



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



**RANCANG BANGUN SISTEM MANAJEMEN
PROYEK PERANGKAT LUNAK DENGAN
IDENTIFIKASI TUGAS KRITIS MENGGUNAKAN
*CRITICAL PATH METHOD***

SKRIPSI

**POLITEKNIK
NEGERI
JAKARTA**

RAGILIAWAN PUTRA RENCANA

2207412013

**PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA DAN
KOMPUTER
POLITEKNIK NEGERI JAKARTA**

2026



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini:

Nama : Ragiliawan Putra Rencana
NIM : 2207412013
Jurusan/Program Studi : T. Informatika dan Komputer/Teknik Informatika
Judul Skripsi : Rancang Bangun Sistem Manajemen Proyek Perangkat Lunak Dengan Identifikasi Tugas Kritis Menggunakan *Critical Path Method*

Menyatakan dengan sebenarnya bahwa skripsi ini benar-benar merupakan hasil karya saya sendiri, bebas dari peniruan terhadap karya dari orang lain. Kutipan pendapat dan tulisan orang lain ditunjuk sesuai dengan cara-cara penulisan karya ilmiah yang berlaku.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa dalam skripsi ini terkandung ciri-ciri plagiat dan bentuk-bentuk peniruan lain yang dianggap melanggar peraturan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

POLITEKNIK
NEGERI
JAKARTA

Depok, 20 Juli 2026

Yang membuat pernyataan,

Ragiliawan Putra Rencana

NIM 2207412013



LEMBAR PENGESAHAN

Skripsi diajukan oleh:

Nama : Ragiliawan Putra Rencana
NIM : 2207412013
Program Studi : T. Informatika dan Komputer/Teknik Informatika
Judul Skripsi : Rancang Bangun Sistem Manajemen Proyek Perangkat Lunak Dengan Identifikasi Tugas Kritis Menggunakan Critical Path Method

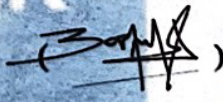
Telah diuji oleh tim penguji dalam Sidang Skripsi pada hari Rabu, tanggal 1 Bulan Juli, Tahun 2026 dan dinyatakan LULUS.

Disahkan oleh

Pembimbing I : Risna Sari, S.Kom., M.T.I. ()

Penguji I : Dr. Anita Hidayati, S.Kom., M.Kom. ()

Penguji II : Ariawan Andi Suhandana, S.Kom., M.T.I. ()

Penguji III : Bambang Warsuta S.Kom., M.T.I. ()

Mengetahui:

Jurusan Teknik Informatika dan Komputer
Ketua



Dr. Anita Hidayati, S.Kom., M.Kom.
NIP. 19790832003122003

© Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

Hak Cipta:

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang menggunakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

KATA PENGANTAR

Segala puji dan syukur saya panjatkan kepada Tuhan Yang Maha Esa atas rahmat dan bimbingan-Nya, sehingga saya dapat menyusun Skripsi ini dengan baik. Skripsi ini disusun sebagai salah satu persyaratan untuk menyelesaikan Program Diploma Empat pada Program Studi Teknik Informatika, Jurusan Teknik Informatika dan Komputer, Politeknik Negeri Jakarta. Penyusunan skripsi ini tidak terlepas dari dukungan dan bantuan berbagai pihak. Oleh karena itu, saya mengucapkan terima kasih kepada:

- a. Ibu Risna Sari, S.Kom., M.T.I., selaku dosen pembimbing yang telah memberikan arahan dan masukan selama proses penyusunan proposal.
- b. PT Lokasolusi Teknologi Indonesia, atas kesempatan dan dukungan yang diberikan selama proses penelitian sehingga penulis dapat memperoleh data dan gambaran permasalahan yang relevan.
- c. Keluarga, orang tua, serta teman-teman yang selalu memberikan doa dan semangat.

Penulis menyadari bahwa skripsi ini masih memiliki keterbatasan dan jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun demi penyempurnaan skripsi ini. Semoga skripsi ini dapat memberikan manfaat bagi pembaca dan berkontribusi pada pengembangan ilmu pengetahuan di bidang teknik informatika.

Depok, 25 Juni 2026

Penulis,

Ragiliawan Putra Rencana



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Politeknik Negeri Jakarta, saya bertanda

Nama : Ragiliawan Putra Rencana

NIM : 2207412013

Jurusan/Program Studi : T. Informatika dan Komputer/Teknik Informatika

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Politeknik Negeri Jakarta Hak Bebas Royalti Non-Eksklusif atas karya ilmiah saya yang berjudul:

**Rancang Bangun Sistem Manajemen Proyek Perangkat Lunak Dengan
Identifikasi Tugas Kritis Menggunakan *Critical Path Method***

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksklusif ini Politeknik Negeri Jakarta Berhak menyimpan, mengalihmediakan/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini Saya buat dengan sebenarnya.

Depok, 20 Juli 2026

Yang membuat pernyataan,



Ragiliawan Putra Rencana

NIM 2207412013



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Rancang Bangun Sistem Manajemen Proyek Perangkat Lunak Dengan Identifikasi Tugas Kritis Menggunakan *Critical Path Method*

ABSTRAK

PT Lokasolusi Teknologi Indonesia menghadapi kendala dalam manajemen proyek perangkat lunak karena informasi proyek tersebar pada media kerja terpisah, kesiapan tugas sulit dipantau, dan jalur kritis proyek belum terpantau secara terpusat. Penelitian ini bertujuan menghasilkan sistem manajemen proyek perangkat lunak berbasis web yang mengidentifikasi tugas kritis menggunakan Critical Path Method (CPM). Penelitian menggunakan pendekatan mixed method yang memadukan tahap kualitatif untuk analisis kebutuhan melalui observasi, wawancara, dan studi dokumentasi dengan tahap kuantitatif untuk pengujian sistem, sedangkan pengembangan perangkat lunak mengikuti model Waterfall. Sistem menghitung nilai earliest start, earliest finish, latest start, latest finish, dan slack untuk menentukan jalur kritis pada level project sehingga kesiapan dan prioritas tugas dapat dipantau secara terpusat. Hasil pengujian menunjukkan black box testing berhasil 100% dari 59 skenario, User Acceptance Testing sesuai 100% dari 25 skenario, dan System Usability Scale memperoleh skor rata-rata 83,1 yang termasuk kategori Acceptable dengan Grade A serta Adjective Rating Excellent. Dengan demikian, Prolytics dinilai layak digunakan untuk mendukung monitoring proyek perangkat lunak dengan identifikasi tugas kritis di PT Lokasolusi Teknologi Indonesia.

Kata kunci: *approval workflow, Critical Path Method, dashboard monitoring, dependency tugas, monitoring proyek, role-based access control, tugas kritis*



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR ISI

SURAT PERNYATAAN BEBAS PLAGIARISME	ii
KATA PENGANTAR.....	iv
SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS.....	v
ABSTRAK	vi
DAFTAR ISI.....	vii
DAFTAR TABEL	xi
DAFTAR GAMBAR.....	xii
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang Masalah	1
1.2 Perumusan Masalah	2
1.3 Batasan Masalah	3
1.4 Tujuan dan Manfaat	3
1.4.1 Tujuan Penelitian	3
1.4.2 Manfaat Penelitian	4
1.5 Sistematika Penulisan	4
BAB 2 TINJAUAN PUSTAKA.....	6
2.1 Tinjauan Teoritis	6
2.1.1 Manajemen Proyek dan <i>Monitoring</i>	6
2.1.2 Manajemen Tugas Phase/Termin, dan Keterkaitan Tugas	6
2.1.3 <i>Critical Path Method</i>	8
2.1.4 Perhitungan CPM dan Perubahan Jalur <i>Kritis</i>	10
2.1.5 Integrasi <i>Version Control System</i> /GitHub untuk <i>Monitoring</i> Aktivitas Teknis.....	11
2.1.6 Basis Data Relasional untuk Pemodelan <i>Project</i> , <i>Phase</i> , Tugas, dan <i>Dependency</i>	12
2.1.7 Large Language Model sebagai Asisten Analisis Requirement	13



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.1.8 <i>Framework Frontend</i> Next.js, React, dan Tailwind CSS	13
2.1.9 Bahasa Pemrograman Go dan <i>Framework</i> Gin	16
2.1.10 <i>RESTful</i> API dan Dokumentasi Swagger.....	19
2.1.11 PostgreSQL, Bun ORM, dan Penyimpanan Data	20
2.1.12 Autentikasi JWT dan Role-Based Access Control	21
2.1.13 Black Box Testing dan User Acceptance Testing.....	22
2.1.14 Usability Testing dan System Usability Scale	22
2.2 Penelitian Terdahulu	24
BAB 3 METODE PENELITIAN.....	28
3.1 Rancangan Penelitian.....	28
3.1.1 Teknik Pengumpulan Data.....	28
3.1.2 Teknik Analisis Data.....	28
3.2 Tahapan Penelitian.....	29
3.2.1 Identifikasi Masalah.....	30
3.2.2 Studi Literatur	30
3.2.3 Pengumpulan Data.....	30
3.2.4 Analisis Kebutuhan.....	30
3.2.5 Perancangan Sistem.....	30
3.2.6 Implementasi.....	30
3.2.7 Pengujian.....	31
3.2.8 Evaluasi.....	31
3.2.9 Penyusunan Laporan.....	31
3.3 Objek Penelitian.....	31
BAB 4 HASIL DAN PEMBAHASAN.....	32
4.1 Analisis Kebutuhan Sistem	32
4.1.1 Analisis Hasil Wawancara dengan <i>Project Manager</i>	32



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.1.2 Analisis Hasil Wawancara dengan <i>Lead Developer</i>	33
4.1.3 Analisis Hasil Wawancara dengan <i>Software Developer</i>	33
4.1.4 Analisis Hasil Observasi	34
4.1.5 Analisis Proses Bisnis Existing	34
4.1.6 Analisis Proses Bisnis Future	37
4.1.7 Perbandingan Proses Bisnis Existing dan Future	40
4.1.8 Identifikasi Permasalahan dan Tema Kebutuhan	43
4.1.9 Analisis Pengguna Sistem	45
4.1.10 Kebutuhan Fungsional Sistem	47
4.1.11 <i>Use Case Diagram</i>	53
4.1.12 <i>Activity Diagram</i>	55
4.1.13 Kebutuhan Non - Fungsional Sistem	65
4.2 Perancangan Sistem	66
4.2.1 Perancangan Arsitektur Sistem	67
4.2.2 Perancangan Antarmuka Pengguna	68
4.2.3 Perancangan Hak Akses Pengguna	78
4.2.4 Perancangan Basis Data	80
4.2.5 Perancangan Algoritma CPM	80
4.2.6 Perhitungan Critical Path Method (CPM)	82
4.3 Implementasi Sistem	85
4.3.1 Teknologi dan Tools Implementasi	85
4.3.2 Struktur <i>Project</i>	87
4.3.3 Implementasi <i>Backend</i> dan Basis Data	88
4.3.4 Implementasi Modul Utama	93
4.3.5 Implementasi Tampilan Visual dan Responsivitas Antarmuka	111
4.4 Pengujian Sistem	113



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.4.1 Deskripsi Pengujian	113
4.4.2 Prosedur Pengujian	113
4.4.3 Hasil Pengujian	115
4.4.4 Analisis Data dan Evaluasi Hasil Pengujian	127

BAB 5 PENUTUP	130
5.1 Kesimpulan	130
5.2 Saran	130
DAFTAR PUSTAKA	131
DAFTAR RIWAYAT HIDUP PENULIS	136
LAMPIRAN	137





© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang menggunakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR TABEL

Tabel 1 Rentang <i>Acceptability System Usability Scale</i>	23
Tabel 2 Grade Scale dan Adjective Rating System Usability Scale.....	23
Tabel 3 Penelitian Terdahulu	24
Tabel 4 Perbandingan Proses Bisnis Existing dan Future	40
Tabel 5 Identifikasi Permasalahan dan Tema Kebutuhan.....	43
Tabel 6 Analisis Pengguna Sistem.....	46
Tabel 7 Kebutuhan Fungsional Sistem	48
Tabel 8 Kebutuhan Non Fungsional Sistem	65
Tabel 9 Rancangan Tabel Hak Akses Pengguna	78
Tabel 10 Rancangan Algoritma Critical Path Method.....	81
Tabel 11 Identifikasi Kegiatan Proyek	82
Tabel 12 Hasil Perhitungan Maju dan Mundur.....	83
Tabel 13 Hasil Perhitungan Critical Path Method	83
Tabel 14 Teknologi dan Tools	86
Tabel 15 Struktur Project Prolytics.....	87
Tabel 16 Daftar Pernyataan SUS	114
Tabel 17 Blackbox Testing	115
Tabel 18 User Acceptance Testing	122
Tabel 19 Peran Responden.....	125
Tabel 20 Rekapitulasi Jawaban Usability Testing Menggunakan SUS.....	126
Tabel 21 Rekapitulasi Skor SUS.....	126



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR GAMBAR

Gambar 2. 1 Skala Penilaian <i>System Usability Scale</i>	24
Gambar 3. 1 Tahapan Penelitian	29
Gambar 4. 1 Proses Bisnis Existing	36
Gambar 4. 2 Proses Bisnis <i>Future</i>	39
Gambar 4. 3 Usecase Diagram.....	54
Gambar 4. 4 <i>Activity Diagram</i> Login Sistem	56
Gambar 4. 5 <i>Activity Diagram</i> Mengelola Proyek	57
Gambar 4. 6 <i>Activity Diagram</i> Mengelola Tugas.....	58
Gambar 4. 7 <i>Activity Diagram Approval</i> Tugas dan Aktivasi CPM.....	59
Gambar 4. 8 <i>Activity Diagram</i> Mengedit Tugas	60
Gambar 4. 9 <i>Activity Diagram</i> Perhitungan CPM	61
Gambar 4. 10 <i>Activity Diagram</i> Melihat Hasil <i>Critical Path Method</i>	62
Gambar 4. 11 <i>Activity Diagram</i> Memperbarui Status Tugas	63
Gambar 4. 12 <i>Activity Diagram</i> Aktivitas GitHub dan Progress Tugas	64
Gambar 4. 13 Arsitektur Sistem.....	67
Gambar 4. 14 Wireframe Halaman <i>Login</i>	69
Gambar 4. 15 <i>Wireframe Dashboard Project Manager</i>	70
Gambar 4. 16 <i>Wireframe</i> Halaman Daftar Proyek.....	71
Gambar 4. 17 <i>Wireframe</i> halaman membuat Proyek.....	72
Gambar 4. 18 <i>Wireframe</i> Halaman Detail Proyek.....	73
Gambar 4. 19 Wireframe Halaman Detail <i>Phase</i> dan CPM	74
Gambar 4. 20 <i>Wireframe</i> Visualisasi <i>Jalur Kritis</i>	75
Gambar 4. 21 <i>Wireframe</i> Visualisasi Gantt <i>Phase</i>	75
Gambar 4. 22 <i>Wireframe</i> Halaman Tugas Saya.....	76
Gambar 4. 23 <i>Wireframe</i> Halaman Persetujuan Tugas.....	77
Gambar 4. 24 <i>Wireframe</i> Halaman Manajemen User.....	78
Gambar 4. 25 Diagram Relasi Tabel.....	80
Gambar 4. 26 Grafik Hasil <i>Critical Path Method</i>	84
Gambar 4. 27 Tampilan Dokumentasi API Swagger Prolytics	89
Gambar 4. 28 Potongan Kode Konfigurasi <i>Database</i> (pkg/config/database.go) ..	91



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4. 29 Potongan Kode Fungsi Init Koneksi Database (pkg/database/postgres.go)	92
Gambar 4. 30 Tampilan Login Sistem Prolytics	94
Gambar 4. 31 Dashboard Project Manager	95
Gambar 4. 32 Detail Project dan Kondisi CPM Proyek	96
Gambar 4. 33 Halaman Create Project Prolytics	97
Gambar 4. 34 Visualisasi Dependency dan Jalur Kritis	98
Gambar 4. 35 Tabel Hasil Perhitungan CPM Project-Wide	99
Gambar 4. 36 Riwayat Snapshot CPM	100
Gambar 4. 37 Struktur Data Engine CPM	101
Gambar 4. 38 Inisialisasi Data dan Pembentukan Graph Dependency	102
Gambar 4. 39 Forward Pass dan Backward Pass	103
Gambar 4. 40 Perhitungan Slack, Tugas Kritis, dan Readiness	104
Gambar 4. 41 Pembentukan Critical Path dan Topological Sort	106
Gambar 4. 42 Halaman Approval Workflow Berdasarkan Phase	108
Gambar 4. 43 Tampilan Repository Terhubung dan Activity Feed GitHub	109
Gambar 4. 44 Detail Aktivitas GitHub yang Ditautkan pada Tugas	110
Gambar 4. 45 Sistem Warna dan Tema	111
Gambar 4. 46 Responsivitas dan Navigasi Mobile	112

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB I PENDAHULUAN

1.1 Latar Belakang Masalah

Dalam pengembangan perangkat lunak, keberhasilan sebuah *project* tidak hanya ditentukan oleh kualitas kode yang dihasilkan, tetapi juga oleh kemampuan tim dalam mengelola pekerjaan, jadwal, dan perubahan kebutuhan. *Project* perangkat lunak umumnya berjalan dalam kondisi yang dinamis karena kebutuhan pengguna, komposisi tugas, estimasi durasi, dan hubungan ketergantungan antarpekerjaan dapat berubah selama proses pengembangan berlangsung. Kondisi tersebut menjadikan *monitoring* proyek sebagai aspek penting agar pekerjaan tetap terkendali sesuai ruang lingkup dan target waktu yang telah ditetapkan.

Seiring meningkatnya kompleksitas proyek, pencatatan progres saja belum cukup apabila sistem belum mampu menunjukkan hubungan antar-tugas dan dampaknya terhadap jadwal penyelesaian. Pada praktiknya, tugas dalam proyek perangkat lunak tidak selalu dapat dikerjakan tanpa urutan tertentu. Sebagian tugas baru dapat dimulai setelah tugas lain selesai, misalnya perancangan basis data harus tersedia sebelum pengembangan fitur tertentu dilakukan. Hubungan ketergantungan atau *dependency* tersebut menyebabkan keterlambatan pada satu tugas dapat merembet ke tugas lain dan memengaruhi estimasi penyelesaian *phase* maupun *project* secara keseluruhan.

Persoalan menjadi lebih kompleks ketika terjadi perubahan di tengah pelaksanaan proyek. Penambahan tugas baru, perubahan estimasi durasi, perubahan *dependency*, atau penyelesaian tugas yang lebih cepat maupun lebih lambat dapat menggeser *timeline* yang sebelumnya telah direncanakan. Tanpa mekanisme analisis ulang yang jelas, *Project Manager* dan *Lead Developer* sulit memastikan apakah perubahan tersebut membuat target *phase* menjadi lebih cepat, tetap, atau justru lebih lama. Perubahan pada pekerjaan yang berada di jalur kritis akan berdampak besar terhadap durasi, sedangkan perubahan pada pekerjaan yang memiliki *slack* belum tentu berpengaruh; perbedaan ini sulit dikenali secara manual.

Permasalahan serupa ditemukan pada lingkungan pengembangan perangkat lunak di PT Lokasolusi Teknologi Indonesia. Berdasarkan observasi awal dan



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

wawancara, proses *monitoring* proyek masih tersebar pada enam media terpisah, yaitu percakapan chat, notulen rapat, *spreadsheet*, dokumen kerja, papan *Trello*, dan pemeriksaan aktivitas *repository* secara manual. Akibatnya, *Project Manager* dan *Lead Developer* perlu melakukan pemeriksaan silang ke banyak sumber hanya untuk mengetahui status aktual pekerjaan, tugas yang dapat dikerjakan, serta tugas yang berpotensi menahan penyelesaian *phase*.

Untuk menjawab kebutuhan tersebut, penelitian ini menggunakan *Critical Path Method* sebagai metode identifikasi tugas kritis. CPM menganalisis aktivitas proyek berdasarkan urutan pekerjaan, *dependency*, dan durasi sehingga dapat diketahui jalur kritis, tugas kritis, serta nilai *slack*. Perencanaan jadwal proyek perangkat lunak dapat disusun melalui pemodelan *dependency* aktivitas dan penentuan *critical path*, sehingga jadwal tidak hanya berupa daftar pekerjaan, melainkan struktur hubungan antaraktivitas yang menentukan urutan pelaksanaan. Lebih lanjut, jalur kritis dan durasi total proyek dapat berubah ketika kondisi proyek berubah, sehingga diperlukan pendekatan yang menghitung ulang jalur kritis saat data tugas diperbarui (Ren and Li, 2023).

Berdasarkan uraian tersebut, penelitian ini mengusulkan rancang bangun sistem *monitoring* proyek perangkat lunak dengan identifikasi tugas kritis menggunakan *Critical Path Method*. Pada sistem yang diusulkan, perhitungan CPM dilakukan pada level *project* dan mencakup seluruh *approved* tugas lintas *phase* dalam satu *project*. Setiap perubahan pada tugas, durasi, *dependency*, maupun status pekerjaan akan memicu perhitungan ulang sehingga sistem dapat menyajikan jalur kritis dan estimasi *timeline* terbaru. Dengan demikian, *Project Manager* dan *Lead Developer* dapat memantau progres proyek secara lebih terpusat dan memahami pekerjaan yang perlu didahulukan agar keterlambatan *phase* dapat ditekan.

1.2 Perumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan, permasalahan utama adalah belum tersedianya sistem yang mampu memantau progres proyek perangkat lunak secara terpusat sekaligus menunjukkan dampak hubungan antar-tugas terhadap *timeline phase*. Untuk menjawab permasalahan tersebut, penelitian ini merumuskan pertanyaan penelitian sebagai berikut:

**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

1. Bagaimana membangun sistem manajemen proyek perangkat lunak berbasis web yang mendukung pengelolaan *project*, *phase*, tugas, dan *dependency* dengan visibilitas berbasis peran di PT Lokasolusi Teknologi Indonesia?
2. Bagaimana membangun fitur identifikasi tugas kritis menggunakan *Critical Path Method* berdasarkan estimasi durasi tugas dan hubungan *dependency* antar-tugas pada level *project*?
3. Bagaimana menerapkan mekanisme perhitungan ulang *Critical Path Method* secara otomatis ketika terjadi perubahan data tugas sehingga sistem dapat menampilkan jalur kritis dan estimasi *timeline* terbaru?

1.3 Batasan Masalah

Agar penelitian ini tetap terfokus dan dapat diselesaikan dengan baik, ruang lingkup pengembangan sistem dibatasi sebagai berikut:

1. Penelitian dilakukan pada studi kasus proses *monitoring* proyek perangkat lunak di PT Lokasolusi Teknologi Indonesia, meliputi pengelolaan *project*, *phase*, tugas, dan *dependency* antar tugas dengan visibilitas berbasis peran.
2. Identifikasi tugas kritis dihitung pada level *project* dengan ruang lingkup *dependency* antar tugas yang dibatasi, *dependency* diperbolehkan lintas *phase* selama berada dalam *project* yang sama dan tidak diperbolehkan lintas *project*. Hanya tugas berstatus *approved* yang masuk perhitungan.
3. Pengguna sistem dibatasi pada tiga peran, yaitu *Project Manager*, *Lead Developer*, dan *Software Developer*, sesuai struktur tim pengembangan pada objek penelitian.

1.4 Tujuan dan Manfaat**1.4.1 Tujuan Penelitian**

Berdasarkan rumusan masalah yang telah ditetapkan, tujuan penelitian ini adalah sebagai berikut:

1. Menghasilkan sistem manajemen proyek perangkat lunak berbasis web yang mendukung pengelolaan *project*, *phase*, tugas, dan *dependency* dengan visibilitas berbasis peran di PT Lokasolusi Teknologi Indonesia.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2. Menghasilkan fitur identifikasi tugas kritis menggunakan *Critical Path Method* berdasarkan estimasi durasi tugas dan hubungan *dependency* antar-tugas pada level *project*.
3. Menghasilkan mekanisme perhitungan ulang *Critical Path Method* secara otomatis ketika terjadi perubahan data tugas sehingga sistem dapat menampilkan jalur kritis dan estimasi *timeline* terbaru.

1.4.2 Manfaat Penelitian

Penelitian ini diharapkan memberikan manfaat bagi PT Lokasolusi Teknologi Indonesia berupa terpusatnya proses *monitoring* proyek yang sebelumnya tersebar pada enam media terpisah menjadi satu sistem, sehingga *Project Manager* dan *Lead Developer* tidak perlu lagi melakukan pemeriksaan silang antar sumber untuk mengetahui status pekerjaan. Sistem juga menampilkan tugas kritis dan urutan pengerjaan berdasarkan *dependency* yang sebelumnya harus ditelusuri secara manual, serta menyajikan estimasi *timeline* terbaru setiap kali terjadi perubahan data tugas, sehingga keputusan pengendalian proyek dapat diambil lebih cepat dan berdasarkan data. Selain itu, penelitian ini diharapkan menjadi referensi penerapan *Critical Path Method* berbasis *dependency* untuk mengidentifikasi tugas kritis pada sistem *monitoring* proyek perangkat lunak.

1.5 Sistematika Penulisan

Laporan penelitian ini disusun dalam lima bab yang saling berkaitan untuk memberikan gambaran yang utuh dan runtut mengenai proses pengerjaan. Penyusunan sistematika ini bertujuan agar pembaca dapat mengikuti alur penelitian secara *logis*, mulai dari identifikasi masalah hingga kesimpulan akhir. Susunan bab dalam laporan ini adalah sebagai berikut:

1. Bab I Pendahuluan, berisi latar belakang masalah, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, serta sistematika penulisan.
2. Bab II Tinjauan Pustaka, berisi teori-teori pendukung dan penelitian terdahulu yang relevan dengan sistem *monitoring* proyek perangkat lunak, *task management*, *dependency*, *Critical Path Method*, integrasi *GitHub*, *Large Language Model*, dan basis data relasional.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

3. Bab III Metode Penelitian, berisi pendekatan dan desain penelitian, fungsi utama sistem, arsitektur sistem, sumber data, teknik pengumpulan data, teknik analisis data, tahapan penelitian, serta objek penelitian.
4. Bab IV Hasil dan Pembahasan, berisi analisis kebutuhan, perancangan sistem, rancangan implementasi, rancangan pengujian, serta pembahasan rancangan sistem *monitoring* proyek perangkat lunak dan *Critical Path Method*.
5. Bab V Penutup, berisi kesimpulan dari penelitian serta saran untuk pengembangan lebih lanjut.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB II TINJAUAN PUSTAKA

2.1 Tinjauan Teoritis

2.1.1 Manajemen Proyek dan *Monitoring*

Manajemen proyek perangkat lunak merupakan proses perencanaan, pengorganisasian, pelaksanaan, pemantauan, dan pengendalian sumber daya untuk menghasilkan perangkat lunak sesuai ruang lingkup, waktu, dan kualitas yang ditetapkan. Dalam proyek perangkat lunak, keberhasilan tidak hanya ditentukan oleh kemampuan teknis pengembang, tetapi juga oleh kemampuan tim dalam mengelola pekerjaan, jadwal, komunikasi, dan perubahan kebutuhan yang muncul selama proses pengembangan berlangsung.

Monitoring proyek berfungsi untuk memastikan bahwa aktivitas pengembangan berjalan sesuai rencana dan penyimpangan dapat diketahui lebih awal. Sistem informasi manajemen proyek berbasis web dapat membantu pencatatan data proyek, penyimpanan informasi, dan penyajian laporan secara lebih terstruktur dibandingkan proses manual (Agustin and Kustiana, 2025). Dengan *monitoring* yang terpusat, *Project Manager* dan *Lead Developer* dapat melihat status pekerjaan, progres *phase*, serta informasi yang dibutuhkan untuk mengambil tindakan pengendalian.

Pada penelitian ini, *monitoring* tidak hanya dipahami sebagai proses melihat persentase progres, tetapi juga sebagai proses memahami kondisi tugas yang memengaruhi *timeline* proyek. Oleh karena itu, sistem *monitoring* yang dirancang perlu mampu menampilkan hubungan antara *project*, *phase*, tugas, *dependency*, durasi, status pekerjaan, dan hasil perhitungan *jalur kritis*. Informasi tersebut menjadi dasar agar pengelolaan proyek tidak hanya bersifat administratif, tetapi juga mendukung pengendalian waktu pengerjaan.

2.1.2 Manajemen Tugas Phase/Termin, dan Keterkaitan Tugas

Manajemen tugas merupakan proses pengelolaan pekerjaan yang mencakup pembentukan tugas, penentuan penanggung jawab, pendefinisian status, penetapan

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

durasi, dan pemantauan penyelesaian pekerjaan(Ren and Li, 2023). Dalam konteks proyek perangkat lunak, tugas biasanya berasal dari kebutuhan bisnis atau *requirement* yang kemudian diterjemahkan menjadi pekerjaan teknis oleh *Lead Developer*. Tugas teknis tersebut dapat berupa perancangan basis data, pembuatan antarmuka, pengembangan API, integrasi modul, pengujian, atau perbaikan fitur.

Phase atau *termin* digunakan sebagai unit kerja yang membagi *project* menjadi bagian yang lebih mudah dikelola(Febriani et al., 2025). Pembagian ini sesuai dengan karakter proyek *vendor-client*, karena dalam satu *project* dapat muncul *phase* pengembangan awal, *phase* penambahan fitur, *phase improvement*, atau *phase bug fixing*. Dengan model tersebut, satu *project* tetap menjadi induk utama, sedangkan *phase* berfungsi sebagai *container* atau pengelompokan tugas dalam satu *project*. *Phase* tidak mengunci urutan kerja; urutan kerja ditentukan oleh *dependency* antar tugas. *Dependency* diperbolehkan lintas *phase* selama masih dalam *project* yang sama.

