftar proyek serta menampilkan pesan sukses.

Nama *class*: *Project*

Nama *method*: *tambah*

Tabel 5.1 Pseudocode Algoritma tambah

Pseudocode algoritma tambah pada controller <i>Project</i>	
1	Mulai
2	Melakukan pengecekan hak akses user
3	Jika akses tidak diizinkan maka akan ditampilkan halaman <i>blocked</i>
4	Deklarasi dan inisialisasi array berisi data yang diperlukan untuk navbar, topbar, sidebar dan juga judul halaman
5	Pemanggilan library <i>form_validasi</i> untuk nama, Startdate, Enddate, description, progress, client dan pm dengan rule tidak kosong dengan rule tidak kosong
6	Seleksi kondisi rule form
7	Jika form dalam keadaan kosong maka sistem memuat halaman form tambah <i>Project</i> dan menampilkan pesan gagal
8	Jika tidak maka sistem akan melakukan pengecekan terhadap data yang dimasukkan

Pseudocode algoritma tambah pada controller Project	
9	Melakukan pengecekan StartDate dan EndDate apakah kurang dari hari ini jika benar maka sistem menampilkan pesan gagal pada halaman tambah proyek
10	Melakukan pengecekan apakah StartDate melebihi EndDate, jika iya maka sistem menampilkan pesan gagal pada halaman tambah proyek
11	Data terisi dengan benar dan tidak memenuhi kondisi pengecekan
12	Data ditambahkan pada database melalui method pada class Model
13	Menambahkan data Board untuk proyek yang baru dibuat
14	Sistem memuat halaman daftar proyek dan menampilkan pesan sukses
15	Selesai

5.1.2.2 Perancangan Komponen *Method tambah()* pada Controller Board

Pada Tabel 5.2 memaparkan algoritma *method tambah* yang digunakan untuk menambahkan data *board* pada tabel *board*. Mula-mula algoritma ini melakukan pengecekan apakah field ada yang belum terisi, jika kondisi pengecekan tidak terpenuhi maka sistem akan menambahkan data *board* melalui *class model*, mengarahkan user kembali ke halaman *Kanban* proyek dan menampilkan pesan sukses. apabila kondisi pengecekan terpenuhi maka algoritma akan mengarahkan user ke halaman daftar proyek dan menampilkan pesan gagal.

Nama *class*: *Board*

Nama *method*: *tambah*

Tabel 5.2 Pseudocode Algoritma tambah

Pseudocode algoritma tambah pada controller Board	
1	Mulai
2	Pemanggilan library form_validasi untuk nama dengan rule tidak kosong
3	Seleksi kondisi form
4	Jika form dalam keadaan kosong maka sistem memuat halaman detail proyek
5	Jika form tidak kosong maka data akan ditambahkan pada database melalui method pada class Model
6	Sistem memuat halaman Kanban proyek dan menampilkan pesan sukses
7	Selesai

5.1.2.3 Perancangan Komponen *Method tambah()* pada Controller Task

Pada Tabel 5.3 memaparkan algoritma *method tambah* yang digunakan untuk menambahkan data tugas pada tabel *task*. Mula-mula algoritma ini melakukan pengecekan terhadap *user* yang mengakses method tersebut, kemudian menyimpan data ‘title’, ‘client’, ‘emp’, ‘user’ pada suatu array. Setelah data disimpan dalam array, algoritma ini akan mengecek apakah algoritma ini diakses untuk memasukkan data ke *database*, jika algoritma ini tidak digunakan untuk memasukkan data ke *database* maka algoritma akan memuat *view* untuk menambahkan proyek baru, apabila algoritma dipanggil untuk memasukkan data

ke *database* maka akan dilakukan pengecekan, pengecekan pertama adalah apakah startdate dan enddate kurang dari hari ini, pengecekan yang kedua adalah apakah startdate melebihi enddate. Jika kondisi pengecekan terpenuhi maka sistem akan melakukan set *session* yang berisi pesan eror yang terjadi, dan apabila kondisi pengecekan tidak terpenuhi maka algoritma menambahkan data ke *database* dan sistem akan mengarahkan user ke halaman daftar proyek serta menampilkan pesan sukses.

Nama *class*: *Task*

Nama *method*: *tambah*

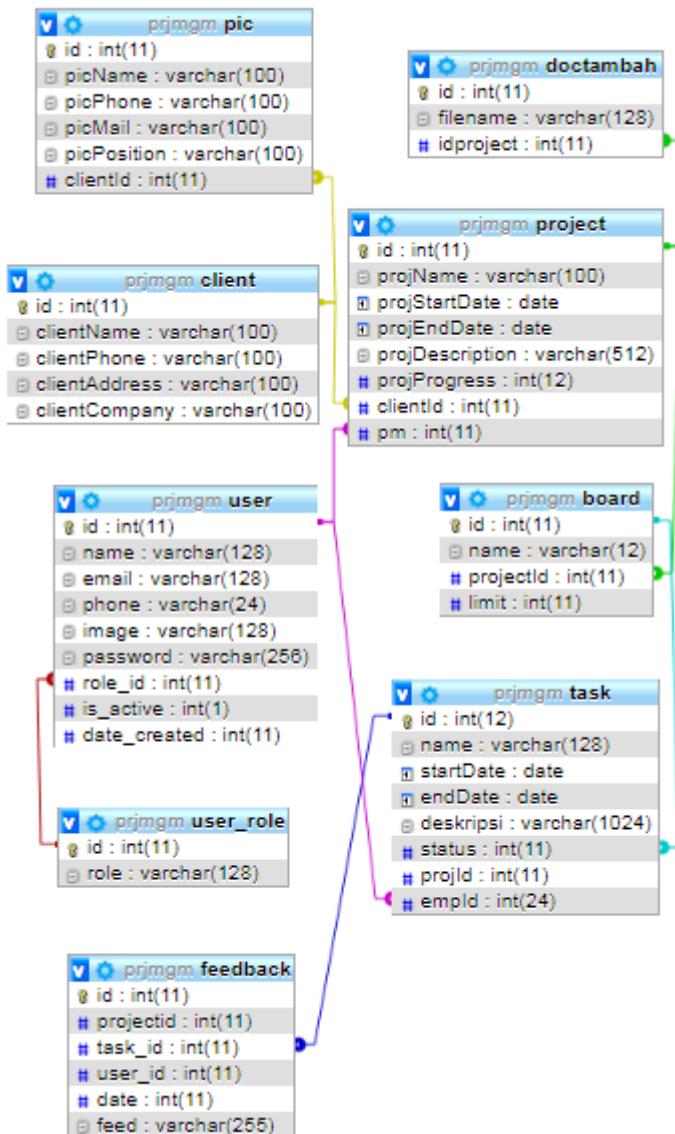
Tabel 5.3 Pseudocode Algoritma tambah

Pseudocode algoritma tambah pada controller Task	
1	Mulai
2	Melakukan pengecekan hak akses user
3	Jika akses tidak diizinkan maka akan ditampilkan halaman <i>blocked</i>
4	Jika hak akses diizinkan, maka dilakukan pemanggilan library <i>form_validasi</i> untuk nama, Startdate, Enddate, description, status, assignee,
5	Deklarasi dan inisialisasi variabel <i>thisproj</i> untuk menyimpan data proyek saat ini
6	Seleksi kondisi form
7	Jika form dalam keadaan kosong maka sistem memuat halaman detail Proyek
8	Jika form tidak kosong maka sistem akan melakukan pengecekan terhadap data yang dimasukkan
9	Melakukan pengecekan StartDate dan EndDate apakah kurang dari hari ini, jika iya maka sistem menampilkan pesan gagal pada halaman detail proyek
10	Melakukan pengecekan apakah StartDate melebihi EndDate, jika iya maka sistem menampilkan pesan gagal pada halaman detail proyek
11	Melakukan pengecekan apakah StartDate kurang dari <i>StartDateProj</i> , jika iya maka sistem menampilkan pesan gagal pada halaman detail proyek
12	Melakukan pengecekan apakah EndDate melebihi <i>EndDateProj</i> , jika iya maka sistem menampilkan pesan gagal pada halaman detail proyek
13	Data tidak memenuhi kondisi pengecekan
14	Data ditambahkan pada <i>database</i> melalui method pada class <i>Model</i>
15	Sistem memuat halaman <i>Kanban</i> proyek dan menampilkan pesan sukses
16	Selesai

5.1.3 Perancangan Data

Perancangan data dilakukan berdasarkan pada hasil dari analisis data yang telah dilakukan, pada sub-bab ini hasil analisis berupa entitas-entitas yang saling berhubungan ditranslasikan menjadi tabel, dan hubungan antar tabel ini dituangkan dalam *physical data model* pada Gambar 5.5. Sesuai dengan analisis data yang telah dilakukan, sembilan entitas yang digunakan telah digunakan sebagai perancangan tabel basis data pada sistem ini, diantaranya adalah tabel *pic*, *client*, *doctambah*, *project*, *board*, *user*, *user_role*, *task*, dan *feedback*.

5.1.3.1 Physical Data Model



Gambar 5.5 Physical Data Model Sistem Manajemen Proyek

5.1.4 Perancangan Antarmuka

Pada bagian perancangan antarmuka memaparkan mengenai rancangan dari tampilan sistem yang akan digunakan sebagai penghubung interaksi dengan pengguna. Rancangan tampilan yang terdapat pada perancangan antarmuka nantinya akan digunakan sebagai acuan implementasi antarmuka dalam tahap implementasi. Pada bagian ini akan dipaparkan tiga gambar perancangan antarmuka yang dapat dilihat pada sub bab 5.1.4.1 hingga sub bab 5.1.4.3. Pada perancangan antarmuka *Sistem Manajemen Proyek Menggunakan Metode Kanban* terdapat tampilan yang mencakup bagian *header*, *sidebar*, dan bagian *body*, namun pada gambar perancangan antarmuka yang dipaparkan hanya bagian *body* dari suatu halaman dan juga *body* dari suatu *modal dialog*.

5.1.4.1 Perancangan Antarmuka Halaman Menambah Data Proyek

Gambar 5.6 adalah perancangan antarmuka dari bagian *body* halaman untuk menambahkan proyek, *body* dari halaman ini berupa *form* yang memiliki tujuh buah *field* yang dapat diisi oleh aktor dan dua buah tombol yang dapat digunakan oleh aktor. Nomor 1 adalah field untuk nama proyek, nomor 2 adalah field untuk tanggal mulai proyek, nomor 3 adalah field untuk tanggal selesai atau tenggat waktu proyek, nomor 4 adalah field untuk deskripsi proyek, nomor 5 adalah field untuk progres proyek, nomor 6 adalah field untuk *client* proyek, dan field nomor 7 adalah field untuk manajer proyek yang dipilih untuk menangani proyek tersebut, nomor 8 adalah tombol kembali untuk kembali ke halaman daftar proyek dan nomor 9 adalah tombol Buat untuk menyimpan data proyek.

Nama Sistem	Nama Perusahaan		Nama Pengguna 
Side Bar	Judul Halaman	<input type="text"/> 1 <input type="text"/> 2 <input type="text"/> 3	
	<input type="text"/> 4		
	<input type="text"/> 5		
	Option 1	6	 
	Option 1	7	 
	Button	8	
		<input type="text"/> 9	
			

Gambar 5.6 Gambar Rancangan Menambah Data Proyek

5.1.4.2 Perancangan Antarmuka Modal Dialog Menambah Data Board

Gambar 5.7 adalah perancangan antarmuka dari bagian *body modal dialog* untuk menambahkan *board*, *body* dari *modal* ini berupa *form* yang memiliki satu buah *field* yang dapat diisi oleh aktor dan dua buah tombol yang dapat digunakan oleh aktor. Nomor 1 adalah field untuk nama *board*, nomor 2 adalah tombol untuk menutup modal dialog, nomor 3 adalah tombol “Add” untuk menyimpan data *board*.

Nama Sistem	Nama Perusahaan	Nama Pengguna 
	<div style="border: 1px solid #ccc; padding: 10px; width: fit-content; margin: auto;"> <p>Add New Board</p> <input type="text" value="1"/> <div style="display: flex; justify-content: space-around; width: 100%;"> Close Add </div> <p>2 3</p> </div>	
Side Bar		

Gambar 5.7 Gambar Rancangan Menambah Data Board

5.1.4.3 Perancangan Antarmuka Modal Dialog Menambah Data Tugas

Gambar 5.8 adalah perancangan antarmuka dari bagian *body modal dialog* untuk menambahkan tugas, *body* dari *modal dialog* ini berupa *form* yang memiliki tujuh buah *field* yang dapat diisi oleh aktor dan dua buah tombol yang dapat digunakan oleh aktor. Nomor 1 adalah label untuk nama proyek, nomor 2 adalah field untuk nama tugas, nomor 3 adalah field untuk tanggal mulai tugas, nomor 4 adalah field untuk tanggal selesai/tenggat waktu tugas, nomor 5 adalah *field* untuk deskripsi tugas, nomor 6 adalah *field* untuk status tugas, dan *field* nomor 7 adalah *field* untuk pegawai yang dipilih untuk mengerjakan tugas tersebut dan nomor 8 adalah tombol Tambah untuk menyimpan data tugas.

Nama Sistem	Nama Perusahaan	Nama Pengguna 
	<div style="border: 1px solid #ccc; padding: 10px; width: fit-content; margin: auto;"> <p>Tambah Task</p> <input type="text" value="1"/> <input type="text" value="2"/> <input type="text" value="3"/> <input type="text" value="4"/> <input type="text" value="5"/> Option 1 <input type="text" value="6"/> Option 1 <input type="text" value="7"/> 8 Button </div>	<p>List Feedback</p> <div style="border: 1px solid #ccc; height: 100px; width: 100%;"></div>
Side Bar		

Gambar 5.8 Gambar Rancangan Menambah Data Tugas

5.2 Implementasi Sistem

Setelah tahap perancangan selesai tahap selanjutnya adalah implementasi sistem. Implementasi dilakukan berdasarkan hasil dari perancangan sistem yang didapatkan dari tahap sebelumnya dan dari tahap analisis kebutuhan sistem. Implementasi sistem dengan kode program yang memiliki dasar dari perancangan algoritma, diikuti dengan implementasi perancangan data yang didasarkan pada hasil analisis data dan perancangan data serta implementasi perancangan antarmuka yang didasarkan pada perancangan antarmuka. Penjelasan mengenai implementasi sistem lebih lanjut akan dimulai dari spesifikasi sistem, kemudian implementasi perancangan data, implementasi kode program, dan implementasi antarmuka.

5.2.1 Spesifikasi Sistem

Bagian ini akan memaparkan spesifikasi yang digunakan oleh sistem mulai dari perangkat lunak hingga perangkat keras yang digunakan selama pengembangan sistem berlangsung. Spesifikasi perangkat keras yang digunakan dapat dilihat pada Tabel 5.4, sementara untuk spesifikasi perangkat lunak dapat dilihat pada Tabel 5.5.

Tabel 5.4 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
<i>Processor</i>	<i>Intel Core i5 with Intel HD Graphics 3000</i>
<i>Hard Disk Drive</i>	298 GB
<i>Main Memory (RAM)</i>	4,00 GB

Tabel 5.5 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
<i>Operating System</i>	Windows 7 64-bit
<i>Text Editor</i>	Sublime Text 3
<i>Programming Language</i>	PHP 7.2.9
<i>Framework</i>	Codeigniter 3.1.10
<i>Editor Dokumentasi</i>	Draw.io Visual Paradigm Online
<i>Editor Perancangan</i>	Microsoft Word 2010

5.2.2 Implementasi Data

Berikut adalah implementasi perancangan data yang telah dilakukan, implementasi dilakukan menggunakan database MySql, dan bahasa pemrograman Sql. Kode sumber DDL implementasi tertuang dalam Tabel 5.6, tabel ini memuat beberapa query untuk membuat tabel-tabel yang digunakan pada sistem ini dan juga memuat penambahan *constraint* yang mendefinisikan aturan pengisian suatu tabel dan hubungan antar tabel yang ada.

Tabel 5.6 Implementasi Rancangan Physical Data Model

No.	Kode
1	CREATE TABLE `board` (
2	`id` int(11) NOT NULL,
3	`name` varchar(12) NOT NULL,
4	`projectId` int(11) NOT NULL,
5	`limit` int(11) NOT NULL
6) ENGINE=InnoDB DEFAULT CHARSET=latin1;
7	
8	CREATE TABLE `client` (
9	`id` int(11) NOT NULL,
10	`clientName` varchar(100) NOT NULL,
11	`clientPhone` varchar(100) NOT NULL,
12	`clientAddress` varchar(100) NOT NULL,
13	`clientCompany` varchar(100) NOT NULL
14) ENGINE=InnoDB DEFAULT CHARSET=latin1;
15	
16	CREATE TABLE `doctambah` (
17	`id` int(11) NOT NULL,
18	`filename` varchar(128) NOT NULL,
19	`idproject` int(11) NOT NULL
20) ENGINE=InnoDB DEFAULT CHARSET=latin1;
21	
22	CREATE TABLE `feedback` (
23	`id` int(11) NOT NULL,
24	`projectid` int(11) NOT NULL,
25	`task_id` int(11) NOT NULL,
26	`user_id` int(11) NOT NULL,
27	`date` int(11) NOT NULL,
28	`feed` varchar(255) NOT NULL
29) ENGINE=InnoDB DEFAULT CHARSET=latin1;
30	
31	CREATE TABLE `pic` (
32	`id` int(11) NOT NULL,
33	`picName` varchar(100) NOT NULL,
34	`picPhone` varchar(100) NOT NULL,
35	`picMail` varchar(100) NOT NULL,
36	`picPosition` varchar(100) NOT NULL,
37	`clientId` int(11) NOT NULL
38) ENGINE=InnoDB DEFAULT CHARSET=latin1;
39	
40	CREATE TABLE `project` (
41	`id` int(11) NOT NULL,
42	`projName` varchar(100) NOT NULL,
43	`projStartDate` date NOT NULL,
44	`projEndDate` date NOT NULL,
45	`projDescription` varchar(512) NOT NULL,
46	`projProgress` int(12) NOT NULL,

No.	Kode
47	`clientId` int(11) NOT NULL,
48	`pm` int(11) NOT NULL
49) ENGINE=InnoDB DEFAULT CHARSET=latin1;
50	
51	
52	CREATE TABLE `task` (
53	`id` int(12) NOT NULL,
54	`name` varchar(128) NOT NULL,
55	`startDate` date NOT NULL,
56	`endDate` date NOT NULL,
57	`deskripsi` varchar(1024) NOT NULL,
58	`status` int(11) NOT NULL,
59	`projId` int(11) NOT NULL,
60	`empId` int(24) NOT NULL
61) ENGINE=InnoDB DEFAULT CHARSET=latin1;
62	
63	CREATE TABLE `user` (
64	`id` int(11) NOT NULL,
65	`name` varchar(128) NOT NULL,
66	`email` varchar(128) NOT NULL,
67	`phone` varchar(24) NOT NULL,
68	`image` varchar(128) NOT NULL,
69	`password` varchar(256) NOT NULL DEFAULT 'primavisi123',
70	`role_id` int(11) NOT NULL,
71	`is_active` int(1) NOT NULL,
72	`date_created` int(11) NOT NULL
73) ENGINE=InnoDB DEFAULT CHARSET=latin1;
74	
75	CREATE TABLE `user_role` (
76	`id` int(11) NOT NULL,
77	`role` varchar(128) NOT NULL
78) ENGINE=InnoDB DEFAULT CHARSET=latin1;
79	
80	ALTER TABLE `board`
81	ADD PRIMARY KEY (`id`),
82	ADD KEY `projectId` (`projectId`),
83	ADD KEY `projectId_2` (`projectId`),
84	ADD KEY `projectId_3` (`projectId`),
85	ADD KEY `projectId_4` (`projectId`);
86	
87	ALTER TABLE `client`
88	ADD PRIMARY KEY (`id`);
89	
90	ALTER TABLE `doctambah`
91	ADD PRIMARY KEY (`id`),
92	ADD KEY `idproject` (`idproject`);
93	
94	ALTER TABLE `feedback`
95	ADD PRIMARY KEY (`id`),
96	ADD KEY `task_id` (`task_id`);
97	
98	ALTER TABLE `pic`
99	ADD PRIMARY KEY (`id`),
100	ADD KEY `clientId` (`clientId`);
101	
102	ALTER TABLE `project`
103	ADD PRIMARY KEY (`id`),
104	ADD KEY `clientId` (`clientId`),

No.	Kode
105	ADD KEY `pm` (`pm`);
106	
107	ALTER TABLE `task`
108	ADD PRIMARY KEY (`id`),
109	ADD KEY `status` (`status`, `projId`),
110	ADD KEY `empId` (`empId`),
111	ADD KEY `status_2` (`status`),
112	ADD KEY `projId` (`projId`);
113	
114	ALTER TABLE `user`
115	ADD PRIMARY KEY (`id`),
116	ADD KEY `role_id` (`role_id`),
117	ADD KEY `role_id_2` (`role_id`);
118	
119	ALTER TABLE `user_role`
120	ADD PRIMARY KEY (`id`);
121	
122	ALTER TABLE `board`
123	ADD CONSTRAINT `board_to_project` FOREIGN KEY
124	(`projectId`) REFERENCES `project` (`id`) ON DELETE CASCADE
125	ON UPDATE CASCADE;
126	
127	ALTER TABLE `doctambah`
128	ADD CONSTRAINT `doctambah_ibfk_1` FOREIGN KEY
129	(`idproject`) REFERENCES `project` (`id`) ON DELETE CASCADE
130	ON UPDATE CASCADE;
131	
132	ALTER TABLE `feedback`
133	ADD CONSTRAINT `feedback_ibfk_1` FOREIGN KEY (`task_id`)
134	REFERENCES `task` (`id`) ON DELETE CASCADE ON UPDATE
135	CASCADE;
136	
137	ALTER TABLE `pic`
138	ADD CONSTRAINT `pic_ibfk_1` FOREIGN KEY (`clientId`)
139	REFERENCES `client` (`id`) ON DELETE CASCADE ON UPDATE
140	CASCADE;
141	
142	ALTER TABLE `project`
143	ADD CONSTRAINT `project_ibfk_1` FOREIGN KEY (`pm`)
144	REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE
145	CASCADE,
146	ADD CONSTRAINT `project_ibfk_2` FOREIGN KEY (`clientId`)
147	REFERENCES `client` (`id`) ON DELETE CASCADE ON UPDATE
148	CASCADE;
149	
150	ALTER TABLE `task`
151	ADD CONSTRAINT `task_ibfk_1` FOREIGN KEY (`empId`)
152	REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE
153	CASCADE,
154	ADD CONSTRAINT `task_ibfk_2` FOREIGN KEY (`status`)
155	REFERENCES `board` (`id`) ON DELETE CASCADE ON UPDATE
156	CASCADE;
157	
158	ALTER TABLE `user`
159	ADD CONSTRAINT `user_ibfk_1` FOREIGN KEY (`role_id`)
160	REFERENCES `user_role` (`id`);
161	COMMIT;

5.2.3 Implementasi Kode Program

Bagian ini akan memaparkan mengenai implementasi kode program. Algoritma yang ada pada perancangan komponen di tahap sebelumnya akan digunakan sebagai dasar penulisan kode program. Bahasa pemrograman yang digunakan pada penelitian ini adalah bahasa PHP. Implementasi kode program dari sistem dapat dilihat pada sub bab 5.2.3.1 hingga sub bab 5.2.3.3.

5.2.3.1 Implementasi *Pseudocode Method tambah()*

Method tambah() pada kelas *controller Project* berfungsi untuk menambahkan data proyek baru kedalam database, sesuai dengan *pseudocode* yang dituliskan, algoritma yang dirancang telah diimplementasikan dan kode sumber implementasi algoritma tersebut terruang dalam Tabel 5.7.

Tabel 5.7 Source Code Method tambah() pada Controller Project

Method tambah()	
1	public function tambah()
2	{
3	is_admin(\$_SESSION['user']['role_id']);
4	\$data['title'] = 'Buat Project';
5	\$data['client'] = \$this->Client_model-
6	>getAllClient();
7	\$data['emp'] = \$this->Employee_model-
8	>getAllEmployee();
9	\$data['user'] = \$this->db-
10	>get_where('user', ['email' => \$_SESSION['user']['email']])->row_array();
11	\$this->form_validation->set_rules('nama',
12	'Nama', 'required');
13	\$this->form_validation-
14	>set_rules('startdate', 'Start Date', 'required');
15	\$this->form_validation-
16	>set_rules('enddate', 'End Date', 'required');
17	\$this->form_validation-
18	>set_rules('description', 'Description', 'required');
19	\$this->form_validation-
20	>set_rules('progress', 'Progress', 'required');
21	\$this->form_validation-
22	>set_rules('client', 'Client', 'required');
23	\$this->form_validation->set_rules('pm',
24	'Project Manager', 'required');
25	if (\$this->form_validation->run() ==
26	FALSE) {
27	\$this->load-
28	>view('templates/header', \$data);
29	\$this->load-
30	>view('templates/sidebar', \$data);
31	\$this->load-
32	>view('templates/topbar', \$data);
33	\$this->load-
34	>view('project/tambahForm', \$data);
35	\$this->load-
36	>view('templates/footer');
37	}else{
38	if (\$this->input->post('startdate')
39	

```

Method tambah()
40 < date("Y-m-d") || $this->input->post('enddate') < date("Y-
41 m-d")) {
42                                     $this->session-
43 >set_flashdata('message', '<div class="alert alert-danger
44 alert-dismissible fade show mt-3" role="alert">Start / end
45 date tidak boleh kurang dari hari ini ! <button
46 type="button" class="close" data-dismiss="alert" aria-
47 label="Close"><span aria-
48 hidden="true">&times;</span></button></div>');
49                                     redirect('project/tambah');
50 }else if($this->input-
51 >post('startdate') > $this->input->post('enddate')){
52                                     $this->session-
53 >set_flashdata('message', '<div class="alert alert-danger
54 alert-dismissible fade show mt-3" role="alert">Start date
55 tidak boleh melebihi end date ! <button type="button"
56 class="close" data-dismiss="alert" aria-label="Close"><span aria-
57 hidden="true">&times;</span></button></div>
58                                     <div class="alert
59 alert-danger mt-3" role="alert">End date tidak boleh kurang
60 dari start date ! <button type="button" class="close" data-
61 dismiss="alert" aria-label="Close"><span aria-
62 hidden="true">&times;</span></button></div>');
63                                     redirect('project/tambah');
64 }else{
65                                     $this->Project_model-
66 >tambahDataProject();
67                                     $this->Board_model-
68 >tambahDataBoardNewPrj('To Do', $this->input->post('nama',
69 true));
70                                     $this->Board_model-
71 >tambahDataBoardNewPrj('In Progress', $this->input-
72 >post('nama', true));
73                                     $this->Board_model-
74 >tambahDataBoardNewPrj('Done', $this->input->post('nama',
75 true));
76                                     $this->session-
77 >set_flashdata('message', '<div class="alert alert-success
78 alert-dismissible fade show mt-3" role="alert">Proyek telah
79 dibuat, notifikasi e-mail telah dikirim ke PM. <button
80 type="button" class="close" data-dismiss="alert" aria-
81 label="Close"><span aria-
82 hidden="true">&times;</span></button></div>');
83                                     redirect('project');
84 }
85 }
86 }

```

5.2.3.2 Implementasi *Pseudocode Method tambah()*

Method tambah() pada kelas *controller Board* berfungsi untuk menambahkan data *board* baru kedalam *database*, sesuai dengan *pseudocode* yang dituliskan, algoritma yang dirancang telah diimplementasikan dan kode sumber implementasi algoritma tersebut tertuang dalam Tabel 5.8.

Tabel 5.8 Source Code Method tambah() pada Controller Board

Method tambah()	
1	public function tambah()
2	{
3	\$this->form_validation->set_rules('nama',
4	'Nama', 'required');
5	if (\$this->form_validation->run() ==
6	FALSE) {
7	\$this->session-
8	>set_flashdata('message', '<div class="alert alert-danger
9	alert-dismissible fade show mt-3" role="alert">Nama board
10	harus diisi ! <button type="button" class="close" data-
11	dismiss="alert" aria-label="Close"><span aria-
12	hidden="true">×</button></div>');
13	redirect('project/view/'.\$this-
14	>input->post('projId'));
15	} else{
16	\$this->Board_model-
17	>tambahDataBoard();
18	\$this->session-
19	>set_flashdata('message', '<div class="alert alert-success
20	alert-dismissible fade show mt-3" role="alert">Board baru
21	telah dibuat ! <button type="button" class="close" data-
22	dismiss="alert" aria-label="Close"><span aria-
23	hidden="true">×</button></div>');
24	redirect('project/view/'.\$this-
25	>input->post('projId'));
26	}
27	}

5.2.3.3 Implementasi Pseudocode Method tambah()

Method tambah() pada kelas controller Task berfungsi untuk menambahkan data tugas baru kedalam database, sesuai dengan pseudocode yang dituliskan, algoritma yang dirancang telah diimplementasikan dan kode sumber implementasi algoritma tersebut tertuang dalam Tabel 5.9.

Tabel 5.9 Source Code Method tambah() pada Controller Task

Method tambah()	
1	public function tambah()
2	{
3	\$this->form_validation->set_rules('nama',
4	'Nama', 'required');
5	\$this->form_validation-
6	>set_rules('startdate', 'Start Date', 'required');
7	\$this->form_validation-
8	>set_rules('enddate', 'End Date', 'required');
9	\$this->form_validation-
10	>set_rules('description', 'Description', 'required');
11	\$this->form_validation-
12	>set_rules('status', 'Status', 'required');
13	\$this->form_validation-
14	>set_rules('assignee', 'Assignee', 'required');
15	\$thisproj = \$this->db-
16	>get_where('project', ['id' => \$this->input->post('projId', true)])->row_array();
17	

```

Method tambah()
18         if ($this->form_validation->run() ==
19 FALSE) {
20             $this->session-
21 >set_flashdata('message','<div class="alert alert-danger
22 alert-dismissible fade show mt-3" role="alert">Task gagal
23 ditambahkan ! <button type="button" class="close" data-
24 dismiss="alert" aria-label="Close"><span aria-
25 hidden="true">&times;</span></button></div>');
26             redirect('project/view/'.$this-
27 >input->post('projId'));
28         } else{
29             if ($this->input->post('startdate') < date("Y-m-d") || $this->input->post('enddate') < date("Y-
30 m-d")) {
31                 $this->session-
32 >set_flashdata('message','<div class="alert alert-danger
33 alert-dismissible fade show mt-3" role="alert">Start / end
34 date tidak boleh kurang dari hari ini ! <button
35 type="button" class="close" data-dismiss="alert" aria-
36 label="Close"><span aria-
37 hidden="true">&times;</span></button></div>');
38
39             redirect('project/view/'.$this->input-
40 >post('projId'));
41         } else if($this->input-
42 >post('startdate') > $this->input->post('enddate')){
43             $this->session-
44 >set_flashdata('message','<div class="alert alert-danger
45 alert-dismissible fade show mt-3" role="alert">Start date
46 tidak boleh melebihi end date ! <button type="button"
47 class="close" data-dismiss="alert" aria-label="Close"><span
48 aria-hidden="true">&times;</span></button></div>
49
50             <div class="alert alert-
51 danger mt-3" role="alert">End date tidak boleh kurang dari
52 start date ! <button type="button" class="close" data-
53 dismiss="alert" aria-label="Close"><span aria-
54 hidden="true">&times;</span></button></div>');
55
56             redirect('project/view/'.$this->input-
57 >post('projId'));
58         }
59     } else if($this->input-
60 >post('startdate') < $this->input->post('startdateprj')){
61         $this->session-
62 >set_flashdata('message','<div class="alert alert-danger
63 alert-dismissible fade show mt-3" role="alert">Start date
64 task tidak boleh kurang dari start date project ! <button
65 type="button" class="close" data-dismiss="alert" aria-
66 label="Close"><span aria-
67 hidden="true">&times;</span></button></div>');
68
69         redirect('project/view/'.$this->input-
70 >post('projId'));
71     }
72     else if($this->input-
73 >post('enddate') > $this->input->post('enddateprj')){
74         $this->session-
75 >set_flashdata('message','<div class="alert alert-danger

```

```

Method tambah()
76    alert-dismissible fade show mt-3" role="alert">End date task
77    tidak boleh melebihi end date project ! <button
78    type="button" class="close" data-dismiss="alert" aria-
79    label="Close"><span aria-
80    hidden="true">&times;</span></button></div>');
81
82        redirect('project/view/'.$this->input-
83    >post('projId'));
84        }
85        else{
86            $this->Task_model->tambahDataTask();
87            $this->session-
88            >set_flashdata('message','<div class="alert alert-success
89            alert-dismissible fade show mt-3" role="alert">Task telah
90            dibuat! Notifikasi e-mail telah dikirim ke employee.<button
91            type="button" class="close" data-dismiss="alert" aria-
92            label="Close"><span aria-
93            hidden="true">&times;</span></button></div>');
94            redirect('project/view/'.$this-
95    >input->post('projId'));
96            }
97        }
98    }
}

```

5.2.4 Implementasi Antarmuka

Pada bagian implementasi antarmuka memaparkan hasil implementasi yang didasarkan dari perancangan antarmuka yang telah dilakukan sebelumnya. Dalam implementasi antarmuka ini akan dipaparkan tiga hasil implementasi antarmuka yang terdapat pada sub bab 5.2.4.1 hingga sub bab 5.2.4.3.

5.2.4.1 Implementasi Antarmuka Halaman Menambah Data Proyek

Buat Project

Nama

Start Date
09/09/2019

End Date
mm/dd/yyyy

Description

Progress (%)
0

Client
RS Universitas Brawijaya

PM
Beni Dektos Heronimus

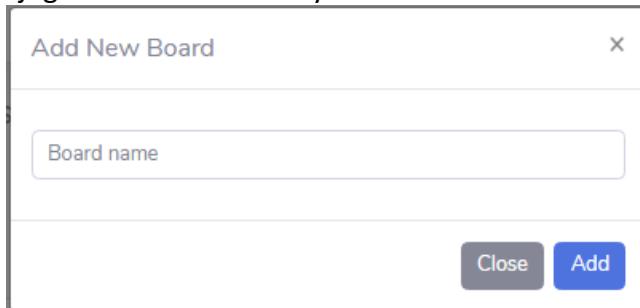
Kembali Buat

Gambar 5.9 Gambar Implementasi Antarmuka Menambah Data Proyek

Pada Gambar 5.9 dapat dilihat hasil implementasi antarmuka body dari rancangan halaman menambah data proyek yang disesuaikan dengan perancangan antarmukanya. Terdapat tujuh buah field dimana tiga diantaranya adalah field bertipe teks, dua field bertipe *date*, dan dua field bertipe *options* dan juga dua buah tombol yaitu tombol kembali dan buat.

5.2.4.2 Implementasi Antarmuka Modal Dialog Menambah Data Board

Pada Gambar 5.10 dapat dilihat hasil implementasi antarmuka modal dialog dari rancangan menambah data data board yang disesuaikan dengan perancangan antarmukanya. Terdapat satu buah field dimana field tersebut bertipe teks dan juga dua buah tombol yaitu *Close* dan *Add*.



Gambar 5.10 Implementasi Antarmuka Menambah Data Board

5.2.4.3 Implementasi Antarmuka Modal Dialog Menambah Data Tugas

Pada Gambar 5.11 dapat dilihat hasil implementasi antarmuka modal dialog dari rancangan menambah data tugas yang disesuaikan dengan perancangan antarmukanya. Terdapat tujuh buah field dimana dua diantaranya adalah field bertipe teks, dua field bertipe *date*, dan dua field bertipe *options*, satu buah label dan juga se buah tombol yaitu tambah.

A screenshot of a modal dialog window titled "Tambah Task". The window contains several input fields: "Nama Project" with value "Project Aplikasi RS", "Nama Task" (empty), "Start Date" set to "02/24/2020" with a calendar icon, "End Date" (empty), "Description" (empty), "Status" (empty dropdown), "Assignee" (empty dropdown), and a "Feedback..." text area. At the bottom right is a blue "Tambah" button.

Gambar 5.11 Implementasi Antarmuka Menambah Data Tugas

BAB 6 PENGUJIAN

6.1 White-box Testing

Pada tahap ini akan dilakukan pengujian unit yang akan dilakukan terhadap setiap unit atau komponen dari sistem menggunakan teknik pengujian *basis path testing* dari metode *white box testing*. Dalam tahap ini akan di uji tiga *method*, yaitu *method tambah()* pada *controller Project*, *method tambah()* pada *controller Board*, dan *method tambah()* pada *controller Task* yang dapat dilihat pada subbab 6.1.1 hingga sub-bab 6.1.3.