Dependency tugas menunjukkan hubungan ketergantungan antara satu tugas dengan tugas lain(Erwin Erwin et al., 2025). Tugas yang menjadi prasyarat disebut *predecessor*, sedangkan tugas yang bergantung pada penyelesaian tugas sebelumnya disebut *successor*. *Dependency* penting karena tidak semua pekerjaan dapat dimulai secara bersamaan. Sebagai contoh, pengembangan fitur tertentu dapat bergantung pada ketersediaan rancangan *database*, *endpoint API*, atau hasil *approval requirement*. Jika *dependency* tidak dikelola dengan baik, urutan kerja menjadi tidak jelas dan estimasi *timeline* dapat menjadi tidak akurat.

Dalam sistem yang dirancang, *dependency* digunakan untuk menentukan tugas yang dapat dikerjakan lebih dulu dan tugas yang masih tertahan. Tugas yang seluruh *predecessor*-nya telah selesai dapat dianggap siap dikerjakan, sedangkan tugas yang masih memiliki *predecessor* belum selesai berada dalam kondisi *blocked*. Model ini menjadi dasar bagi perhitungan CPM karena jalur pekerjaan dibentuk dari hubungan antar tugas, bukan dari penilaian skor atau bobot prioritas.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.1.3 Critical Path Method

Critical Path Method (CPM) adalah metode penjadwalan proyek yang digunakan untuk menganalisis urutan aktivitas, hubungan ketergantungan, durasi, serta jalur pekerjaan yang menentukan durasi total proyek. Dalam CPM, setiap aktivitas atau tugas memiliki durasi tertentu dan dapat memiliki hubungan predecessor-successor dengan aktivitas lain. Rangkaian hubungan tersebut dapat digambarkan sebagai *network diagram* agar alur pekerjaan dan ketergantungan antaraktivitas dapat terlihat secara sistematis.

Perencanaan jadwal proyek perangkat lunak dapat dilakukan melalui representasi *dependency* menggunakan *network chart*, estimasi waktu aktivitas, dan penentuan *critical path* (Febriani et al., 2025). Dengan pendekatan tersebut, jadwal proyek tidak hanya disusun berdasarkan daftar pekerjaan, tetapi juga berdasarkan struktur hubungan antaraktivitas yang menentukan urutan pelaksanaan. CPM relevan digunakan dalam penelitian ini karena permasalahan utama sistem adalah memahami tugas yang berdampak langsung terhadap penyelesaian *project secara keseluruhan*.

Perhitungan CPM secara konseptual dilakukan melalui dua proses utama, yaitu *forward pass* dan *backward pass*. *Forward pass* digunakan untuk menghitung *earliest start* (ES) dan *earliest finish* (EF), yaitu waktu paling awal suatu tugas dapat dimulai dan diselesaikan berdasarkan predecessor-nya. *Backward pass* digunakan untuk menghitung *latest start* (LS) dan *latest finish* (LF), yaitu waktu paling akhir suatu tugas dapat dimulai dan diselesaikan tanpa menunda penyelesaian *project*. Dari nilai tersebut, sistem dapat menghitung *slack* atau *float* sebagai selisih antara waktu paling akhir dan waktu paling awal tugas.

Tugas yang memiliki *slack* atau *float* bernilai nol termasuk *tugas kritis* karena keterlambatan pada tugas tersebut akan memengaruhi durasi penyelesaian *project*. Rangkaian *tugas kritis* membentuk *jalur kritis* atau *critical path*. *Jalur kritis* menunjukkan urutan pekerjaan yang harus dijaga agar *project* tidak mengalami keterlambatan. Oleh karena itu, CPM tidak menghasilkan penilaian berbasis skor, melainkan menunjukkan tugas yang perlu mendapat perhatian lebih karena posisinya dalam alur *dependency* dan pengaruhnya terhadap *timeline*.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Dalam penelitian ini, CPM dihitung pada level project. Input perhitungan meliputi `project_start_date` sebagai anchor timeline, daftar approved tugas, `estimated_duration` setiap tugas, dan dependency antar tugas dalam satu project. Tugas dengan status pending dan rejected tidak dimasukkan ke dalam active CPM. Dependency diperbolehkan lintas phase selama masih dalam project yang sama, sedangkan dependency lintas project tidak diperkenankan. Output perhitungan CPM meliputi nilai Early Start (ES), Early Finish (EF), Late Start (LS), dan Late Finish (LF) yang direpresentasikan sebagai tanggal kalender, nilai slack atau float dalam satuan hari, tugas kritis, serta critical path. Selain itu, phase memperoleh `calculated_start_date` dari nilai ES minimum tugas approved dalam phase, dan `calculated_end_date` dari nilai EF maksimum tugas approved dalam phase. Output tersebut digunakan dalam dashboard monitoring agar Project Manager dan Lead Developer dapat memahami urutan pengerjaan dan tugas yang berisiko menunda penyelesaian project.

Dalam perhitungan CPM dikenal beberapa istilah, yaitu *earliest start* (ES) sebagai waktu paling awal suatu tugas dapat dimulai, *earliest finish* (EF) sebagai waktu paling awal tugas dapat diselesaikan, *latest start* (LS) sebagai waktu paling lambat tugas boleh dimulai tanpa memperlambat proyek, *latest finish* (LF) sebagai waktu paling lambat tugas boleh diselesaikan, *t* sebagai durasi tugas, serta *float* atau *slack* sebagai kelonggaran waktu pengerjaan (Febriani et al., 2025).

Perhitungan maju (*forward pass*) dilakukan dari tugas awal hingga akhir untuk memperoleh nilai ES dan EF menggunakan Persamaan (1).

$$EF = ES + t \quad (1)$$

Apabila suatu tugas memiliki lebih dari satu *predecessor*, nilai ES yang digunakan adalah EF terbesar di antara *predecessor*-nya.

Perhitungan mundur (*backward pass*) dilakukan dari tugas akhir hingga awal untuk memperoleh nilai LS dan LF menggunakan Persamaan (2).

$$LS = LF - t \quad (2)$$

Apabila suatu tugas memiliki lebih dari satu *successor*, nilai LF yang digunakan adalah LS terkecil di antara *successor*-nya.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Total float atau *slack* setiap tugas dihitung dengan Persamaan (3):

$$TS = LS - ES = LF - EF \quad (3)$$

Tugas yang memiliki *total float* bernilai nol merupakan tugas kritis. Rangkaian tugas kritis yang saling berhubungan dari awal hingga akhir proyek membentuk jalur kritis (*critical path*), yaitu urutan pekerjaan yang menentukan durasi total proyek dan harus dijaga agar proyek tidak terlambat (Febriani et al., 2025).

2.1.4 Perhitungan CPM dan Perubahan Jalur Kritis

Proyek perangkat lunak bersifat dinamis karena perubahan dapat terjadi selama pengerjaan berlangsung. Perubahan tersebut dapat berupa penambahan tugas baru, perubahan estimasi durasi, perubahan *dependency*, perubahan status penyelesaian, atau *request* lanjutan dari *client*. Ketika data tersebut berubah, hasil perhitungan CPM sebelumnya dapat menjadi tidak relevan karena *jalur kritis* dan estimasi penyelesaian project juga dapat berubah.

critical path dapat berubah ketika kondisi proyek berubah, sehingga pendekatan dinamis diperlukan untuk mengidentifikasi *jalur kritis* dan durasi proyek secara lebih cepat. Konsep ini menjadi dasar penting bagi penelitian ini karena sistem yang dirancang tidak hanya menghitung CPM sekali pada tahap perencanaan awal, tetapi juga melakukan perhitungan ulang ketika terjadi perubahan data tugas (Erwin Erwin et al., 2025).

Perhitungan ulang CPM pada sistem dilakukan dengan membaca ulang daftar tugas, *dependency*, durasi, dan status pekerjaan pada *phase* yang dipilih. Setelah data berubah, sistem menghitung kembali nilai ES, EF, LS, LF, *slack* atau *float*, durasi internal CPM project, *readiness*, dan *jalur kritis*. Nilai-nilai ES, EF, LS, dan LF tersebut direpresentasikan sebagai tanggal kalender yang dianchor pada *project_start_date*, sehingga dapat langsung digunakan sebagai gambaran timeline pengerjaan. Perlu dicatat bahwa versi ini belum mencakup working calendar, hari libur, maupun resource availability; aspek-aspek tersebut direncanakan sebagai future enhancement.

Mekanisme perhitungan ulang ini penting bagi *Project Manager* dan *Lead Developer* karena perubahan kecil pada tugas tertentu dapat berdampak besar

apabila tugas tersebut berada pada *jalur kritis*. Sebaliknya, perubahan pada tugas yang memiliki *slack* mungkin tidak langsung mengubah durasi internal *phase*. Informasi ini membantu tim membedakan perubahan administratif dengan perubahan yang benar-benar memengaruhi struktur CPM.

2.1.5 Integrasi *Version Control System*/GitHub untuk *Monitoring* Aktivitas Teknis

Version Control System (VCS) merupakan teknologi yang digunakan dalam pengembangan perangkat lunak untuk menyimpan riwayat perubahan kode menurut (Github, n.d.), mengelola versi, dan mendukung kolaborasi antaranggota tim. GitHub sebagai platform berbasis Git menyediakan fitur seperti *repository*, *branch*, *commit*, *pull request*, *issue*, dan *webhook* yang dapat dimanfaatkan untuk membaca aktivitas teknis pengembangan perangkat lunak.

Dalam sistem *monitoring* proyek, aktivitas *repository* dapat menjadi informasi pendukung untuk melihat hubungan antara tugas yang tercatat di sistem dengan aktivitas teknis yang dilakukan developer. Misalnya, *branch* atau *pull request* dapat dikaitkan dengan tugas tertentu, sedangkan *commit* dapat menjadi indikator adanya pekerjaan teknis yang sedang atau telah dilakukan. Informasi ini membantu mengurangi kesenjangan antara laporan manajerial dan aktivitas teknis di *repository*.

GitHub menyediakan *webhook* yang memungkinkan sistem menerima pembaruan ketika suatu *event* terjadi. Dokumentasi GitHub menjelaskan bahwa *webhook* dapat memberikan pembaruan hampir *real-time* karena dipicu saat *event* terjadi, serta lebih efisien dibanding polling API berulang untuk memantau banyak resource (Github, n.d.). Dengan pendekatan ini, sistem dapat memperbarui informasi aktivitas teknis tanpa harus bergantung sepenuhnya pada *input* manual.

Pada penelitian ini, integrasi GitHub diposisikan sebagai pendukung *monitoring* aktivitas teknis, bukan sebagai metode utama penentuan *jalur kritis*. CPM tetap dihitung berdasarkan data *approved tugas*, *dependency*, dan durasi pada *level project*. Data *repository* berfungsi untuk memperkaya informasi progres, membantu keterlacakan pekerjaan, dan mendukung validasi aktivitas pengembangan yang berkaitan dengan tugas.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.1.6 Basis Data Relasional untuk Pemodelan *Project, Phase, Tugas, dan Dependency*

Basis data relasional digunakan untuk menyimpan data dalam bentuk tabel yang saling berhubungan melalui *primary key* dan *foreign key*. Pada sistem *monitoring* proyek perangkat lunak, pendekatan relasional sesuai digunakan karena data yang dikelola memiliki struktur yang jelas, seperti *project, phase, requirement, technical tugas, dependency*, status pekerjaan, aktivitas *repository*, dan hasil perhitungan CPM.

Model relasional mendukung integritas data melalui relasi antarentitas. *Project* dapat memiliki banyak *phase*, *phase* dapat memiliki banyak tugas, dan tugas dapat memiliki *dependency* terhadap tugas lain. Relasi tersebut penting karena perhitungan CPM membutuhkan data *dependency* yang konsisten. Jika struktur data tidak terjaga, hasil perhitungan *jalur kritis* dapat menjadi tidak valid.

Selain menyimpan data operasional, basis data juga digunakan untuk menyimpan hasil perhitungan authoritative seperti nilai ES, EF, LS, LF, *slack* atau *float*, status *tugas kritis, readiness*, durasi internal CPM, critical paths, dan riwayat perubahan CPM. Selain tabel operasional, basis data juga mencakup tabel *project_cpm_snapshots* sebagai penyimpan *snapshot* CPM pada level project, tabel *task_change_requests* untuk menampung permintaan perubahan tugas approved yang berdampak pada CPM sebelum disetujui Project Manager, serta kolom *calculated_start_date* dan *calculated_end_date* pada tabel *phase* sebagai hasil rollup dari tugas approved dalam *phase* tersebut. Pada sistem ini, PostgreSQL dipilih karena mendukung pemodelan data relasional, integritas referensial, tipe data JSONB, serta transaksi yang diperlukan untuk menjaga konsistensi tugas, *dependency*, dan *snapshot* CPM (PostgreSQL Global Development Group, 2026).

Dengan demikian, basis data relasional pada penelitian ini bukan hanya berfungsi sebagai tempat penyimpanan, tetapi juga sebagai fondasi model informasi proyek. Struktur data yang rapi memungkinkan sistem mengelola *dependency*, menghitung ulang CPM, menghubungkan aktivitas teknis dari GitHub, dan menyajikan *monitoring* proyek secara terpusat.

2.1.7 Large Language Model sebagai Asisten Analisis Requirement

Large Language Model (LLM) merupakan model kecerdasan buatan yang mampu memproses dan menghasilkan teks alami. Dalam konteks rekayasa perangkat lunak, LLM dapat digunakan untuk membantu aktivitas yang berkaitan dengan pemahaman teks, seperti merangkum kebutuhan, mengidentifikasi poin penting dari *brief*, atau memberikan saran awal struktur *requirement*. Pemanfaatan otomatisasi pada *requirements engineering* menjadi relevan karena banyak kebutuhan perangkat lunak dinyatakan dalam bahasa alami dan dokumen teks yang volumenya besar (Umar and Lano, 2024).

Kajian tentang penggunaan Natural Language Processing dalam software *requirements engineering* menunjukkan bahwa teknik komputasional dapat membantu proses *requirement* melalui ekstraksi, formalisation, klasifikasi, dan analisis teks kebutuhan (Umar and Lano, 2024). Namun, hasil yang diberikan oleh teknologi semacam ini tetap perlu divalidasi karena *requirement* perangkat lunak memiliki konteks bisnis, teknis, dan operasional yang tidak selalu dapat diputuskan secara otomatis.

Pada penelitian ini, LLM diposisikan sebagai asisten *requirement*. *Project Manager* dapat memasukkan *brief* atau *request* fitur, kemudian LLM membantu memberikan saran awal berupa ringkasan kebutuhan atau kandidat *requirement*/tugas. Hasil tersebut tidak langsung menjadi keputusan final, melainkan harus ditinjau dan divalidasi oleh *Project Manager* atau *Lead Developer* sebelum masuk ke proses pembentukan *technical tugas* dan perhitungan CPM.

Dengan posisi tersebut, LLM mendukung efisiensi analisis awal tanpa menggantikan peran manusia dalam validasi kebutuhan dan penyusunan *dependency* teknis. Hal ini penting agar sistem tetap mempertahankan akurasi konteks proyek, sekaligus menjaga agar *input* CPM yang digunakan berasal dari tugas, durasi, dan *dependency* yang telah diverifikasi oleh aktor yang berwenang.

2.1.8 Framework Frontend Next.js, React, dan Tailwind CSS

Pengembangan antarmuka membutuhkan teknologi *frontend* yang mampu mendukung halaman sistem yang terstruktur, penggunaan komponen secara

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

berulang, serta konsistensi tampilan pada berbagai fitur seperti *dashboard*, *project detail*, *phase detail*, *task board*, *approval queue*, halaman profil, dan visualisasi hasil perhitungan *Critical Path Method* (CPM). Untuk memenuhi kebutuhan tersebut, pemilihan teknologi *frontend* dilakukan dengan mempertimbangkan kemampuan *framework* dalam mengelola struktur halaman yang kompleks, dukungan terhadap komponen yang dapat digunakan ulang, serta kemudahan pemeliharaan antarmuka secara menyeluruh.

1. Next.js dibandingkan React + Vite

Next.js merupakan *framework* berbasis React yang menyediakan struktur tambahan untuk pengembangan aplikasi web, mencakup *routing*, *rendering*, optimasi, dan pengelolaan konfigurasi aplikasi secara terintegrasi (Next.js, n.d.). Dengan struktur tersebut, pengembang dapat mendefinisikan halaman dan navigasi antar halaman secara deklaratif tanpa konfigurasi tambahan yang kompleks.

Sebagai perbandingan, Vite lebih berperan sebagai *build tool* dan *development server* yang berfokus pada kecepatan proses pengembangan, termasuk dukungan *Hot Module Replacement* (HMR) (Vite, 2026). Vite tidak menyediakan struktur *routing* atau pola manajemen halaman secara bawaan, sehingga pengembang perlu mengonfigurasi sendiri struktur aplikasi ketika kompleksitas meningkat.

Next.js dipilih karena penelitian membutuhkan struktur aplikasi yang jelas untuk mengelola banyak halaman dan fitur yang saling terhubung. Halaman seperti *dashboard*, *project detail*, *phase detail*, *task board*, *approval queue*, dan halaman profil membutuhkan pengaturan *routing* yang rapi agar alur antarmuka mudah dikembangkan dan dipelihara. React + Vite tetap dapat digunakan untuk membangun aplikasi *frontend*, tetapi membutuhkan konfigurasi tambahan untuk struktur *routing* dan pola pengembangan aplikasi yang lebih kompleks. Dengan demikian, Next.js lebih sesuai digunakan sebagai *framework frontend* karena menyediakan struktur pengembangan yang lebih lengkap dan terintegrasi.

2. React dibandingkan Angular dan Vue.js

React merupakan pustaka JavaScript untuk membangun antarmuka pengguna berbasis komponen (React, 2026). Pendekatan berbasis komponen memungkinkan



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

pengembang membagi tampilan menjadi elemen-elemen kecil yang dapat digunakan ulang, seperti tabel, *form*, *card*, *badge*, *modal*, dan komponen navigasi. Pendekatan berbasis komponen tersebut sesuai karena sistem yang dikembangkan memiliki banyak komponen antarmuka yang berulang, seperti tabel tugas, *badge readiness*, daftar *project*, tampilan *critical path*, dan *form* pengelolaan tugas (Puan Maharani, 2025).

Angular juga dapat digunakan untuk membangun aplikasi web karena menyediakan *framework* yang lebih *opinionated* dan terstruktur, termasuk dukungan *dependency injection*(Angular, 2026). Namun, pada konteks penelitian ini, penggunaan Angular dapat menambah kompleksitas pengembangan karena sistem lebih membutuhkan fleksibilitas komponen antarmuka dan pengembangan fitur secara bertahap.

Vue.js menawarkan pendekatan yang sederhana dan fleksibel untuk membangun antarmuka web(Vue.js, 2026). Namun, React dipilih karena memiliki integrasi langsung dengan Next.js serta mendukung pola komponen yang sesuai dengan kebutuhan pengembangan *dashboard* monitoring.

Menurut (Puan Maharani, 2025)React dipilih sebagai dasar pengembangan antarmuka karena mendukung penyusunan UI yang modular, *reusable*, dan mudah dikembangkan secara bertahap. Kesesuaian React dengan Next.js juga menjadi pertimbangan penting karena keduanya digunakan untuk membangun struktur *frontend* secara terintegrasi.

3. Tailwind CSS dibandingkan CSS konvensional

Tailwind CSS merupakan *utility-first CSS framework* yang menyediakan kelas-kelas utilitas untuk membangun tampilan antarmuka secara langsung pada elemen UI(TailwindCSS, 2026). Berbeda dengan CSS konvensional yang umumnya membutuhkan penulisan aturan *style* secara terpisah dalam berkas tersendiri, Tailwind CSS memungkinkan pengembang menyusun tampilan melalui kombinasi *class utility* yang konsisten langsung pada elemen HTML(Benz and Testiana, 2025). Pendekatan ini mengurangi kebutuhan untuk membuat banyak berkas *style* terpisah, sehingga pengembangan antarmuka dapat dilakukan lebih cepat dan konsisten.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Pada sistem yang dikembangkan, Tailwind CSS digunakan untuk membangun tampilan seperti *dashboard*, tabel CPM, *task board*, *badge readiness*, *form* pengelolaan tugas, dan komponen navigasi. Penggunaan Tailwind CSS membantu menjaga konsistensi *layout*, *spacing*, warna, dan responsivitas antarmuka tanpa perlu membuat banyak berkas CSS terpisah. Oleh karena itu, Tailwind CSS dipilih karena sesuai dengan kebutuhan sistem yang memiliki banyak halaman berbasis *role* dan komponen visual yang digunakan secara berulang.

Berdasarkan pertimbangan kebutuhan pengembangan antarmuka, kombinasi Next.js, React, dan Tailwind CSS dipilih sebagai teknologi *frontend* utama. Next.js menyediakan struktur *routing* dan manajemen halaman yang sesuai untuk sistem dengan banyak fitur yang saling terhubung. React mendukung pengembangan antarmuka berbasis komponen yang modular dan *reusable*. Tailwind CSS memastikan konsistensi *style*, *layout*, dan responsivitas antarmuka secara efisien. Ketiga teknologi ini digunakan secara terintegrasi untuk membangun antarmuka sistem yang terstruktur, mudah dikembangkan, dan konsisten di seluruh halaman.

2.1.9 Bahasa Pemrograman Go dan Framework Gin

Pengembangan *backend* sistem yang dikembangkan membutuhkan bahasa pemrograman dan *framework* yang mampu mendukung pembuatan REST API, pengelolaan autentikasi, pemrosesan aturan bisnis, integrasi eksternal, serta perhitungan Critical Path Method (CPM) secara deterministik. *Backend* berperan sebagai lapisan utama yang mengelola data *project*, *phase*, tugas, *dependency*, *approval*, integrasi GitHub, autentikasi JWT, dan dokumentasi API. Oleh karena itu, pemilihan bahasa pemrograman dan *framework backend* dilakukan dengan mempertimbangkan stabilitas implementasi, kemampuan pemrosesan aturan bisnis, dan kesesuaian untuk membangun layanan REST API yang terstruktur.

1. Go dibandingkan Node.js

Go merupakan bahasa pemrograman yang dirancang untuk membangun perangkat lunak yang sederhana, efisien, dan mendukung pemrosesan konkuren melalui *goroutine* (Uptrace, 2026). Mekanisme *goroutine* memungkinkan Go menjalankan beberapa proses secara bersamaan tanpa memerlukan overhead yang besar,



sehingga mendukung *concurrency* secara bawaan (Go Programming Language, 2026). Karakteristik ini sesuai untuk membangun layanan *backend* yang membutuhkan pemrosesan aturan bisnis, REST API, serta perhitungan CPM secara stabil.

Sebagai perbandingan, Node.js merupakan *runtime* JavaScript yang banyak digunakan untuk membangun aplikasi *backend* berbasis *event-driven* dan *non-blocking I/O* (Node, 2026). Node.js cocok untuk aplikasi yang membutuhkan banyak operasi I/O secara bersamaan, namun pada penelitian ini Go dipilih karena struktur bahasa yang eksplisit, performa yang stabil, dan kesesuaian untuk membangun layanan *backend* dengan logika bisnis yang deterministik. Go dipilih karena sistem yang dikembangkan membutuhkan *backend* yang dapat menangani pengelolaan data *project*, validasi *dependency*, *approval workflow*, perhitungan CPM, dan integrasi eksternal secara konsisten. Node.js tetap dapat digunakan untuk membangun REST API, tetapi Go lebih sesuai untuk kebutuhan penelitian ini karena memberikan struktur implementasi yang lebih eksplisit dan mendukung pemrosesan *concurrency* secara bawaan.

2. Go dibandingkan Python

Menurut Alfarizi M. R. S. et al. (2023) python merupakan bahasa pemrograman yang banyak digunakan karena sintaksnya sederhana dan ekosistemnya luas, terutama untuk analisis data, otomasi, dan pengembangan aplikasi berbasis kecerdasan buatan. Penggunaan Python untuk berbagai *domain* tersebut telah banyak didokumentasikan dalam literatur ilmiah (Alfarizi M. R. S. et al., 2023). Namun, dalam penelitian ini, kebutuhan utama *backend* adalah menjalankan REST API, validasi data, pengelolaan *workflow*, dan perhitungan CPM secara konsisten.

Go dipilih karena lebih sesuai untuk membangun layanan *backend* yang terstruktur dan efisien. Dalam sistem yang dikembangkan, perhitungan CPM membutuhkan proses yang deterministik berdasarkan data tugas, durasi, dan *dependency*. Oleh karena itu, Go digunakan untuk menjaga struktur implementasi tetap eksplisit, mudah diuji, dan sesuai dengan kebutuhan layanan *backend*. Python memiliki keunggulan pada kemudahan penulisan kode dan ekosistem pustaka yang luas, tetapi Go lebih sesuai digunakan pada penelitian ini karena sistem membutuhkan



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

layanan *backend* yang stabil, terstruktur, dan berorientasi pada pengelolaan REST API serta aturan bisnis.

3. Gin dibandingkan Echo atau Fiber

Gin merupakan *framework* web untuk Go yang menyediakan fitur untuk membangun layanan HTTP, seperti *routing*, *middleware*, *request binding*, dan *response handling*(Gin, 2026). Fitur tersebut sesuai untuk membangun REST API pada sistem ini, terutama untuk *endpoint* autentikasi, *project*, *phase*, tugas, *approval*, CPM, *repository* GitHub, dan dokumentasi Swagger.

Sebagai perbandingan, Echo juga merupakan *framework backend* untuk Go yang menyediakan fitur *routing*, *middleware*, dan dukungan pengembangan API(Echo, 2026). Fiber dikenal sebagai *framework* yang berorientasi pada performa dan memiliki gaya penggunaan yang mirip dengan Express.js (Fiber, 2026). Namun, pada penelitian ini Gin dipilih karena sederhana, ringan, cukup populer dalam ekosistem Go, dan sesuai untuk kebutuhan REST API sistem yang dikembangkan(Encore, 2026).

Gin dipilih karena sistem yang dikembangkan membutuhkan *framework backend* yang dapat mendukung pemisahan lapisan *controller*, *service*, *repository*, dan *domain*. Dengan Gin, implementasi *endpoint* dapat disusun secara jelas sehingga proses autentikasi, validasi, pengelolaan data *project*, dan perhitungan CPM dapat dikelola secara lebih terstruktur.

Berdasarkan pertimbangan kebutuhan pengembangan *backend* pada penelitian ini, kombinasi Go dan Gin dipilih sebagai teknologi *backend* utama. Go menyediakan struktur bahasa yang eksplisit, mendukung *concurrency* melalui *goroutine*, dan sesuai untuk membangun layanan *backend* dengan logika bisnis yang deterministik. Gin sebagai *framework backend* melengkapi Go dengan menyediakan fitur *routing*, *middleware*, dan pengelolaan *endpoint* yang memudahkan pengembangan REST API secara terstruktur. Kedua teknologi ini digunakan secara terintegrasi untuk membangun lapisan *backend* sistem yang stabil, terstruktur, dan mudah dipelihara.

2.1.10 *RESTful* API dan Dokumentasi Swagger

Sistem yang dikembangkan membutuhkan mekanisme komunikasi data yang jelas antara *frontend* dan *backend*. *RESTful* API digunakan sebagai pendekatan komunikasi antara antarmuka Next.js dengan layanan *backend* Go/Gin melalui *endpoint* HTTP. Dokumentasi Swagger digunakan agar setiap *endpoint*, parameter, metode HTTP, format *request*, dan format *response* dapat terdokumentasi secara terstruktur, sehingga pengembangan dan pengujian sistem dapat dilakukan secara lebih sistematis.

1. *RESTful* API dibandingkan GraphQL

RESTful API merupakan gaya arsitektur yang menggunakan *resource* sebagai pusat pertukaran data dan memanfaatkan metode HTTP seperti GET, POST, PUT, PATCH, dan DELETE (AWS, 2026). Setiap *resource* pada *RESTful* API dapat diakses melalui *endpoint* yang terdefinisi, sehingga alur komunikasi antara *client* dan *server* dapat dipetakan secara eksplisit (Ikhwan, 2026). Pendekatan ini sesuai untuk sistem yang dibangun karena memiliki *resource* yang jelas, seperti *project*, *phase*, tugas, *dependency*, *approval*, CPM, pengguna, dan aktivitas GitHub.

Sebagai perbandingan, GraphQL merupakan pendekatan API yang memungkinkan *client* menentukan struktur data yang ingin diambil melalui satu *endpoint* (GraphQL, 2026). GraphQL dapat mengurangi *over-fetching* atau *under-fetching* pada sisi *client*, namun membutuhkan skema, resolver, dan pengelolaan query yang lebih kompleks. *RESTful* API dipilih karena kebutuhan sistem ini lebih sesuai dengan struktur *resource* yang jelas dan operasi CRUD yang terpisah berdasarkan fungsi sistem. *RESTful* API juga lebih mudah diintegrasikan dengan dokumentasi Swagger dan pengujian *endpoint* menggunakan *tools* seperti Postman. GraphQL tetap dapat digunakan untuk kebutuhan pengambilan data yang lebih fleksibel, tetapi pada penelitian ini *RESTful* API lebih sesuai karena alur komunikasi sistem dapat dipetakan secara eksplisit melalui metode HTTP dan *endpoint* yang terdokumentasi.