6.1.1 White-box Testing Method tambah() pada Controller Project

1. Pseudocode

Pengujian unit *method tambah* yang terdapat pada kelas *Project* akan dijelaskan dengan algortime yang ditunjukkan pada Tabel 6.1.

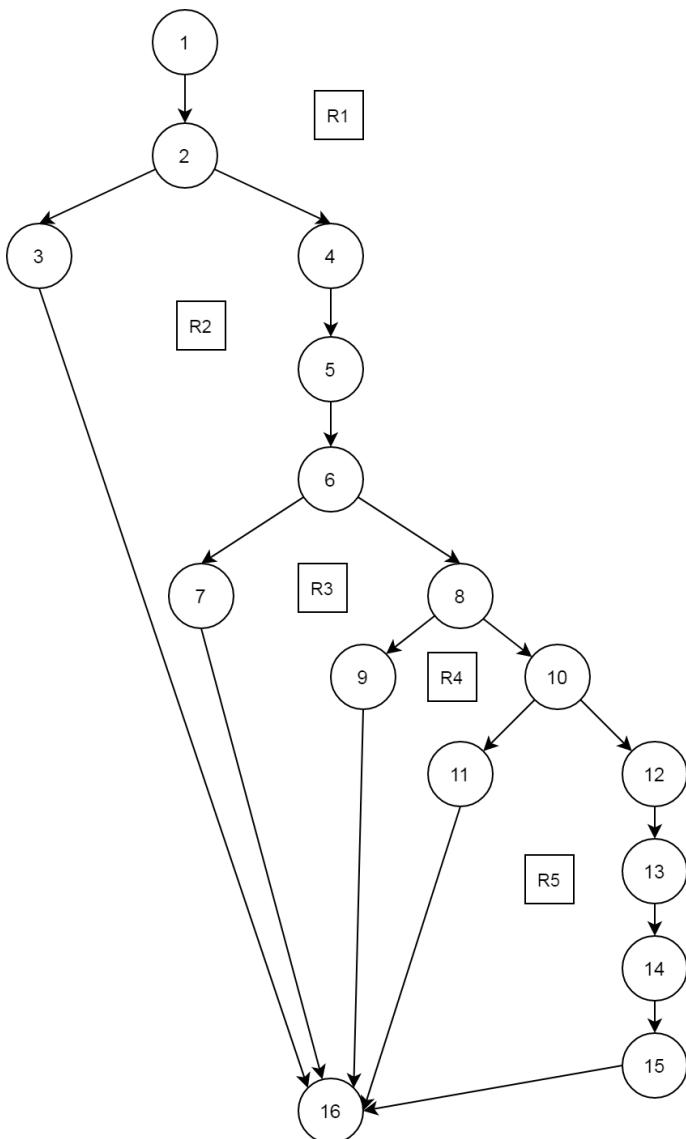
Tabel 6.1 Pseudocode Algoritme tambah

Pseudocode algoritma tambah pada controller Project	
1	Mulai
2	Melakukan pengecekan hak akses user
3	Jika akses tidak diizinkan maka akan ditampilkan halaman <i>blocked</i>
4	Deklarasi dan inisialisasi array berisi data yang diperlukan untuk navbar, topbar, sidebar dan juga judul halaman
5	Pemanggilan library <i>form_validation</i> untuk nama, Startdate, Enddate, description, progress, client dan pm dengan rule tidak kosong dengan rule tidak kosong
6	Seleksi kondisi rule <i>form</i>
7	Jika form dalam keadaan kosong maka sistem memuat halaman <i>form tambah Project</i> dan menampilkan pesan gagal
8	Form dalam keadaan terisi, sistem melakukan pengecekan StartDate atau EndDate apakah kurang dari hari ini
9	Jika kondisi pengecekan poin 8 terpenuhi, maka sistem menampilkan pesan gagal pada halaman tambah proyek
10	Melakukan pengecekan apakah StartDate melebihi EndDate
11	Jika kondisi pengecekan poin 10 terpenuhi, maka sistem menampilkan pesan gagal pada halaman tambah proyek
12	Data terisi dengan benar dan tidak memenuhi kondisi pengecekan poin 8 dan 10
13	Data ditambahkan pada database melalui method pada class Model
14	Menambahkan data <i>Board</i> untuk proyek yang baru dibuat
15	Sistem memuat halaman daftar proyek dan menampilkan pesan sukses
16	Selesai

2. Basis Path Testing

1.1. Flow Graph

Menggunakan pseudocode yang dipaparkan sebelumnya, dapat dibuat *flow graph* dari *method tambah* yang terdapat pada kelas *Project* yang ditunjukkan pada Gambar 6.1.



Gambar 6.1 Flow Graph Method `tambah()` pada kelas `Project`

1.2. Cyclomatic Complexity

Pada Gambar 6.1 akan didapatkan jumlah jalur independen dengan menggunakan kalkulasi *cyclomatic complexity*. Hasil dari kalkulasi *cyclomatic complexity flow graph method* tambah yang terdapat pada kelas `Project` dipaparkan oleh perhitungan berikut ini.

- $V(G) = \text{jumlah region} = 5$
- $V(G) = \text{jumlah edge} - \text{jumlah node} + 2 = 19 - 16 + 2 = 5$
- $V(G) = \text{jumlah predicate node} + 1 = 4 + 1 = 5$

1.3. Independent Path

- Jalur 1: 1-2-3-16
- Jalur 2: 1-2-4-5-6-7-16

- Jalur 3: 1-2-4-5-6-8-9-16
- Jalur 4: 1-2-4-5-6-8-10-11-16
- Jalur 5: 1-2-4-5-6-8-10-12-13-14-15-16

Dari jalur independen yang didapatkan, kemudian akan dijadikan dasar dari pembuatan kasus uji. Hasil pengujian menggunakan kasus uji dipaparkan melalui Tabel 6.2.

Tabel 6.2 Hasil Pengujian Unit *Method tambah*

Jalur	Prosedur Uji	Hasil Pengujian yang diharapkan	Hasil Pengujian	Status
1	<i>Method tambah()</i> diakses menggunakan peramban dengan <i>url</i> secara langsung, method ini diakses dengan menggunakan akun pengguna yang tidak berperan sebagai admin. Seleksi kondisi yang terjadi akan mengakibatkan sistem melakukan <i>return view blocked</i> untuk pengguna.	<i>Method tambah()</i> dijalankan dan menampilkan halaman <i>blocked</i> untuk pengguna.	<i>Method tambah()</i> dijalankan dan menampilkan halaman <i>blocked</i> untuk pengguna.	Valid
2	<i>Method tambah()</i> diakses menggunakan peramban, method ini diakses dengan menggunakan akun pengguna yang berperan sebagai admin. Kemudian sistem menampilkan halaman tambah proyek dan pesan <i>error</i> pada <i>field</i> yang masih kosong.	<i>Method tambah()</i> dijalankan seuai dengan kondisi yang diuji kemudian sistem menampilkan halaman tambah proyek dan pesan <i>error</i> pada <i>field</i> yang masih kosong.	<i>Method tambah()</i> dijalankan seuai dengan kondisi yang diuji kemudian sistem menampilkan halaman tambah proyek dan pesan <i>error</i> pada <i>field</i> yang masih kosong.	Valid
3	<i>Method tambah()</i> diakses menggunakan peramban, method ini diakses dengan menggunakan akun pengguna yang berperan sebagai admin. Lalu sistem menampilkan halaman tambah proyek.	<i>Method tambah()</i> dijalankan dan menampilkan halaman tambah proyek dan pesan “ <i>Start date atau End date</i> tidak boleh kurang dari hari ini”.	<i>Method tambah()</i> dijalankan dan menampilkan halaman tambah proyek dan pesan “ <i>Start date atau End date</i> tidak boleh kurang dari hari ini”.	Valid

Jalur	Prosedur Uji	Hasil Pengujian yang diharapkan	Hasil Pengujian	Status
	Admin mengisi <i>field</i> yang tersedia dan mengisi <i>field Start date</i> atau <i>End date</i> dengan nilai kurang dari hari ini.			
4	<i>Method tambah()</i> diakses menggunakan peramban, method ini diakses dengan menggunakan akun pengguna yang berperan sebagai admin. Lalu sistem menampilkan halaman tambah proyek. Admin mengisi <i>field</i> yang tersedia dan mengisi <i>field Start date</i> melebihi nilai <i>End date</i> .	<i>Method tambah()</i> dijalankan dan menampilkan halaman tambah proyek dan pesan nilai <i>Start date</i> melebihi nilai <i>End date</i> .	<i>Method tambah()</i> dijalankan dan menampilkan halaman tambah proyek dan pesan nilai <i>Start date</i> melebihi nilai <i>End date</i> .	Valid
5	<i>Method tambah()</i> diakses menggunakan peramban, method ini diakses dengan menggunakan akun pengguna yang berperan sebagai admin. Lalu sistem menampilkan halaman tambah proyek. Admin mengisi seluruh <i>field</i> dengan benar dan menekan tombol <i>buat</i> .	<i>Method tambah()</i> dijalankan dengan kondisi yang diujikan, data proyek berhasil ditambahkan kedalam basis data, data <i>board</i> proyek berhasil dibuat, dan sistem menampilkan halaman daftar proyek dan pesan “proyek berhasil dibuat” dan notifikasi <i>email</i> berhasil dikirimkan ke manajer proyek”.	<i>Method tambah()</i> dijalankan dengan kondisi yang diujikan, data proyek berhasil ditambahkan kedalam basis data, data <i>board</i> proyek berhasil dibuat, dan sistem menampilkan halaman daftar proyek dan pesan “proyek berhasil dibuat” dan notifikasi <i>email</i> berhasil dikirimkan ke manajer proyek”.	Valid

6.1.2 White-box Testing Method tambah() pada Controller Board

1. Pseudocode

Pengujian unit *method* tambah yang terdapat pada kelas *Board* akan dijelaskan dengan algortime yang ditunjukkan pada Tabel 6.2.

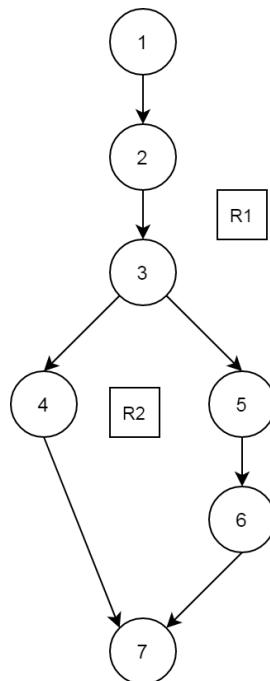
Tabel 6.3 Pseudocode Algoritme tambah

Pseudocode algoritma tambah pada controller Board	
1	Mulai
2	Pemanggilan library form_validasi untuk nama dengan rule tidak kosong
3	Seleksi kondisi form
4	Jika form dalam keadaan kosong maka sistem memuat halaman detail proyek
5	Jika form tidak kosong maka data akan ditambahkan pada database melalui method pada class Model
6	Sistem memuat halaman Kanban proyek dan menampilkan pesan sukses
7	Selesai

2. Basis Path Testing

1.1. Flow Graph

Menggunakan pseudocode yang dipaparkan sebelumnya, dapat dibuat *flow graph* dari *method* tambah yang terdapat pada kelas *Board* yang ditunjukkan pada Gambar 6.2.



Gambar 6.2 Flow Graph Method tambah() pada kelas Board

1.2. Cyclomatic Complexity

Pada Gambar 6.2 akan didapatkan jumlah jalur independen dengan menggunakan kalkulasi *cyclomatic complexity*. Hasil dari kalkulasi *cyclomatic complexity flow graph method* tambah yang terdapat pada kelas *Board* dipaparkan oleh perhitungan berikut ini.

- $V(G) = \text{jumlah region} = 2$
- $V(G) = \text{jumlah edge} - \text{jumlah node} + 2 = 7 - 7 + 2 = 2$
- $V(G) = \text{jumlah predicate node} + 1 = 1 + 1 = 2$

1.3. Independent Path

- Jalur 1: 1-2-3-4-7
- Jalur 2: 1-2-3-5-6-7

Dari jalur independen yang didapatkan, kemudian akan dijadikan dasar dari pembuatan kasus uji. Hasil pengujian menggunakan kasus uji dipaparkan melalui Tabel 6.4.

Tabel 6.4 Hasil Pengujian Unit *Method tambah*

Jalur	Prosedur Uji	Hasil Pengujian yang diharapkan	Hasil Pengujian	Status
1	<i>Method tambah()</i> diakses dengan cara menekan tombol “+Board” pada halaman Kanban proyek, kemudian aktor tidak mengisi <i>field</i> nama dan langsung menekan tombol “Add”.	<i>Method tambah()</i> dijalankan dengan kondisi yang diujikan, kemudian sistem menampilkan halaman detail proyek dan juga pesan bahwa <i>board</i> gagal ditambahkan.	<i>Method tambah()</i> dijalankan dengan kondisi yang diujikan, kemudian sistem menampilkan halaman detail proyek dan juga pesan bahwa <i>board</i> gagal ditambahkan.	Valid
2	<i>Method tambah()</i> diakses dengan cara menekan tombol “+Board” pada halaman Kanban proyek, kemudian aktor mengisi <i>field</i> nama dan menekan tombol “Add”.	<i>Method tambah()</i> dijalankan dengan kondisi yang diujikan, kemudian sistem menambahkan data <i>board</i> ke database, dan menampilkan halaman detail proyek dan juga pesan bahwa <i>board</i> berhasil ditambahkan.	<i>Method tambah()</i> dijalankan dengan kondisi yang diujikan, kemudian sistem menambahkan data <i>board</i> ke database, dan menampilkan halaman detail proyek dan juga pesan bahwa <i>board</i> berhasil ditambahkan.	Valid

6.1.3 White-box Testing Method tambah() pada Controller Task

1. Pseudocode

Pengujian unit *method* tambah yang terdapat pada kelas *Task* akan dijelaskan dengan algortime yang ditunjukkan pada Tabel 6.1.

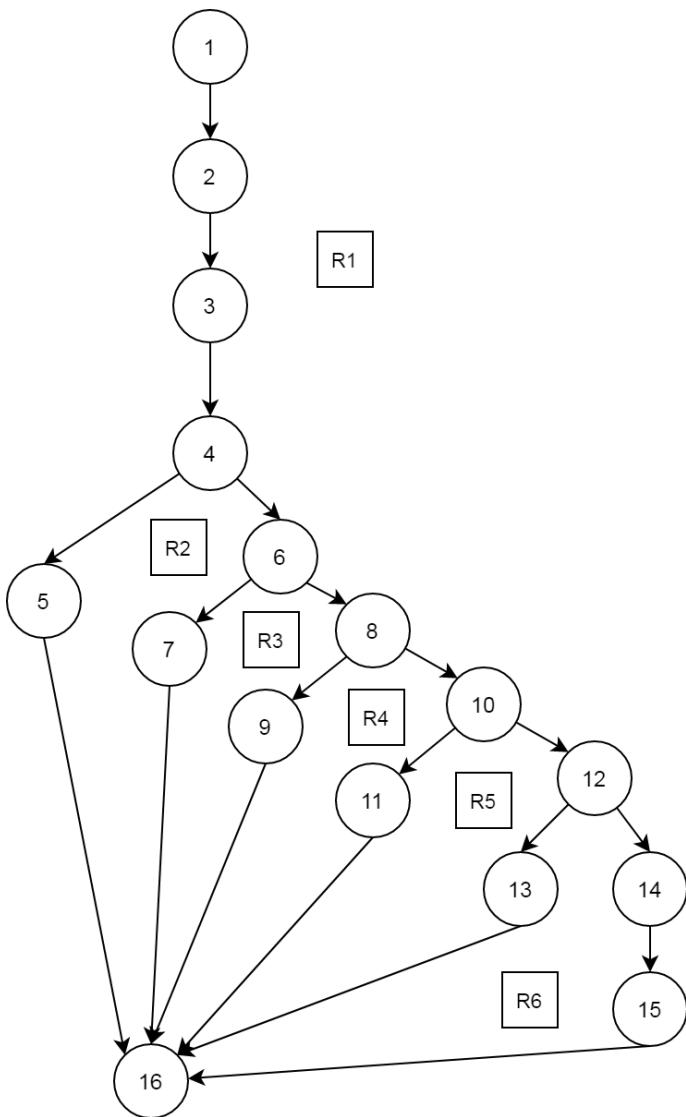
Tabel 6.5 Pseudocode Algoritme tambah

Pseudocode algoritma tambah pada controller Task	
1	Mulai
2	Melakukan pemanggilan library form_validasi untuk nama, Startdate, Enddate, description, status dan assignee
3	Deklarasi dan inisialisasi variabel thisproj untuk menyimpan data proyek saat ini
4	Seleksi kondisi form
5	Jika form dalam keadaan kosong maka sistem memuat halaman detail Proyek
6	Jika form tidak kosong maka sistem akan melakukan pengecekan StartDate dan EndDate apakah kurang dari hari ini
7	Jika kondisi poin 8 terpenuhi, maka sistem menampilkan halaman detail proyek dan pesan gagal
8	Melakukan pengecekan apakah StartDate melebihi EndDate
9	Jika kondisi poin 10 terpenuhi, maka sistem menampilkan halaman detail proyek dan pesan gagal
10	Melakukan pengecekan apakah StartDate kurang dari StartDateProj
11	Jika kondisi poin 12 terpenuhi, maka sistem menampilkan halaman detail proyek dan pesan gagal
12	Melakukan pengecekan apakah EndDate melebihi EndDateProj
13	Jika kondisi poin 14 terpenuhi, maka sistem menampilkan halaman detail proyek dan pesan gagal
14	Jika kondisi 8.4 tidak terpenuhi, maka data task akan dimasukkan kedalam database melalui method pada class Model
15	Sistem memuat halaman Kanban proyek dan menampilkan pesan sukses
16	Selesai

2. Basis Path Testing

1.4. Flow Graph

Menggunakan *pseudocode* yang dipaparkan sebelumnya, dapat dibuat *flow graph* dari *method* tambah yang terdapat pada kelas *Task* yang ditunjukkan pada Gambar 6.3.



Gambar 6.3 Flow Graph Method `tambah()` pada kelas `Task`

1.5. Cyclomatic Complexity

Pada Gambar 6.3 akan didapatkan jumlah jalur independen dengan menggunakan kalkulasi *cyclomatic complexity*. Hasil dari kalkulasi *cyclomatic complexity flow graph method tambah* yang terdapat pada kelas `Taks` dipaparkan oleh perhitungan berikut ini.

- $V(G) = \text{jumlah region} = 6$
- $V(G) = \text{jumlah edge} - \text{jumlah node} + 2 = 20 - 16 + 2 = 6$
- $V(G) = \text{jumlah predicate node} + 1 = 5 + 1 = 6$

1.6. Independent Path

- Jalur 1: 1-2-3-4-5-16
- Jalur 2: 1-2-3-4-5-6-7-16
- Jalur 3: 1-2-3-4-5-6-8-9-16

- Jalur 4: 1-2-3-4-5-6-8-10-11-16
- Jalur 5: 1-2-3-4-5-6-8-10-12-13-16
- Jalur 6: 1-2-3-4-5-6-8-10-12-14-15-16

Dari jalur independen yang didapatkan, kemudian akan dijadikan dasar dari pembuatan kasus uji. Hasil pengujian menggunakan kasus uji dipaparkan melalui Tabel 6.6.

Tabel 6.6 Hasil Pengujian Unit *Method tambah*

Jalur	Prosedur Uji	Hasil Pengujian yang diharapkan	Hasil Pengujian	Status
1	<i>Method tambah()</i> diakses dengan cara menekan tombol “+Tugas” pada halaman detail proyek, aktor mengosongkan seluruh <i>field</i> yang tersedia atau terlupa tidak mengisi salah satu <i>field</i> yang ada dan kemudian menekan tombol “Tambah”.	<i>Method tambah()</i> dijalankan dan menampilkan halaman detail proyek dan pesan bahwa tugas gagal ditambahkan. Tugas gagal ditambahkan ke basis data.	<i>Method tambah()</i> dijalankan dan menampilkan halaman detail proyek dan pesan bahwa tugas gagal ditambahkan. Tugas gagal ditambahkan ke basis data.	Valid
2	<i>Method tambah()</i> diakses dengan cara menekan tombol “+Tugas” pada halaman detail proyek, aktor mengisi seluruh <i>field</i> yang tersedia dan mengisi <i>field Start date</i> atau <i>End date</i> dengan nilai kurang dari hari ini dan kemudian menekan tombol “Tambah”.	<i>Method tambah()</i> dijalankan dan menampilkan halaman detail proyek dan pesan “Start date atau End date tidak boleh kurang dari hari ini”. Tugas gagal ditambahkan ke basis data.	<i>Method tambah()</i> dijalankan dan menampilkan halaman detail proyek dan pesan “Start date atau End date tidak boleh kurang dari hari ini”. Tugas gagal ditambahkan ke basis data.	Valid
3	<i>Method tambah()</i> diakses dengan cara menekan tombol “+Tugas” pada halaman detail proyek, aktor mengisi seluruh <i>field</i> yang tersedia Admin mengisi <i>field</i> yang tersedia dan mengisi <i>field Start date</i> melebihi nilai <i>End date</i> , kemudian aktor menekan tombol “Tambah”.	<i>Method tambah()</i> dijalankan dan menampilkan halaman detail proyek dan pesan “Start date tidak boleh melebihi End date”. Tugas gagal ditambahkan ke basis data.	<i>Method tambah()</i> dijalankan dan menampilkan halaman detail proyek dan pesan “Start date tidak boleh melebihi End date”. Tugas gagal ditambahkan ke basis data.	Valid
4	<i>Method tambah()</i> diakses dengan cara menekan tombol “+Tugas” pada	<i>Method tambah()</i> dijalankan dan menampilkan	<i>Method tambah()</i> dijalankan dan menampilkan	Valid

Jalur	Prosedur Uji	Hasil Pengujian yang diharapkan	Hasil Pengujian	Status
	halaman detail proyek, aktor mengisi seluruh <i>field</i> yang tersedia Admin mengisi <i>field</i> yang tersedia dan mengisi <i>field Start date</i> kurang dari nilai <i>Start date</i> proyek, kemudian aktor menekan tombol “Tambah”.	halaman detail proyek dan pesan “ <i>Start date tugas tidak boleh kurang dari Start date proyek</i> ”. Tugas gagal ditambahkan ke basis data.	halaman detail proyek dan pesan “ <i>Start date tugas tidak boleh kurang dari Start date proyek</i> ”. Tugas gagal ditambahkan ke basis data.	
5	<i>Method tambah()</i> diakses dengan cara menekan tombol “+Tugas” pada halaman detail proyek, aktor mengisi seluruh <i>field</i> yang tersedia Admin mengisi <i>field</i> yang tersedia dan mengisi <i>field End date</i> melebihi nilai <i>Date date</i> proyek, kemudian aktor menekan tombol “Tambah”.	<i>Method tambah()</i> dijalankan dan menampilkan halaman detail proyek dan pesan “ <i>End date tugas tidak boleh melebihi End date proyek</i> ”. Tugas gagal ditambahkan ke basis data.	<i>Method tambah()</i> dijalankan dan menampilkan halaman detail proyek dan pesan “ <i>End date tugas tidak boleh melebihi End date proyek</i> ”. Tugas gagal ditambahkan ke basis data.	Valid
6	<i>Method tambah()</i> diakses dengan cara menekan tombol “+Tugas” pada halaman detail proyek, aktor mengisi seluruh <i>field</i> yang tersedia dengan benar dan kemudian menekan tombol “Tambah”.	<i>Method tambah()</i> dijalankan dengan kondisi yang diujikan, data tugas berhasil ditambahkan kedalam basis data dan sistem menampilkan halaman detail proyek dan pesan “tugas berhasil dibuat dan notifikasi email berhasil dikirimkan ke employee”.	<i>Method tambah()</i> dijalankan dengan kondisi yang diujikan, data tugas berhasil ditambahkan kedalam basis data dan sistem menampilkan halaman detail proyek dan pesan “tugas berhasil dibuat dan notifikasi email berhasil dikirimkan ke employee”.	Valid

6.2 Black-box Testing

Setelah tahap implementasi selesai barulah tahap pengujian *Black-box* dapat dilakukan. Pengujian ini dilakukan untuk mengetahui apakah sistem berjalan seperti yang diharapkan, pengujian ini dilakukan secara terperinci dengan cara menguji setiap fungsi sistem dengan seluruh kemungkinan kasus uji yang dapat terjadi. *Black-box* memungkinkan untuk melakukan pengujian untuk menghasilkan sebuah kondisi yang diharapkan dengan suatu masukan untuk semua fungsi sistem.

6.2.1 Pengujian Validasi Login

1. Kasus uji berhasil melakukan *login*.

Tabel 6.7 Kasus uji berhasil melakukan login

Nama Kasus Uji	Kasus uji berhasil melakukan <i>login</i> .
Prosedur	<ol style="list-style-type: none">1. Mengisi kolom <i>email</i>.2. Mengisi kolom <i>password</i>.3. Menekan tombol <i>login</i>.
Hasil yang diharapkan	Menampilkan halaman <i>dashboard</i> pengguna sesuai dengan <i>role</i> yang dimiliki.
Hasil	Menampilkan halaman <i>dashboard</i> pengguna.
Status	Valid

2. Kasus uji gagal melakukan *login* dengan tidak mengisi kolom *password* dan *email*.

Tabel 6.8 Kasus uji gagal melakukan login dengan tidak mengisi kolom password dan email

Nama Kasus Uji	Kasus uji gagal melakukan <i>login</i> .
Prosedur	<ol style="list-style-type: none">1. Tidak mengisi kolom <i>email</i>.2. Tidak mengisi kolom <i>password</i>.3. Menekan tombol <i>login</i>.
Hasil yang diharapkan	Menampilkan pesan “ <i>the email field is required</i> ” pada kolom <i>email</i> dan pesan “ <i>the password field is required</i> ” pada kolom <i>password</i> .
Hasil	Menampilkan pesan “ <i>the email field is required</i> ” pada kolom <i>email</i> dan pesan “ <i>the password field is required</i> ” pada kolom <i>password</i> .
Status	Valid

3. Kasus uji gagal melakukan *login* dengan tidak mengisi kolom *password*.

Tabel 6.9 Kasus uji gagal melakukan login dengan tidak mengisi kolom password

Nama Kasus Uji	Kasus uji gagal melakukan <i>login</i> .
Prosedur	<ol style="list-style-type: none">1. Mengisi kolom <i>email</i>.

	2. Tidak mengisi kolom <i>password</i> . 3. Menekan tombol <i>login</i> .
Hasil yang diharapkan	Menampilkan pesan “ <i>the password field is required</i> ” pada kolom <i>password</i> .
Hasil	Menampilkan pesan “ <i>the password field is required</i> ” pada kolom <i>password</i> .
Status	Valid

4. Kasus uji gagal melakukan *login* dengan tidak mengisi kolom *email*.

Tabel 6.10 Kasus uji gagal melakukan *login* dengan tidak mengisi kolom *email*

Nama Kasus Uji	Kasus uji gagal melakukan <i>login</i> .
Prosedur	1. Tidak mengisi kolom <i>email</i> . 2. Mengisi kolom <i>password</i> . 3. Menekan tombol <i>login</i> .
Hasil yang diharapkan	Menampilkan pesan “ <i>the email field is required</i> ” pada kolom <i>email</i> .
Hasil	Menampilkan pesan “ <i>the email field is required</i> ” pada kolom <i>email</i> .
Status	Valid.

5. Kasus uji gagal melakukan *login* dengan masukan *email* yang tidak valid pada kolom *email*.

Tabel 6.11 Kasus uji gagal melakukan *login* dengan *email* yang tidak valid

Nama Kasus Uji	Kasus uji gagal melakukan <i>login</i>
Prosedur	1. Mengisi kolom <i>email</i> dengan <i>email</i> yang tidak valid. 2. Mengisi kolom <i>password</i> . 3. Menekan tombol <i>login</i> .
Hasil yang diharapkan	Menampilkan pesan “ <i>The Email field must contain a valid email address</i> ” pada kolom <i>email</i> .
Hasil	Menampilkan pesan “ <i>The Email field must contain a valid email address</i> ” pada kolom <i>email</i> .
Status	Valid

6. Kasus uji gagal melakukan *login* dengan masukan *email* yang belum diaktifkan pada kolom *email*.

Tabel 6.12 Kasus uji gagal melakukan *login* dengan emali yang belum aktif

Nama Kasus Uji	Kasus uji gagal melakukan <i>login</i>
Prosedur	<ol style="list-style-type: none"> 1. Mengisi kolom <i>email</i> dengan <i>email</i> yang belum aktif. 2. Mengisi kolom <i>password</i>. 3. Menekan tombol <i>login</i>.
Hasil yang diharapkan	Menampilkan pesan “ <i>Email is not activated</i> ”.
Hasil	Menampilkan pesan “ <i>The Email field must contain a valid email address</i> ” pada kolom <i>email</i> .
Status	Valid

7. Kasus uji gagal melakukan *login* dengan *email* yang tidak terdaftar dalam sistem.

Tabel 6.13 Kasus uji gagal melakukan *login* dengan *email* yang tidak terdaftar dalam sistem

Nama Kasus Uji	Kasus uji gagal melakukan <i>login</i> .
Prosedur	<ol style="list-style-type: none"> 1. Mengisi kolom <i>email</i>. 2. Mengisi kolom <i>password</i>. 3. Menekan tombol <i>login</i>.
Hasil yang diharapkan	Menampilkan pesan “ <i>Email not registered!</i> ”.
Hasil	Menampilkan pesan “ <i>Email not registered!</i> ”.
Status	Valid.

8. Kasus uji gagal melakukan *login* dengan *password* yang tidak sesuai.

Tabel 6.14 Kasus uji gagal melakukan *login* dengan *password* yang tidak sesuai

Nama Kasus Uji	Kasus uji gagal melakukan <i>login</i> .
Prosedur	<ol style="list-style-type: none"> 1. Mengisi kolom <i>email</i>. 2. Mengisi kolom <i>password</i> yang tidak sesuai. 3. Menekan tombol <i>login</i>.
Hasil yang diharapkan	Menampilkan pesan “ <i>Wrong password!</i> ”.
Hasil	Menampilkan pesan “ <i>Wrong password!</i> ”.
Status	Valid.

6.2.2 Pengujian Validasi Lupa Password

1. Kasus uji berhasil melakukan *reset password*.

Tabel 6.15 Kasus uji berhasil melakukan *reset password*

Nama Kasus Uji	Kasus uji berhasil melakukan <i>reset password</i> .
Prosedur	1. Mengisi kolom <i>email</i> . 2. Menekan tombol <i>reset password</i> .
Hasil yang diharapkan	Menampilkan pesan “Silakan cek email untuk reset password”.
Hasil	Menampilkan pesan “Silakan cek email untuk reset password”.
Status	Valid

2. Kasus uji gagal melakukan *reset password* dengan tidak mengisi kolom *email*.

Tabel 6.16 Kasus uji gagal melakukan *reset password* dengan tidak mengisi kolom *email*

Nama Kasus Uji	Kasus uji gagal melakukan <i>reset password</i> .
Prosedur	1. Mengisi kolom <i>email</i> . 2. Menekan tombol <i>reset password</i> .
Hasil yang diharapkan	Menampilkan pesan “the email field is required” pada kolom <i>email</i> .
Hasil	Menampilkan pesan “the email field is required” pada kolom <i>email</i> .
Status	Valid

3. Kasus uji gagal melakukan *reset password* dengan *email* yang tidak valid.

Tabel 6.17 Kasus uji gagal melakukan *reset password* dengan *email* yang tidak valid

Nama Kasus Uji	Kasus uji gagal melakukan <i>reset password</i>
Prosedur	1. Mengisi kolom <i>email</i> dengan <i>email</i> yang tidak valid. 2. Menekan tombol <i>reset password</i> .
Hasil yang diharapkan	Menampilkan pesan “The Email field must contain a valid email address” pada kolom <i>email</i> .
Hasil	Menampilkan pesan “The Email field must contain a valid email address” pada kolom <i>email</i> .
Status	Valid

4. Kasus uji gagal melakukan *reset password* dengan *email* yang belum aktif.

Tabel 6.18 Kasus uji gagal melakukan *reset password* dengan *email* yang belum aktif

Nama Kasus Uji	Kasus uji gagal melakukan <i>reset password</i>
Prosedur	<ol style="list-style-type: none"> 1. Mengisi kolom <i>email</i> dengan <i>email</i> yang belum aktif. 2. Menekan tombol <i>reset password</i>.
Hasil yang diharapkan	Menampilkan pesan “ <i>Email is not activated!</i> ”.
Hasil	Menampilkan pesan “ <i>Email is not activated!</i> ”.
Status	Valid

5. Kasus uji gagal melakukan *reset password* dengan *email* yang tidak terdaftar.

Tabel 6.19 Kasus uji gagal melakukan *reset password* dengan *email* yang tidak terdaftar

Nama Kasus Uji	Kasus uji gagal melakukan <i>reset password</i>
Prosedur	<ol style="list-style-type: none"> 4. Mengisi kolom <i>email</i> dengan <i>email</i> yang tidak terdaftar. 5. Menekan tombol <i>reset password</i>.
Hasil yang diharapkan	Menampilkan pesan “ <i>Email tidak terdaftar!</i> ”.
Hasil	Menampilkan pesan “ <i>Email tidak terdaftar!</i> ”.
Status	Valid

6.2.3 Pengujian Validasi *Logout*

1. Kasus uji berhasil melakukan *logout*.

Tabel 6.20 Kasus uji berhasil melakukan *logout*

Nama Kasus Uji	Kasus uji berhasil melakukan <i>logout</i> .
Prosedur	<ol style="list-style-type: none"> 1. Pengguna berhasil login kedalam sistem. 2. Menekan menu <i>logout</i> pada <i>sidebar</i>.
Hasil yang diharapkan	Menampilkan halaman <i>login</i> .
Hasil	Menampilkan halaman <i>login</i> .
Status	Valid

6.2.4 Pengujian Validasi Melihat Dashboard

1. Kasus uji berhasil melihat *dashboard*.

Tabel 6.21 Kasus uji berhasil melihat dashboard

Nama Kasus Uji	Kasus uji berhasil melakukan <i>logout</i> .
Prosedur	1. Pengguna berhasil login kedalam sistem. 2. Menekan menu <i>dashboard</i> pada <i>sidebar</i> .
Hasil yang diharapkan	Menampilkan halaman <i>dashboard</i> .
Hasil	Menampilkan halaman <i>dashboard</i> .
Status	Valid

6.2.5 Pengujian Validasi Mencari Proyek

1. Kasus uji berhasil mencari proyek.

Tabel 6.22 Kasus uji berhasil mencari proyek

Nama Kasus Uji	Kasus uji berhasil mencari proyek.
Prosedur	1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>dashboard</i> pada <i>sidebar</i> . 3. Admin memasukkan kata kunci pencarian pada <i>field</i> “Cari”. 4. Admin menekan tombol “Loupe” untuk melakukan pencarian. 5. Menampilkan daftar proyek yang sesuai dengan kata kunci yang dimasukkan.
Hasil yang diharapkan	Menampilkan daftar proyek yang sesuai.
Hasil	Menampilkan daftar proyek yang sesuai.
Status	Valid

2. Kasus uji gagal mencari proyek.

Tabel 6.23 Kasus uji gagal mencari proyek

Nama Kasus Uji	Kasus uji gagal mencari proyek.
Prosedur	1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>dashboard</i> pada <i>sidebar</i> . 3. Admin memasukkan kata kunci pencarian yang tidak sesuai pada <i>field</i> “Cari”. 4. Admin menekan tombol “Loupe” untuk melakukan pencarian. 5. Menampilkan .
Hasil yang diharapkan	Menampilkan pesan “ <i>Proyek tidak ditemukan!</i> ”.
Hasil	Menampilkan pesan “ <i>Proyek tidak ditemukan!</i> ”.
Status	Valid

3. Kasus uji menampilkan semua proyek.

Tabel 6.24 Kasus uji menampilkan semua proyek

Nama Kasus Uji	Kasus uji menampilkan semua proyek.
Prosedur	<ol style="list-style-type: none"> Admin berhasil login kedalam sistem. Admin menekan menu <i>dashboard</i> pada <i>sidebar</i>. Admin mengosongkan <i>field</i> “Cari”. Admin menekan tombol “Loupe”. Menampilkan semua proyek.
Hasil yang diharapkan	Menampilkan semua proyek.
Hasil	Menampilkan semua proyek.
Status	Valid

6.2.6 Pengujian Validasi Menambah Data Proyek

1. Kasus uji berhasil menambah data proyek.

Tabel 6.25 Kasus uji Kasus uji berhasil menambah data proyek

Nama Kasus Uji	Kasus uji berhasil menambah data proyek.
Prosedur	<ol style="list-style-type: none"> Admin berhasil login kedalam sistem. Admin menekan menu <i>project</i> pada <i>sidebar</i>. Admin menekan tombol “+ Proyek”. Admin mengisi seluruh <i>field</i> dengan benar. Admin menekan tombol “Buat”.
Hasil yang diharapkan	Menampilkan halaman <i>Project</i> dan menampilkan pesan “ <i>Proyek telah dibuat, notifikasi e-mail telah dikirim ke PM</i> ”.
Hasil	Menampilkan halaman <i>Project</i> dan menampilkan pesan “ <i>Proyek telah dibuat, notifikasi e-mail telah dikirim ke PM</i> ”.
Status	Valid

2. Kasus uji gagal menambah data proyek dengan seluruh *field* belum terisi atau terdapat *field* belum terisi.

Tabel 6.26 Kasus uji gagal menambah data proyek dengan seluruh *field* belum terisi atau terdapat *field* belum terisi

Nama Kasus Uji	Kasus uji gagal menambah data proyek.
Prosedur	<ol style="list-style-type: none"> Admin berhasil login kedalam sistem. Admin menekan menu <i>project</i> pada <i>sidebar</i>. Admin menekan tombol “+ Proyek”. Admin mengosongkan seluruh <i>field</i> atau tidak mengisi salah satu <i>field</i>. Admin menekan tombol “Buat”.
Hasil yang diharapkan	Menampilkan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.