2. Swagger dibandingkan dokumentasi API manual

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Swagger atau OpenAPI digunakan untuk mendeskripsikan API secara terstruktur, termasuk daftar *endpoint*, parameter, metode HTTP, format *request*, format *response*, dan skema data (OpenAPI, 2026). Dokumentasi yang terstruktur penting dalam pengembangan sistem ini karena *frontend* dan *backend* perlu memiliki acuan yang sama dalam proses integrasi. Dengan Swagger, pengembang dapat melihat spesifikasi *endpoint* secara langsung dan melakukan pengujian awal terhadap *request* dan *response* tanpa harus membaca kode sumber (Swagger, 2026).

Sebagai perbandingan, dokumentasi API manual yang biasanya dibuat dalam dokumen terpisah tetap dapat digunakan, tetapi lebih rentan tidak sinkron ketika terjadi perubahan *endpoint*, parameter, atau format data. Menurut Firdaus and Afwani (2024) pada sistem yang memiliki banyak fitur seperti autentikasi, *project*, *phase*, tugas, *approval*, CPM, dan integrasi GitHub, dokumentasi yang terstruktur membantu proses pengujian dan pemeliharaan sistem. Swagger dipilih karena dapat membantu mendokumentasikan API sistem secara lebih konsisten dan mudah diuji.

2.1.11 PostgreSQL, Bun ORM, dan Penyimpanan Data

Sistem yang dikembangkan membutuhkan penyimpanan data yang mampu menjaga relasi antarentitas seperti *project*, *phase*, tugas, *dependency*, *approval*, pengguna, aktivitas GitHub, dan *snapshot* hasil perhitungan CPM. PostgreSQL digunakan sebagai sistem manajemen basis data relasional, Bun ORM digunakan untuk memetakan query database pada struktur data Go, dan penyimpanan objek digunakan untuk kebutuhan file seperti avatar atau lampiran jika tersedia pada implementasi sistem. Kombinasi ketiga komponen ini dipilih agar pengelolaan data sistem dapat dilakukan secara terstruktur dan konsisten.

1. PostgreSQL dibandingkan MySQL

PostgreSQL merupakan sistem manajemen basis data relasional yang mendukung SQL, transaksi, integritas referensial, indeks, serta tipe data lanjutan seperti JSONB (PostgreSQL, 2026). Dukungan terhadap JSONB memungkinkan penyimpanan data semi-terstruktur seperti konfigurasi atau *snapshot* hasil perhitungan dalam satu kolom (PostgreSQL, 2026). Karakteristik tersebut relevan untuk sistem yang dikembangkan karena data *project*, *phase*, tugas, *dependency*, *approval*, *snapshot*

CPM, dan aktivitas teknis memiliki relasi yang kuat serta membutuhkan konsistensi data. MySQL juga merupakan sistem manajemen basis data relasional yang banyak digunakan untuk aplikasi web (MySQL, 2026). MySQL tetap dapat digunakan untuk menyimpan data relasional, tetapi PostgreSQL dipilih karena sistem yang dibangun membutuhkan pemodelan relasi yang kuat, transaksi, dan dukungan tipe data lanjutan untuk menyimpan data yang lebih fleksibel. Menurut Aji Bayu Prasetyo and Alfa Susetyo (2022) PostgreSQL lebih sesuai dengan kebutuhan sistem ini yang mengelola banyak relasi antarentitas dan membutuhkan konsistensi data, terutama pada proses validasi *dependency*, perubahan status tugas, *approval workflow*, dan penyimpanan hasil perhitungan CPM.

2. Bun ORM dibandingkan GORM

Bun ORM merupakan ORM untuk Go yang menggunakan pendekatan SQL-first, sehingga pengembang tetap dapat mengontrol query SQL dengan lebih eksplisit sambil memetakan hasil query ke struktur data Go (Uptrace, 2026). Pendekatan ini sesuai untuk sistem yang dikembangkan karena beberapa proses membutuhkan query yang jelas, terutama pada pengambilan data tugas, *dependency*, *approval*, dan *snapshot* CPM. Sebagai perbandingan, GORM merupakan ORM Go yang populer dan menyediakan abstraksi pemodelan data yang lebih tinggi (Gorm, 2026). GORM dapat mempercepat pengembangan dengan abstraksi yang lebih tinggi, tetapi pada konteks sistem yang dikembangkan, Bun ORM dipilih karena lebih sesuai dengan kebutuhan pengendalian query yang eksplisit dan struktur *repository* yang terpisah dari lapisan *service* maupun *domain*. Bun ORM membantu implementasi lapisan *repository* dalam membaca dan menulis data tanpa menghilangkan kontrol terhadap query SQL yang digunakan.

2.1.12 Autentikasi JWT dan Role-Based Access Control

Sistem yang dikembangkan membutuhkan mekanisme keamanan untuk memastikan setiap pengguna hanya dapat mengakses fitur sesuai perannya. JWT digunakan untuk autentikasi pengguna dan membawa klaim identitas, sedangkan Role-Based Access Control digunakan untuk mengatur otorisasi berdasarkan *role* seperti *Project Manager*, *Lead Developer*, dan *Software Developer*. Pengendalian akses penting karena sistem mengelola data *project*, tugas, *approval*, *dependency*,

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

dan hasil perhitungan CPM yang hanya boleh diakses oleh pengguna dengan kewenangan yang sesuai.

2.1.13 Black Box Testing dan User Acceptance Testing

Black Box Testing merupakan teknik pengujian yang menilai perilaku sistem berdasarkan *input*, kondisi, dan *output* tanpa melihat struktur internal kode. mendefinisikan black-box test technique sebagai teknik yang didasarkan pada analisis spesifikasi komponen atau sistem (Muhammad Arofiq et al., 2023). Pada penelitian ini, *Black Box Testing* digunakan untuk memeriksa apakah fitur utama sistem berjalan sesuai kebutuhan fungsional, seperti *login*, *role access*, pengelolaan *project*, tugas, *dependency*, *approval*, CPM *preview*, CPM *authoritative*, *snapshot*, dan aktivitas teknis.

Acceptance Testing merupakan pengujian formal untuk menentukan apakah suatu sistem memenuhi kriteria penerimaan dan memungkinkan pengguna atau pihak terkait memutuskan penerimaan sistem (Abraham and Ismail, 2021). *User Acceptance Testing* (UAT) pada penelitian ini digunakan untuk menilai kesesuaian sistem terhadap kebutuhan *Project Manager*, *Lead Developer*, dan *Software Developer*. Skenario UAT disusun berdasarkan tugas masing-masing *role* agar evaluasi tidak hanya bersifat teknis, tetapi juga sesuai dengan alur kerja pengguna.

2.1.14 Usability Testing dan System Usability Scale

Usability Testing merupakan pengujian yang digunakan untuk mengevaluasi kemudahan penggunaan suatu sistem berdasarkan pengalaman pengguna setelah mencoba aplikasi. Salah satu metode *usability testing* yang banyak digunakan adalah *System Usability Scale* (SUS). SUS merupakan metode evaluasi *usability* yang melibatkan pengguna akhir secara langsung untuk menilai sistem yang telah dibangun (Raharja and Pramudya, 2022).

Pengujian SUS dilakukan menggunakan kuesioner yang terdiri atas sepuluh pernyataan dengan skala Likert 1 (Sangat Tidak Setuju) sampai 5 (Sangat Setuju). Perhitungan skor dilakukan pada setiap pernyataan. Untuk pernyataan bernomor ganjil, skor kontribusi diperoleh dari nilai jawaban dikurangi 1, sedangkan untuk pernyataan bernomor genap, skor kontribusi diperoleh dari 5 dikurangi nilai

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

jawaban. Seluruh skor kontribusi dari sepuluh pernyataan dijumlahkan, kemudian dikalikan 2,5 sehingga menghasilkan skor SUS pada rentang 0 sampai 100.

Hasil akhir SUS diinterpretasikan melalui tiga aspek, yaitu *Acceptability*, *Grade Scale*, dan *Adjective Rating*. *Acceptability* menunjukkan tingkat penerimaan sistem oleh pengguna, dengan rentang Not Acceptable, Marginal, dan Acceptable sebagaimana disajikan pada Tabel 1. Selain itu, hasil SUS juga dipetakan ke dalam nilai huruf melalui *Grade Scale* dan predikat kualitas *usability* melalui *Adjective Rating* sebagaimana disajikan pada Tabel 1. Kombinasi ketiga aspek tersebut divisualisasikan dalam skala penilaian SUS yang menjadi acuan interpretasi hasil pengujian pada penelitian ini .

Tabel 1 Rentang *Acceptability System Usability Scale*

Rentang Skor	Acceptability
0 – 50	Not Acceptable
50 – 70	Marginal
70 – 100	Acceptable

Selain *Acceptability*, hasil SUS juga dapat diinterpretasikan menggunakan *Grade Scale* dan *Adjective Rating*. *Grade Scale* digunakan untuk memetakan skor ke dalam nilai huruf, sedangkan *Adjective Rating* digunakan untuk memberikan predikat kualitas *usability*. Interpretasi *Grade Scale* dan *Adjective Rating* dapat dilihat pada Tabel 2.

Tabel 2 Grade Scale dan Adjective Rating System Usability Scale

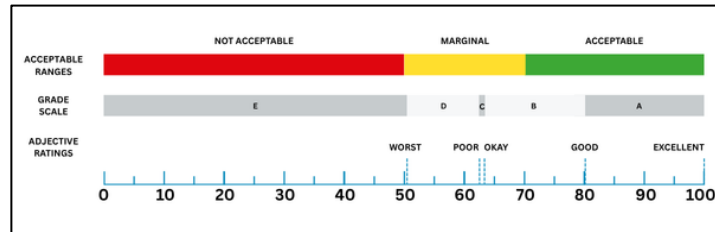
Rentang Skor	Grade	Adjective Rating
> 80,2	A	Excellent
68 – 80,2	B	Good
67	C	Okay
51 – 66	D	Poor
< 51	E	Worst

Gabungan rentang *Acceptability*, *Grade Scale*, dan *Adjective Rating* divisualisasikan dalam skala penilaian SUS pada Gambar 2.1. Gambar tersebut digunakan sebagai acuan interpretasi hasil *Usability Testing* pada penelitian ini.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 2. 1 Skala Penilaian *System Usability Scale*

Sumber: Olahan Peneliti berdasarkan Pramudya and Raharja (2022) dan (Ismail et al., 2021)

2.2 Penelitian Terdahulu

Penelitian terdahulu digunakan untuk memahami posisi penelitian ini terhadap studi yang telah ada, khususnya yang berkaitan dengan sistem manajemen proyek, *Critical Path Method*, perubahan *jalur kritis*, dan dukungan otomatis pada analisis *requirement*. Ringkasan penelitian terdahulu dapat dilihat pada Tabel 3.

Tabel 3 Penelitian Terdahulu

Peneliti/Tahun	Fokus Penelitian	Metode/Teknologi	Hasil Utama	Relevansi dan Gap
Astari, Subagyo, dan Kusnadi (2021)	Perencanaan proyek Museum XYZ untuk mengatasi keterlambatan waktu dan mengukur dampak biaya proyek.	CPM, PERT, WBS, Kurva S, Gantt Chart, dan analisis sensitivitas.	Penelitian menghasilkan lintasan kritis dan perbandingan durasi proyek menggunakan CPM dan PERT.	Relevan sebagai dasar penggunaan CPM untuk menentukan lintasan kritis. Namun, penelitian masih berfokus pada proyek konstruksi dan belum berupa sistem monitoring proyek perangkat lunak berbasis web.
Ren & Li (2023)	Perencanaan jadwal proyek perangkat lunak dengan	PERT, CPM, network chart, PDM, ADM, dan	Penggunaan metode perencanaan jadwal dapat	Relevan karena membahas CPM pada konteks proyek perangkat

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Peneliti/Tahun	Fokus Penelitian	Metode/Teknologi	Hasil Utama	Relevansi dan Gap
	memperhatikan aktivitas, durasi, dan ketergantungan pekerjaan.	duration compression.	membantu meningkatkan akurasi rencana proyek perangkat lunak.	lunak. Namun, penelitian ini masih bersifat kajian metode, belum mengembangkan sistem monitoring dengan fitur tugas, phase, dan dependency.
Agustha et al. (2024)	Pengembangan aplikasi monitoring informasi proyek berbasis web untuk mengatasi laporan manual dan data proyek yang belum terintegrasi.	ICONIX Process, UML, Laravel 10, MVC, dan Black Box Testing.	Aplikasi monitoring proyek berhasil dikembangkan dan dinyatakan sesuai kebutuhan pengguna berdasarkan pengujian black box.	Relevan karena sama-sama membahas aplikasi monitoring proyek berbasis web. Namun, penelitian ini belum membahas identifikasi tugas kritis berdasarkan estimasi durasi dan dependency antar tugas.
Erwin et al. (2025)	Penerapan CPM pada manajemen servis genset untuk mengetahui lintasan kritis dan efisiensi waktu pengerjaan.	CPM dengan bantuan Microsoft Project.	Penelitian menghasilkan lintasan kritis berdurasi 38 hari dan efisiensi waktu pengerjaan selama 8 hari.	Relevan untuk menunjukkan fungsi CPM dalam menentukan aktivitas kritis. Namun, objek penelitian bukan proyek perangkat lunak dan belum diimplementasikan dalam sistem monitoring berbasis web.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Peneliti/Tahun	Fokus Penelitian	Metode/Teknologi	Hasil Utama	Relevansi dan Gap
Agustin & Kustiana (2025)	Rancang bangun sistem manajemen proyek perangkat lunak berbasis web untuk mengatasi penundaan jadwal, koordinasi tim, dan pelacakan progres yang masih semi-manual.	Rational Unified Process, UML, Black Box Testing, dan System Usability Scale.	Sistem berhasil dikembangkan, seluruh fungsi utama valid, dan hasil SUS menunjukkan sistem berada pada kategori baik.	Sangat relevan karena membahas sistem manajemen proyek perangkat lunak berbasis web. Namun, penelitian ini belum secara khusus membahas identifikasi tugas kritis berbasis dependency dan integrasi aktivitas teknis developer.

Berdasarkan penelitian terdahulu pada Tabel 1, dapat disimpulkan bahwa kajian mengenai manajemen proyek umumnya terbagi ke dalam dua fokus utama. Pertama, penelitian yang menerapkan metode *CPM* atau *PERT* untuk menentukan lintasan kritis, menghitung durasi proyek, serta membantu perencanaan dan pengendalian waktu proyek. Kedua, penelitian yang mengembangkan sistem monitoring atau sistem manajemen proyek berbasis web untuk meningkatkan pencatatan, pemantauan progres, integrasi data, dan transparansi informasi proyek. Hasil penelitian terdahulu menunjukkan bahwa metode *CPM* efektif digunakan untuk mengidentifikasi aktivitas kritis, sedangkan sistem berbasis web dapat membantu perusahaan dalam mengelola informasi proyek secara lebih terstruktur.

Berdasarkan Relevansi atau Gap yang telah dijelaskan pada Tabel 1, pembeda utama penelitian ini terletak pada penggabungan sistem *monitoring* proyek perangkat lunak berbasis *web* dengan identifikasi *tugas* kritis menggunakan *Critical Path Method* (CPM) sebagai fitur utama. Penelitian terdahulu yang menerapkan CPM atau PERT umumnya berfokus pada perhitungan lintasan kritis dan durasi proyek, tetapi belum mengintegrasikannya ke dalam sistem *monitoring*



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

proyek perangkat lunak yang mengelola proyek, *phase*, *tugas*, dan *dependency* secara terpusat. Sementara itu, penelitian yang mengembangkan sistem *monitoring* proyek berbasis *web* umumnya masih berfokus pada pencatatan data, pemantauan progres, dan penyajian informasi proyek, tetapi belum secara khusus menampilkan *tugas* yang berdampak langsung terhadap estimasi penyelesaian proyek. Oleh karena itu, penelitian ini tidak hanya berfokus pada pengelolaan data proyek, tetapi juga menghitung dan menampilkan *tugas* kritis berdasarkan estimasi durasi dan hubungan *dependency* antar *tugas* pada level proyek, sehingga dapat membantu pengguna memahami pekerjaan yang berpengaruh langsung terhadap estimasi penyelesaian proyek.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB III METODE PENELITIAN

3.1 Rancangan Penelitian

Penelitian ini menggunakan pendekatan *mixed method* (kualitatif dan kuantitatif) dengan jenis penelitian studi kasus (*case study*) pada PT Lokasolusi Teknologi Indonesia. Pendekatan ini dipilih karena penelitian dijalankan dalam dua tahap yang saling melengkapi (*sequential exploratory*). Tahap kualitatif dilakukan pada awal penelitian untuk memahami alur kerja, peran pengguna, serta kebutuhan sistem secara mendalam melalui observasi, wawancara, dan studi dokumentasi, yang kemudian menjadi dasar perancangan dan pembangunan sistem. Tahap kuantitatif dilakukan pada akhir penelitian untuk mengukur kualitas sistem yang dihasilkan melalui data terukur, yaitu tingkat keberhasilan *black box testing* dan *User Acceptance Testing* (UAT) serta skor *System Usability Scale* (SUS). Jenis studi kasus digunakan karena penelitian difokuskan pada satu objek spesifik, yaitu proses pengelolaan dan pemantauan proyek perangkat lunak di PT Lokasolusi Teknologi Indonesia. Karena luaran penelitian berupa sistem, proses pengembangan perangkat lunak mengikuti model *Waterfall*, yaitu tahapan yang dikerjakan secara berurutan mulai dari analisis kebutuhan, perancangan, implementasi, hingga pengujian.

3.1.1 Teknik Pengumpulan Data

Data dikumpulkan melalui tiga teknik. Pertama, observasi terhadap proses *monitoring* proyek yang sedang berjalan di PT Lokasolusi Teknologi Indonesia. Kedua, wawancara semi-terstruktur dengan *Project Manager*, *Lead Developer*, dan *Software Developer* untuk menggali alur kerja, pembagian peran, pengelolaan *project*, *phase*, dan tugas, pengaturan *dependency*, serta pemantauan progres. Ketiga, studi dokumentasi terhadap artefak kerja seperti dokumen *requirement*, catatan tugas, media koordinasi tim, dan aktivitas teknis pada *repository*.

3.1.2 Teknik Analisis Data

Data hasil pengumpulan dianalisis menggunakan dua cara sesuai pendekatan *mixed method*. Data kualitatif dari observasi, wawancara, dan studi dokumentasi dianalisis secara deskriptif kualitatif melalui tahap transkripsi, reduksi data, penyajian data, dan penarikan kesimpulan untuk memetakan proses bisnis berjalan,

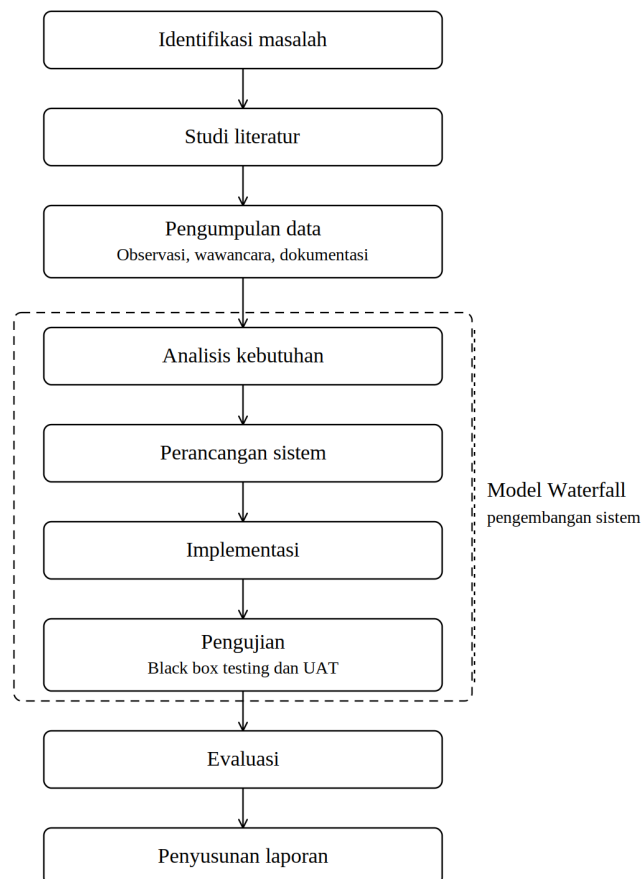
Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

mengidentifikasi kendala, serta merumuskan kebutuhan fungsional dan nonfungsional sistem. Khusus untuk identifikasi tugas kritis, data tugas, estimasi durasi, dan hubungan *dependency* dianalisis menggunakan *Critical Path Method* (CPM) untuk menentukan jalur kritis pada level *project*. Adapun data kuantitatif dianalisis secara statistik deskriptif, yaitu menghitung persentase keberhasilan *black box testing* dan *User Acceptance Testing* (UAT) terhadap seluruh skenario, serta menghitung skor *System Usability Scale* (SUS) beserta nilai terendah, tertinggi, dan rata-rata untuk menilai tingkat kemudahan penggunaan sistem.

3.2 Tahapan Penelitian

Tahapan penelitian merupakan langkah-langkah yang dilakukan secara berurutan dari awal hingga akhir penelitian, dengan setiap tahap menjadi landasan bagi tahap berikutnya. Alur tahapan penelitian ditunjukkan pada Gambar 3.1.



Gambar 3. 1 Tahapan Penelitian



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

3.2.1 Identifikasi Masalah

Tahap identifikasi masalah dilakukan melalui observasi awal dan wawancara untuk memahami kendala *monitoring* proyek di PT Lokasolusi Teknologi Indonesia. Keluaran tahap ini berupa daftar permasalahan utama dan ruang lingkup penelitian.

3.2.2 Studi Literatur

Tahap studi literatur dilakukan untuk mengumpulkan dasar teori yang relevan, meliputi *monitoring* proyek perangkat lunak, pengelolaan *dependency*, *Critical Path Method*, serta penelitian terdahulu yang berkaitan. Keluaran tahap ini berupa landasan teori yang digunakan dalam analisis dan perancangan.

3.2.3 Pengumpulan Data

Tahap pengumpulan data dilakukan untuk memperoleh data proses bisnis yang sedang berjalan melalui observasi, wawancara semi-terstruktur, dan studi dokumentasi. Keluaran tahap ini berupa data kondisi kerja *existing* sebagai dasar analisis kebutuhan.

3.2.4 Analisis Kebutuhan

Tahap analisis kebutuhan dilakukan untuk memetakan proses bisnis berjalan dan merumuskan kebutuhan sistem. Keluaran tahap ini berupa kebutuhan fungsional dan nonfungsional serta spesifikasi awal sistem.

3.2.5 Perancangan Sistem

Tahap perancangan sistem dilakukan untuk menerjemahkan kebutuhan ke dalam rancangan teknis, meliputi arsitektur sistem, basis data, antarmuka pengguna, dan mekanisme identifikasi tugas kritis menggunakan *Critical Path Method*. Keluaran tahap ini berupa artefak desain yang menjadi dasar implementasi.

3.2.6 Implementasi

Tahap implementasi dilakukan dengan membangun sistem sesuai rancangan, mencakup pengelolaan *project*, *phase*, dan tugas, *approval workflow*, perhitungan CPM, serta integrasi aktivitas teknis *repository* sebagai fitur pendukung. Keluaran tahap ini berupa *prototype* sistem yang siap diuji.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

3.2.7 Pengujian

Tahap pengujian dilakukan untuk memverifikasi kesesuaian sistem terhadap kebutuhan melalui *black box testing*, *User Acceptance Testing* (UAT) dan *System Usability Scale* (SUS). Keluaran tahap ini berupa hasil pengujian dan catatan perbaikan.

3.2.8 Evaluasi

Tahap evaluasi dilakukan untuk menilai kesesuaian sistem terhadap kebutuhan pengguna berdasarkan hasil pengujian. Keluaran tahap ini berupa rekomendasi penyempurnaan sistem.

3.2.9 Penyusunan Laporan

Tahap penyusunan laporan merupakan tahap akhir yang mendokumentasikan seluruh proses penelitian ke dalam bentuk skripsi. Keluaran tahap ini berupa laporan penelitian beserta lampiran pendukung.

3.3 Objek Penelitian

Objek penelitian ini adalah PT Lokasolusi Teknologi Indonesia, perusahaan pengembang perangkat lunak yang mengelola beberapa proyek sekaligus dan menghadapi kendala dalam *monitoring* progres serta pengelolaan *dependency* antar-tugas. Secara lebih spesifik, sasaran penelitian difokuskan pada proses pengelolaan dan pemantauan proyek perangkat lunak di perusahaan tersebut beserta penggunanya, yaitu *Project Manager*, *Lead Developer*, dan *Software Developer*, yang menjadi sumber kebutuhan sekaligus pengguna sistem yang dirancang.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB IV HASIL DAN PEMBAHASAN

4.1 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem dilakukan untuk memastikan sistem *monitoring* proyek perangkat lunak yang dikembangkan mampu menjawab permasalahan tim di PT Lokasolusi Teknologi Indonesia. Analisis disusun berdasarkan wawancara dengan *Project Manager*, *Lead Developer*, dan *Software Developer*, studi dokumentasi, serta hasil pemetaan proses kerja. Hasil analisis digunakan untuk merumuskan kebutuhan sistem yang mendukung pengelolaan *project*, *phase*, tugas, *dependency*, *approval*, *monitoring* aktivitas teknis, dan identifikasi *tugas kritis* menggunakan *Critical Path Method* (CPM).

4.1.1 Analisis Hasil Wawancara dengan *Project Manager*

Wawancara dilakukan dengan Sita Amira Syarifah selaku *Project Manager* PT Lokasolusi Teknologi Indonesia. Narasumber bertanggung jawab memastikan proyek berjalan sesuai target dan kebutuhan bisnis, mulai dari penerimaan proyek, validasi *backlog*, pembagian pekerjaan, hingga pelaporan progres. Proyek pada umumnya diterima terlebih dahulu oleh *Chief Executive Officer*, kemudian ditugaskan kepada *Project Manager* untuk direncanakan dan dikoordinasikan kepada *Lead Developer* dan *Software Developer*. Dokumentasi hasil wawancara dapat dilihat pada Lampiran 1, sedangkan validasi lanjutan dari *Project Manager* dapat dilihat pada Lampiran 6.

Hasil wawancara menunjukkan bahwa *monitoring* proyek masih dilakukan melalui media yang terfragmentasi, seperti *spreadsheet*, dokumen, *Trello*, chat, dan *GitHub*. *Project Manager* perlu melakukan *cross-check* ke beberapa sumber untuk mengetahui status aktual tugas. Selain itu, *dependency* antar tugas belum dipetakan secara eksplisit sehingga tugas yang terlihat kecil tetapi berdampak besar terhadap pekerjaan lain sering baru diketahui ketika proyek telah berjalan.

Kebutuhan utama dari sisi *Project Manager* adalah tersedianya sistem terpusat yang menampilkan *project*, *phase*, tugas, status, *dependency*, progres, dan tugas kritis. Informasi tersebut dibutuhkan agar *Project Manager* dapat memahami kondisi proyek dan mengambil keputusan berdasarkan data yang terstruktur. *Project*

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Manager juga mengelola beban proyek setiap anggota tim, yaitu *Software Developer* yang saat ini dibatasi maksimal tiga proyek dalam satu waktu, sedangkan *Lead Developer* masih bersifat tentatif dan belum diberlakukan batasan maksimal. Kondisi tersebut memperkuat kebutuhan sistem yang mampu menampilkan keterlibatan setiap anggota tim pada berbagai proyek secara terpusat.

4.1.2 Analisis Hasil Wawancara dengan *Lead Developer*

Wawancara dilakukan dengan Cahya Nugroho selaku *Lead Developer* PT Lokasolusi Teknologi Indonesia. *Lead Developer* bertanggung jawab menerjemahkan kebutuhan proyek menjadi technical tugas, memantau pekerjaan teknis, melakukan *review*, serta menghubungkan kebutuhan manajerial dengan pelaksanaan teknis. Dokumentasi hasil wawancara dapat dilihat pada Lampiran 2.

Hasil wawancara menunjukkan bahwa *dependency* antar tugas masih berada dalam pemahaman internal *Lead Developer* atau dibahas secara lisan. Tidak adanya dokumentasi *dependency* menyebabkan developer berpotensi memulai tugas yang sebenarnya masih menunggu tugas lain. *Lead Developer* juga perlu berpindah antara GitHub dan media komunikasi untuk memperoleh gambaran progres seluruh pekerjaan.

Kebutuhan utama dari sisi *Lead Developer* adalah fitur untuk membentuk tugas teknis lengkap dengan estimasi durasi, assignee, dan *dependency*, serta kemampuan sistem untuk menunjukkan readiness dan *jalur kritis* pada level project.

4.1.3 Analisis Hasil Wawancara dengan *Software Developer*

Wawancara dilakukan dengan Gema Hafiz Maulidy selaku Software Engineer PT Lokasolusi Teknologi Indonesia. *Software Developer* bertanggung jawab mengerjakan tugas teknis, memperbarui progres, dan mengelola aktivitas pengembangan melalui GitHub. Dokumentasi hasil wawancara dapat dilihat pada Lampiran 3.

Hasil wawancara menunjukkan bahwa penerimaan tugas dan pelaporan progres masih dilakukan melalui beberapa saluran. Developer tidak selalu memperoleh penanda eksplisit mengenai tugas yang sudah siap dikerjakan atau tugas yang masih



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

menunggu *predecessor*. Kondisi ini menyebabkan konfirmasi tambahan dan meningkatkan risiko pengerjaan tugas dalam kondisi blocked.

Kebutuhan utama dari sisi *Software Developer* adalah halaman tugas pribadi yang menampilkan detail tugas, status readiness, *dependency*, dan mekanisme pembaruan status yang dapat dipantau oleh *Project Manager* dan *Lead Developer*.