Hasil	Menampilkan pesan “The ‘nama-field’ field is required” pada kolom yang masih kosong.
Status	Valid

3. Kasus uji gagal menambah data proyek dengan tanggal mulai proyek melebihi tanggal selesai proyek.

Tabel 6.27 Kasus uji gagal menambah data proyek dengan tanggal mulai proyek melebihi tanggal selesai proyek

Nama Kasus Uji	Kasus uji gagal menambah data proyek.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>project</i> pada <i>sidebar</i>. 3. Admin menekan tombol “+ Proyek”. 4. Admin mengisi seluruh <i>field</i>. 5. Admin mengisi tanggal mulai proyek melebihi tanggal selesai proyek. 6. Admin menekan tombol “Buat”.
Hasil yang diharapkan	Menampilkan pesan “Start date tidak boleh melebihi end date !” dan “End date tidak boleh kurang dari start date !”.
Hasil	Menampilkan pesan “Start date tidak boleh melebihi end date !” dan “End date tidak boleh kurang dari start date !”.
Status	Valid

4. Kasus uji gagal menambah data proyek dengan tanggal mulai proyek atau tanggal selesai proyek kurang dari hari ini.

Tabel 6.28 Kasus uji gagal menambah data proyek dengan tanggal mulai proyek atau tanggal selesai proyek kurang dari hari ini

Nama Kasus Uji	Kasus uji gagal menambah data proyek.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>project</i> pada <i>sidebar</i>. 3. Admin menekan tombol “+ Proyek”. 4. Admin mengisi seluruh <i>field</i>. 5. Admin mengisi tanggal mulai proyek dan tanggal selesai proyek kurang dari hari ini. 6. Admin menekan tombol “Buat”.
Hasil yang diharapkan	Menampilkan pesan “Start / end date tidak boleh kurang dari hari ini !”.
Hasil	Menampilkan pesan “Start / end date tidak boleh kurang dari hari ini !”.
Status	Valid

6.2.7 Pengujian Validasi Melihat Proyek

1. Kasus uji berhasil melihat proyek.

Tabel 6.29 Kasus uji berhasil melihat proyek

Nama Kasus Uji	Kasus uji berhasil melakukan <i>logout</i> .
Prosedur	1. Pengguna berhasil login kedalam sistem. 2. Pengguna menekan menu <i>project</i> pada <i>sidebar</i> .
Hasil yang diharapkan	Menampilkan halaman <i>Project</i> berisi daftar proyek yang ditangani perusahaan.
Hasil	Menampilkan halaman <i>Project</i> berisi daftar proyek yang ditangani perusahaan.
Status	Valid

6.2.8 Pengujian Validasi Mengubah Data Proyek

1. Kasus uji berhasil mengubah data proyek.

Tabel 6.30 Kasus uji berhasil mengubah data proyek

Nama Kasus Uji	Kasus uji berhasil mengubah data proyek.
Prosedur	1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>project</i> pada <i>sidebar</i> . 3. Admin menekan tombol dengan icon “Pencil” untuk mengubah data proyek. 4. Admin mengubah data proyek. 5. Admin menekan tombol “Update”.
Hasil yang diharapkan	Menampilkan halaman <i>Project</i> dan menampilkan pesan “ <i>Project telah diupdate!</i> ”.
Hasil	Menampilkan halaman <i>Project</i> dan menampilkan pesan “ <i>Project telah diupdate!</i> ”.
Status	Valid

2. Kasus uji gagal mengubah data proyek.

Tabel 6.31 Kasus uji gagal mengubah data proyek

Nama Kasus Uji	Kasus uji gagal mengubah data proyek.
Prosedur	1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>project</i> pada <i>sidebar</i> . 3. Admin menekan tombol dengan icon “Pencil” untuk mengubah data proyek. 4. Admin tidak mengisi salah satu <i>field</i> . 5. Admin menekan tombol “Update”.
Hasil yang diharapkan	Menampilkan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.
Hasil	Menampilkan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.
Status	Valid

6.2.9 Pengujian Validasi Menghapus Data Proyek

1. Kasus uji berhasil menghapus data proyek.

Tabel 6.32 Kasus uji berhasil menghapus data proyek

Nama Kasus Uji	Kasus uji berhasil menghapus data proyek.
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>project</i> pada <i>sidebar</i>.3. Admin menekan tombol dengan icon “Trash” untuk menghapus data proyek.
Hasil yang diharapkan	Menampilkan halaman <i>Project</i> dan pesan “ <i>Semua data yang berhubungan dengan projek telah dihapus !</i> ”.
Hasil	Menampilkan halaman <i>Project</i> dan pesan “ <i>Semua data yang berhubungan dengan projek telah dihapus !</i> ”.
Status	Valid

6.2.10 Pengujian Validasi Melihat Detail Proyek

1. Kasus uji berhasil melihat detail proyek.

Tabel 6.33 Kasus uji berhasil melihat detail proyek

Nama Kasus Uji	Kasus uji berhasil melihat detail proyek.
Prosedur	<ol style="list-style-type: none">1. Pengguna berhasil login kedalam sistem.2. Pengguna menekan menu <i>project</i> pada <i>sidebar</i>.3. Pengguna menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> .
Hasil	Menampilkan halaman <i>Detail Project</i> .
Status	Valid

6.2.11 Pengujian Validasi Menambah Tugas

1. Kasus uji berhasil menambah tugas.

Tabel 6.34 Kasus uji berhasil menambah tugas

Nama Kasus Uji	Kasus uji berhasil menambahkan tugas.
Prosedur	<ol style="list-style-type: none">1. Manajer proyek berhasil login kedalam sistem.2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>.3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek.4. Manajer proyek menekan tombol “+ Tugas” untuk menambah tugas.

	5. Manajer proyek mengisi seluruh <i>field</i> dengan benar. 6. Manajer proyek menekan tombol "Tambah".
Hasil yang diharapkan	Menampilkan halaman <i>Project</i> dan menampilkan pesan " <i>Task telah dibuat! Notifikasi e-mail telah dikirim ke employee.</i> ".
Hasil	Menampilkan halaman <i>Project</i> dan menampilkan pesan " <i>Task telah dibuat! Notifikasi e-mail telah dikirim ke employee.</i> ".
Status	Valid

2. Kasus uji gagal menambah tugas dengan seluruh *field* belum terisi atau salah satu *field* belum terisi.

Tabel 6.35 Kasus uji gagal menambah tugas dengan seluruh *field* belum terisi atau salah satu *field* belum terisi

Nama Kasus Uji	Kasus uji gagal menambahkan tugas.
Prosedur	1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i> . 3. Manajer proyek menekan tombol dengan <i>icon</i> "Eye" untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol "+ Tugas" untuk menambah tugas. 5. Manajer proyek tidak mengisi seluruh <i>field</i> atau membiarkan salah satu <i>field</i> tidak terisi. 6. Manajer proyek menekan tombol "Tambah".
Hasil yang diharapkan	Menampilkan halaman <i>Project</i> dan menampilkan pesan " <i>Task gagal ditambahkan!</i> ".
Hasil	Menampilkan halaman <i>Project</i> dan menampilkan pesan " <i>Task gagal ditambahkan!</i> ".
Status	Valid

3. Kasus uji gagal menambah tugas dengan *end date* tugas melebihi *end date* proyek.

Tabel 6.36 Kasus uji gagal menambah tugas dengan *end date* tugas melebihi *end date* proyek

Nama Kasus Uji	Kasus uji gagal menambahkan tugas.
Prosedur	1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i> . 3. Manajer proyek menekan tombol dengan <i>icon</i> "Eye" untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol "+ Tugas" untuk menambah tugas.

	5. Manajer proyek mengisi seluruh <i>field</i> . 6. Manajer proyek mengisi <i>field end date</i> tugas dengan tanggal lebih dari <i>start date</i> proyek. 7. Manajer proyek menekan tombol "Tambah".
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan " <i>End date task</i> tidak boleh melebihi <i>end date project</i> !".
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan " <i>End date task</i> tidak boleh melebihi <i>end date project</i> !".
Status	Valid

4. Kasus uji gagal menambah tugas dengan *start date* tugas kurang dari *start date* proyek.

Tabel 6.37 Kasus uji gagal menambah tugas dengan *start date* tugas kurang dari *start date* proyek

Nama Kasus Uji	Kasus uji gagal menambahkan tugas.
Prosedur	1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i> . 3. Manajer proyek menekan tombol dengan <i>icon</i> "Eye" untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol "+ Tugas" untuk menambah tugas. 5. Manajer proyek mengisi seluruh <i>field</i> . 6. Manajer proyek mengisi <i>field start date</i> tugas dengan tanggal kurang dari <i>start date</i> proyek. 7. Manajer proyek menekan tombol "Tambah".
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan " <i>Start date task</i> tidak boleh kurang dari <i>start date project</i> !".
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan " <i>End date task</i> tidak boleh melebihi <i>end date project</i> !".
Status	Valid

5. Kasus uji gagal menambah tugas dengan *start date* tugas kurang dari *end date* tugas.

Tabel 6.38 Kasus uji gagal menambah tugas dengan *start date* tugas kurang dari *end date* tugas

Nama Kasus Uji	Kasus uji gagal menambahkan tugas.
Prosedur	1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i> .

	<ol style="list-style-type: none"> 3. Manajer proyek menekan tombol dengan <i>icon "Eye"</i> untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol “+ Tugas” untuk menambah tugas. 5. Manajer proyek mengisi seluruh <i>field</i>. 6. Manajer proyek mengisi <i>field start date</i> tugas dengan tanggal kurang dari <i>end date</i> tugas. 7. Manajer proyek menekan tombol “Tambah”.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Start date tidak boleh melebihi end date !</i> ” dan “ <i>End date tidak boleh kurang dari start date !</i> ”.
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>End date task tidak boleh melebihi end date project !</i> ”.
Status	Valid

6. Kasus uji gagal menambah tugas dengan *end date* tugas dan *start date* tugas kurang dari hari ini.

Tabel 6.39 Kasus uji gagal menambah tugas dengan *end date* tugas dan *start date* tugas kurang dari hari ini

Nama Kasus Uji	Kasus uji gagal menambahkan tugas.
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan <i>icon "Eye"</i> untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol “+ Tugas” untuk menambah tugas. 5. Manajer proyek mengisi <i>field end date</i> tugas atau <i>start date</i> tugas kurang dari hari ini. 6. Manajer proyek menekan tombol “Tambah”.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Start / end date tidak boleh kurang dari hari ini !</i> ”.
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Start / end date tidak boleh kurang dari hari ini !</i> ”.
Status	Valid

6.2.12 Pengujian Validasi Melihat Daftar Tugas

1. Kasus uji berhasil melihat daftar tugas.

Tabel 6.40 Kasus uji berhasil melihat daftar tugas

Nama Kasus Uji	Kasus uji berhasil melihat daftar tugas.
Prosedur	<ol style="list-style-type: none">1. Pengguna berhasil login kedalam sistem.2. Pengguna menekan menu <i>project</i> pada <i>sidebar</i>.3. Pengguna menekan tombol “Daftar Tugas”.
Hasil yang diharapkan	Menampilkan halaman daftar tugas.
Hasil	Menampilkan halaman daftar tugas.
Status	Valid

6.2.13 Pengujian Validasi Mengupdate Tugas

1. Kasus uji berhasil mengupdate tugas.

Tabel 6.41 Kasus uji berhasil mengupdate tugas

Nama Kasus Uji	Kasus uji berhasil melakukan <i>logout</i> .
Prosedur	<ol style="list-style-type: none">1. <i>Employee</i> berhasil login kedalam sistem.2. <i>Employee</i> menekan menu <i>project</i> pada <i>sidebar</i>.3. <i>Employee</i> menekan tombol dengan <i>icon</i> “Eye” untuk melihat detail proyek suatu proyek.4. <i>Employee</i> menekan tombol dengan <i>icon</i> “Eye” untuk melihat detail tugas suatu tugas.5. <i>Employee</i> mengubah status tugas.6. <i>Employee</i> menekan tombol “Update”.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “Task berhasil diupdate !”.
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “Task berhasil diupdate !”.
Status	Valid

6.2.14 Pengujian Validasi Menghapus Data Tugas

1. Kasus uji berhasil menghapus data tugas.

Tabel 6.42 Kasus uji berhasil menghapus data tugas

Nama Kasus Uji	Kasus uji berhasil menghapus data tugas.
Prosedur	<ol style="list-style-type: none">1. Manajer proyek berhasil login kedalam sistem.2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>.3. Manajer proyek menekan tombol dengan <i>icon</i> “Eye” untuk melihat detail proyek suatu proyek.4. Manajer proyek menekan tombol dengan <i>icon</i> “Trash” untuk menghapus tugas.

Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Data task telah dihapus!</i> ”.
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Data task telah dihapus!</i> ”.
Status	Valid

6.2.15 Pengujian Validasi Melihat Detail Tugas

1. Kasus uji berhasil melihat detail tugas.

Tabel 6.43 Kasus uji berhasil melihat detail tugas

Nama Kasus Uji	Kasus uji berhasil melihat detail tugas.
Prosedur	<ol style="list-style-type: none"> 1. <i>Employee</i> berhasil login kedalam sistem. 2. <i>Employee</i> menekan menu <i>project</i> pada <i>sidebar</i>. 3. <i>Employee</i> menekan tombol dengan <i>icon</i> “Eye” untuk melihat detail proyek suatu proyek. 4. <i>Employee</i> menekan tombol dengan <i>icon</i> “Eye” untuk melihat detail tugas suatu tugas.
Hasil yang diharapkan	Menampilkan modal dialog berisi detail tugas.
Hasil	Menampilkan modal dialog berisi detail tugas.
Status	Valid

6.2.16 Pengujian Validasi Menambah Feedback Tugas

1. Kasus uji berhasil menambah feedback tugas.

Tabel 6.44 Kasus uji berhasil menambah feedback tugas

Nama Kasus Uji	Kasus uji berhasil menambah feedback tugas.
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan <i>icon</i> “Eye” untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol dengan <i>icon</i> “Eye” untuk melihat detail tugas suatu tugas. 5. Manajer proyek mengisi <i>field feedback</i>. 6. Manajer proyek menekan tombol “Tambah”.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Feedback berhasil dibuat! Notifikasi e-mail telah dikirim ke pegawai.</i> ”.
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Feedback berhasil dibuat! Notifikasi e-mail telah dikirim ke pegawai.</i> ”.
Status	Valid

2. Kasus uji gagal menambah feedback tugas.

Tabel 6.45 Kasus uji gagal menambah feedback tugas

Nama Kasus Uji	Kasus uji gagal menambah feedback tugas.
Prosedur	<ol style="list-style-type: none"> Manajer proyek berhasil login kedalam sistem. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. Manajer proyek menekan tombol dengan <i>icon "Eye"</i> untuk melihat detail proyek suatu proyek. Manajer proyek menekan tombol dengan <i>icon "Eye"</i> untuk melihat detail tugas suatu tugas. Manajer proyek tidak mengisi <i>field feedback</i>. Manajer proyek menekan tombol "Tambah".
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan " <i>Feedback gagal ditambahkan!</i> ".
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan " <i>Feedback gagal ditambahkan!</i> ".
Status	Valid

6.2.17 Pengujian Validasi Melihat Feedback Tugas

1. Kasus uji berhasil melihat feedback tugas.

Tabel 6.46 Kasus uji berhasil melihat feedback tugas

Nama Kasus Uji	Kasus uji berhasil melihat feedback tugas.
Prosedur	<ol style="list-style-type: none"> <i>Employee</i> berhasil login kedalam sistem. <i>Employee</i> menekan menu <i>project</i> pada <i>sidebar</i>. <i>Employee</i> menekan tombol dengan <i>icon "Eye"</i> untuk melihat detail proyek suatu proyek. <i>Employee</i> menekan tombol dengan <i>icon "Eye"</i> untuk melihat detail tugas suatu tugas beserta <i>feedback</i> yang diterimanya.
Hasil yang diharapkan	Menampilkan detail tugas dan <i>feedback</i> tugas tersebut.
Hasil	Menampilkan detail tugas dan <i>feedback</i> tugas tersebut.
Status	Valid

6.2.18 Pengujian Validasi Menghapus Feedback Tugas

1. Kasus uji berhasil menghapus *feedback* tugas.

Tabel 6.47 Kasus uji berhasil menghapus feedback tugas

Nama Kasus Uji	Kasus uji berhasil menghapus feedback tugas.
Prosedur	<ol style="list-style-type: none"> Manajer proyek berhasil login kedalam sistem. Manajer proyek menekan menu <i>project</i> pada

	<p><i>sidebar.</i></p> <ol style="list-style-type: none"> 3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail tugas suatu tugas beserta <i>feedback</i> tugas tersebut. 5. Manajer proyek menekan tombol dengan icon “Trash” untuk menghapus <i>feedback</i>.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Data feed telah dihapus!</i> ”.
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Data feed telah dihapus!</i> ”.
Status	Valid

6.2.19 Pengujian Validasi Menambah *Board*

1. Kasus uji berhasil menambah *board*.

Tabel 6.48 Kasus uji berhasil menambah *board*

Nama Kasus Uji	Kasus uji berhasil menambah <i>board</i> .
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol “Kanban” untuk melihat halaman <i>Kanban</i> proyek. 5. Manajer proyek menekan tombol “+ Board” untuk menambahkan <i>board</i>. 6. Manajer proyek mengisi <i>field</i> nama. 7. Manajer proyek menekan tombol “Add”.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Board baru telah dibuat!</i> ”.
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “ <i>Board baru telah dibuat!</i> ”.
Status	Valid

2. Kasus uji gagal menambah *board*.

Tabel 6.49 Kasus uji gagal menambah *board*

Nama Kasus Uji	Kasus uji gagal menambah <i>board</i> .
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek.

	<ol style="list-style-type: none"> 4. Manajer proyek menekan tombol “Kanban” untuk melihat halaman <i>Kanban</i> proyek. 5. Manajer proyek menekan tombol “+ Board” untuk menambahkan <i>board</i>. 6. Manajer proyek mengosongkan <i>field</i> nama. 7. Manajer proyek menekan tombol “Add”.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “Nama board harus diisi!”.
Hasil	Menampilkan halaman <i>Detail Project</i> dan menampilkan pesan “Nama board harus diisi!”.
Status	Valid

6.2.20 Pengujian Validasi Melihat *Kanban Board*

1. Kasus uji berhasil melihat *kanban board*.

Tabel 6.50 Kasus uji berhasil melihat *kanban board*

Nama Kasus Uji	Kasus uji berhasil melihat <i>kanban board</i> .
Prosedur	<ol style="list-style-type: none"> 1. Pengguna berhasil login kedalam sistem. 2. Pengguna menekan menu <i>project</i> pada <i>sidebar</i>. 3. Pengguna menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Pengguna menekan tombol “Kanban” untuk melihat halaman <i>Kanban</i> proyek.
Hasil yang diharapkan	Menampilkan halaman <i>kanban</i> proyek.
Hasil	Menampilkan halaman <i>kanban</i> proyek.
Status	Valid

6.2.21 Pengujian Validasi Mengubah Limit

1. Kasus uji berhasil mengubah limit.

Tabel 6.51 Kasus uji berhasil mengubah limit

Nama Kasus Uji	Kasus uji berhasil mengubah limit.
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol “Kanban” untuk melihat halaman <i>Kanban</i> proyek. 5. Manajer proyek mengubah limit yang tertera pada suatu <i>Kanban board</i>. 6. Manajer proyek menekan tombol dengan icon “Check”.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Proyek</i> dan pesan “Limit board telah diubah!”.

Hasil	Menampilkan halaman <i>Detail Proyek</i> dan pesan " <i>Limit board telah diubah!</i> ".
Status	Valid

6.2.22 Pengujian Validasi Menghapus Board

1. Kasus uji berhasil menghapus *board*.

Tabel 6.52 Kasus uji berhasil menghapus *board*

Nama Kasus Uji	Kasus uji berhasil menghapus <i>board</i> .
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan icon "Eye" untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol "<i>Kanban</i>" untuk melihat halaman <i>Kanban</i> proyek. 5. Manajer proyek menekan tombol "x" pada suatu <i>board</i>.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Proyek</i> dan pesan " <i>Board telah dihapus!</i> ".
Hasil	Menampilkan halaman <i>Detail Proyek</i> dan pesan " <i>Board telah dihapus!</i> ".
Status	Valid

6.2.23 Pengujian Validasi Melihat *Gantt Chart* Proyek

1. Kasus uji berhasil melihat *gantt chart* proyek.

Tabel 6.53 Kasus uji berhasil melihat *gantt chart* proyek

Nama Kasus Uji	Kasus uji berhasil melihat <i>gantt chart</i> proyek.
Prosedur	<ol style="list-style-type: none"> 1. Pengguna berhasil login kedalam sistem. 2. Pengguna menekan menu <i>project</i> pada <i>sidebar</i>. 3. Pengguna menekan tombol dengan icon "Eye" untuk melihat detail proyek suatu proyek. 4. Pengguna menekan tombol "<i>Grafik Proyek</i>" untuk melihat halaman grafik proyek.
Hasil yang diharapkan	Menampilkan halaman grafik proyek.
Hasil	Menampilkan halaman grafik proyek.
Status	Valid

6.2.24 Pengujian Validasi Melihat *Pie Chart* Proyek

1. Kasus uji berhasil melihat *pie chart* proyek.

Tabel 6.54 Kasus uji berhasil melihat *pie chart* proyek

Nama Kasus Uji	Kasus uji berhasil melihat <i>pie chart</i> proyek.
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>project</i> pada <i>sidebar</i>.3. Admin menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek.4. Admin menekan tombol “<i>Grafik Proyek</i>” untuk melihat halaman grafik proyek.
Hasil yang diharapkan	Menampilkan halaman grafik proyek.
Hasil	Menampilkan halaman grafik proyek.
Status	Valid

6.2.25 Pengujian Validasi Mengupload Dokumen

1. Kasus uji berhasil mengupload dokumen.

Tabel 6.55 Kasus uji berhasil mengupload dokumen

Nama Kasus Uji	Kasus uji berhasil mengupload dokumen.
Prosedur	<ol style="list-style-type: none">1. Manajer proyek berhasil login kedalam sistem.2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>.3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek.4. Manajer proyek menekan tombol “<i>Dokumen</i>” untuk melihat halaman dokumen proyek.5. Manajer proyek menekan <i>field file</i>.6. Manajer proyek memilih dokumen.7. Manajer proyek menekan tombol “+ Dokumen”
Hasil yang diharapkan	Menampilkan halaman <i>Detail Proyek</i> dan pesan “ <i>Berkas berhasil diupload!</i> ”.
Hasil	Menampilkan halaman <i>Detail Proyek</i> dan pesan “ <i>Berkas berhasil diupload!</i> ”.
Status	Valid

2. Kasus uji gagal mengupload dokumen dengan format berkas tidak sesuai.

Tabel 6.56 Kasus uji gagal mengupload dokumen dengan format berkas tidak sesuai

Nama Kasus Uji	Kasus uji gagal mengupload dokumen.
Prosedur	<ol style="list-style-type: none">1. Manajer proyek berhasil login kedalam sistem.2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>.

	<ol style="list-style-type: none"> 3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol “Dokumen” untuk melihat halaman dokumen proyek. 5. Manajer proyek menekan <i>field file</i>. 6. Manajer proyek memilih dokumen dengan format selain pdf. 7. Manajer proyek menekan tombol “+ Dokumen”
Hasil yang diharapkan	Menampilkan halaman <i>Detail Proyek</i> dan pesan “ <i>Berkas gagal diupload ! The filetype you are attempting to upload is not allowed.</i> ”.
Hasil	Menampilkan halaman <i>Detail Proyek</i> dan pesan “ <i>Berkas gagal diupload ! The filetype you are attempting to upload is not allowed.</i> ”.
Status	Valid

3. Kasus uji gagal mengupload dokumen dengan *field* berkas tidak diisi.

Tabel 6.57 Kasus uji gagal mengupload dokumen dengan *field* berkas tidak diisi

Nama Kasus Uji	Kasus uji gagal mengupload dokumen.
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol “Dokumen” untuk melihat halaman dokumen proyek. 5. Manajer proyek mengosongkan <i>field file</i>. 6. Manajer proyek menekan tombol “+ Dokumen”
Hasil yang diharapkan	Menampilkan halaman <i>Detail Proyek</i> dan pesan “ <i>Berkas gagal diupload ! You did not select a file to upload.</i> ”.
Hasil	Menampilkan halaman <i>Detail Proyek</i> dan pesan “ <i>Berkas gagal diupload ! You did not select a file to upload.</i> ”.
Status	Valid

4. Kasus uji gagal mengupload dokumen dengan ukuran berkas melebihi batas.

Tabel 6.58 Kasus uji gagal mengupload dokumen dengan ukuran berkas melebihi batas

Nama Kasus Uji	Kasus uji gagal mengupload dokumen.
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan icon

	<p>“Eye” untuk melihat detail proyek suatu proyek.</p> <ol style="list-style-type: none"> 4. Manajer proyek menekan tombol “Dokumen” untuk melihat halaman dokumen proyek. 5. Manajer proyek memilih berkas dengan ukuran melebihi batas. 6. Manajer proyek menekan tombol “+ Dokumen”
Hasil yang diharapkan	Menampilkan halaman <i>Detail Proyek</i> dan pesan <i>“Berkas gagal diupload ! The uploaded file exceeds the maximum allowed size in your PHP configuration file.”</i> .
Hasil	Menampilkan halaman <i>Detail Proyek</i> dan pesan <i>“Berkas gagal diupload ! The uploaded file exceeds the maximum allowed size in your PHP configuration file.”</i> .
Status	Valid

6.2.26 Pengujian Validasi Melihat Dokumen Proyek

1. Kasus uji berhasil melihat dokumen proyek.

Tabel 6.59 Kasus uji berhasil melihat dokumen proyek

Nama Kasus Uji	Kasus uji berhasil melihat dokumen proyek.
Prosedur	<ol style="list-style-type: none"> 1. Pengguna berhasil login kedalam sistem. 2. Pengguna menekan menu <i>project</i> pada <i>sidebar</i>. 3. Pengguna menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Pengguna menekan tombol “Dokumen” untuk melihat halaman dokumen proyek.
Hasil yang diharapkan	Menampilkan halaman dokumen proyek.
Hasil	Menampilkan halaman dokumen proyek.
Status	Valid

6.2.27 Pengujian Validasi Membaca Dokumen Proyek

1. Kasus uji berhasil membaca dokumen proyek.

Tabel 6.60 Kasus uji berhasil membaca dokumen proyek

Nama Kasus Uji	Kasus uji berhasil membaca dokumen proyek.
Prosedur	<ol style="list-style-type: none"> 1. Pengguna berhasil login kedalam sistem. 2. Pengguna menekan menu <i>project</i> pada <i>sidebar</i>. 3. Pengguna menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Pengguna menekan tombol “Dokumen” untuk melihat halaman dokumen proyek. 5. Pengguna menekan tombol dengan icon “Eye” untuk melakukan <i>preview</i> pada dokumen.
Hasil yang diharapkan	Menampilkan modal dialog berisi dokumen yang dipilih.

Hasil	Menampilkan modal dialog berisi dokumen yang dipilih.
Status	Valid

6.2.28 Pengujian Validasi Menghapus Dokumen

1. Kasus uji berhasil menghapus dokumen.

Tabel 6.61 Kasus uji berhasil menghapus dokumen

Nama Kasus Uji	Kasus uji berhasil menghapus dokumen.
Prosedur	<ol style="list-style-type: none"> 1. Manajer proyek berhasil login kedalam sistem. 2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>. 3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Manajer proyek menekan tombol “<i>Dokumen</i>” untuk melihat halaman dokumen proyek. 5. Manajer proyek menekan tombol “x” pada salah satu dokumen.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Proyek</i> dan pesan “ <i>Berkas telah dihapus!</i> ”.
Hasil	Menampilkan halaman <i>Detail Proyek</i> dan pesan “ <i>Berkas telah dihapus!</i> ”.
Status	Valid

6.2.29 Pengujian Validasi *Download* Dokumen

1. Kasus uji berhasil mengunduh dokumen.

Tabel 6.62 Kasus uji berhasil download dokumen

Nama Kasus Uji	Kasus uji berhasil mengunduh dokumen.
Prosedur	<ol style="list-style-type: none"> 1. Pengguna berhasil login kedalam sistem. 2. Pengguna menekan menu <i>project</i> pada <i>sidebar</i>. 3. Pengguna menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek. 4. Pengguna menekan tombol “<i>Dokumen</i>” untuk melihat halaman dokumen proyek. 5. Pengguna menekan tombol dengan icon “Eye” untuk melakukan <i>preview</i> pada dokumen. 6. Pengguna menekan icon “<i>Download</i>” pada <i>preview</i> dokumen.
Hasil yang diharapkan	Dokumen berhasil diunduh.
Hasil	Dokumen berhasil diunduh.
Status	Valid

6.2.30 Pengujian Validasi Melihat Tim Proyek

1. Kasus uji berhasil melihat tim proyek.

Tabel 6.63 Kasus uji berhasil melihat tim proyek

Nama Kasus Uji	Kasus uji berhasil melihat tim proyek.
Prosedur	<ol style="list-style-type: none">1. Manajer proyek berhasil login kedalam sistem.2. Manajer proyek menekan menu <i>project</i> pada <i>sidebar</i>.3. Manajer proyek menekan tombol dengan icon “Eye” untuk melihat detail proyek suatu proyek.4. Manajer proyek menekan tombol “<i>Tim Proyek</i>” untuk melihat halaman tim proyek.
Hasil yang diharapkan	Menampilkan halaman tim proyek.
Hasil	Menampilkan halaman tim proyek.
Status	Valid

6.2.31 Pengujian Validasi Menambah Data *Client*

1. Kasus uji berhasil menambah data *client*.

Tabel 6.64 Kasus uji berhasil menambah data *client*

Nama Kasus Uji	Kasus uji berhasil menambah data <i>client</i> .
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>client</i> pada <i>sidebar</i>.3. Admin menekan tombol “+ <i>Client</i>” untuk menambah <i>client</i> baru.4. Admin mengisi seluruh <i>field</i> dengan benar.5. Admin menekan tombol “Tambah”.
Hasil yang diharapkan	Menampilkan halaman <i>Client</i> dan menampilkan pesan “ <i>Client</i> baru telah ditambahkan! Silakan tambahkan PIC pada detail client.”.
Hasil	Menampilkan halaman <i>Client</i> dan menampilkan pesan “ <i>Client</i> baru telah ditambahkan! Silakan tambahkan PIC pada detail client.”.
Status	Valid

2. Kasus uji gagal menambah data *client*.

Tabel 6.65 Kasus uji gagal menambah data *client*

Nama Kasus Uji	Kasus uji gagal menambah data <i>client</i> .
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>client</i> pada <i>sidebar</i>.3. Admin menekan tombol “+ <i>Client</i>” untuk menambah <i>client</i> baru.

	4. Admin tidak mengisi <i>field</i> dengan benar dan membiarkan terdapat <i>field</i> kosong. 5. Admin menekan tombol "Tambah".
Hasil yang diharapkan	Menampilkan pesan " <i>The 'nama-field' field is required</i> " pada kolom yang masih kosong.
Hasil	Menampilkan pesan " <i>The 'nama-field' field is required</i> " pada kolom yang masih kosong.
Status	Valid

6.2.32 Pengujian Validasi Melihat Client

1. Kasus uji berhasil melihat *client*.

Tabel 6.66 Kasus uji berhasil melihat client

Nama Kasus Uji	Kasus uji berhasil melihat <i>client</i> .
Prosedur	1. Pengguna berhasil login kedalam sistem. 2. Pengguna menekan menu <i>client</i> pada <i>sidebar</i> .
Hasil yang diharapkan	Menampilkan halaman daftar <i>client</i> .
Hasil	Menampilkan halaman daftar <i>client</i> .
Status	Valid

6.2.33 Pengujian Validasi Mengubah Data Client

1. Kasus uji berhasil mengubah data *client*.

Tabel 6.67 Kasus uji berhasil mengubah data client

Nama Kasus Uji	Kasus uji berhasil mengubah data <i>client</i> .
Prosedur	1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada <i>sidebar</i> . 3. Admin menekan tombol dengan icon "Pencil" untuk mengubah data <i>client</i> . 4. Admin mengubah data <i>client</i> . 5. Admin menekan tombol "Update".
Hasil yang diharapkan	Menampilkan halaman <i>client</i> .
Hasil	Menampilkan halaman <i>client</i> .
Status	Valid

2. Kasus uji gagal mengubah data *client*.

Tabel 6.68 Kasus uji gagal mengubah data client

Nama Kasus Uji	Kasus uji gagal mengubah data <i>client</i> .
Prosedur	1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada <i>sidebar</i> . 3. Admin menekan tombol dengan icon "Pencil" untuk mengubah data <i>client</i> . 4. Admin mengosongkan salah satu <i>field</i> . 5. Admin menekan tombol "Update".

Hasil yang diharapkan	Menampilkan pesan “The ‘nama-field’ field is required” pada kolom yang masih kosong.
Hasil	Menampilkan pesan “The ‘nama-field’ field is required” pada kolom yang masih kosong.
Status	Valid

6.2.34 Pengujian Validasi Menghapus Data Client

1. Kasus uji berhasil menghapus data *client*.

Tabel 6.69 Kasus uji berhasil menghapus data client

Nama Kasus Uji	Kasus uji berhasil menghapus data <i>client</i> .
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada sidebar. 3. Admin menekan tombol dengan icon “Trash” untuk menghapus data <i>client</i>.
Hasil yang diharapkan	Menampilkan halaman <i>client</i> dan pesan “Client telah dihapus!”.
Hasil	Menampilkan halaman <i>client</i> dan pesan “Client telah dihapus!”.
Status	Valid

6.2.35 Pengujian Validasi Menambah Data PIC

1. Kasus uji berhasil menambah data *PIC*.

Tabel 6.70 Kasus uji berhasil menambah data PIC

Nama Kasus Uji	Kasus uji berhasil menambah data <i>PIC</i> .
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada sidebar. 3. Admin menekan tombol dengan icon “Eye” untuk melihat halaman detail <i>client</i>. 4. Admin mengisi seluruh <i>field</i> pada form tambah <i>PIC</i> dengan benar. 5. Admin menekan tombol “Tambah”.
Hasil yang diharapkan	Menampilkan halaman <i>Detail Client</i> dan menampilkan pesan “PIC berhasil ditambahkan!”.
Hasil	Menampilkan halaman <i>Detail Client</i> dan menampilkan pesan “PIC berhasil ditambahkan!”.
Status	Valid

2. Kasus uji gagal menambah data *PIC*.

Tabel 6.71 Kasus uji gagal menambah data client

Nama Kasus Uji	Kasus uji gagal menambah data <i>client</i> .
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada sidebar.