4.1.4 Analisis Hasil Observasi

Selain wawancara, dilakukan observasi langsung terhadap proses pengelolaan proyek di PT Lokasolusi Teknologi Indonesia. Sebagai perusahaan *software house*, PT Lokasolusi Teknologi Indonesia mengerjakan proyek perangkat lunak berbasis kebutuhan klien, sehingga setiap proyek berawal dari penugasan formal di tingkat manajemen. Berdasarkan hasil observasi, proses pengelolaan proyek diawali dengan pemberian arahan dan penugasan proyek oleh *Chief Executive Officer* (CEO) kepada *Project Manager*. Arahan tersebut mencakup tujuan proyek, kebutuhan klien, ruang lingkup pekerjaan, target penyelesaian, serta sumber daya yang akan digunakan. Setelah menerima penugasan tersebut, *Project Manager* bertanggung jawab merencanakan, mengoordinasikan, dan mengawasi pelaksanaan proyek bersama *Lead Developer* dan *Software Developer* hingga proyek selesai. Observasi ini menegaskan bahwa kebutuhan monitoring proyek berangkat dari alur penugasan berjenjang, dari CEO hingga tim pelaksana, yang selama ini belum terpantau secara terpusat.

4.1.5 Analisis Proses Bisnis Existing

Analisis proses bisnis existing dilakukan berdasarkan hasil wawancara dengan *Project Manager*, *Lead Developer*, dan *Software Developer* PT Lokasolusi Teknologi Indonesia. Kondisi saat ini menunjukkan bahwa pengelolaan proyek perangkat lunak masih mengandalkan media yang terfragmentasi, mencakup *spreadsheet*, dokumen, platform manajemen tugas seperti *Trello*, saluran komunikasi berbasis chat, serta *GitHub* untuk pencatatan aktivitas teknis. *Dependency* antar tugas belum dipetakan secara eksplisit, sehingga kesiapan pekerjaan ditentukan melalui konfirmasi manual. Tidak terdapat satu sistem terpusat yang merangkum seluruh informasi *project*, *phase*, tugas, dan status



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

pengerjaan dalam satu tampilan yang dapat diakses oleh seluruh pihak yang terlibat. Alur proses bisnis existing secara keseluruhan ditunjukkan pada Gambar 4.1.

Pada proses bisnis existing, terdapat tiga aktor utama yang terlibat, yaitu Project Manager, Lead Developer, dan Software Developer. Project Manager menerima kebutuhan dari klien atau pihak internal, menyusun ringkasan kebutuhan, kemudian menyimpannya pada media kerja terpisah seperti *spreadsheet*, dokumen, atau platform manajemen tugas. Kebutuhan tersebut selanjutnya didiskusikan dengan tim teknis dan diteruskan kepada Lead Developer untuk diterjemahkan menjadi *backlog* serta dibagi menjadi *tugas* teknis. Setelah *tugas* diberikan, Software Developer mengerjakan pekerjaan tersebut, melakukan *commit*, dan mengirimkan *pull request* melalui GitHub untuk ditinjau oleh Lead Developer.

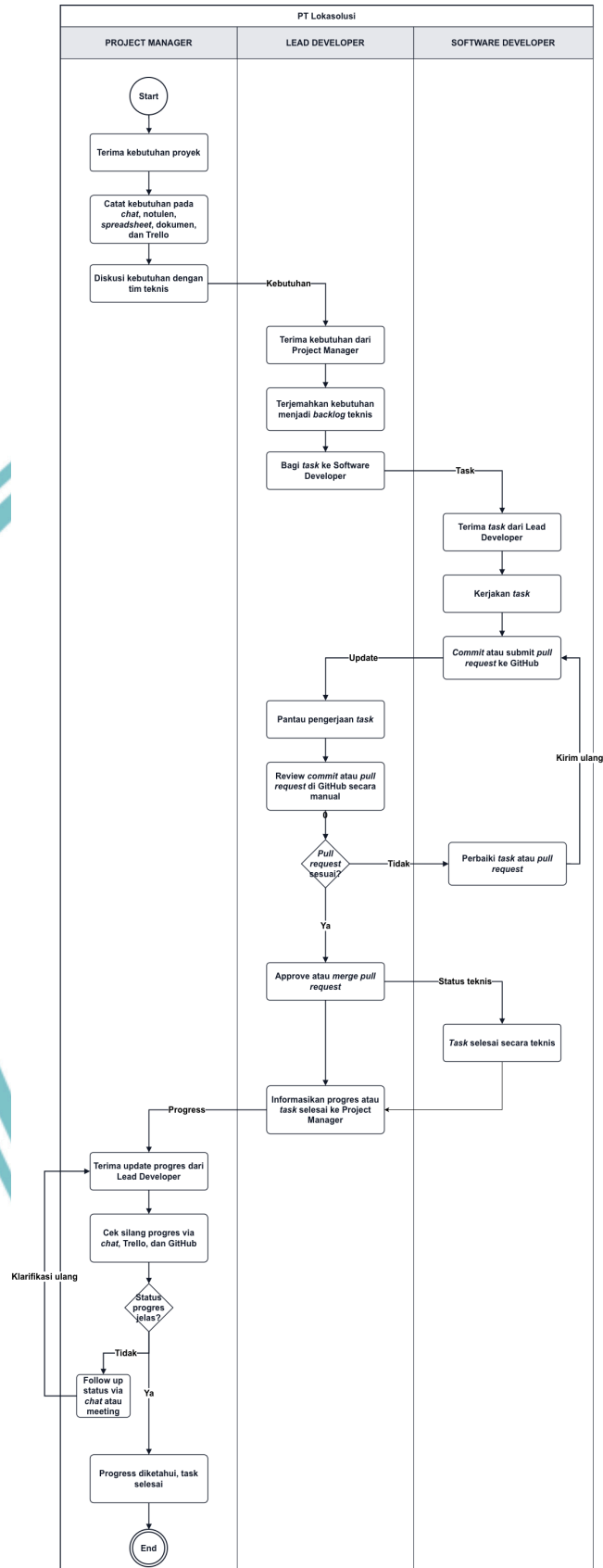
Apabila hasil pekerjaan belum sesuai, Lead Developer memberikan instruksi revisi kepada Software Developer hingga *tugas* dinyatakan sesuai dan dapat diterima. Setelah itu, progres pekerjaan diinformasikan kembali kepada Project Manager. Namun, karena belum tersedia sistem terpusat yang mencatat hubungan antara *project*, *phase*, *tugas*, *dependency*, dan status pengerjaan, proses pemantauan masih memerlukan konfirmasi manual. Hal ini menyebabkan validasi progres dan kejelasan status pekerjaan sangat bergantung pada komunikasi antar pihak. Alur tersebut divisualisasikan secara lebih rinci pada Gambar 4.1.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 4. 1 Proses Bisnis Existing

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.1 menunjukkan alur proses bisnis *existing* yang berjalan di PT Lokasolusi Teknologi Indonesia. Proses dimulai dari penerimaan kebutuhan oleh *Project Manager*, dilanjutkan dengan pembentukan daftar pekerjaan secara manual, kemudian pembagian tugas kepada tim melalui koordinasi lisan maupun *chat*, pelaksanaan oleh *Software Developer*, hingga pelaporan progres yang tersebar di berbagai media. Karena tidak terdapat mekanisme *approval workflow* yang terdokumentasi, validasi kelayakan tugas sebelum pengerjaan bergantung pada penilaian masing-masing individu. Kondisi ini mengakibatkan *Project Manager* dan *Lead Developer* harus melakukan *cross-check* ke beberapa sumber untuk memperoleh gambaran status proyek yang akurat dan terkini.

4.1.6 Analisis Proses Bisnis Future

Pada proses bisnis future, alur kerja dirancang agar proses monitoring proyek dapat berlangsung lebih terstruktur melalui sistem Prolytics. *Project Manager* memulai proses dengan memasukkan data utama proyek, seperti *project*, *start date*, dan *phase* ke dalam sistem. Data tersebut kemudian disimpan secara terpusat agar dapat digunakan oleh pihak yang terlibat dalam proses perencanaan dan pemantauan proyek. Setelah *project* tersedia, *Lead Developer* menyusun *technical tugas* dengan menentukan durasi, *assignee*, serta *dependency* antar *tugas*. Informasi tersebut menjadi dasar bagi sistem untuk melakukan validasi hubungan antar pekerjaan dan menghasilkan *CPM preview* sebelum tugas diajukan untuk disetujui.

Tahap berikutnya adalah peninjauan rencana *tugas* oleh *Project Manager*. Apabila *tugas* belum sesuai, *Project Manager* dapat meminta revisi sehingga *Lead Developer* perlu memperbaiki rencana *tugas* sebelum diajukan kembali. Apabila *tugas* disetujui, sistem akan menetapkan *tugas* sebagai *approved*, menghitung CPM aktif, serta menyimpan hasil persetujuan tersebut sebagai acuan pelaksanaan proyek. *Software Developer* kemudian mengerjakan *approved tugas*, memperbarui status pengerjaan, dan mengirimkan *pull request* untuk ditinjau oleh *Lead Developer*. Setelah pekerjaan diterima, status *tugas* diperbarui dalam sistem sehingga *Project Manager* dapat memantau CPM, progres, dan status *tugas* melalui *dashboard*. Gambaran alur proses bisnis *future* dengan Prolytics dapat dilihat pada Gambar 4.2.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



- [illegible]

Jurusan Teknik Informatika dan Komputer – Politeknik Negeri Jakarta



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.2 menunjukkan alur proses bisnis *future* yang dirancang menggunakan sistem Prolytics. Dibandingkan dengan proses *existing*, seluruh tahapan kini terintegrasi dalam satu *platform* sehingga tidak diperlukan perpindahan antara beberapa media. *Project Manager* dapat memantau kondisi *project* dan *phase* dari satu tampilan, *Lead Developer* dapat mendefinisikan *dependency* secara eksplisit, dan *Software Developer* memperoleh informasi *readiness* yang jelas tanpa perlu konfirmasi manual. Perubahan yang memengaruhi *jalur kritis* disimpan dalam bentuk *snapshot* pada level project (*project_cpm_snapshots*) untuk keperluan audit dan *monitoring* riwayat.

4.1.7 Perbandingan Proses Bisnis Existing dan Future

Perbandingan proses bisnis *existing* dan *future* dilakukan untuk mengidentifikasi perubahan konkret yang dihasilkan dari penerapan sistem Prolytics. Perbandingan ini disusun berdasarkan kondisi aktual yang diperoleh melalui wawancara dengan *Project Manager*, *Lead Developer*, dan *Software Developer* PT Lokasolusi Teknologi Indonesia. Berdasarkan hasil wawancara tersebut, *Project Manager* menghadapi kendala utama berupa belum tersedianya sumber informasi terpusat yang dapat menampilkan kondisi proyek secara menyeluruh. *Lead Developer* mengalami keterbatasan dalam mendokumentasikan *dependency* antar-tugas, sehingga kesiapan suatu *tugas* sering kali masih perlu dipastikan melalui konfirmasi manual. Sementara itu, *Software Developer* tidak selalu mengetahui apakah *tugas* yang akan dikerjakan sudah siap atau masih berada dalam kondisi *blocked*. Perbandingan dari setiap aspek tersebut disajikan secara terstruktur pada Tabel 4.

Tabel 4 Perbandingan Proses Bisnis Existing dan Future

Aspek	Existing	Future dengan Prolytics
Pemusatan informasi	Data tersebar pada spreadsheet, chat, dokumen, Trello, dan GitHub.	<i>Project</i> , <i>phase</i> , tugas, <i>approval</i> , dan <i>monitoring</i> tersedia dalam satu sistem.
Pembentukan tugas	Tugas dibentuk melalui koordinasi manual dan	<i>Lead Developer</i> membuat tugas lengkap

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Aspek	Existing	Future dengan Prolytics
	<i>dependency</i> tidak selalu terdokumentasi.	dengan durasi, assignee, dan <i>dependency</i> .
Validasi tugas	Tidak terdapat <i>approval</i> yang terdokumentasi.	Tugas diajukan oleh <i>Lead Developer</i> dan disetujui atau ditolak oleh <i>Project Manager</i> .
Kesiapan tugas	Developer harus memastikan kesiapan tugas melalui koordinasi manual.	Sistem menghitung readiness ready atau blocked berdasarkan predecessor.
<i>Tugas kritis</i>	Ditentukan berdasarkan pengalaman dan kondisi situasional.	Sistem menghitung ES, EF, LS, LF, <i>slack</i> , dan <i>critical path</i> menggunakan CPM.
Riwayat perubahan	Perubahan struktur pekerjaan sulit ditelusuri.	CPM authoritative menghasilkan <i>snapshot</i> dan ringkasan perubahan.
Aktivitas teknis	Aktivitas GitHub diperiksa secara terpisah.	<i>Aktivitas GitHub dapat ditautkan ke tugas sebagai monitoring-only.</i>
Scope CPM	Tidak ada tugas kritis ditentukan secara situasional tanpa metode formal, tugas kritis ditentukan berdasarkan hasil pembicaraan.	CPM dihitung pada level project, mencakup seluruh approved tugas lintas phase dalam satu project.
Phase date	Tanggal mulai dan selesai phase diinput	Phase memiliki <i>calculated_start_date</i> dan <i>calculated_end_date</i>



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Aspek	Existing	Future dengan Prolytics
	manual atau tidak tersedia.	yang dihitung otomatis dari rollup tugas approved.
Dependency lintas phase	Dependency antar tugas tidak terdokumentasi; batasan lintas phase tidak ditetapkan.	Dependency diperbolehkan lintas phase dalam satu project; dependency lintas project ditolak; circular dependency dicegah oleh backend.
Change request tugas aktif	Perubahan tugas dilakukan langsung tanpa mekanisme review dampak CPM.	Perubahan approved tugas yang berdampak CPM masuk task_change_requests dan menunggu persetujuan Project Manager sebelum diterapkan.
Snapshot project-level	Tidak terdapat riwayat snapshot jalur kritis.	Hasil CPM project-wide disimpan sebagai project_cpm_snapshots untuk audit dan penelusuran historis.

Tabel 4 menunjukkan bahwa terdapat perbedaan mendasar antara proses *existing* dan *future* pada seluruh aspek yang diidentifikasi. Dengan penerapan Prolytics, permasalahan fragmentasi informasi diatasi melalui pemusatan *project*, *phase*, tugas, *approval*, dan *monitoring* dalam satu sistem. Mekanisme *approval workflow* memastikan setiap tugas telah divalidasi sebelum dikerjakan, sehingga *dependency* antar tugas dapat terdokumentasi dan status *readiness* dapat diketahui secara

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

otomatis. Perhitungan *Critical Path Method (CPM)* menggantikan penilaian situasional dalam menentukan *tugas kritis*, dan penyimpanan *snapshot* memungkinkan perubahan *jalur kritis* ditelusuri secara historis.

4.1.8 Identifikasi Permasalahan dan Tema Kebutuhan

Berdasarkan hasil analisis proses bisnis berjalan, ditemukan beberapa permasalahan utama dalam *monitoring* proyek perangkat lunak. Permasalahan tersebut kemudian dikelompokkan ke dalam tema kebutuhan agar dapat diturunkan menjadi rancangan fitur pada sistem Prolytics. Identifikasi ini bertujuan untuk memastikan bahwa fitur yang dikembangkan memiliki keterkaitan dengan masalah yang ditemukan pada proses berjalan. Hasil identifikasi permasalahan dan pengelompokan tema kebutuhan tersebut disajikan pada Tabel 5.

Tabel 5 Identifikasi Permasalahan dan Tema Kebutuhan

Kode Masalah	Permasalahan pada Proses Bisnis <i>Existing</i>	Dampak Permasalahan	Tema Kebutuhan Sistem
P-01	Informasi proyek masih tersebar pada <i>chat</i> , notulen, <i>spreadsheet</i> , dokumen, Trello, dan pemeriksaan GitHub secara manual.	Project Manager dan Lead Developer perlu melakukan pengecekan silang untuk mengetahui kondisi proyek.	Sentralisasi data proyek dan <i>monitoring</i>
P-02	Data proyek, anggota tim, <i>phase</i> , dan tugas belum dikelola dalam satu sistem terpusat.	Struktur pekerjaan sulit ditelusuri secara konsisten dari proyek sampai tugas teknis.	Pengelolaan proyek, anggota, <i>phase</i> , dan tugas
P-03	Pembentukan <i>phase</i> dan pembagian tugas teknis belum terdokumentasi secara sistematis.	Lead Developer sulit memastikan pembagian pekerjaan teknis dan anggota <i>phase</i> tercatat dengan baik.	Pengelolaan <i>phase</i> , anggota <i>phase</i> , dan tugas teknis



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode Masalah	Permasalahan pada Proses Bisnis <i>Existing</i>	Dampak Permasalahan	Tema Kebutuhan Sistem
P-04	Hubungan ketergantungan antar-tugas belum dinyatakan secara eksplisit.	Urutan pekerjaan sulit dipahami dan tugas yang tertahan oleh <i>dependency</i> harus ditelusuri secara manual.	Pengelolaan <i>dependency</i> tugas
P-05	Belum tersedia mekanisme identifikasi tugas kritis dan jalur kritis.	Project Manager dan Lead Developer sulit mengetahui pekerjaan yang berdampak langsung terhadap estimasi penyelesaian proyek.	Perhitungan <i>Critical Path Method</i> dan identifikasi tugas kritis
P-06	Belum ada mekanisme <i>approval</i> terhadap tugas sebelum digunakan sebagai dasar pelaksanaan dan perhitungan CPM.	Perubahan atau penambahan tugas dapat memengaruhi rencana proyek tanpa validasi Project Manager.	<i>Approval workflow</i> tugas
P-07	Perubahan terhadap tugas aktif belum memiliki alur persetujuan yang jelas.	Perubahan <i>planning</i> dapat sulit dikendalikan dan berpotensi mengubah jadwal tanpa proses <i>review</i> .	<i>Change request</i> dan persetujuan perubahan tugas
P-08	Aktivitas teknis pada GitHub masih diperiksa secara manual.	Kesesuaian antara laporan progres dan aktivitas teknis perlu diverifikasi secara terpisah.	Integrasi aktivitas teknis GitHub

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode Masalah	Permasalahan pada Proses Bisnis <i>Existing</i>	Dampak Permasalahan	Tema Kebutuhan Sistem
P-09	Status progres tugas dan kesiapan pekerjaan belum dapat dipantau secara otomatis berdasarkan kondisi tugas.	Project Manager dan Lead Developer sulit mengetahui status aktual, <i>readiness</i> , dan hambatan pengerjaan.	<i>Dashboard monitoring</i> , status tugas, dan <i>readiness</i>
P-10	Belum ada penyimpanan riwayat hasil perhitungan CPM.	Perubahan jalur kritis dan estimasi proyek sulit ditelusuri kembali.	Riwayat <i>snapshot</i> CPM

Berdasarkan Tabel 5, kebutuhan utama sistem tidak hanya berkaitan dengan pencatatan *project* dan tugas, tetapi juga dengan kemampuan sistem untuk mendukung *monitoring* berbasis *dependency* dan CPM. Tema kebutuhan seperti *monitoring* terpusat, pengelolaan *dependency*, *readiness* tugas, *approval* tugas, dan riwayat CPM menjadi dasar dalam penyusunan kebutuhan fungsional sistem. Dengan demikian, fitur yang dirancang pada Prolytics diarahkan untuk menjawab permasalahan proses berjalan secara lebih terstruktur dan terukur.

4.1.9 Analisis Pengguna Sistem

Analisis pengguna sistem dilakukan untuk mengidentifikasi aktor-aktor yang terlibat dalam penggunaan Prolytics beserta tanggung jawab dan kebutuhan utama dari masing-masing aktor. Identifikasi ini disusun berdasarkan hasil wawancara dengan seluruh narasumber di PT Lokasolusi Teknologi Indonesia, sehingga peran yang didefinisikan mencerminkan kondisi dan kebutuhan nyata di lapangan. Berdasarkan hasil wawancara tersebut, *Project Manager* membutuhkan visibilitas menyeluruh terhadap kondisi proyek dan *tugas kritis* agar dapat mengambil keputusan berdasarkan data yang terstruktur. *Lead Developer* memerlukan fitur pengelolaan *tugas* teknis yang mendukung pendefinisian *dependency* serta



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

pemantauan *jalur kritis*. Sementara itu, *Software Developer* membutuhkan informasi *readiness* yang jelas serta mekanisme pembaruan status *tugas* yang dapat dipantau oleh pihak terkait. Selain ketiga pengguna tersebut, sistem Prolytics juga berperan sebagai aktor otomatis yang menjalankan perhitungan *Critical Path Method* (CPM), menjaga konsistensi *Directed Acyclic Graph* (DAG), dan menyimpan *snapshot* perubahan *jalur kritis*. Rincian aktor, tanggung jawab, dan kebutuhan utama masing-masing disajikan pada Tabel 6

Tabel 6 Analisis Pengguna Sistem

Aktor	Peran dalam Sistem	Kebutuhan Pengguna	Hak Akses Utama
Project Manager	Pengguna yang bertanggung jawab terhadap pengelolaan proyek, penentuan anggota proyek, <i>review tugas</i> , aktivasi proyek, serta pemantauan progres dan CPM.	Membuat proyek, memilih anggota proyek, melihat <i>phase</i> dan tugas, meninjau tugas yang diajukan, menyetujui atau menolak tugas, mengaktifkan proyek, meninjau <i>change request</i> , dan memantau <i>dashboard</i> .	Mengelola proyek, mengelola anggota proyek, mengelola pengguna, menyetujui atau menolak tugas, menyetujui atau menolak perubahan tugas, melihat hasil CPM, melihat <i>dashboard</i> , dan melihat aktivitas teknis.
Lead Developer	Pengguna yang bertanggung jawab terhadap pembentukan <i>phase</i> , pengelolaan anggota <i>phase</i> , penyusunan <i>technical tugas</i> , <i>dependency</i> , durasi,	Membuat <i>phase</i> , mengatur anggota <i>phase</i> , membuat dan mengatur tugas, mengatur <i>dependency</i> , mengajukan tugas, merevisi tugas, mengajukan	Mengelola <i>phase</i> , mengelola anggota <i>phase</i> , mengelola tugas, mengatur <i>dependency</i> , mengajukan tugas, merevisi tugas, mengajukan <i>change request</i> , melihat

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Aktor	Peran dalam Sistem	Kebutuhan Pengguna	Hak Akses Utama
	<i>assignee</i> , serta <i>review</i> aktivitas teknis developer.	perubahan tugas, memantau progres developer, dan menautkan aktivitas teknis ke tugas.	CPM, melihat aktivitas teknis, serta menautkan atau melepas aktivitas teknis.
Software Developer	Pengguna yang bertanggung jawab terhadap pengerjaan tugas yang telah disetujui dan ditugaskan.	Melihat tugas yang ditugaskan, mengerjakan tugas, memperbarui status tugas, melakukan <i>commit</i> atau <i>pull request</i> , dan melihat aktivitas teknis yang berkaitan dengan pekerjaannya.	Melihat <i>dashboard</i> , melihat profil, memperbarui profil, melihat tugas, memperbarui status tugas, dan melihat aktivitas teknis.

Tabel 6 menunjukkan bahwa setiap aktor memiliki peran dan kebutuhan yang berbeda namun saling berkaitan dalam mendukung proses *monitoring* proyek perangkat lunak. *Project Manager* membutuhkan visibilitas menyeluruh agar dapat mengambil keputusan berdasarkan kondisi *project* yang terstruktur. *Lead Developer* memerlukan mekanisme pengelolaan tugas teknis yang mendukung pendefinisian *dependency* dan pemantauan *critical path* pada setiap *phase*. *Software Developer* membutuhkan informasi *readiness* yang jelas serta saluran pembaruan status yang dapat dipantau oleh tim.

4.1.10 Kebutuhan Fungsional Sistem

Kebutuhan fungsional disusun berdasarkan permasalahan dan tema kebutuhan yang telah diidentifikasi sebelumnya. Kebutuhan ini menggambarkan fungsi-fungsi utama yang harus disediakan oleh sistem Prolytics agar dapat mendukung proses *monitoring* proyek perangkat lunak, pengelolaan tugas, *approval*, aktivitas teknis, serta identifikasi *tugas kritis* menggunakan *Critical Path Method*.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Tabel 7 Kebutuhan Fungsional Sistem

Kode	Dasar Permasalahan	Tema Kebutuhan	Kebutuhan Fungsional	Aktor	Use Case Terkait
KF-01	P-01	Akses sistem	Sistem menyediakan fitur <i>login</i> untuk pengguna.	Project Manager, Lead Developer, Software Developer	<i>Login</i>
KF-02	P-01	Akses sistem	Sistem menyediakan fitur <i>logout</i> untuk mengakhiri sesi pengguna.	Project Manager, Lead Developer, Software Developer	<i>Logout</i>
KF-03	P-09	<i>Dashboard monitoring</i>	Sistem menyediakan <i>dashboard</i> sesuai peran pengguna.	Project Manager, Lead Developer, Software Developer	Melihat <i>Dashboard</i>
KF-04	P-01	Profil pengguna	Sistem menyediakan fitur melihat profil pengguna.	Project Manager, Lead Developer, Software Developer	Melihat Profil
KF-05	P-01	Profil pengguna	Sistem menyediakan fitur memperbarui profil pengguna.	Project Manager, Lead Developer, Software Developer	Memperbarui Profil
KF-06	P-01	Profil pengguna	Sistem menyediakan fitur mengunggah avatar pengguna.	Project Manager, Lead Developer,	Mengunggah Avatar

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode	Dasar Permasalahan	Tema Kebutuhan	Kebutuhan Fungsional	Aktor	Use Case Terkait
				Software Developer	
KF-07	P-02	Manajemen pengguna	Sistem menyediakan fitur melihat daftar pengguna.	Project Manager	Melihat Daftar User
KF-08	P-02	Manajemen pengguna	Sistem menyediakan fitur membuat pengguna baru.	Project Manager	Membuat User
KF-09	P-02	Manajemen pengguna	Sistem menyediakan fitur memperbaiki data pengguna.	Project Manager	Memperbarui Data User
KF-10	P-02	Manajemen pengguna	Sistem menyediakan fitur menghapus pengguna.	Project Manager	Menghapus User
KF-11	P-02	Manajemen proyek	Sistem menyediakan fitur membuat proyek dan menentukan anggota proyek.	Project Manager	Membuat Proyek
KF-12	P-02	Manajemen proyek	Sistem menyediakan fitur memperbaiki informasi proyek.	Project Manager	Memperbarui Proyek
KF-13	P-02	Manajemen proyek	Sistem menyediakan fitur menghapus proyek.	Project Manager	Menghapus Proyek
KF-14	P-02	Manajemen proyek	Sistem menyediakan fitur mengelola anggota proyek.	Project Manager	Mengelola Anggota Proyek



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode	Dasar Permasalahan	Tema Kebutuhan	Kebutuhan Fungsional	Aktor	Use Case Terkait
KF-15	P-06	Aktivasi proyek	Sistem menyediakan fitur mengaktifkan proyek setelah tugas siap dan disetujui.	Project Manager	Mengaktifkan Proyek
KF-16	P-03	Manajemen <i>phase</i>	Sistem menyediakan fitur membuat <i>phase</i> pada proyek berstatus <i>draft</i> .	Lead Developer	Membuat Fase
KF-17	P-03	Manajemen <i>phase</i>	Sistem menyediakan fitur memperbarui metadata <i>phase</i> .	Lead Developer	Memperbarui Metadata Fase
KF-18	P-03	Manajemen <i>phase</i>	Sistem menyediakan fitur mengelola anggota <i>phase</i> .	Lead Developer	Mengelola Anggota Fase
KF-19	P-03	Manajemen tugas	Sistem menyediakan fitur membuat tugas teknis.	Lead Developer	Membuat Tugas
KF-20	P-03	Manajemen tugas	Sistem menyediakan fitur memperbarui tugas.	Lead Developer	Memperbarui Tugas
KF-21	P-03	Manajemen tugas	Sistem menyediakan fitur menghapus tugas.	Lead Developer	Menghapus Tugas
KF-22	P-04	<i>Dependency tugas</i>	Sistem menyediakan fitur mengatur <i>dependency</i> antar-tugas.	Lead Developer	Mengatur Dependensi Tugas



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode	Dasar Permasalahan	Tema Kebutuhan	Kebutuhan Fungsional	Aktor	Use Case Terkait
KF-23	P-06	<i>Approval workflow</i> tugas	Sistem menyediakan fitur mengajukan tugas kepada Project Manager.	Lead Developer	Mengajukan Tugas
KF-24	P-06	<i>Approval workflow</i> tugas	Sistem menyediakan fitur menampilkan antrean persetujuan tugas.	Project Manager	Mengelola Persetujuan Tugas
KF-25	P-06	<i>Approval workflow</i> tugas	Sistem menyediakan fitur menyetujui tugas.	Project Manager	Menyetujui Tugas
KF-26	P-06	<i>Approval workflow</i> tugas	Sistem menyediakan fitur menolak tugas.	Project Manager	Menolak Tugas
KF-27	P-06	<i>Approval workflow</i> tugas	Sistem menyediakan fitur meminta revisi tugas.	Project Manager	Meminta Revisi Tugas
KF-28	P-06	<i>Approval workflow</i> tugas	Sistem menyediakan fitur merevisi tugas yang ditolak atau diminta revisi.	Lead Developer	Merevisi Tugas
KF-29	P-07	<i>Change request</i>	Sistem menyediakan fitur mengajukan perubahan tugas aktif.	Lead Developer	Mengajukan Perubahan Tugas
KF-30	P-07	<i>Change request</i>	Sistem menyediakan fitur menampilkan antrean persetujuan perubahan tugas.	Project Manager	Mengelola Persetujuan Perubahan Tugas



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode	Dasar Permasalahan	Tema Kebutuhan	Kebutuhan Fungsional	Aktor	Use Case Terkait
KF-31	P-07	<i>Change request</i>	Sistem menyediakan fitur menyetujui perubahan tugas.	Project Manager	Menyetujui Perubahan Tugas
KF-32	P-07	<i>Change request</i>	Sistem menyediakan fitur menolak perubahan tugas.	Project Manager	Menolak Perubahan Tugas
KF-33	P-09	Status tugas	Sistem menyediakan fitur memperbarui status tugas.	Software Developer	Memperbarui Status Tugas
KF-34	P-05	Perhitungan CPM	Sistem menyediakan fitur melihat hasil <i>Critical Path Method</i> pada level proyek.	Project Manager, Lead Developer	Melihat Hasil CPM
KF-35	P-05	Perhitungan CPM	Sistem menyediakan fitur melihat jalur kritis.	Project Manager, Lead Developer	Melihat Jalur Kritis
KF-36	P-05	Perhitungan CPM	Sistem menyediakan fitur melihat visualisasi CPM.	Project Manager, Lead Developer	Melihat Visualisasi CPM
KF-37	P-10	Riwayat CPM	Sistem menyediakan fitur melihat riwayat <i>snapshot</i> CPM.	Project Manager, Lead Developer	Melihat Riwayat Snapshot CPM
KF-38	P-05, P-07	Perhitungan ulang CPM	Sistem menyediakan fitur memicu perhitungan ulang CPM ketika terdapat	Project Manager, Lead Developer	Memicu Perhitungan Ulang CPM

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode	Dasar Permasalahan	Tema Kebutuhan	Kebutuhan Fungsional	Aktor	Use Case Terkait
			perubahan yang disetujui.		
KF-39	P-08	Aktivitas teknis	Sistem menyediakan fitur melihat aktivitas teknis dari repository.	Lead Developer, Software Developer	Melihat Aktivitas Teknis
KF-40	P-08	Aktivitas teknis	Sistem menyediakan fitur menautkan aktivitas teknis ke tugas.	Lead Developer	Menautkan Aktivitas ke Tugas
KF-41	P-08	Aktivitas teknis	Sistem menyediakan fitur melepas tautan aktivitas teknis dari tugas.	Lead Developer	Melepas Tautan Aktivitas

Berdasarkan Tabel 7, kebutuhan fungsional Prolytics berfokus pada pengelolaan *project*, *phase*, tugas, *dependency*, *approval*, dan perhitungan CPM. Selain itu, sistem juga mendukung *monitoring* aktivitas teknis melalui integrasi GitHub serta pengelolaan profil pengguna. Kebutuhan fungsional tersebut menjadi dasar dalam perancangan *use case*, arsitektur sistem, basis data, dan implementasi modul utama Prolytics.