	<ol style="list-style-type: none"> 3. Admin menekan tombol dengan icon “Eye” untuk melihat halaman detail <i>client</i>. 4. Admin tidak mengisi seluruh <i>field</i> pada form tambah PIC dengan benar. 5. Admin menekan tombol “Tambah”.
Hasil yang diharapkan	Menampilkan pesan “PIC gagal ditambahkan!”.
Hasil	Menampilkan pesan “PIC gagal ditambahkan!”.
Status	Valid

6.2.36 Pengujian Validasi Melihat PIC

1. Kasus uji berhasil melihat PIC.

Tabel 6.72 Kasus uji berhasil melihat PIC

Nama Kasus Uji	Kasus uji berhasil melihat PIC.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada <i>sidebar</i>. 3. Admin menekan tombol dengan icon “Eye” untuk melihat halaman detail <i>client</i>.
Hasil yang diharapkan	Menampilkan halaman detail <i>client</i> berserta daftar PIC <i>client</i> tersebut.
Hasil	Menampilkan halaman detail <i>client</i> berserta daftar PIC <i>client</i> tersebut.
Status	Valid

6.2.37 Pengujian Validasi Mengubah Data PIC

1. Kasus uji berhasil mengubah data PIC.

Tabel 6.73 Kasus uji berhasil mengubah data PIC

Nama Kasus Uji	Kasus uji berhasil mengubah data PIC.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada <i>sidebar</i>. 3. Admin menekan tombol dengan icon “Eye” untuk melihat halaman detail <i>client</i>. 4. Admin menekan tombol dengan icon “Pencil” untuk mengubah data PIC. 5. Admin mengubah data PIC. 6. Admin menekan tombol “Update”.
Hasil yang diharapkan	Menampilkan halaman detail <i>client</i> dan pesan “Data PIC berhasil diupdate!”.
Hasil	Menampilkan halaman detail <i>client</i> dan pesan “Data PIC berhasil diupdate!”.
Status	Valid

2. Kasus uji gagal mengubah data PIC.

Tabel 6.74 Kasus uji gagal mengubah data PIC

Nama Kasus Uji	Kasus uji gagal mengubah data PIC.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada <i>sidebar</i>. 3. Admin menekan tombol dengan icon “Eye” untuk melihat halaman detail <i>client</i>. 4. Admin menekan tombol dengan icon “Pencil” untuk mengubah data PIC. 5. Admin tidak mengisi salah satu <i>field</i> update PIC. 6. Admin menekan tombol “Update”.
Hasil yang diharapkan	Menampilkan halaman detail <i>client</i> dan pesan “Data PIC gagal diupdate!”.
Hasil	Menampilkan halaman detail <i>client</i> dan pesan “Data PIC gagal diupdate!”.
Status	Valid

6.2.38 Pengujian Validasi Menghapus Data PIC

1. Kasus uji berhasil menghapus data PIC.

Tabel 6.75 Kasus uji berhasil menghapus data PIC

Nama Kasus Uji	Kasus uji berhasil menghapus data PIC.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>client</i> pada <i>sidebar</i>. 3. Admin menekan tombol dengan icon “Eye” untuk melihat halaman detail <i>client</i>. 4. Admin menekan tombol dengan icon “Trash” untuk menghapus data PIC.
Hasil yang diharapkan	Menampilkan halaman detail <i>client</i> dan pesan “PIC berhasil dihapus!”.
Hasil	Menampilkan halaman detail <i>client</i> dan pesan “PIC berhasil dihapus!”.
Status	Valid

6.2.39 Pengujian Validasi Menambah Data Pegawai

1. Kasus uji berhasil menambah data pegawai.

Tabel 6.76 Kasus uji berhasil menambah data pegawai

Nama Kasus Uji	Kasus uji berhasil menambah data pegawai.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>employee</i> pada <i>sidebar</i>. 3. Admin menekan tombol “+ Akun” untuk menambah <i>employee</i> baru.

	4. Admin mengisi seluruh <i>field</i> dengan benar. 5. Admin menekan tombol "Tambah".
Hasil yang diharapkan	Menampilkan halaman <i>Employee</i> dan menampilkan pesan " <i>Akun employee berhasil dibuat. Silakan minta employee cek email / spam untuk aktivasi akun.</i> ".
Hasil	Menampilkan halaman <i>Employee</i> dan menampilkan pesan " <i>Akun employee berhasil dibuat. Silakan minta employee cek email / spam untuk aktivasi akun.</i> ".
Status	Valid

2. Kasus uji gagal menambah data pegawai.

Tabel 6.77 Kasus uji gagal menambah data pegawai

Nama Kasus Uji	Kasus uji gagal menambah data <i>client</i> .
Prosedur	1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>employee</i> pada <i>sidebar</i> . 3. Admin menekan tombol "+ Akun" untuk menambah <i>employee</i> baru. 4. Admin tidak mengisi seluruh <i>field</i> dengan benar dan membiarkan terdapat <i>field</i> kosong. 5. Admin menekan tombol "Tambah".
Hasil yang diharapkan	Menampilkan pesan " <i>The 'nama-field' field is required</i> " pada kolom yang masih kosong.
Hasil	Menampilkan pesan " <i>The 'nama-field' field is required</i> " pada kolom yang masih kosong.
Status	Valid

6.2.40 Pengujian Validasi Melihat Data Pegawai

1. Kasus uji berhasil melihat *data pegawai*.

Tabel 6.78 Kasus uji berhasil melihat data pegawai

Nama Kasus Uji	Kasus uji berhasil melihat data pegawai.
Prosedur	1. Pengguna berhasil login kedalam sistem. 2. Pengguna menekan menu <i>employee</i> pada <i>sidebar</i> .
Hasil yang diharapkan	Menampilkan halaman daftar <i>employee</i> .
Hasil	Menampilkan halaman daftar <i>employee</i> .
Status	Valid

6.2.41 Pengujian Validasi Mengubah Data Pegawai

1. Kasus uji berhasil mengubah data pegawai.

Tabel 6.79 Kasus uji berhasil mengubah data pegawai

Nama Kasus Uji	Kasus uji berhasil mengubah data pegawai.
-----------------------	---

Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>employee</i> pada sidebar. 3. Admin menekan tombol dengan icon “<i>Pencil</i>” untuk mengubah data <i>employee</i>. 4. Admin mengisi seluruh <i>field</i> dengan benar. 5. Admin menekan tombol “Update”.
Hasil yang diharapkan	Menampilkan halaman <i>Employee</i> dan menampilkan pesan “ <i>Akun employee berhasil diupdate!</i> ”.
Hasil	Menampilkan halaman <i>Employee</i> dan menampilkan pesan “ <i>Akun employee berhasil diupdate!</i> ”.
Status	Valid

2. Kasus uji gagal mengubah data pegawai.

Tabel 6.80 Kasus uji gagal mengubah data pegawai

Nama Kasus Uji	Kasus uji gagal mengubah data pegawai.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>employee</i> pada sidebar. 3. Admin menekan tombol dengan icon “<i>Pencil</i>” untuk mengubah data <i>employee</i>. 4. Admin tidak mengisi seluruh <i>field</i> dengan benar dan membiarkan terdapat <i>field</i> kosong. 5. Admin menekan tombol “Update”.
Hasil yang diharapkan	Menampilkan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.
Hasil	Menampilkan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.
Status	Valid

6.2.42 Pengujian Validasi Menghapus Data Pegawai

1. Kasus uji berhasil menghapus data pegawai.

Tabel 6.81 Kasus uji berhasil menghapus data pegawai

Nama Kasus Uji	Kasus uji berhasil menghapus data pegawai.
Prosedur	<ol style="list-style-type: none"> 1. Admin berhasil login kedalam sistem. 2. Admin menekan menu <i>employee</i> pada sidebar. 3. Admin menekan tombol dengan icon “<i>Trash</i>” untuk menghapus data <i>employee</i>.
Hasil yang diharapkan	Menampilkan halaman <i>Employee</i> dan pesan “ <i>Akun employee berhasil dihapus!</i> ”.
Hasil	Menampilkan halaman <i>Employee</i> dan pesan “ <i>Akun employee berhasil dihapus!</i> ”.
Status	Valid

6.2.43 Pengujian Validasi Melihat Detail Data Pegawai

1. Kasus uji berhasil melihat detail data pegawai.

Tabel 6.82 Kasus uji berhasil melihat detail data pegawai

Nama Kasus Uji	Kasus uji berhasil melihat detail data pegawai.
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>employee</i> pada sidebar.3. Admin menekan tombol dengan icon “Eye” untuk melihat detail data <i>employee</i>.
Hasil yang diharapkan	Menampilkan halaman <i>detail employee</i> .
Hasil	Menampilkan halaman <i>detail employee</i> .
Status	Valid

6.2.44 Pengujian Validasi Menambah Data Role Pegawai

1. Kasus uji berhasil menambah data role pegawai.

Tabel 6.83 Kasus uji berhasil menambah data role pegawai

Nama Kasus Uji	Kasus uji berhasil menambah data role pegawai.
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>employee</i> pada sidebar.3. Admin menekan tombol “+ Role” untuk menambah <i>role employee</i>.4. Admin mengisi <i>field</i> nama <i>role</i>.5. Admin menekan tombol “Add”.
Hasil yang diharapkan	Menampilkan halaman <i>Employee</i> dan menampilkan pesan “ <i>Role baru telah dibuat!</i> ”.
Hasil	Menampilkan halaman <i>Employee</i> dan menampilkan pesan “ <i>Role baru telah dibuat!</i> ”.
Status	Valid

2. Kasus uji gagal menambah data pegawai.

Tabel 6.84 Kasus uji gagal menambah data pegawai

Nama Kasus Uji	Kasus uji gagal menambah data <i>client</i> .
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>employee</i> pada sidebar.3. Admin menekan tombol “+ Role” untuk menambah <i>role employee</i>.4. Admin tidak mengisi <i>field</i> nama <i>role</i>.5. Admin menekan tombol “Add”.
Hasil yang diharapkan	Menampilkan halaman <i>Employee</i> dan menampilkan pesan “ <i>Nama role harus diisi!</i> ”.
Hasil	Menampilkan halaman <i>Employee</i> dan menampilkan pesan “ <i>Nama role harus diisi!</i> ”.
Status	Valid

6.2.45 Pengujian Validasi Melihat Data Role Pegawai

1. Kasus uji berhasil melihat data role pegawai.

Tabel 6.85 Kasus uji berhasil melihat data role pegawai

Nama Kasus Uji	Kasus uji berhasil melihat data role pegawai.
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>employee</i> pada <i>sidebar</i>.3. Admin menekan tombol “+ Role” untuk melihat daftar <i>role employee</i>.
Hasil yang diharapkan	Menampilkan modal dialog berisi <i>role</i> yang saat ini ada di perusahaan.
Hasil	Menampilkan modal dialog berisi <i>role</i> yang saat ini ada di perusahaan.
Status	Valid

6.2.46 Pengujian Validasi Menghapus Data Role Pegawai

1. Kasus uji berhasil menghapus data role pegawai.

Tabel 6.86 Kasus uji berhasil menghapus data role pegawai

Nama Kasus Uji	Kasus uji berhasil menghapus data role pegawai.
Prosedur	<ol style="list-style-type: none">1. Admin berhasil login kedalam sistem.2. Admin menekan menu <i>employee</i> pada <i>sidebar</i>.3. Admin menekan tombol “+ Role” untuk melihat daftar <i>role employee</i>.4. Admin menekan tombol dengan icon “Trash” untuk menghapus <i>role</i>.
Hasil yang diharapkan	Menampilkan halaman <i>Employee</i> dan menampilkan pesan “Role telah dihapus!”.
Hasil	Menampilkan halaman <i>Employee</i> dan menampilkan pesan “Role telah dihapus!”.
Status	Valid

6.2.47 Pengujian Validasi Melihat Profil

1. Kasus uji berhasil melihat profil.

Tabel 6.87 Kasus uji berhasil melihat profil

Nama Kasus Uji	Kasus uji berhasil melihat profil.
Prosedur	<ol style="list-style-type: none">1. Pengguna berhasil login kedalam sistem.2. Pengguna menekan menu <i>My Profile</i> pada <i>sidebar</i>.
Hasil yang diharapkan	Menampilkan halaman <i>My Profile</i> .
Hasil	Menampilkan halaman <i>My Profile</i> .
Status	Valid

6.2.48 Pengujian Validasi Mengubah Data Profil

1. Kasus uji berhasil mengubah data profil.

Tabel 6.88 Kasus uji berhasil mengubah data profil

Nama Kasus Uji	Kasus uji berhasil mengubah data profil.
Prosedur	<ol style="list-style-type: none">Pengguna berhasil login kedalam sistem.Pengguna menekan menu <i>Edit Profile</i> pada sidebar.Pengguna mengisi seluruh <i>field</i> dengan benar.Pengguna menekan tombol "Simpan".
Hasil yang diharapkan	Menampilkan halaman <i>My Profile</i> dan menampilkan pesan " <i>Profile updated!</i> ".
Hasil	Menampilkan halaman <i>My Profile</i> dan menampilkan pesan " <i>Profile updated!</i> ".
Status	Valid

2. Kasus uji gagal mengubah data profil.

Tabel 6.89 Kasus uji gagal mengubah data profil

Nama Kasus Uji	Kasus uji gagal mengubah data profil.
Prosedur	<ol style="list-style-type: none">Pengguna berhasil login kedalam sistem.Pengguna menekan menu <i>Edit Profile</i> pada sidebar.Pengguna tidak mengisi seluruh <i>field</i> dengan benar dengan membiarkan terdapat <i>field</i> kosong.Pengguna menekan tombol "Simpan".
Hasil yang diharapkan	Menampilkan pesan " <i>The 'nama-field' field is required</i> " pada kolom yang masih kosong.
Hasil	Menampilkan pesan " <i>The 'nama-field' field is required</i> " pada kolom yang masih kosong.
Status	Valid

6.2.49 Pengujian Validasi Mengubah Password

1. Kasus uji berhasil mengubah password.

Tabel 6.90 Kasus uji berhasil mengubah password

Nama Kasus Uji	Kasus uji berhasil melakukan <i>logout</i> .
Prosedur	<ol style="list-style-type: none">Pengguna berhasil login kedalam sistem.Pengguna menekan menu <i>Change Password</i> pada sidebar.Pengguna mengisi seluruh <i>field</i> dengan benar.Pengguna menekan tombol "<i>Change Password</i>".
Hasil yang diharapkan	Menampilkan halaman <i>My Profile</i> dan menampilkan pesan " <i>Password changed!</i> ".

Hasil	Menampilkan halaman <i>My Profile</i> dan menampilkan pesan “ <i>Password changed!</i> ”.
Status	Valid

2. Kasus uji gagal mengubah password dengan terdapat *field* yang masih kosong.

Tabel 6.91 Kasus uji gagal mengubah password

Nama Kasus Uji	Kasus uji gagal mengubah password.
Prosedur	<ol style="list-style-type: none"> Pengguna berhasil login kedalam sistem. Pengguna menekan menu <i>Change Password</i> pada <i>sidebar</i>. Pengguna tidak mengisi seluruh <i>field</i> dengan benar dengan membiarkan terdapat <i>field</i> kosong. Pengguna menekan tombol “<i>Change Password</i>”.
Hasil yang diharapkan	Menampilkan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.
Hasil	Menampilkan pesan “ <i>The ‘nama-field’ field is required</i> ” pada kolom yang masih kosong.
Status	Valid

3. Kasus uji gagal mengubah password dengan inputan password sekarang tidak sesuai.

Tabel 6.92 Kasus uji gagal mengubah password dengan inputan password saat ini tidak sesuai

Nama Kasus Uji	Kasus uji gagal mengubah password.
Prosedur	<ol style="list-style-type: none"> Pengguna berhasil login kedalam sistem. Pengguna menekan menu <i>Change Password</i> pada <i>sidebar</i>. Pengguna mengisi <i>field</i> “<i>Current Password</i>” dengan data yang tidak sesuai. Pengguna menekan tombol “<i>Change Password</i>”.
Hasil yang diharapkan	Menampilkan pesan “ <i>Wrong current password!</i> ”.
Hasil	Menampilkan pesan “ <i>Wrong current password!</i> ”.
Status	Valid

4. Kasus uji gagal mengubah password dengan inputan password baru tidak sesuai dengan inputan *field repeat password* dan sebaliknya.

Tabel 6.93 Kasus uji gagal mengubah password dengan inputan password baru tidak sesuai dengan inputan *field repeat password* dan sebaliknya

Nama Kasus Uji	Kasus uji gagal mengubah password.
Prosedur	<ol style="list-style-type: none"> Pengguna berhasil login kedalam sistem. Pengguna menekan menu <i>Change Password</i> pada <i>sidebar</i>.

	3. Pengguna mengisi field "New Password" berbeda dengan field "Repeat Password". 4. Pengguna menekan tombol "Change Password".
Hasil yang diharapkan	Menampilkan pesan " <i>The New Password field does not match the Repeat Password field.</i> " pada kolom <i>New password</i> dan pesan " <i>The Repeat Password field does not match the New Password field.</i> " pada kolom <i>Repat password</i> .
Hasil	Menampilkan pesan " <i>The New Password field does not match the Repeat Password field.</i> " pada kolom <i>New password</i> dan pesan " <i>The Repeat Password field does not match the New Password field.</i> " pada kolom <i>Repat password</i> .
Status	Valid

5. Kasus uji gagal mengubah password dengan inputan password baru kurang dari enam karakter.

Tabel 6.94 Kasus uji gagal mengubah password dengan inputan password baru kurang dari enam karakter.

Nama Kasus Uji	Kasus uji gagal mengubah password.
Prosedur	5. Pengguna berhasil login kedalam sistem. 6. Pengguna menekan menu <i>Change Password</i> pada sidebar. 7. Pengguna mengisi field "New Password" dengan kurang dari enam karakter. 8. Pengguna menekan tombol "Change Password".
Hasil yang diharapkan	Menampilkan pesan " <i>The New Password field must be at least 6 character in length</i> " pada kolom <i>New password</i> .
Hasil	Menampilkan pesan " <i>The New Password field must be at least 6 character in length</i> " pada kolom <i>New password</i> .
Status	Valid

6.3 Usability Testing

Pada tahap ini akan dilakukan pengujian *usability* dengan melibatkan beberapa responden yang dipilih secara acak dan akan mencoba menggunakan sistem ini. Pengujian *usability* terhadap sistem akan menggunakan metode *System Usability Scale (SUS)*.

6.3.1 Prosedur Pengujian Usability

Pengujian *usability* menggunakan metode SUS dilakukan dengan memberikan beberapa pertanyaan kepada sejumlah responden yang diberikan kesempatan untuk mencoba mengoperasikan sistem. Nielsen (2020) menyatakan

bahwa setidaknya terdapat 5 orang yang dijadikan responden agar pengujian ini dapat dikatakan valid. Dalam pengujian ini, responden yang berpartisipasi adalah orang yang akan menjadi calon pengguna dan dipilih secara acak. Responden dipersilakan untuk menjawab pertanyaan-pertanyaan yang diberikan setelah selesai menggunakan sistem. Daftar pertanyaan menggunakan metode SUS dapat dilihat pada Tabel 6.95.

Tabel 6.95 Daftar Pertanyaan Pengujian *Usability* menggunakan Metode SUS

No.	Pertanyaan
1.	Saya merasa akan menggunakan sistem ini secara berkelanjutan
2.	Saya merasa penggunaan sistem ini cukup rumit
3.	Saya merasa sistem ini mudah untuk digunakan
4.	Saya merasa membutuhkan bantuan teknis dari orang yang ahli untuk menggunakan sistem ini
5.	Saya menemukan beragam fitur yang terdapat dalam sistem terintegrasi dengan baik
6.	Saya merasa banyak inkonsistensi yang terdapat dalam sistem ini
7.	Saya merasa orang lain akan mempelajari cara penggunaan sistem ini dengan mudah dan cepat
8.	Saya berpikir bahwa sistem ini tidak praktis untuk digunakan
9.	Saya merasa sangat percaya diri dalam menggunakan sistem ini
10.	Saya merasa harus belajar lebih banyak untuk menggunakan sistem ini

6.3.2 Analisis dan Hasil Pengujian *Usability*

Setelah prosedur pengujian usability selesai dilakukan dan didapatkan nilai respon yang diberikan oleh responden, proses selanjutnya adalah menganalisa hasil pengujian tersebut. Hasil pengujian berdasarkan pertanyaan yang diberikan kepada responden dapat dilihat pada Tabel 6.96.

Tabel 6.96 Hasil Kuisioner Responden terhadap Pertanyaan Metode SUS

No	Nama	Pertanyaan									
		1	2	3	4	5	6	7	8	9	10
1	Bobby Ardian Ekaputra	4	3	4	2	4	4	4	2	4	2
2	Yudha Pratama	5	3	5	2	5	1	5	1	5	2
3	GUNAWAN FAHMI	5	3	3	2	4	3	3	2	5	4
4	Anjumi Kholifatu Rahmatika	4	2	5	2	5	2	5	1	4	2
5	Nadir Basalamah	4	2	3	2	4	2	5	2	3	2

Setelah kuisioner diisi oleh responden, kemudian dilakukan perhitungan skor akhir yang didasarkan penilaian yang diberikan responden dalam kuisioner tersebut. Perhitungan jumlah skor akhir mengambil nilai yang terdapat pada Tabel 6.96 dengan ketentuan nilai dari pertanyaan ganjil akan dikurangi 1 (nilai) dan 5 (lima) nilai akan dikurangi nilai dari pertanyaan genap. Hasil perhitungan skor akhir dapat dilihat pada Tabel 6.97.

Tabel 6.97 Hasil Perhitungan Skor Akhir Penilaian Kuisioner

No	Nama	Pertanyaan										Jumlah Skor	Dikali 2.5
		1	2	3	4	5	6	7	8	9	10		
1	Bobby Ardian Ekaputra	3	2	3	3	3	1	3	3	3	3	27	67,5
2	Yudha Pratama	4	2	4	3	4	4	4	4	4	3	36	90
3	GUNAWAN FAHMI	4	2	2	3	3	2	2	3	4	1	26	65
4	Anjumi Kholifatu Rahmatika	3	3	4	3	4	3	4	4	3	3	34	85
5	Nadir Basalamah	3	3	2	3	3	3	4	3	2	3	29	72,5
Rata-rata													76

Berdasarkan hasil skor akhir yang didapatkan pada Tabel 6.97, skor akhir yang diberikan dari kelima responden dalam pengujian usability sistem adalah 76. Dengan rata-rata skor tersebut dapat disimpulkan bahwa sistem masuk ke dalam kategori *acceptable* yang berarti sistem yang diimplementasikan telah sesuai dengan kebutuhan pengguna.

BAB 7 PENUTUP

7.1 Kesimpulan

Analisis kebutuhan yang dilakukan menghasilkan 4 aktor sebagai pengguna sistem, 9 entitas sebagai dasar perancangan data sistem, 49 kebutuhan fungsional, dan 1 kebutuhan non-fungsional. Perancangan yang dilakukan menghasilkan pemodelan 3 alur fitur utama sistem (tambah proyek, tambah *board*, dan tambah tugas) yang dituangkan dalam *sequence diagram*, pemodelan kelas yang berinteraksi didalam sistem dengan *class diagram*, rancangan algoritma fitur utama sistem yang dituangkan dalam bentuk *pseudocode*, rancangan data sistem yang dituangkan dalam wujud *Physical Data Model* (PDM), dan rancangan antarmuka dari 3 fitur utama sistem. Implementasi sistem menghasilkan 3 jenis implementasi diantaranya adalah implementasi data yang menggunakan bahasa pemrograman Sql dan DBMS MySql, implementasi kode program menggunakan bahasa pemrograman PHP dengan menggunakan *framework* CodeIgniter, dan implementasi antarmuka yang menggunakan HTML, Javascript, dan *framework* Bootstrap.

7.2 Saran

Saran yang diberikan untuk pengembangan lanjut sistem ini adalah dengan memberikan fitur notifikasi yang disesuaikan dengan role pengguna, memperjelas dependency atau hubungan antar tugas dalam proyek, dan pengembangan sistem menggunakan framework yang lebih mutakhir dan handal.

DAFTAR REFERENSI

- Ahmad, Muhammad Ovais. 2016. *Exploring Kanban in Software Engineering*. Oulu: University of Oulu.
- Ahmad, Muhammad Ovais & Dennehy, Denis & Conboy, Kieran & Oivo, Markku. 2017. *Kanban in Software Engineering: A Systematic Mapping Study*. *Journal of Systems and Software*. 137. 96-113.
- Alqudah, Mashal & Razali, Rozilawati. 2017. *A Comparison of Scrum and Kanban for Identifying Their Selection Factors*. 1-6. 10.1109/ICEEI.2017.8312434.
- Apriyanto, R. D. & P. Putro ,Hanson., 2018. *Tingkat Kegagalan dan Keberhasilan Proyek Sistem Informasi Di Indonesia*. Yogyakarta: SENTIKA.
- Bellah, Jeremy C. & Chen, Liang & Zimmer, J. Christopher. 2018. *Development of a Project Management Software Tool: A Design Case*. International Journal of Design for Learning. 09. 158-170.
- Booch, Grady & Rumbaugh, James & Jacobson, Ivar. 1999. *Unified Modeling Language User Guide, The (2nd Edition) (Addison-Wesley Object Technology Series)*. J. Database Manag.. 10.
- Bootstrap. 2019. *SB-Admin-2 Description*. Diakses 2 November 2019, dari <https://startbootstrap.com/themes/sb-admin-2/>
- Bootstrap. 2019. *User Guide – Introduction*. Diakses 25 Oktober 2019, dari <https://getbootstrap.com/docs/4.4/getting-started/introduction/>
- Brooke, J. 2011. *Measuring Usability With The System Usability Scale (SUS)*. [online] Tersedia di: <<http://www.measuringu.com/sus.php>> [Diakses pada 05 Juli 2020]
- Codeigniter. 2020. *CodeIgniter Documentation*. Diakses 20 Oktober 2019, dari <https://codeigniter.com/docs>
- Corona, Erika & Filippo Eros Pani. 2013. *A Review of Lean-Kanban Approaches in the Software Development*. Cagliari: DEE University of Cagliari.
- Fahat, M. Faisal. & P., Bayu & P., Fajar. 2018, *Pengembangan Aplikasi Manajemen Proyek Perangkat Lunak Berbasis Scrum Studi Kasus CV. Nusantara Media Mandiri (CVNMM)*, Malang: Universitas Brawijaya.
- Google Developers. 2019. *Using Google Charts*. Diakses 30 Oktober 2019, dari <https://developers.google.com/chart/interactive/docs>
- Hack Reactor. 2019. *What is JavaScript Used For?*. Diakses 2 November 2019, dari <https://www.hackreactor.com/blog/what-is-javascript-used-for>
- Hughes, B. & Cotterell, M., 1999. *Software Project Management*. 2nd ed. Berkshire: McGraw-Hill.
- Ikonen, Marko & Pirinen, Elena & Fagerholm, Fabian & Kettunen, Petri & Abrahamsson, Pekka. 2011. *On the Impact of Kanban on Software Project*

Work An Empirical Case Study Investigation. Proceedings - 2011 16th IEEE International Conference on Engineering of Complex Computer Systems, ICECCS 2011. 305-314.

- Kumar, M., Professor, A., Kumar Singh, S., Dwivedi, R. K., & Professor, A. 2015. A Comparative Study of Black Box Testing and White Box Testing Techniques. International Journal of Advance Research in Computer Science and Management Studies, 3(10), 32–44. [online] Tersedia di: <http://www.ijarcsms.com/docs/paper/volume3/issue10/V3I10-0018.pdf>
- Munir, 2015. *Manajemen Proyek Perangkat Lunak*. Bandung: UPI Press.
- M, Niranjanamurthy, Nagaraj, A., Gattu, H., & Shetty, P. K. (2014). Research Study on Importance of Usability Testing/ User Experience (UX) Testing. International Journal of Computer Science and Mobile Computing, 310(10), 78–85..
- Nielsen, J.. 2020. *Usability 101: Introduction to Usability* [online] Tersedia di: <<http://www.nngroup.com/articles/usability-101-introduction-to-usability/>> [Diakses 05 April 2020]
- Object Management Group. 2017. *Unified Modelling Language (OMG UML Version 2.5.1)*. [online] Tersedia di: <<https://www.omg.org/spec/UML/2.5.1/PDF/>> [Diakses 8 November 2019]
- Pressman, R., 2010. *Software Engineering A Practitioner's Approach*. 7th ed. New York: McGraw-Hill.
- Project Management Institute, 2013. *A Guide To The Project Management Body Of Knowledge*. 5th ed. Newton Square: Project Management.
- Raut, Laukik & Wakode, Rajat & Talmale, Pravin. 2015. *Overview on Kanban Methodology and its Implementation*. International Journal for Scientific Research & Development. 03. 2518-2521.
- Santosa, Budi. 2009. *Manajemen Proyek Konsep & Implementasi*. Edisi Pertama. Yogyakarta: GRAHA ILMU.
- Satzinger, J. W. & Jackson, R. B. & Burd, S. D.. 2010. *Systems Analysis and Design in a Changing World*. 5th ed. New York: Cengage Learning.
- Sommerville, I., 2011. *Software Engineering*. 9th ed. New York: Addison-Wesley.

LAMPIRAN

A. Kasus Perencanaan Jadwal dan Pembagian Tugas Proyek

Perencanaan Jadwal dan Pembagian Tugas Proyek

Nama Proyek : Pengembangan Management System Using Kanban
Client : CV. Karya Maju Bersama
Manajer Proyek : M F S
Jangka Waktu Proyek : 20 Mei 2020 – 20 Agustus 2020
Anggota Proyek :
1. B D H – Analis Sistem
2. N A A K – Perancangan Data
3. M R F – Perancangan UI
4. D B E N – Perancangan UI
5. M A I – Programmer
6. R N A F – Programmer
7. M R – Tester

No.	Tugas	Sub-tugas	Start Date	End Date	Penerima Tugas
1.	Melakukan rapat proyek dengan stakeholder terkait	1. Menjadwalkan pertemuan seluruh stakeholder 2. Memimpin rapat	20/05/2020	23/05/2020	M F S
2.	Melakukan rekayasa kebutuhan sistem	1. Melakukan survey kebutuhan sistem lebih lanjut dengan client 2. Mencatat masalah client terkait sasaran sistem dan spesifikasi kebutuhan sistem 3. Melaporkan hasil survey kepada direktur	23/05/2020	28/05/2020	B D H
3.	Menyusun dokumen kebutuhan sistem dan gambaran sistem	1. Membuat Business Solution Design 2. Membuat Technical Solution Design 3. Mengunggah dokumen ke sistem manajemen proyek	28/05/2020	30/05/2020	B D H
4.	Melakukan perancangan data	1. Membuat Entity Relationship Diagram (ERD) sesuai dokumen kebutuhan sistem 2. Membuat Mapping table sesuai dokumen kebutuhan sistem	30/05/2020	04/06/2020	N A A K
5.	Melakukan perancangan komponen sistem	1. Membuat rancangan class diagram sesuai dokumen kebutuhan sistem 2. Membuat sequence diagram sesuai dokumen kebutuhan sistem	30/05/2020	04/06/2020	N A A K
6.	Melakukan Perancangan UI	1. Membuat mockup halaman login	30/05/2020	04/06/2020	M R F

Lampiran 1 Kasus Penjadwalan

No.	Tugas	Sub-tugas	Start Date	End Date	Penerima Tugas
	Autentikasi sistem	2. Membuat mockup halaman registrasi			
7.	Melakukan Perancangan UI Dashboard sistem	1. Membuat mockup dashboard admin 2. Membuat mockup dashboard manajer proyek 3. Membuat mockup dashboard employee	04/06/2020	06/06/2020	M R F
8.	Melakukan Perancangan UI CRUD data Proyek	1. Membuat mockup halaman tambah proyek 2. Membuat mockup halaman lihat daftar proyek 3. Membuat mockup halaman update proyek	30/05/2020	04/06/2020	D B E N
9.	Melakukan Perancangan UI Kanban proyek	1. Membuat mockup halaman Kanban proyek 2. Membuat mockup form tambah dokumen 3. Membuat mockup form tambah Kanban board 4. Membuat mockup form tambah tugas proyek	04/06/2020	07/06/2020	D B E N
10.	Melakukan Perancangan UI CRUD data Client dan PIC	1. Membuat mockup halaman tambah client 2. Membuat mockup halaman lihat daftar client 3. Membuat mockup halaman update client 4. Membuat mockup form tambah dan edit PIC 5. Membuat mockup halaman lihat PIC	06/06/2020	09/06/2020	M R F
11.	Melakukan Perancangan UI Profil pengguna	1. Membuat mockup halaman profil pengguna 2. Membuat mockup halaman edit profil pengguna 3. Membuat mockup halaman ubah password akun pengguna	05/06/2020	14/06/2020	M R F
12.	Melakukan implementasi rancangan basis data	1. Melakukan instalasi basis data 2. Mengimplementasikan rancangan ERD menjadi basis data yang siap digunakan	04/06/2020	07/06/2020	M A I
13.	Melakukan implementasi	1. Mengimplementasikan mockup halaman login	04/06/2020	11/06/2020	M A I

Lampiran 2 Lampiran Kasus Penjadwalan (lanjutan)

No.	Tugas	Sub-tugas	Start Date	End Date	Penerima Tugas
	perancangan UI Autentikasi sistem	2. Mengimplementasikan mockup halaman registrasi			
14.	Melakukan implementasi rancangan UI Dashboard sistem	1. Mengimplementasikan mockup dashboard admin 2. Mengimplementasikan mockup dashboard support 3. Mengimplementasikan mockup dashboard manajer proyek 4. Mengimplementasikan mockup dashboard employee	06/06/2020	15/06/2020	R N A F
15.	Melakukan implementasi rancangan UI CRUD data Proyek	1. Mengimplementasikan mockup halaman tambah proyek 2. Mengimplementasikan mockup halaman lihat daftar proyek 3. Mengimplementasikan mockup halaman update proyek	05/06/2020	20/06/2020	M A I
16.	Melakukan implementasi rancangan UI Kanban proyek	1. Mengimplementasikan mockup halaman Kanban proyek 2. Mengimplementasikan mockup form tambah dokumen 3. Mengimplementasikan mockup form tambah Kanban board 4. Mengimplementasikan mockup form tambah tugas proyek	07/06/2020	14/07/2020	R N A F
17.	Melakukan implementasi rancangan UI CRUD data Client dan PIC	1. Mengimplementasikan mockup halaman tambah client 2. Mengimplementasikan mockup halaman lihat daftar client 3. Mengimplementasikan mockup halaman update client 4. Mengimplementasikan mockup form tambah dan edit PIC 5. Mengimplementasikan mockup halaman lihat PIC	05/06/2020	17/07/2020	M A I
18.	Melakukan implementasi	1. Mengimplementasikan mockup halaman profil	14/06/2020	21/07/2020	R N A F

Lampiran 3 Lampiran Kasus Penjadwalan (lanjutan)

No.	Tugas	Sub-tugas	Start Date	End Date	Penerima Tugas
	rancangan UI Profil pengguna	<ul style="list-style-type: none"> pengguna 2. Mengimplementasikan mockup halaman edit profil pengguna 3. Mengimplementasikan mockup halaman ubah password akun pengguna 			
19.	Melakukan implementasi class diagram	Mengimplementasikan seluruh class diagram	04/06/2020	18/07/2020	R N A F
20.	Melakukan integrasi implementasi class diagram dan implementasi UI	<ul style="list-style-type: none"> 1. Integrasi modul A (Class diagram 1 dan UI 1) 2. Integrasi modul B (Class diagram 2 dan UI 2) 3. Integrasi modul C (Class diagram 3 dan UI 3) 4. Dst. 	18/06/2020	24/07/2020	M A I
21.	Melakukan Pengujian terhadap sistem	<ul style="list-style-type: none"> 1. Pengujian unit 2. Pengujian integrasi 3. Pengujian kompatibilitas 4. Pengujian acceptance 	24/06/2020	03/08/2020	M R
22.	Membuat dokumen pengujian sistem	<ul style="list-style-type: none"> 1. Dokumentasi Pengujian unit 2. Dokumentasi Pengujian integrasi 3. Dokumentasi Pengujian kompatibilitas 4. Dokumentasi Pengujian acceptance 5. Upload ke sistem manajemen proyek 	03/07/2020	05/08/2020	M R
23.	Melaksanakan presentasi ke client	<ul style="list-style-type: none"> 1. Menjadwalkan pertemuan dengan client 2. Mempersiapkan powerpoint 3. Melaksanakan presentasi 	05/07/2020	17/08/2020	M F S
24.	Menutup proyek	<ul style="list-style-type: none"> 1. Menjadwalkan pertemuan dengan client 2. Mempersiapkan powerpoint 3. Melaksanakan presentasi 	17/07/2020	20/08/2020	M F S

Lampiran 4 Lampiran Kasus Penjadwalan (lanjutan)

Saran untuk perbaikan:

- Tampilan diperantik
- Laporan dilengkapi

Demikian data ini kami telah kami validasi dan verifikasi dan dapat digunakan untuk sebagaimana mestinya.

Surabaya, 20 Maret 2020



Prima Vista
DAVID
Malinto

Lampiran 5 Lampiran Kasus Penjadwalan (lanjutan)