4.1.11 Use Case Diagram

Use case diagram digunakan untuk menggambarkan hubungan antara aktor pengguna dengan fungsi-fungsi utama yang tersedia pada sistem Prolytics. Diagram ini menunjukkan batasan sistem, peran setiap pengguna, serta layanan yang dapat diakses sesuai dengan hak akses masing-masing *role*. Dalam sistem Prolytics, terdapat tiga aktor utama, yaitu *Project Manager*, *Lead Developer*, dan *Software Developer*. Ketiga aktor tersebut memiliki tanggung jawab yang berbeda dalam proses *monitoring* proyek perangkat lunak, mulai dari pengelolaan *project*, penyusunan tugas teknis, *approval* tugas, pemantauan hasil CPM, hingga

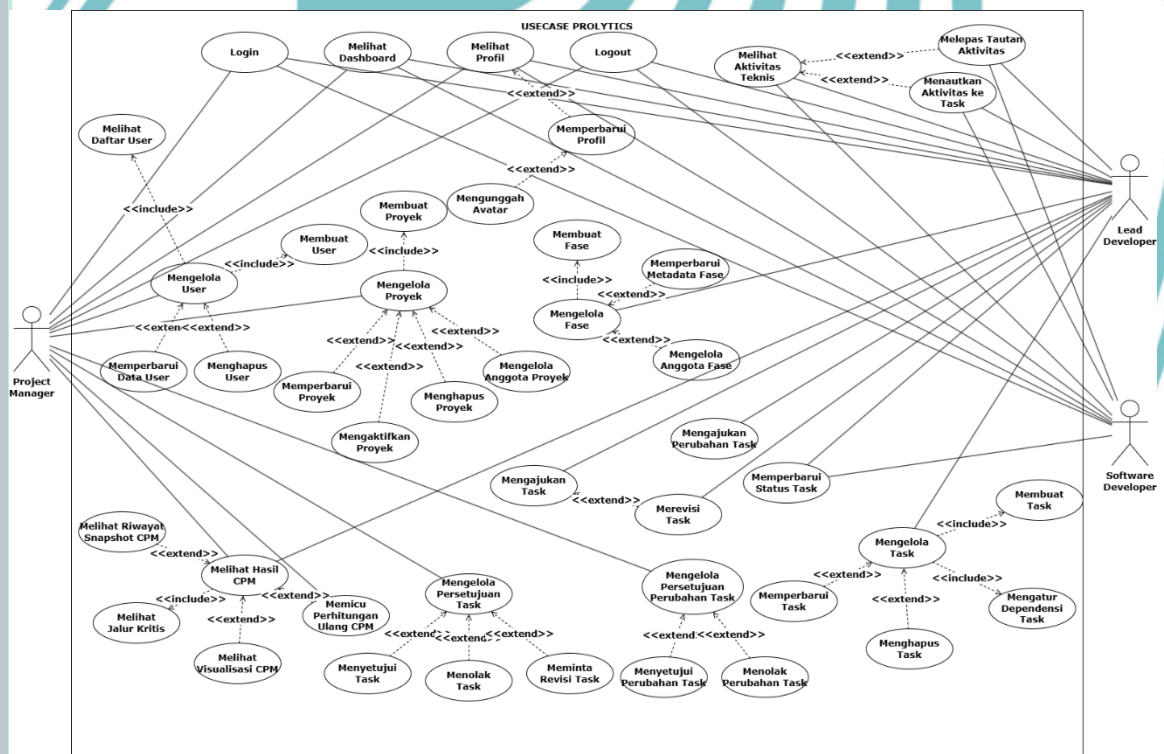


© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

pembaruan status pekerja *Project Manager* berperan sebagai aktor yang memiliki kewenangan utama dalam pengelolaan *project*. Aktor ini dapat mengelola *project*, *phase*, user, anggota *project*, anggota *phase*, serta melakukan *approval* terhadap tugas yang diajukan oleh *Lead Developer*. *Lead Developer* berperan dalam menerjemahkan kebutuhan menjadi tugas teknis, mengatur assignee, menentukan *dependency*, mengajukan tugas untuk persetujuan, serta memantau aktivitas teknis yang berkaitan dengan pengerjaan tugas. *Software Developer* berperan sebagai pelaksana tugas yang telah ditugaskan, sehingga fungsi utamanya adalah melihat tugas, memperbarui status tugas, dan menautkan aktivitas teknis yang relevan. Seluruh *role* juga memiliki akses untuk *login*, *logout*, melihat *dashboard*, melihat profil, memperbarui profil, serta melihat informasi CPM sesuai dengan akses *project* yang dimiliki.



Gambar 4. 3 Usecase Diagram

Berdasarkan Gambar 4.3, sistem Prolytics memiliki tiga aktor utama, yaitu *Project Manager*, *Lead Developer*, dan *Software Developer*. Setiap aktor memiliki hak

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

akses yang berbeda sesuai perannya dalam proses *monitoring* proyek perangkat lunak.

Project Manager berperan dalam mengelola *project*, *phase*, user, anggota *project*, anggota *phase*, serta melakukan *approval* terhadap tugas yang diajukan. *Lead Developer* berperan dalam mengelola tugas teknis, menentukan assignee, mengatur *dependency*, mengajukan tugas, dan memantau aktivitas teknis. Sementara itu, *Software Developer* berperan dalam melihat tugas yang ditugaskan, memperbarui status tugas, serta menautkan aktivitas teknis yang berkaitan dengan pengerjaan tugas.

Diagram tersebut juga menunjukkan bahwa informasi CPM dapat diakses sesuai hak akses pengguna. Fitur CPM mencakup hasil perhitungan, *jalur kritis*, visualisasi CPM, riwayat *snapshot*, dan perhitungan ulang. Dengan demikian, *use case* diagram ini menggambarkan bahwa Prolytics mendukung pengelolaan *project*, tugas, *approval*, aktivitas teknis, dan *monitoring* CPM secara terpusat berdasarkan peran masing-masing pengguna.

4.1.12 Activity Diagram

Activity diagram digunakan untuk menggambarkan alur aktivitas utama pada sistem Prolytics berdasarkan interaksi antara aktor dan sistem. Pada penelitian ini, *activity diagram* disusun untuk menjelaskan sembilan alur utama, yaitu proses *login*, pengelolaan *project*, pengelolaan *tugas*, *approval tugas* dan aktivasi CPM, pengeditan *tugas*, perhitungan CPM, melihat hasil *Critical Path Method*, memperbarui status *tugas*, serta integrasi aktivitas GitHub dengan *progress tugas*. Setiap diagram menunjukkan urutan aktivitas, keputusan yang terjadi, serta respons sistem terhadap tindakan pengguna sesuai dengan peran masing-masing.

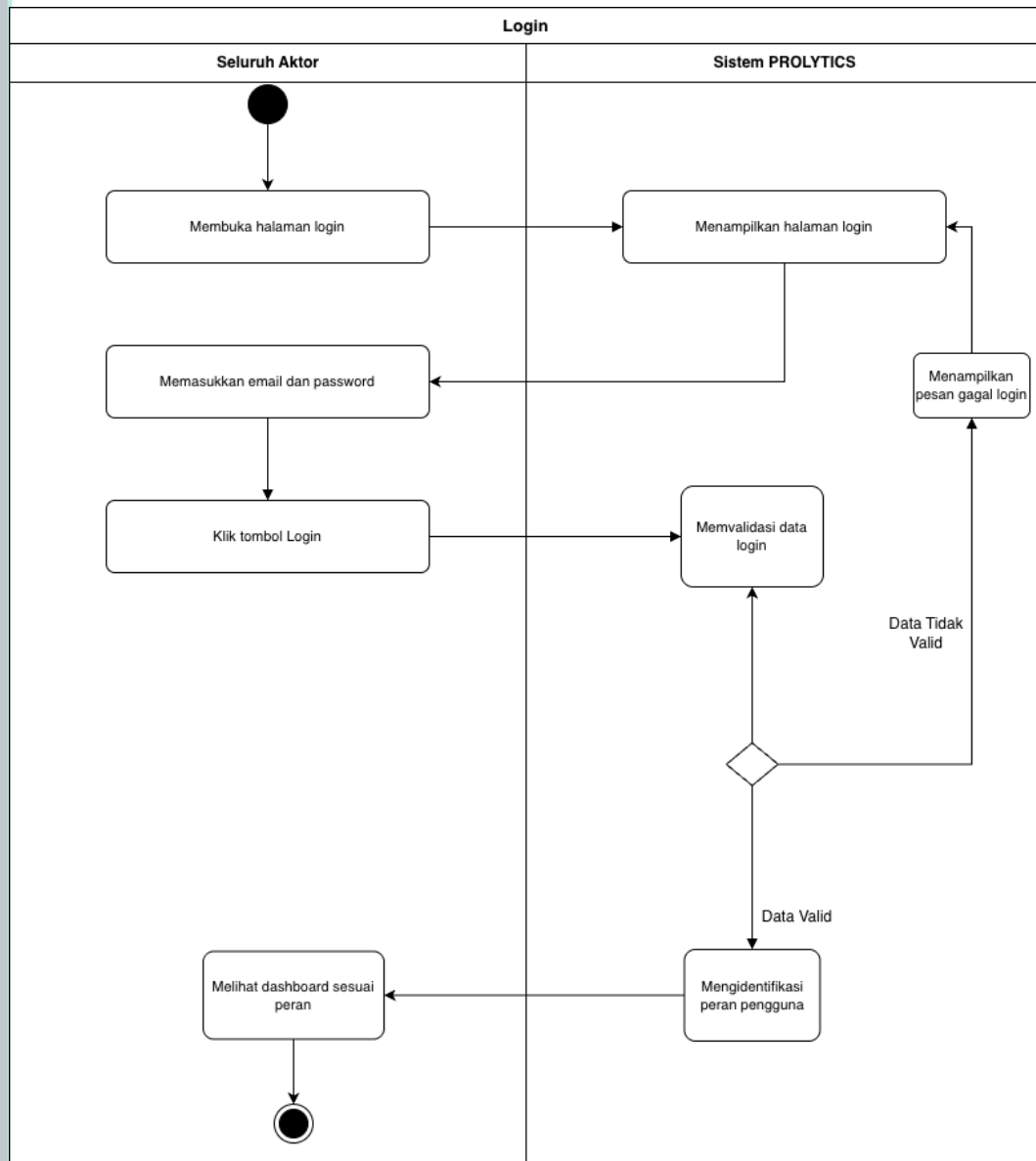
Rangkaian *activity diagram* tersebut digunakan untuk memperjelas bagaimana kebutuhan fungsional sistem diterjemahkan menjadi alur kerja yang dapat dijalankan oleh Project Manager, Lead Developer, Software Developer, dan sistem Prolytics. Diagram pertama menjelaskan proses autentikasi pengguna sebelum aktor dapat mengakses fitur sesuai kewenangannya. Alur proses *login* pada sistem Prolytics ditunjukkan pada Gambar 4.4.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



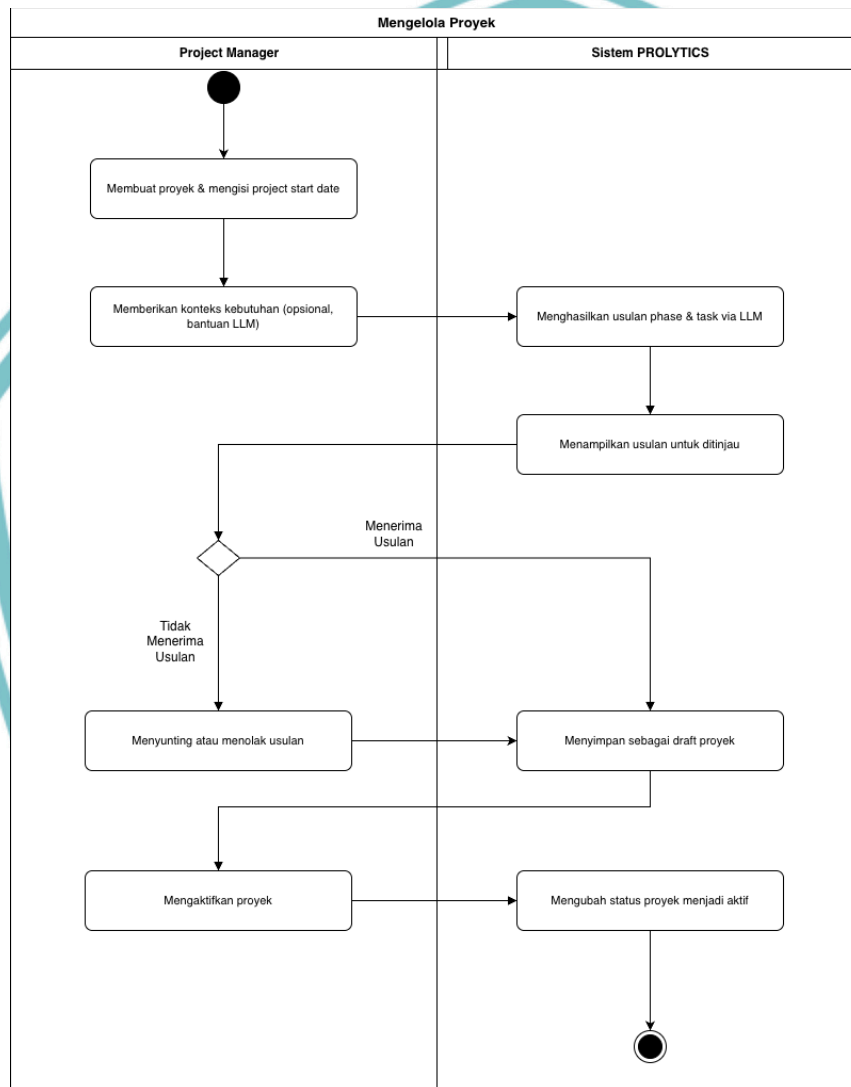
Gambar 4. 4 Activity Diagram Login Sistem

Gambar 4.4 menggambarkan alur autentikasi pengguna pada sistem Prolytics. Proses dimulai ketika pengguna membuka halaman *login*, kemudian sistem menampilkan halaman *login* agar pengguna dapat memasukkan *email* dan *password*. Setelah tombol *login* ditekan, sistem melakukan validasi terhadap data *login* yang dimasukkan. Apabila data valid, sistem mengidentifikasi peran pengguna dan mengarahkan pengguna ke *dashboard* sesuai kewenangannya. Sebaliknya, apabila data tidak valid, sistem menampilkan pesan gagal *login* sehingga pengguna perlu mengulang proses autentikasi.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang menggunakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Setelah pengguna berhasil masuk ke sistem, aktivitas berikutnya adalah pengelolaan *project* oleh Project Manager. Pada proses ini, Project Manager dapat membuat *project*, mengisi *project start date*, serta memberikan konteks kebutuhan yang dapat digunakan oleh sistem untuk menghasilkan usulan *phase* dan *tugas* melalui bantuan LLM. Alur pengelolaan *project* ditunjukkan pada Gambar 4.5.



Gambar 4. 5 Activity Diagram Mengelola Proyek

Gambar 4.5 menggambarkan alur pengelolaan *project* yang dibantu oleh *Large Language Model* (LLM). Project Manager membuat *project*, mengisi *project start date*, dan memberikan konteks kebutuhan *project*. Sistem kemudian menggunakan konteks tersebut untuk menghasilkan usulan *phase* dan *tugas*. Usulan yang



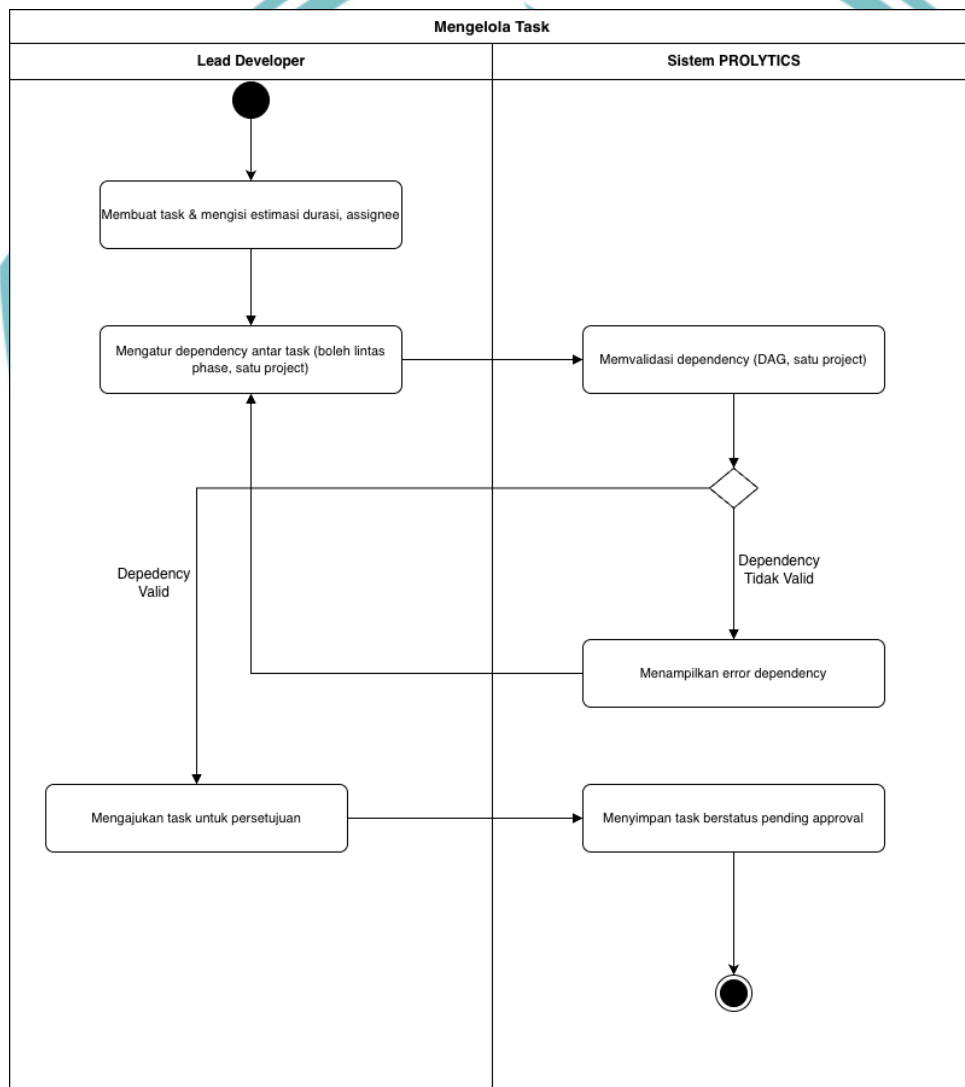
© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

ditampilkan dapat diterima, diubah, atau ditolak oleh Project Manager sebelum disimpan sebagai *draft project*. Setelah data sesuai, Project Manager dapat mengaktifkan *project* sehingga status *project* berubah menjadi aktif. Diagram aktivitas pengelolaan *tugas* ditunjukkan pada Gambar 4.6.

Diagram aktivitas pengelolaan tugas menggambarkan alur *Software Developer* dalam membuat dan mengatur tugas di dalam suatu *phase* pada *project* aktif. Alur proses pengelolaan tugas secara lengkap ditunjukkan pada Gambar 4.6.



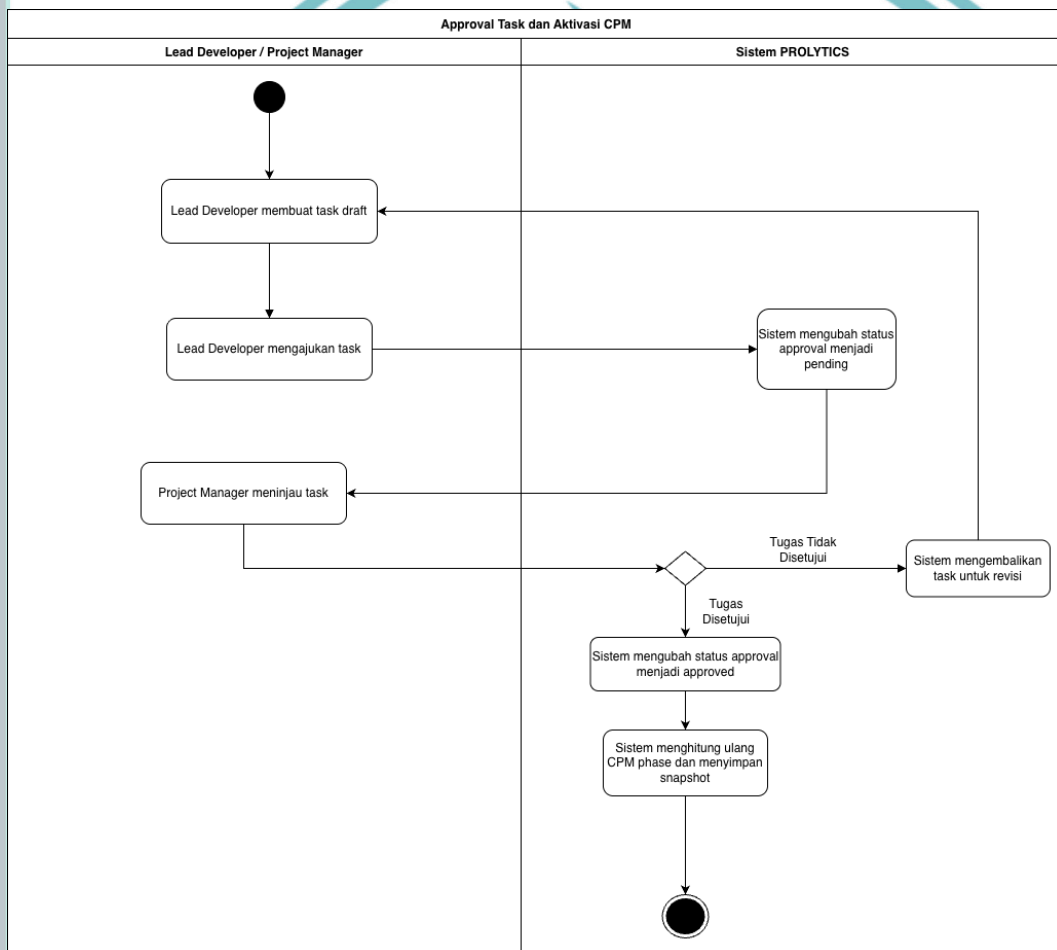
Gambar 4. 6 Activity Diagram Mengelola Tugas

Gambar 4.6 menunjukkan alur pengelolaan *tugas* pada sistem Prolytics. Proses dimulai ketika Lead Developer atau Software Developer membuat *tugas* pada

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

phase tertentu dengan mengisi informasi seperti nama *tugas*, estimasi durasi, *assignee*, dan relasi *dependency* terhadap *tugas* lain. Data *tugas* yang telah dibuat disimpan oleh sistem sebagai rancangan pekerjaan teknis. Setelah *tugas* dinilai siap, *tugas* dapat diajukan untuk proses *approval* agar dapat masuk ke perhitungan CPM dan menjadi bagian dari alur pelaksanaan *project*. Diagram aktivitas *approval tugas* dan aktivasi CPM ditunjukkan pada Gambar 4.7.



Gambar 4. 7 Activity Diagram Approval Tugas dan Aktivasi CPM

Gambar 4.7 menggambarkan alur persetujuan *technical tugas* dan aktivasi CPM. Proses dimulai ketika Lead Developer mengajukan *tugas* yang telah disusun untuk ditinjau oleh Project Manager. Project Manager kemudian memeriksa kelengkapan dan kesesuaian *tugas*, termasuk durasi, *assignee*, dan *dependency* yang telah

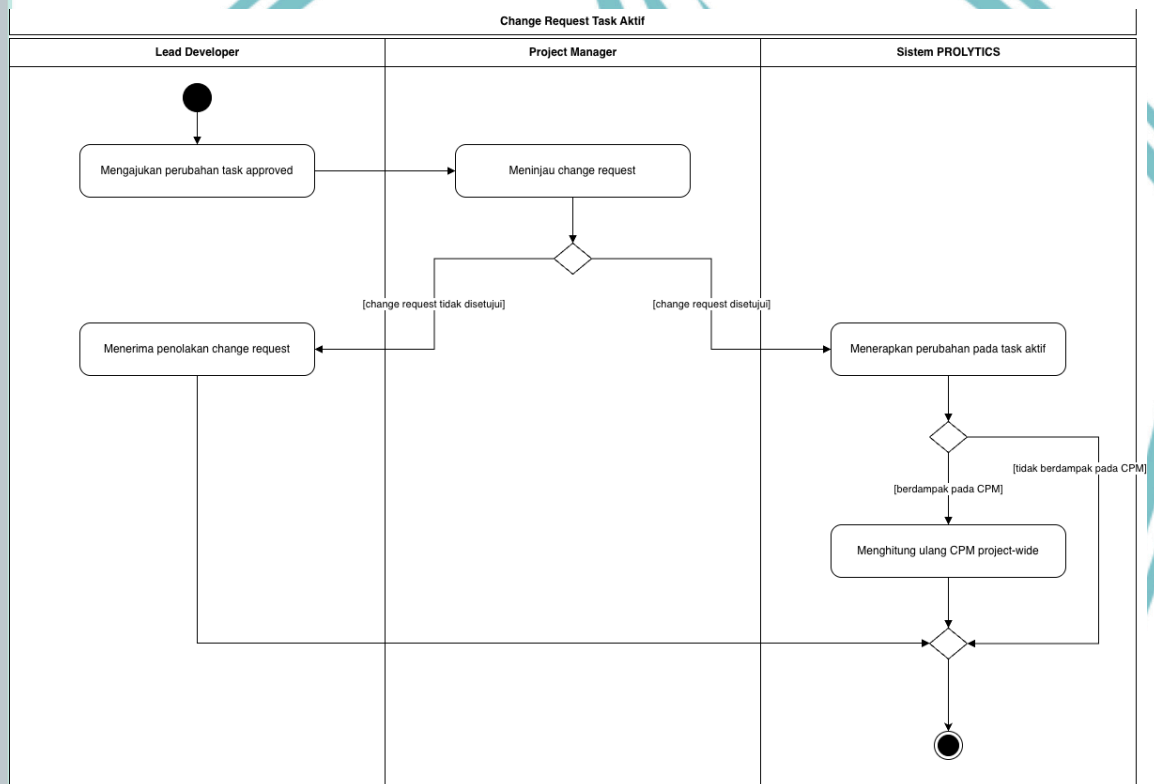


Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

ditentukan. Apabila *task* disetujui, sistem memperbarui status *task* menjadi *approved* dan memicu perhitungan CPM aktif pada *project*. Sebaliknya, apabila *task* belum sesuai, Project Manager dapat menolak atau meminta revisi sehingga *task* dikembalikan kepada Lead Developer untuk diperbaiki. Diagram aktivitas pengeditan *task* ditunjukkan pada Gambar 4.8.

Diagram aktivitas pengeditan *task* menggambarkan alur *Software Developer* dalam memperbarui atribut *task* yang telah dibuat sebelumnya. Proses pengeditan *task* tersebut ditunjukkan pada Gambar 4.8.



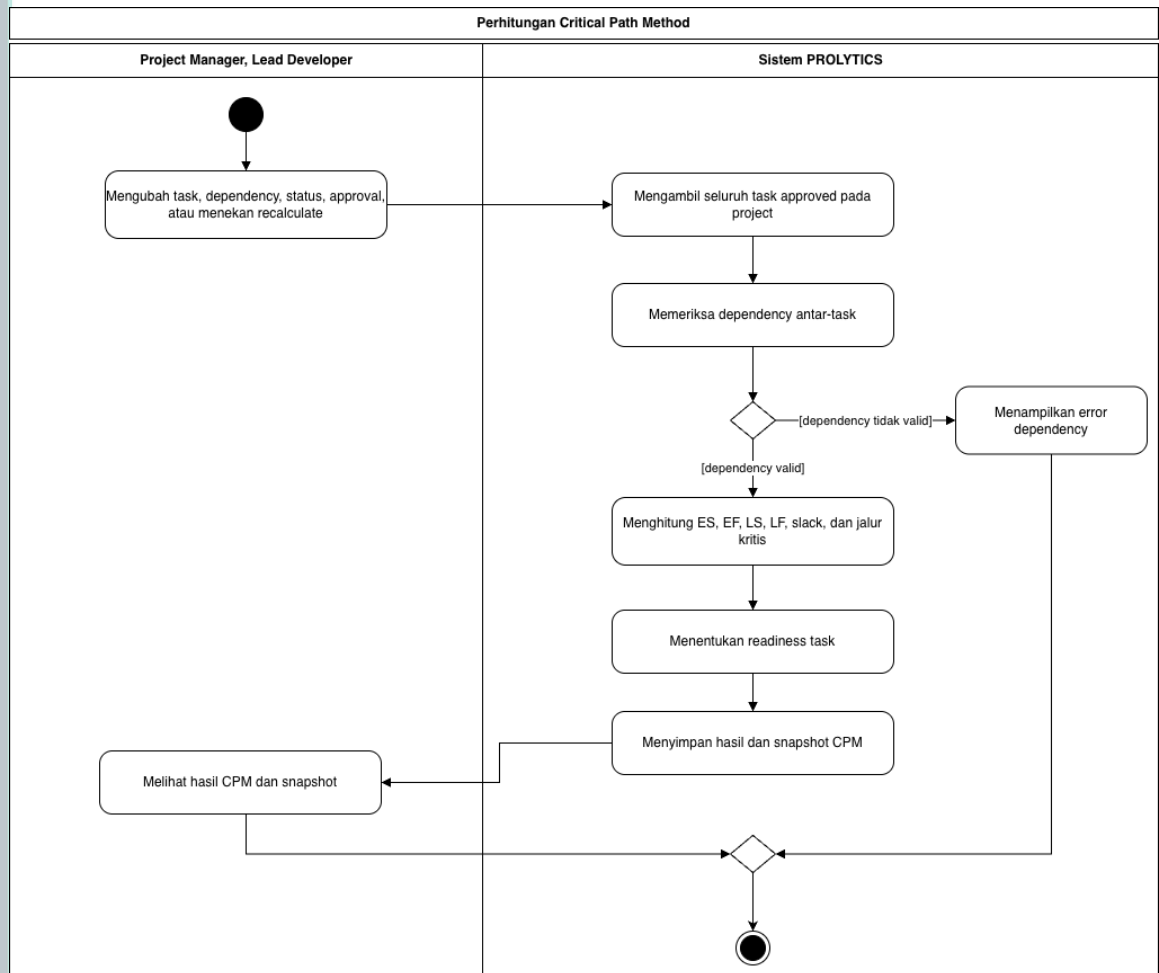
Gambar 4. 8 Activity Diagram Mengedit Tugas

Gambar 4.8 menunjukkan alur pengeditan *task* pada sistem Prolytics. Pengeditan dilakukan ketika terdapat perubahan informasi pekerjaan, seperti perubahan nama *task*, estimasi durasi, *assignee*, atau relasi *dependency*. Apabila *task* yang diedit masih berstatus *draft* atau belum masuk ke CPM aktif, perubahan dapat disimpan sebagai pembaruan data *task*. Namun, apabila *task* yang diedit telah berstatus *approved*, sistem perlu memperbarui data terkait dan memicu ulang perhitungan

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

CPM agar informasi jalur kritis tetap sesuai dengan kondisi terbaru. Diagram aktivitas perhitungan CPM ditunjukkan pada Gambar 4.9.



Gambar 4. 9 Activity Diagram Perhitungan CPM

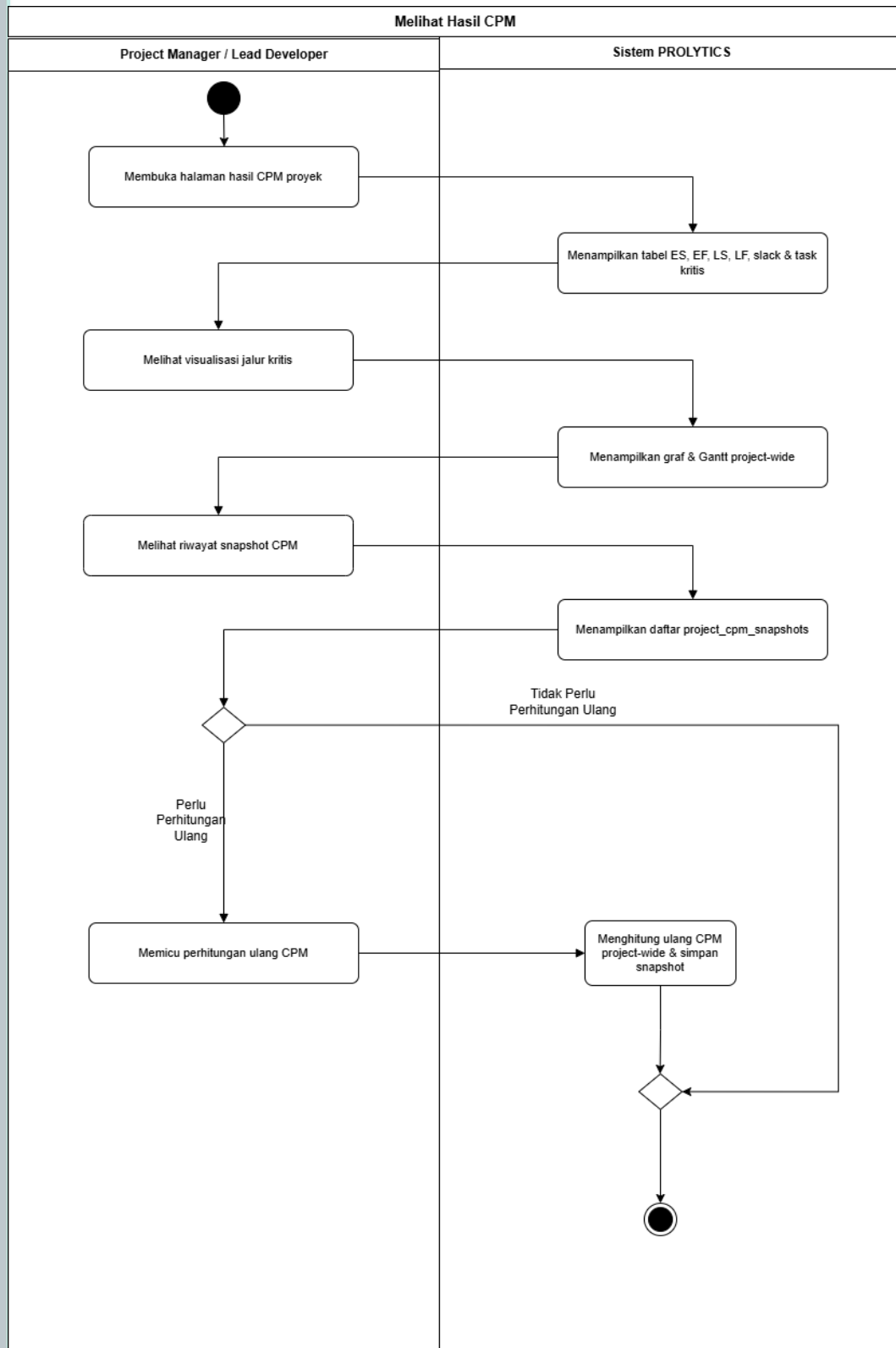
Gambar 4.9 menggambarkan proses perhitungan CPM pada sistem Prolytics. Sistem membaca data *approved tugas*, estimasi durasi, *project start date*, serta relasi *dependency* antar *tugas* dalam satu *project*. Data tersebut kemudian digunakan untuk membentuk struktur hubungan pekerjaan sebelum dilakukan perhitungan *forward pass* dan *backward pass*. Hasil perhitungan menghasilkan nilai *Early Start* (ES), *Early Finish* (EF), *Late Start* (LS), *Late Finish* (LF), *slack*, *tugas* kritis, dan *critical path*. Informasi tersebut disimpan oleh sistem agar dapat digunakan pada halaman monitoring dan visualisasi CPM. Diagram aktivitas melihat hasil *Critical Path Method* ditunjukkan pada Gambar 4.10.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

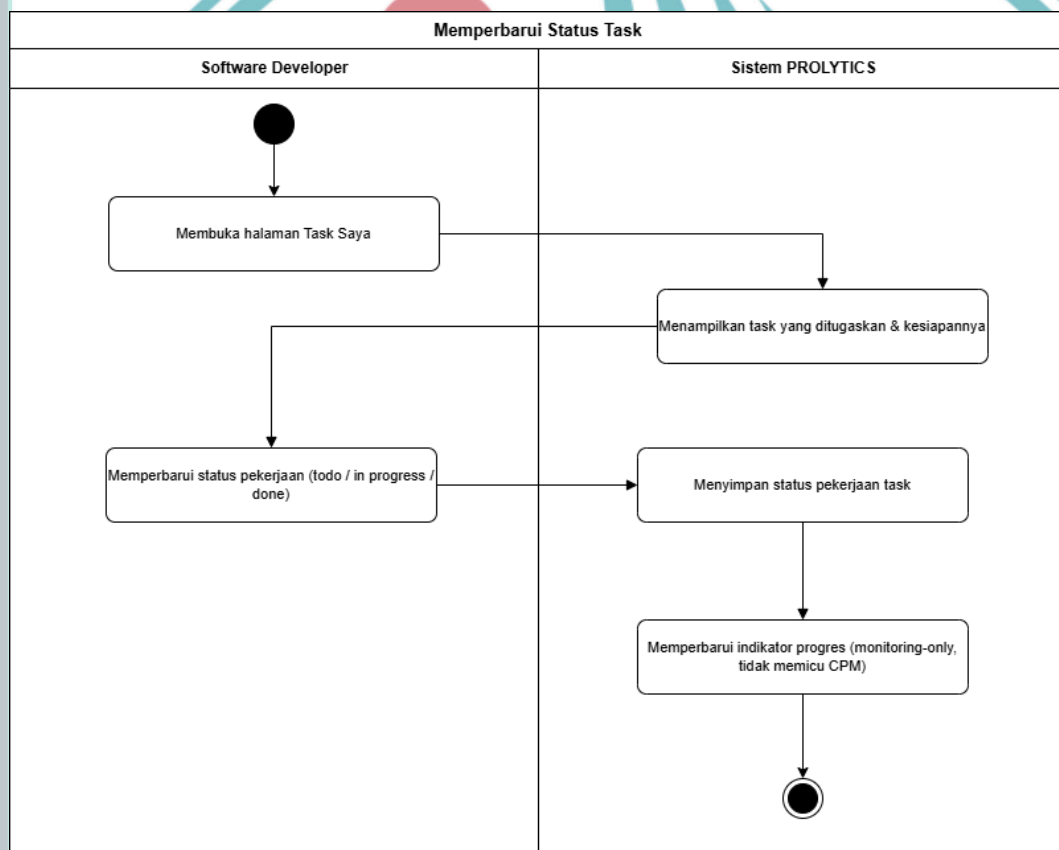


Gambar 4. 10 Activity Diagram Melihat Hasil *Critical Path Method*

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.10 menunjukkan alur pengguna dalam melihat hasil *Critical Path Method* pada sistem Prolytics. Pengguna dapat membuka halaman hasil CPM untuk melihat visualisasi jalur kritis, daftar *tugas* kritis, nilai *slack*, dan informasi waktu pengerjaan berdasarkan hasil perhitungan sistem. Informasi tersebut membantu Project Manager dan Lead Developer memahami *tugas* yang memiliki pengaruh langsung terhadap durasi penyelesaian *project*. Dengan adanya visualisasi ini, proses monitoring tidak hanya menampilkan progres pekerjaan, tetapi juga menunjukkan bagian pekerjaan yang perlu memperoleh perhatian lebih tinggi. Diagram aktivitas pembaruan status *tugas* ditunjukkan pada Gambar 4.11.



Gambar 4. 11 Activity Diagram Memperbarui Status Tugas

Gambar 4.11 menggambarkan alur pembaruan status *tugas* oleh Software Developer. Proses dimulai ketika Software Developer membuka daftar *tugas* yang ditugaskan, kemudian memperbarui status atau persentase progres sesuai perkembangan pengerjaan. Sistem menyimpan perubahan tersebut dan memperbarui informasi progres pada *dashboard* agar Project Manager dan Lead

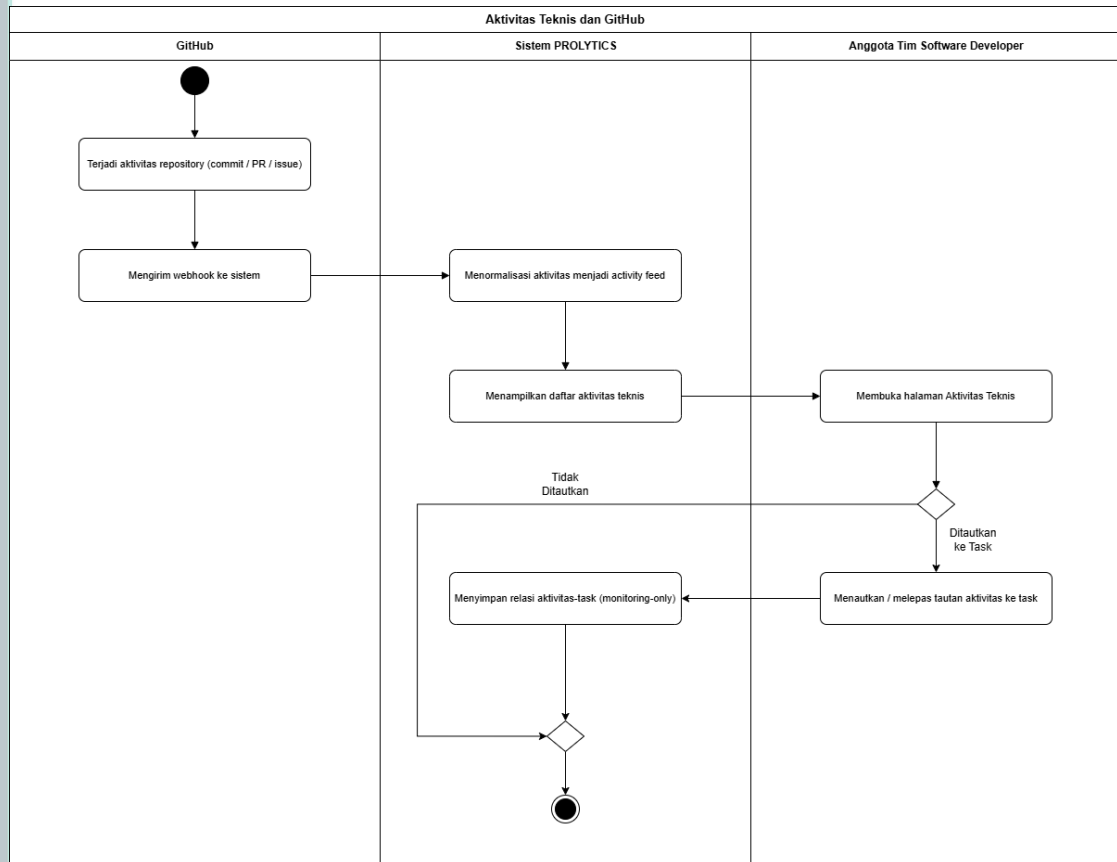


© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Developer dapat memantau realisasi pekerjaan secara berkala. Pembaruan status ini menjadi bagian dari proses monitoring harian karena menunjukkan kondisi aktual pengerjaan *tugas* di dalam *project*. Diagram aktivitas GitHub dan *progress tugas* ditunjukkan pada Gambar 4.12.



Gambar 4. 12 *Activity Diagram* Aktivitas GitHub dan Progress Tugas

Gambar 4.12 menggambarkan alur integrasi aktivitas GitHub dengan *progress tugas* pada sistem Prolytics. Ketika terjadi aktivitas teknis pada *repository*, seperti *commit*, *pull request*, atau *issue*, GitHub mengirimkan *webhook* ke sistem. Sistem kemudian menormalisasi data aktivitas tersebut menjadi *activity feed* dan menautkannya ke *project*, *phase*, atau *tugas* yang relevan. Dengan mekanisme ini, progres yang dilaporkan pada sistem dapat didukung oleh bukti aktivitas teknis dari GitHub. Integrasi tersebut membantu Project Manager dan Lead Developer memantau keterkaitan antara laporan progres dan aktivitas pengembangan yang terjadi pada *repository*.

4.1.13 Kebutuhan Non - Fungsional Sistem

Tabel 8 Kebutuhan Non Fungsional Sistem

Kode	Kebutuhan	Penjelasan
NFR-01	Keamanan akses	<i>Backend</i> dan <i>frontend</i> membatasi akses berdasarkan <i>role</i> dan <i>membership</i> .
NFR-02	Integritas graf	<i>Dependency</i> wajib satu <i>project</i> dan tidak membentuk <i>circular dependency</i> .
NFR-03	Keterlacakan	<i>Snapshot</i> , <i>approval timeline</i> , dan <i>project log</i> menyimpan perubahan penting.
NFR-04	Konsistensi perhitungan	CPM dihitung oleh <i>backend</i> secara deterministik dan transaksional.
NFR-05	Kemudahan penggunaan	Antarmuka menggunakan tabel, badge, visualisasi, dan navigasi berbasis <i>role</i> .
NFR-06	Maintainability	<i>Backend</i> menggunakan pemisahan <i>route</i> , <i>controller</i> , <i>service</i> , <i>repository</i> , dan <i>engine</i> .
NFR-07	Pergantian tema	Sistem menyediakan fitur switch theme (mode terang dan gelap) yang dapat diubah pengguna

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode	Kebutuhan	Penjelasan
		sesuai preferensi tampilan.
NFR-08	Dokumentasi API	<i>Backend</i> menyediakan dokumentasi API berbasis Swagger/OpenAPI untuk memudahkan pemahaman dan pengujian <i>endpoint</i> .
NFR-09	Paginasi data	Daftar data dalam jumlah besar ditampilkan menggunakan pagination agar pemuatan tetap efisien dan terstruktur.
NFR-10	Pencarian data	Sistem menyediakan fitur pencarian untuk mempercepat pengguna menemukan data yang dibutuhkan.
NFR-11	Filtering data	Sistem menyediakan fitur filtering agar pengguna dapat menyaring data berdasarkan kriteria tertentu.

4.2 Perancangan Sistem

Perancangan sistem dilakukan untuk menerjemahkan kebutuhan pengguna dan proses bisnis yang telah dianalisis pada subbab sebelumnya ke dalam rancangan teknis yang dapat diimplementasikan. Pada tahap ini, rancangan tidak hanya menggambarkan tampilan antarmuka, tetapi juga menjelaskan arsitektur sistem,

Hak Cipta :

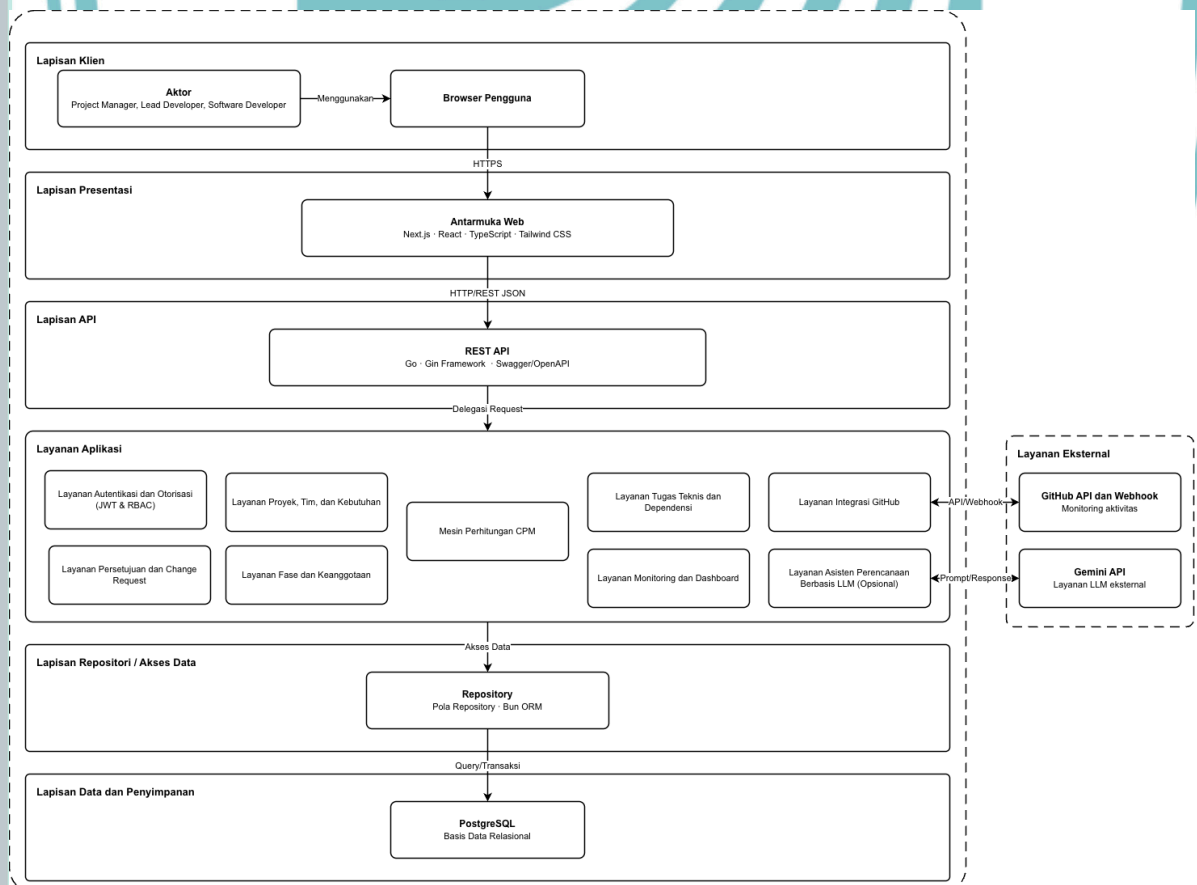
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

pembagian hak akses, basis data, dan alur perhitungan *Critical Path Method* (CPM) yang menjadi bagian utama penelitian.

Rancangan sistem Prolytics disusun dengan pendekatan modular agar setiap kebutuhan dapat dipetakan ke komponen yang sesuai. Modul *project*, *phase*, tugas, *dependency*, *approval*, GitHub, LLM, dan CPM dirancang saling terhubung, tetapi tetap memiliki tanggung jawab yang jelas sehingga proses pengembangan dan pengujian sistem dapat dilakukan secara lebih terarah.

4.2.1 Perancangan Arsitektur Sistem

Arsitektur sistem dirancang dengan pola pemisahan antara sisi *frontend*, *backend*, *database*, dan layanan eksternal. Pemisahan ini diperlukan karena Prolytics melibatkan interaksi pengguna, pengolahan data proyek, perhitungan CPM, serta integrasi eksternal yang memiliki tanggung jawab berbeda.



Gambar 4. 13 Arsitektur Sistem



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.13 menunjukkan bahwa *frontend* berperan sebagai antarmuka pengguna, sedangkan *backend* menyediakan *RESTful* API, validasi hak akses, pengolahan data, dan layanan domain seperti *project*, tugas, *approval*, CPM, GitHub, serta LLM. PostgreSQL digunakan sebagai penyimpanan utama, sementara integrasi GitHub dan LLM ditempatkan sebagai layanan pendukung yang tidak menggantikan keputusan pengguna.

4.2.2 Perancangan Antarmuka Pengguna

Perancangan antarmuka pengguna berfokus pada kemudahan navigasi berdasarkan peran pengguna. *Project Manager* membutuhkan tampilan untuk mengelola proyek, *phase*, anggota, persetujuan, dan kondisi CPM. *Lead Developer* membutuhkan tampilan untuk menyusun technical tugas, *dependency*, assignee, dan mengajukan tugas. *Software Developer* membutuhkan tampilan untuk melihat tugas yang ditugaskan, memperbarui status pekerjaan, serta menautkan aktivitas teknis jika diperlukan.

Rancangan antarmuka dituangkan dalam bentuk wireframe untuk seluruh halaman utama sistem sebelum diimplementasikan. Wireframe digunakan sebagai acuan tata letak, alur navigasi, dan penempatan komponen pada setiap halaman, sehingga implementasi antarmuka pada subbab berikutnya memiliki dasar perancangan yang jelas. Gambar 4.14 sampai dengan Gambar 4.24 menyajikan wireframe untuk setiap halaman utama beserta penjelasan rancangannya.

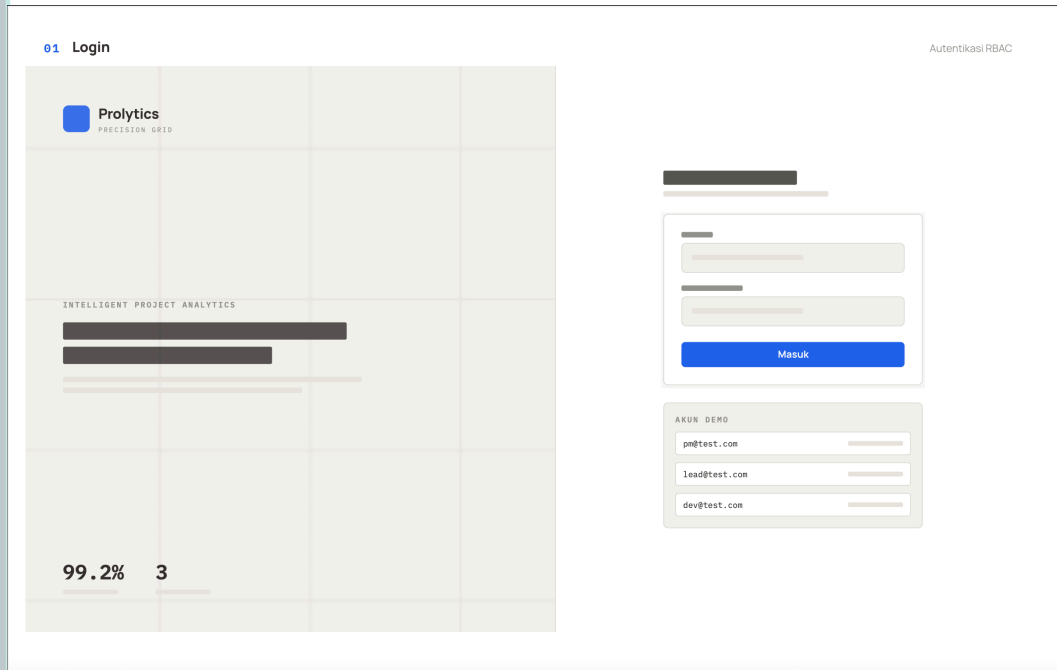
Halaman login merupakan titik masuk sistem yang menerapkan autentikasi berbasis peran. Wireframe pada Gambar 4.14 menempatkan logo dan judul sistem di bagian atas, diikuti formulir masuk yang terdiri atas kolom email dan kata sandi serta tombol Masuk. Tata letak dibuat terpusat agar fokus pengguna tertuju pada proses autentikasi sebelum diarahkan ke *dashboard* sesuai peran. Pada sisi kanan wireframe ditampilkan panel identitas sistem beserta ringkasan nilai yang ditawarkan, sedangkan indikator status autentikasi berbasis RBAC diletakkan pada bagian atas untuk menegaskan bahwa hak akses ditentukan sejak proses masuk. Rancangan ini menjaga kesederhanaan tampilan sekaligus memastikan pengguna langsung memahami langkah yang harus dilakukan tanpa elemen yang mengganggu.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 4. 14 Wireframe Halaman Login

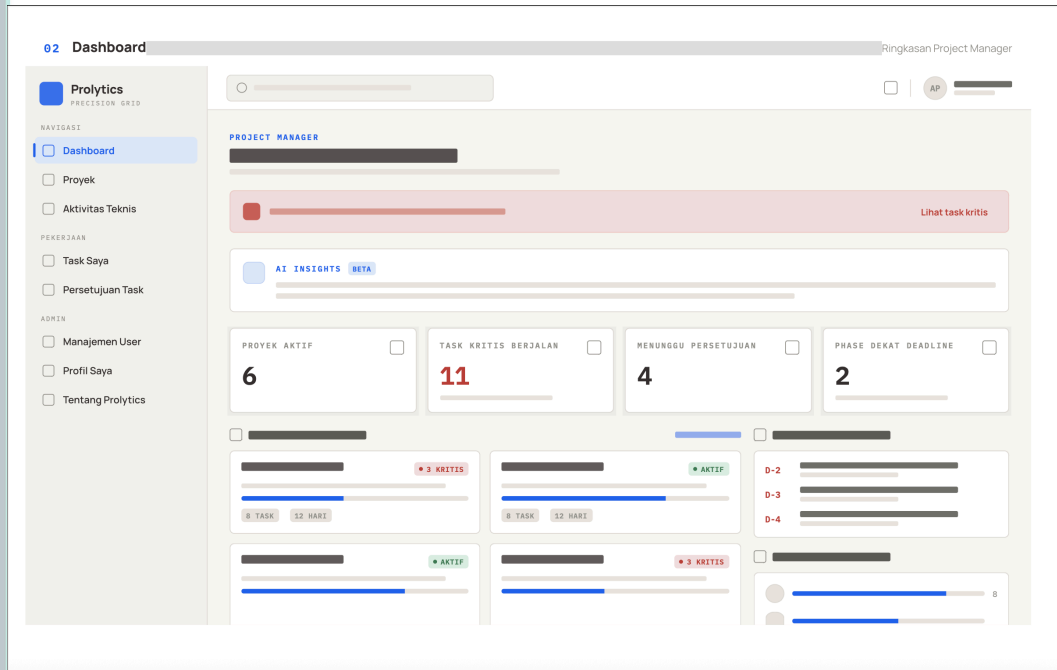
Dashboard Project Manager menyajikan ringkasan kondisi seluruh proyek dalam satu tampilan. Wireframe pada Gambar 4.15 menempatkan kartu indikator utama di bagian atas, yaitu Proyek Aktif, Tugas Kritis, Tugas Menunggu Persetujuan, dan *Phase* Mendekati Tenggat, diikuti panel AI Insights serta banner peringatan tugas kritis. Tata letak ini membantu *Project Manager* memahami kondisi umum sistem tanpa harus membuka setiap proyek satu per satu. Kartu indikator dirancang menonjol dengan angka berukuran besar agar mudah dibaca sekilas, sementara panel di bawahnya menyediakan konteks tambahan berupa daftar proyek dan aktivitas terkini. Penempatan banner peringatan pada posisi yang menonjol bertujuan menarik perhatian terhadap tugas kritis yang membutuhkan tindakan segera.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

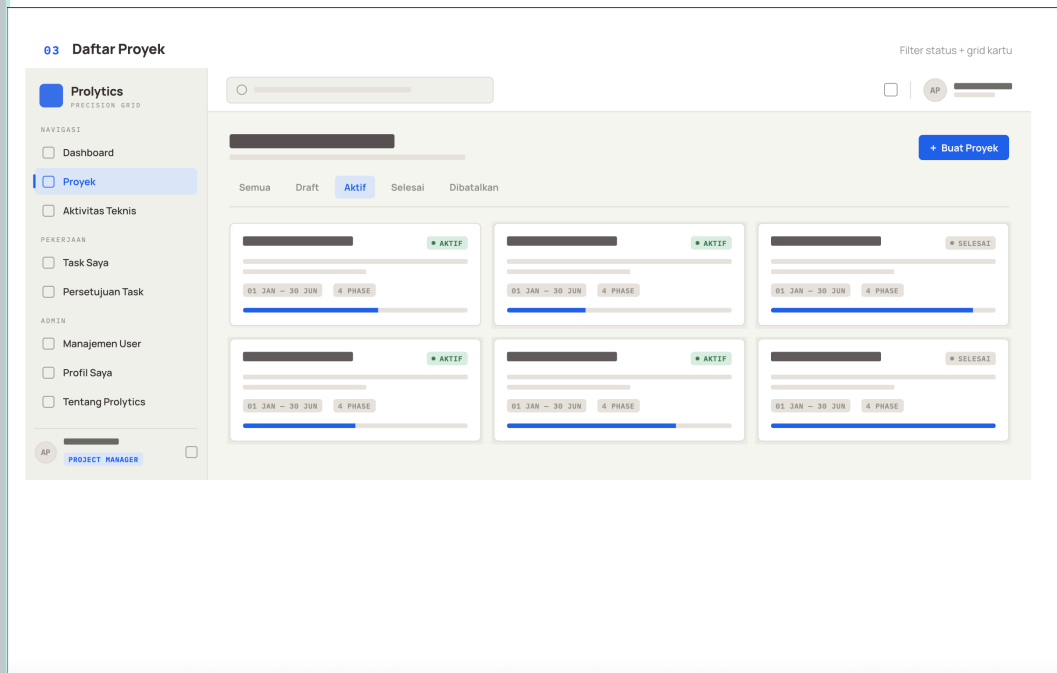


Gambar 4. 15 Wireframe Dashboard Project Manager

Gambar 4.15 menunjukkan tata letak *dashboard Project Manager* secara menyeluruh, termasuk daftar aktivitas terbaru dan ringkasan progres antarproyek. Bagian ini melengkapi indikator utama pada halaman sebelumnya dengan informasi yang lebih rinci sebagai dasar pemantauan harian. Susunan komponen mengikuti pola dari ringkasan menuju detail, sehingga *Project Manager* dapat menelusuri kondisi proyek dari gambaran umum hingga informasi tugas tertentu. Konsistensi tata letak, warna, dan komponen dijaga di seluruh halaman agar pengalaman penggunaan tetap seragam pada setiap peran.

Halaman daftar proyek digunakan untuk menelusuri seluruh proyek berdasarkan statusnya. Wireframe pada Gambar 4.16 menyediakan filter status (Semua, Draft, Aktif, Selesai, Dibatalkan), grid kartu proyek yang menampilkan ringkasan tiap proyek, serta tombol Buat Proyek untuk menambah proyek baru. Rancangan ini menekankan kemudahan navigasi antarstatus proyek. Setiap kartu proyek memuat nama proyek, status, dan ringkasan progres sehingga pengguna dapat menilai kondisi proyek tanpa membuka halaman detail. Filter status ditempatkan pada bagian atas agar penyaringan dapat dilakukan dengan cepat, sedangkan grid kartu

disusun responsif mengikuti lebar layar untuk menjaga keterbacaan pada berbagai perangkat.



Gambar 4. 16 *Wireframe* Halaman Daftar Proyek

Gambar 4.16 menunjukkan tata letak halaman daftar proyek yang memungkinkan pengguna memfilter dan menavigasi antar proyek berdasarkan status. Filter status yang tersedia memudahkan Project Manager menemukan proyek yang sedang aktif, dalam tahap *draft* atau telah selesai.

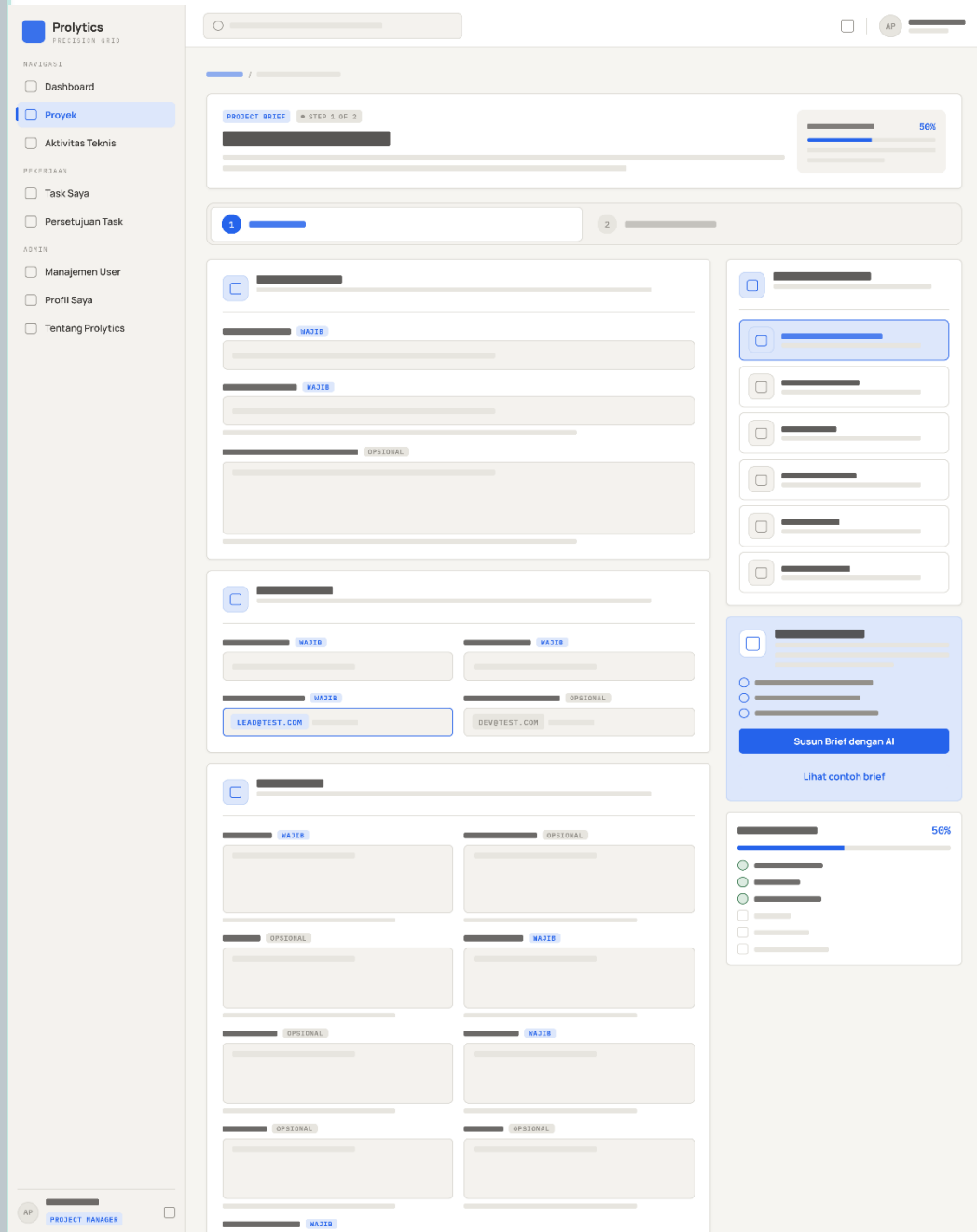
Halaman membuat proyek digunakan oleh Project Manager untuk menginisiasi proyek baru dengan mengisi data awal yang diperlukan. *Wireframe* pada Gambar 4.17 menampilkan formulir pembuatan proyek yang terdiri atas kolom nama proyek, deskripsi singkat, tanggal mulai, dan pengaturan anggota tim. Rancangan ini dirancang agar Project Manager dapat menyiapkan kerangka proyek sebelum breakdown teknis dibuat oleh Lead Developer.



Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



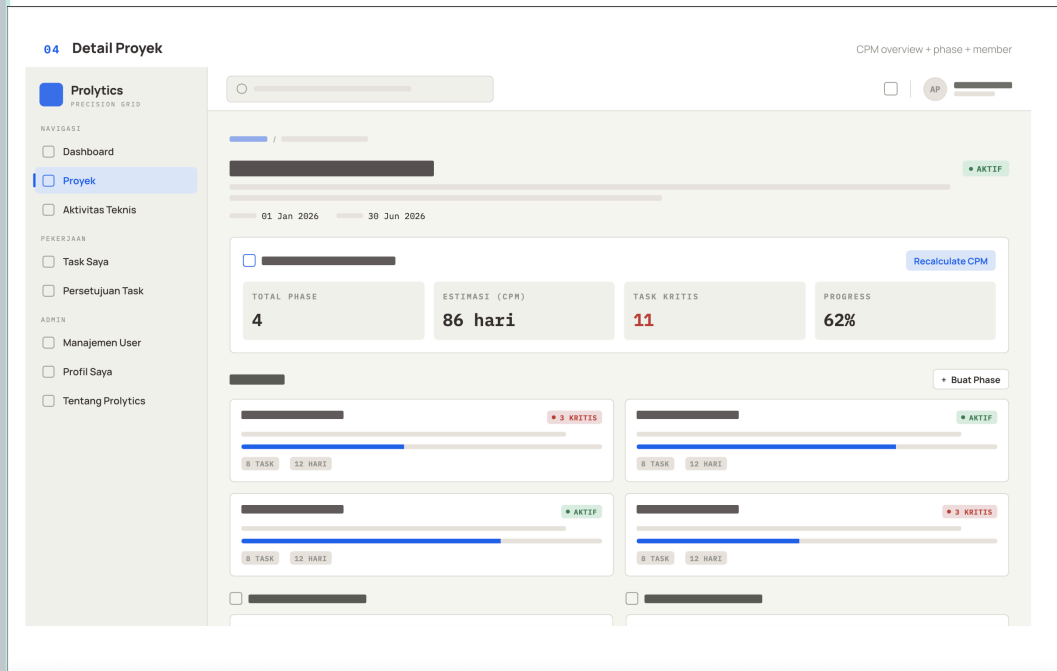
Gambar 4. 17 Wireframe halaman membuat Proyek

Gambar 4.17 menunjukkan tata letak formulir pembuatan proyek yang sederhana dan terstruktur, memungkinkan Project Manager mengisi seluruh data awal proyek dalam satu halaman sebelum melanjutkan ke tahap penyusunan breakdown teknis. Halaman detail proyek menjadi pusat pemantauan satu proyek tertentu. Wireframe pada Gambar 4.18 menampilkan metrik utama proyek Total Phase, Estimasi CPM,

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

jumlah Tugas Kritis, dan Progress diikuti daftar kartu phase serta tombol Recalculate CPM dan Buat Phase. Tata letak ini memfokuskan pengguna pada kondisi CPM proyek tanpa menampilkan klaim prediksi tanggal selesai.



Gambar 4. 18 Wireframe Halaman Detail Proyek

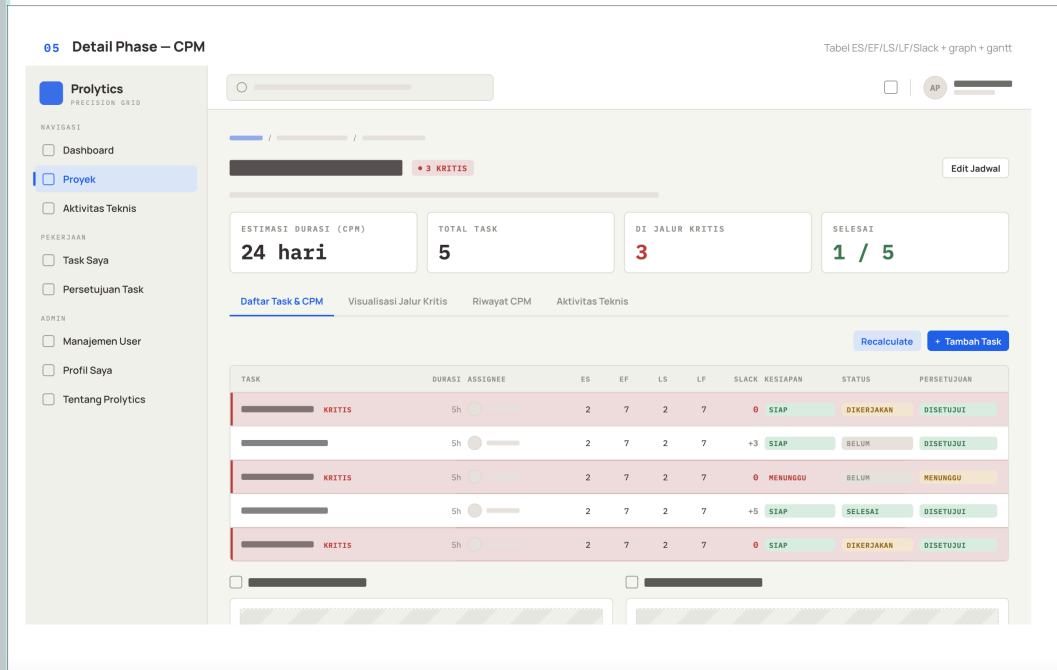
Gambar 4.18 menunjukkan halaman detail project sebagai pusat pemantauan kondisi project secara menyeluruh. Metrik utama yang ditampilkan mencakup Total Phase, jumlah Tugas Kritis, dan progres keseluruhan project, sehingga Project Manager dapat memperoleh gambaran kondisi project tanpa harus menelusuri detail setiap phase.

Halaman detail phase menampilkan hasil perhitungan *Critical Path Method (CPM) project-wide* yang dikelompokkan berdasarkan phase sebagai *container*. Wireframe pada Gambar 4.19 menampilkan kartu ringkasan *phase* (Estimasi durasi, Total Tugas, Tugas di *Jalur Kritis*, dan Tugas Selesai) serta tabel hasil CPM dengan kolom Tugas, Durasi, Assignee, ES, EF, LS, LF, *Slack*, Kesiapan, Status, dan Persetujuan. Tersedia pula tab Visualisasi *Jalur Kritis* dan Riwayat CPM.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



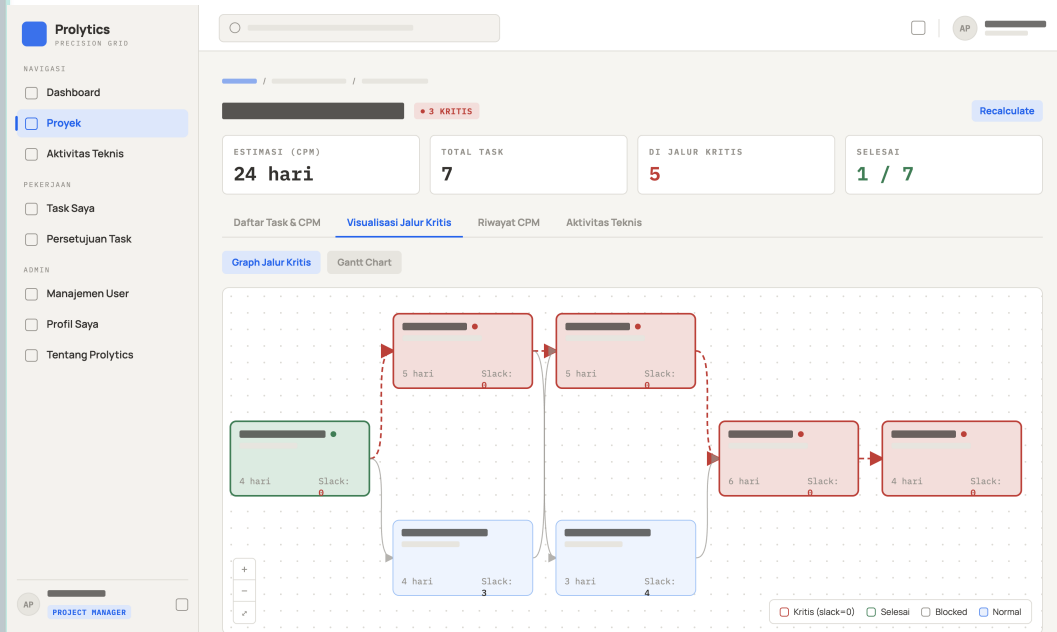
Gambar 4. 19 Wireframe Halaman Detail *Phase* dan CPM

Gambar 4.19 menunjukkan tata letak halaman detail phase yang merangkum hasil perhitungan *CPM project-wide* dalam satu tampilan. Tabel hasil CPM menampilkan nilai ES, EF, LS, LF, slack, dan status approval setiap tugas yang dihitung oleh backend, sedangkan tab tambahan menyediakan akses ke visualisasi jalur kritis dan riwayat *snapshot CPM*.

Selain tabel hasil CPM, sistem juga menyediakan tampilan visual untuk memudahkan pemahaman jalur kritis. Visualisasi jalur kritis dalam bentuk graf ditunjukkan pada Gambar 4.20, sedangkan diagram Gantt yang menampilkan timeline tugas berdasarkan hasil CPM ditunjukkan pada Gambar 4.17.

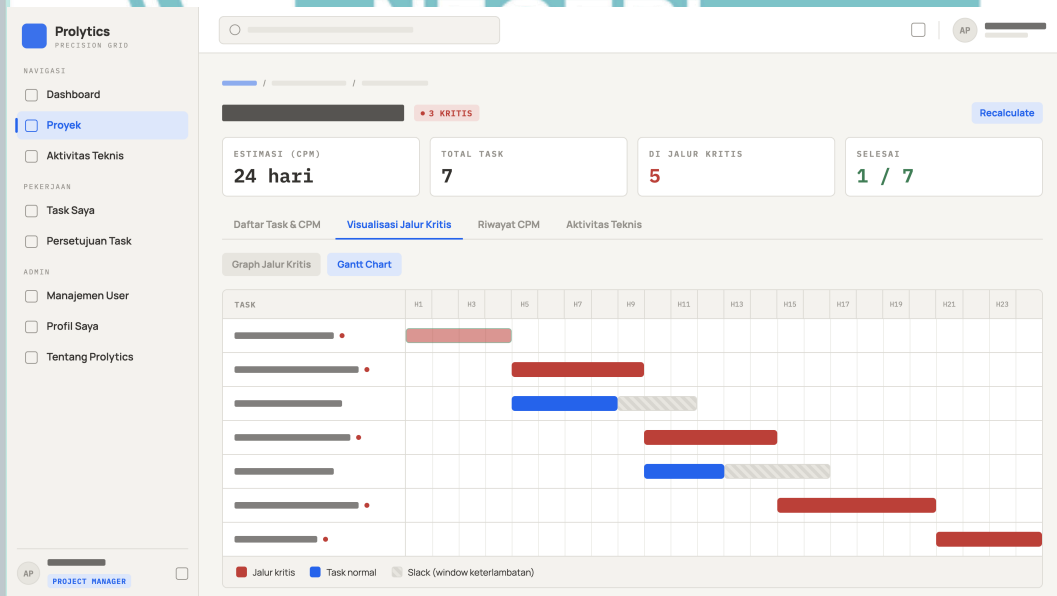
Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 4. 20 *Wireframe Visualisasi Jalur Kritis*

Gambar 4.20 menunjukkan visualisasi *dependency* dan jalur kritis dalam bentuk graf. Node merepresentasikan *tugas* dan *edge* menunjukkan relasi *dependency* lintas phase tugas yang masuk ke jalur kritis ditandai secara visual sehingga pengguna dapat langsung mengidentifikasi urutan pengerjaan yang paling *kritikal* dalam satu project.



Gambar 4. 21 *Wireframe Visualisasi Gantt Phase*



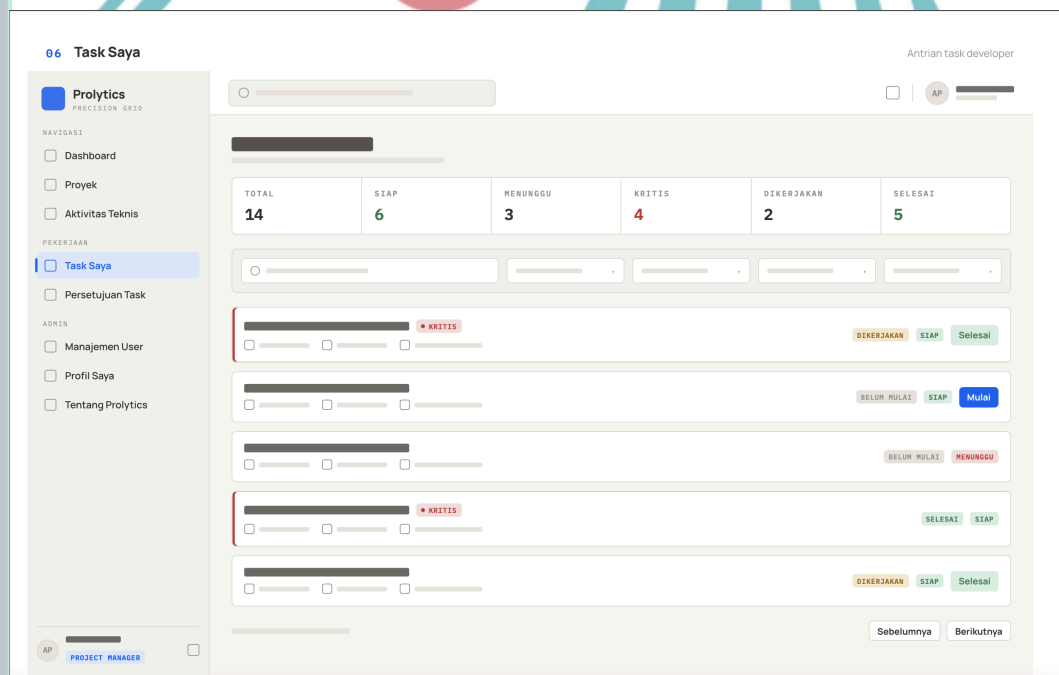
© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.21 menunjukkan diagram Gantt yang memvisualisasikan timeline tugas berdasarkan nilai ES dan EF hasil perhitungan *CPM project-wide*. Tugas dalam jalur kritis ditandai berbeda dengan tugas non-kritis, membantu Project Manager dan Lead Developer memahami distribusi *timeline tugas* secara keseluruhan.

Halaman Tugas Saya digunakan oleh *Software Developer* untuk memantau tugas yang ditugaskan kepadanya. *Wireframe* pada Gambar 4.22 menampilkan kartu ringkasan status tugas (Total, Siap, Menunggu, Kritis, Dikerjakan, dan Selesai) diikuti antrian tugas yang dapat diperbarui statusnya. Rancangan ini memudahkan developer fokus pada pekerjaan yang menjadi tanggung jawabnya.



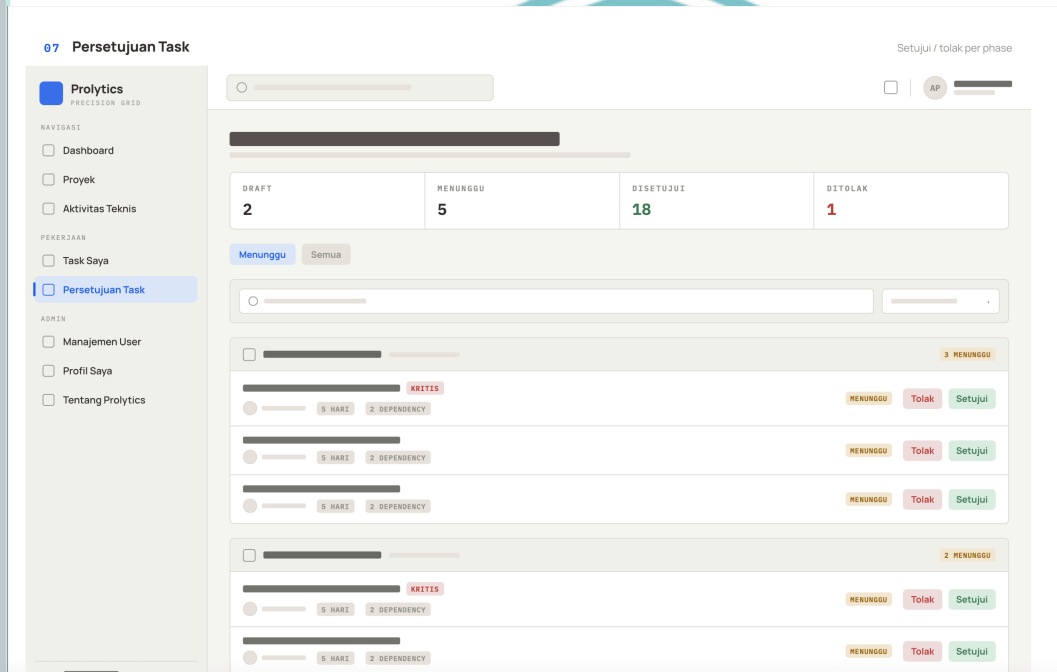
Gambar 4. 22 *Wireframe* Halaman Tugas Saya

Gambar 4.22 menunjukkan tata letak halaman Tugas Saya yang berpusat pada daftar tugas yang ditugaskan kepada *Software Developer*. Kartu ringkasan status memudahkan developer mengetahui jumlah tugas yang siap dikerjakan, masih menunggu *predecessor*, atau sedang dalam pengerjaan, sehingga prioritas pekerjaan dapat ditentukan dengan jelas.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Halaman persetujuan tugas digunakan untuk meninjau technical tugas yang diajukan sebelum menjadi tugas resmi. Wireframe pada Gambar 4.23 menampilkan ringkasan status persetujuan (Draft, Menunggu, Disetujui, Ditolak) serta daftar tugas yang dikelompokkan berdasarkan *phase* beserta aksi setuju atau tolak. Pengelompokan ini membantu peninjauan tugas sesuai konteks *phase*.



Gambar 4. 23 *Wireframe* Halaman Persetujuan Tugas

Gambar 4.23 menunjukkan tata letak halaman persetujuan tugas yang dikelompokkan berdasarkan *phase*. Project Manager dapat meninjau setiap tugas yang diajukan oleh Lead Developer dan memberikan keputusan setuju, tolak, atau minta revisi; hanya tugas berstatus approved yang dimasukkan ke dalam *active* CPM proyek.

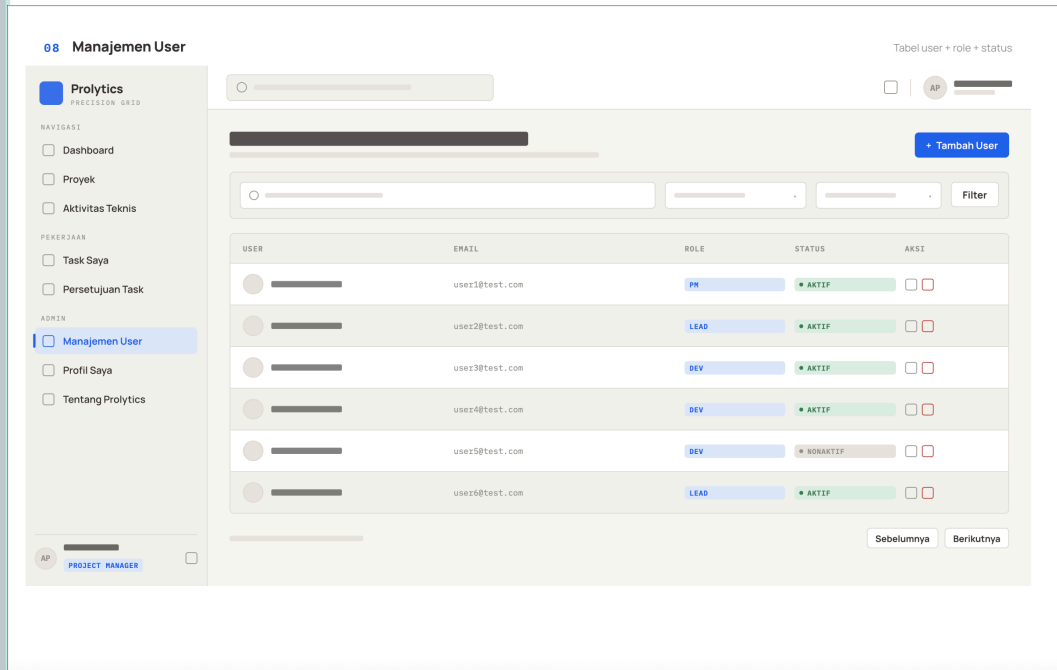
Halaman manajemen user digunakan oleh *Project Manager* untuk mengelola akun pengguna sistem. Wireframe pada Gambar 4.24 menampilkan tabel pengguna dengan kolom User, Email, *Role*, dan Status, serta kolom Aksi untuk mengaktifkan atau menonaktifkan akun. Rancangan ini mendukung pengaturan hak akses pengguna sesuai perannya.



Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 4. 24 *Wireframe* Halaman Manajemen User

Gambar 4.24 menunjukkan tata letak halaman manajemen pengguna yang memungkinkan Project Manager mengatur keanggotaan dan hak akses tim. Tabel pengguna menampilkan identitas, role, dan status akun, serta menyediakan aksi untuk mengaktifkan atau menonaktifkan akun sesuai kebutuhan proyek.

4.2.3 Perancangan Hak Akses Pengguna

Hak akses pengguna dirancang berdasarkan tiga *role* utama yang digunakan dalam sistem. Pembagian ini diperlukan agar setiap pengguna hanya dapat mengakses fungsi yang sesuai dengan tanggung jawabnya dalam proses *monitoring* proyek.

Tabel 9 Rancangan Tabel Hak Akses Pengguna

Fitur	<i>Project Manager</i>	<i>Lead Developer</i>	<i>Software Developer</i>
Manajemen user	Mengelola akun pengguna	Tidak memiliki akses	Tidak memiliki akses
Manajemen <i>project</i>	Membuat, mengubah, menghapus, dan	Mengakses <i>project</i> sesuai membership	Mengakses <i>project</i> sesuai membership

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

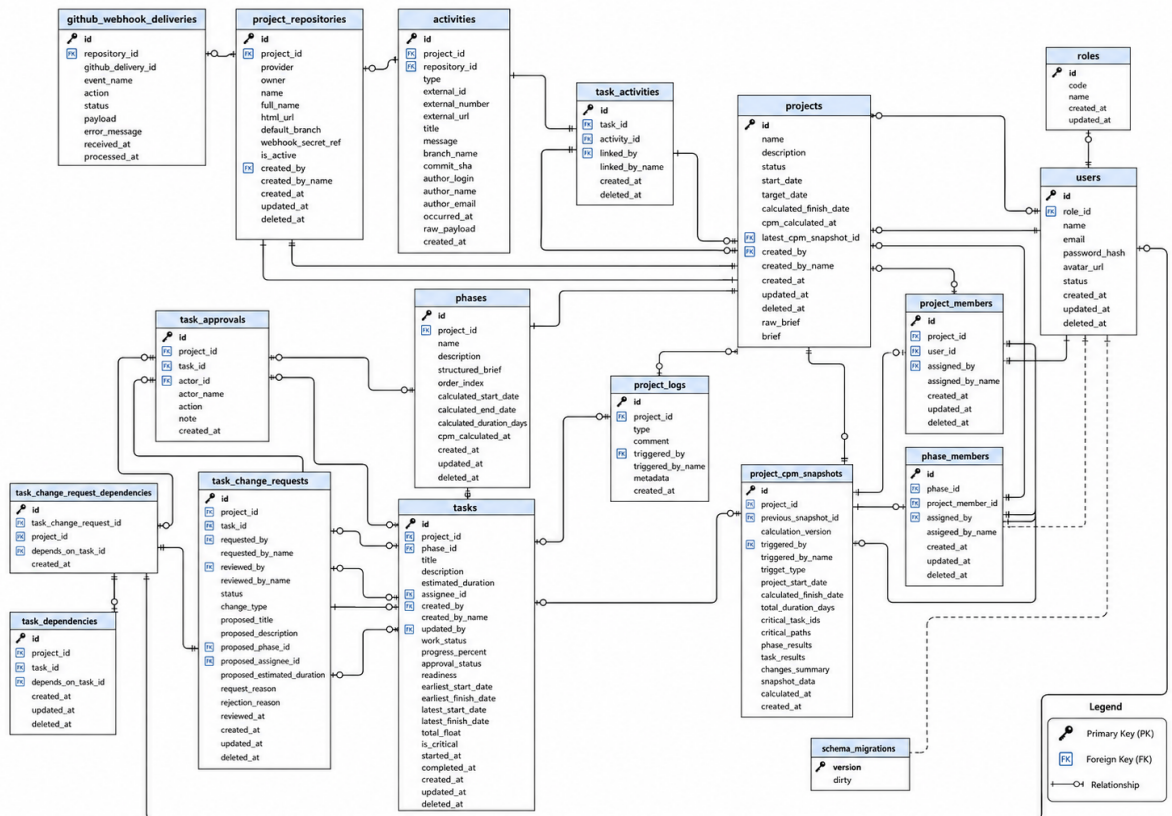
Fitur	<i>Project Manager</i>	<i>Lead Developer</i>	<i>Software Developer</i>
	mengaktifkan <i>project</i>		
Manajemen <i>phase</i>	Mengelola <i>phase</i> dan jadwal <i>phase</i>	Membuat atau mengubah metadata <i>phase</i> sesuai kewenangan	Melihat <i>phase</i> yang dapat diakses
<i>Requirement</i> dan technical tugas	Melihat dan meninjau tugas	Membuat, mengubah, menghapus, dan mengatur <i>dependency</i> tugas	Melihat dan memperbarui status tugas yang ditugaskan
<i>Approval workflow</i>	Menyetujui, menolak, atau meminta revisi tugas	Mengajukan tugas dan merevisi tugas	Tidak memiliki akses <i>approval</i>
CPM dan riwayat <i>snapshot</i>	Melihat hasil CPM dan riwayat	Melihat hasil CPM dan memicu perhitungan sesuai kewenangan	Melihat informasi tugas yang relevan
GitHub activity	Melihat aktivitas teknis proyek	Mengelola <i>repository</i> dan menautkan aktivitas tugas	Menautkan aktivitas pada tugas yang terkait

Tabel 7 memperlihatkan bahwa *Project Manager* memiliki kewenangan tertinggi pada pengelolaan proyek dan persetujuan, *Lead Developer* berfokus pada *breakdown* teknis dan pengaturan tugas, sedangkan *Software Developer* berfokus pada pelaksanaan tugas. Rancangan ini menjadi dasar implementasi *Role-Based Access Control* (RBAC) pada *backend* dan penyesuaian menu pada *frontend*.



4.2.4 Perancangan Basis Data

Basis data dirancang untuk menyimpan data *project*, *phase*, tugas, *dependency*, user, membership, *approval*, *snapshot* CPM, *repository* GitHub, aktivitas teknis, dan *log* perencanaan. Relasi antar entitas dibutuhkan agar sistem dapat menjaga konsistensi alur kerja dari pembuatan *project* hingga *monitoring jalur kritis*.



Gambar 4. 25 Diagram Relasi Tabel

Gambar 4.25 digunakan untuk menunjukkan hubungan data utama pada sistem, terutama relasi *project* dengan *phase*, *phase* dengan tugas, tugas dengan *dependency*, serta tugas dengan *approval* dan *snapshot* CPM. Relasi tersebut penting karena perhitungan CPM membutuhkan data *approved tugas* dan *dependency* yang valid dalam satu *project*.

4.2.5 Perancangan Algoritma CPM

Perancangan algoritma CPM disusun berdasarkan input *project_start_date*, *approved tugas*, *estimated_duration*, dan *dependency* pada level *project*. Backend

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

menjadi satu-satunya sumber perhitungan CPM frontend hanya merender hasil yang dikirimkan oleh backend. Algoritma ini tidak digunakan untuk menggantikan keputusan Project Manager atau Lead Developer, tetapi untuk mengidentifikasi tugas yang berpengaruh langsung terhadap durasi penyelesaian project.

Tabel 10 Rancangan Algoritma Critical Path Method

Tahap	Deskripsi
<i>Input data</i>	Sistem mengambil <code>project_start_date</code> , daftar approved tugas, <code>estimated_duration</code> , dan <code>dependency predecessor-successor</code> dalam satu project. Tugas dengan status pending atau rejected dikecualikan dari perhitungan.
Validasi graf	<i>Dependency</i> tugas harus membentuk <i>Directed Acyclic Graph</i> (DAG) sehingga tidak terdapat <i>circular dependency</i> .
<i>Forward pass</i>	Sistem menghitung Early Start (ES) dan Early Finish (EF) berbasis tanggal kalender. Tugas tanpa predecessor: $ES = \text{project_start_date}$; $EF = ES + \text{estimated_duration} - 1$ hari. Tugas dengan predecessor: $ES = \max(EF \text{ predecessor}) + 1$ hari; $EF = ES + \text{estimated_duration} - 1$ hari.
<i>Backward pass</i>	Sistem menghitung Late Finish (LF) dan Late Start (LS) berbasis tanggal kalender. $LF \text{ tugas akhir} = \text{project calculated finish date}$ (max EF seluruh tugas). $LS = LF - \text{estimated_duration} + 1$ hari. Untuk tugas di tengah jalur: $LF = \min(LS \text{ successor}) - 1$ hari.
<i>Slack/float</i>	<i>Slack</i> dihitung dari selisih tanggal LS dan ES dalam hari kalender. Tugas dengan <i>slack</i> bernilai nol ditandai sebagai <i>tugas kritis</i> .
<i>Critical path</i>	Sistem membentuk satu atau lebih <i>jalur kritis</i> berdasarkan <i>tugas kritis</i> yang saling terhubung.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Tahap	Deskripsi
<i>Snapshot</i>	Pada <i>project</i> aktif, hasil authoritative disimpan sebagai <i>project_cpm_snapshots</i> agar perubahan CPM pada level <i>project</i> dapat ditelusuri secara historis.

Tabel 7 menjelaskan bahwa keluaran CPM berupa nilai ES, EF, LS, LF (dalam tanggal kalender), slack/float dalam satuan hari, tugas kritis, critical path, serta *calculated_start_date* dan *calculated_end_date* phase dari rollup tugas approved. Keluaran tersebut digunakan sebagai informasi *monitoring* agar pengguna dapat mengetahui tugas mana yang perlu diprioritaskan.

4.2.6 Perhitungan Critical Path Method (CPM)

Untuk menunjukkan cara kerja algoritma CPM yang telah dirancang, dilakukan contoh perhitungan menggunakan data salah satu proyek perangkat lunak yang dikelola, yaitu rencana pengembangan sistem Prolytics. Rincian backlog pengembangan sistem tersebut disajikan pada Lampiran 5. Tahap pertama adalah mengidentifikasi kegiatan, durasi, dan ketergantungan antar-kegiatan sebagaimana ditunjukkan pada Tabel 11.

Tabel 11 Identifikasi Kegiatan Proyek

Kode	Kegiatan	Durasi (hari)	Pendahulu
A	Perancangan dan setup basis data	4	-
B	Autentikasi dan RBAC	5	A
C	Manajemen pengguna dan profil	4	B
D	Manajemen project, phase, dan tugas	8	B
E	Pengaturan dependency dan validasi graf	5	D
F	Modul perhitungan CPM	9	E
G	Approval workflow dan change request	6	D
H	Integrasi GitHub	5	D
I	Modul AI Brief	3	D
J	Antarmuka frontend	8	C, F, G
K	Integrasi, dokumentasi API, dan pengujian	5	H, I, J

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Setelah kegiatan dan ketergantungannya ditentukan, dilakukan perhitungan maju untuk memperoleh ES dan EF sesuai persamaan (2.1), dengan hasil sebagai berikut:

$EF(A) = 0 + 4 = 4$; $EF(B) = 4 + 5 = 9$; $EF(C) = 9 + 4 = 13$; $EF(D) = 9 + 8 = 17$; $EF(E) = 17 + 5 = 22$; $EF(F) = 22 + 9 = 31$; $EF(G) = 17 + 6 = 23$; $EF(H) = 17 + 5 = 22$; $EF(I) = 17 + 3 = 20$; $EF(J) = \max(EF(C)=13, EF(F)=31, EF(G)=23) + 8 = 31 + 8 = 39$; $EF(K) = \max(EF(H)=22, EF(I)=20, EF(J)=39) + 5 = 39 + 5 = 44$.

Selanjutnya dilakukan perhitungan mundur untuk memperoleh LS dan LF sesuai persamaan (2.2), dimulai dari durasi total proyek sebesar 44 hari, dengan hasil sebagai berikut: $LS(K) = 44 - 5 = 39$; $LS(J) = 39 - 8 = 31$; $LS(I) = 39 - 3 = 36$; $LS(H) = 39 - 5 = 34$; $LS(G) = 31 - 6 = 25$; $LS(F) = 31 - 9 = 22$; $LS(E) = 22 - 5 = 17$; $LS(D) = \min(LS(E)=17, LS(G)=25, LS(H)=34, LS(I)=36) - 8 = 17 - 8 = 9$; $LS(C) = 31 - 4 = 27$; $LS(B) = \min(LS(C)=27, LS(D)=9) - 5 = 9 - 5 = 4$; $LS(A) = 4 - 4 = 0$.

Tabel 12 Hasil Perhitungan Maju dan Mundur

Kode	Durasi	Pendahulu	ES	EF	LS	LF
A	4	-	0	4	0	4
B	5	A	4	9	4	9
C	4	B	9	13	27	31
D	8	B	9	17	9	17
E	5	D	17	22	17	22
F	9	E	22	31	22	31
G	6	D	17	23	25	31
H	5	D	17	22	34	39
I	3	D	17	20	36	39
J	8	C, F, G	31	39	31	39
K	5	H, I, J	39	44	39	44

Untuk menentukan jalur kritis, dihitung *total float* setiap kegiatan sesuai persamaan (2.3). Hasil perhitungan CPM beserta nilai *float* disajikan pada Tabel 13.

Tabel 13 Hasil Perhitungan Critical Path Method

Kode	Durasi	ES	EF	LS	LF	Float
A	4	0	4	0	4	0
B	5	4	9	4	9	0



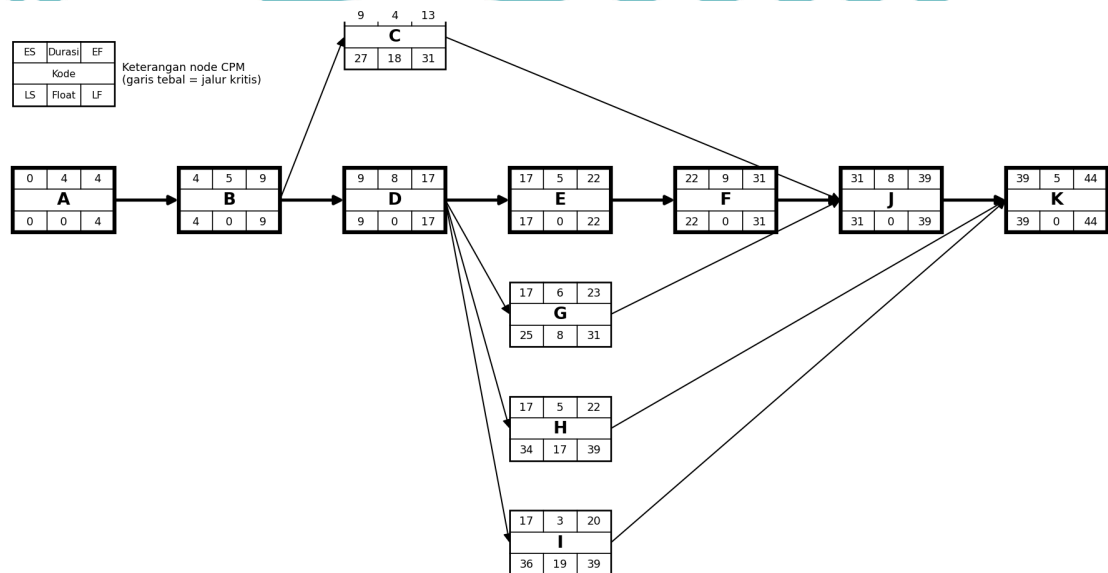
© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode	Durasi	ES	EF	LS	LF	Float
C	4	9	13	27	31	18
D	8	9	17	9	17	0
E	5	17	22	17	22	0
F	9	22	31	22	31	0
G	6	17	23	25	31	8
H	5	17	22	34	39	17
I	3	17	20	36	39	19
J	8	31	39	31	39	0
K	5	39	44	39	44	0

Grafik hasil perhitungan CPM ditunjukkan pada Gambar 4.26. Setiap node menampilkan nilai ES, durasi, EF, LS, *float*, dan LF, sedangkan garis tebal menandai jalur kritis.



Gambar 4. 26 Grafik Hasil *Critical Path Method*

Berdasarkan Tabel 13 dan Gambar 4.26, kegiatan yang memiliki *float* bernilai nol adalah A, B, D, E, F, J, K. Kegiatan tersebut membentuk jalur kritis A - B - D - E - F - J - K dengan durasi total proyek sebesar 44 hari. Jalur kritis ini menunjukkan urutan pekerjaan yang tidak boleh terlambat, yaitu dari perancangan basis data, autentikasi, manajemen *project* dan tugas, pengaturan *dependency*, modul

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

perhitungan CPM, antarmuka, hingga integrasi dan pengujian akhir. Kegiatan di luar jalur kritis, seperti manajemen pengguna dan profil, *approval workflow*, integrasi GitHub, serta modul AI *Brief*, memiliki *float* sehingga keterlambatannya tidak langsung memengaruhi durasi total proyek selama masih dalam batas *float*. Nilai ES, EF, LS, dan LF pada contoh ini dinyatakan dalam satuan hari menggunakan konvensi elapsed-day ($EF = ES + \text{durasi}$) sesuai persamaan (2.1) sampai (2.3). Pada implementasi, sistem memakai konvensi hari kalender inklusif sebagaimana Tabel 10, yaitu $EF = ES + \text{durasi} - 1$ dan $LF = LS + \text{durasi} - 1$, sehingga nilai EF dan LF menjadi satu hari lebih awal dibandingkan tabel di atas, sedangkan nilai ES, LS, total float, jalur kritis, dan durasi total proyek sebesar 44 hari tetap sama. Nilai tersebut kemudian dipetakan ke tanggal kalender berdasarkan `project_start_date`.

4.3 Implementasi Sistem

Implementasi sistem dilakukan berdasarkan hasil analisis kebutuhan dan perancangan yang telah dijelaskan sebelumnya. Sistem Prolytics dikembangkan sebagai aplikasi web yang menghubungkan *frontend*, *backend*, basis data, integrasi GitHub, layanan LLM, CPM engine, dan mekanisme RBAC dalam satu alur *monitoring* proyek perangkat lunak.

Agar pembahasan lebih runtut, implementasi dijelaskan berdasarkan teknologi, struktur *project*, hak akses, dan modul utama sistem. Penyusunan modular ini bertujuan untuk memperlihatkan hubungan antara kebutuhan pengguna dengan komponen sistem yang dibangun tanpa menjadikan laporan sebagai dokumentasi teknis kode secara mentah.

4.3.1 Teknologi dan Tools Implementasi

Teknologi yang digunakan dipilih berdasarkan kebutuhan sistem yang memerlukan antarmuka web interaktif, API terstruktur, penyimpanan relasional, validasi akses, serta dukungan integrasi eksternal. Tabel berikut merangkum peran setiap teknologi dalam implementasi Prolytics.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Tabel 14 Teknologi dan Tools

Teknologi/Tools	Peran dalam Implementasi
Next.js	Membangun <i>frontend</i> berbasis React, routing halaman, server component, server action, dan integrasi antarmuka dengan API <i>backend</i> .
TypeScript	Memberikan tipe data pada <i>frontend</i> agar kontrak data <i>project</i> , tugas, CPM, dan <i>approval</i> lebih konsisten.
Tailwind CSS	Membantu penyusunan tampilan antarmuka yang konsisten, responsif, dan mudah disesuaikan.
Golang	Membangun <i>backend</i> dengan performa baik dan struktur service yang eksplisit untuk kebutuhan domain proyek.
Gin	Menyediakan routing <i>RESTful</i> API, <i>middleware</i> autentikasi, dan pengelolaan <i>request-response backend</i> .
PostgreSQL	Menyimpan data utama sistem seperti user, <i>project</i> , <i>phase</i> , tugas, <i>dependency</i> , <i>approval</i> , <i>snapshot</i> , dan aktivitas GitHub.
Bun ORM	Memetakan struktur data Go dengan tabel PostgreSQL serta menyederhanakan <i>query repository</i> .
JWT	Menerapkan autentikasi pengguna dan membawa identitas <i>role</i> untuk validasi akses.
Swagger/OpenAPI	Mendokumentasikan kontrak API agar <i>endpoint backend</i> dapat diuji dan dipahami secara konsisten.
GitHub <i>Webhook</i>	Menerima aktivitas teknis <i>repository</i> sebagai data pendukung <i>monitoring</i> proyek.
Gemini/LLM	Membantu penyusunan draft <i>requirement</i> dan rencana tugas yang tetap harus divalidasi pengguna.

Berdasarkan Tabel 14, pemilihan Next.js dan TypeScript mendukung pengembangan antarmuka yang terstruktur, sedangkan Golang, Gin, dan PostgreSQL digunakan untuk membangun *backend* yang kuat pada sisi domain bisnis. Swagger digunakan untuk menjaga keterbacaan kontrak API, sementara

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

GitHub *Webhook* dan LLM diposisikan sebagai fitur pendukung yang memperkaya proses *monitoring* dan perencanaan.

4.3.2 Struktur *Project*

Struktur *project* dipisahkan berdasarkan tanggung jawab *frontend* dan *backend*. Pemisahan ini membantu pengembangan sistem karena *logika* antarmuka, komunikasi API, aturan bisnis, akses data, *middleware*, integrasi eksternal, dan engine CPM ditempatkan pada bagian yang berbeda.

Tabel 15 Struktur Project Prolytics

Bagian <i>Project</i>	Deskripsi Tanggung Jawab
Prolytics-fe/src/app	Mengatur halaman aplikasi, routing, layout, dan server action pada <i>frontend</i> .
Prolytics-fe/src/components	Menyimpan komponen antarmuka seperti tabel CPM, <i>approval</i> queue, dialog tugas, <i>dashboard</i> , dan form <i>project</i> .
Prolytics-fe/src/lib/ <i>backend</i>	Menyediakan adapter API untuk menghubungkan <i>frontend</i> dengan <i>endpoint backend</i> .
Prolytics-fe/src/types	Menyimpan tipe data <i>frontend</i> agar kontrak <i>response project</i> , tugas, CPM, <i>approval</i> , dan GitHub konsisten.
Prolytics-be/internal/controllers	Menerima <i>request</i> API dan mengirimkan <i>response</i> sesuai hasil service.
Prolytics-be/internal/services	Menangani aturan bisnis seperti <i>project</i> , <i>phase</i> , tugas, <i>approval</i> , CPM, LLM, GitHub, dan access policy.
Prolytics-be/internal/repository	Mengakses data PostgreSQL dan menyusun <i>query</i> untuk kebutuhan domain.
Prolytics-be/internal/domain	Mendefinisikan model data yang merepresentasikan tabel utama sistem.
Prolytics-be/internal/routes	Mendaftarkan <i>endpoint</i> API dan <i>middleware</i> sesuai kebutuhan modul.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Bagian <i>Project</i>	Deskripsi Tanggung Jawab
Prolytics-be/internal/services/cpm	Menyimpan engine CPM murni untuk validasi graf, <i>forward pass</i> , <i>backward pass</i> , <i>slack</i> , dan <i>critical path</i> .

Tabel 15 menunjukkan bahwa *frontend* berperan sebagai lapisan presentasi dan interaksi pengguna, sedangkan *backend* berperan sebagai pusat validasi, orkestrasi proses bisnis, dan perhitungan CPM. Struktur ini juga memudahkan pelacakan perubahan karena modul utama seperti *approval*, GitHub, LLM, dan CPM memiliki lokasi implementasi yang jelas.

4.3.3 Implementasi *Backend* dan Basis Data

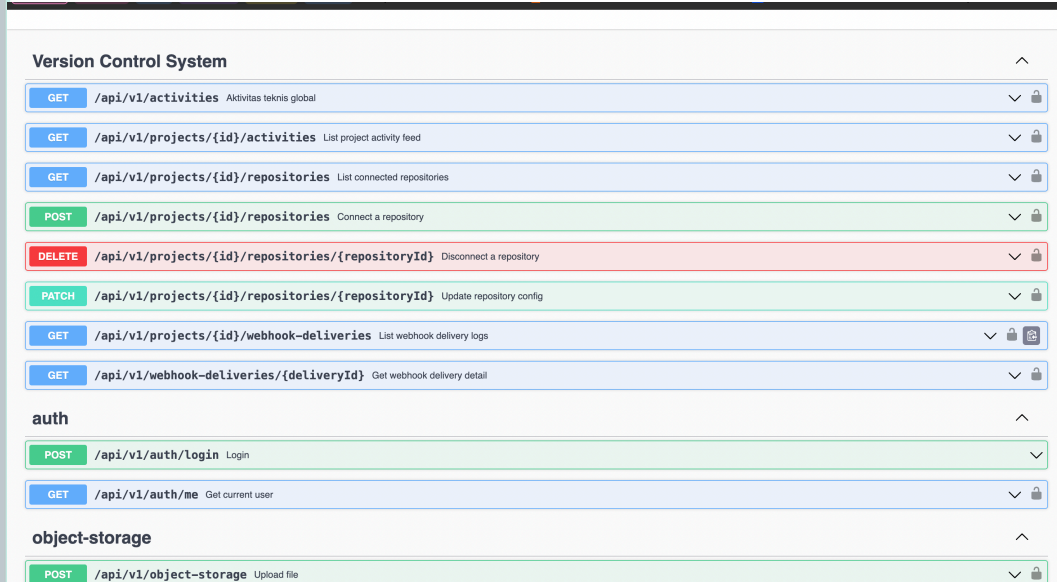
Backend Prolytics dibangun menggunakan bahasa pemrograman Go dengan *framework* Gin dan berperan sebagai pusat pengolahan data, orkestrasi proses bisnis, validasi akses, serta penyedia *RESTful API* bagi *frontend*. Seluruh komunikasi antara *frontend* dan data sistem dilakukan melalui *endpoint* API yang disediakan *backend*, mulai dari pengelolaan *project* dan *phase*, manajemen tugas dan *dependency*, *approval workflow*, perhitungan *Critical Path Method* (CPM), integrasi GitHub, hingga fitur perencanaan berbantuan *Large Language Model* (LLM).

Dokumentasi API Prolytics disusun menggunakan *Swagger/OpenAPI*. Dokumentasi ini mendeskripsikan seluruh *endpoint* yang tersedia, parameter *request*, format *response*, serta kebutuhan *autentikasi* pada setiap *endpoint*. Dengan dokumentasi yang terstandarisasi, proses pengujian *endpoint*, validasi kontrak API, dan konsistensi komunikasi antara *backend* dan *frontend* dapat terjaga selama proses pengembangan.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 4. 27 Tampilan Dokumentasi API Swagger Prolytics

Gambar 4.27 menampilkan antarmuka dokumentasi *Swagger Prolytics* yang dapat diakses melalui *browser*. Halaman tersebut mengelompokkan *endpoint* berdasarkan modul, menampilkan parameter *request* yang diperlukan, format *response* yang dikembalikan, serta keterangan apakah *endpoint* membutuhkan *token JWT* sebagai syarat *autentikasi*. Dokumentasi ini digunakan untuk memastikan bahwa setiap *endpoint* terdefinisi dengan jelas sebelum diintegrasikan dengan *frontend*.

Pola kerja *backend* Prolytics mengikuti alur yang terstruktur dan berlapis. Setiap *request* dari *frontend* masuk ke *router Gin* dan melewati *middleware autentikasi* berbasis *JWT* serta *middleware otorisasi* berbasis *role* dan konteks *project*. Setelah melewati *middleware*, *request* diteruskan ke *controller* yang bertugas menerima *input*, melakukan parsing data, dan mendelegasikan pemrosesan ke *service*. *Controller* tidak menyimpan *logika* bisnis utama; seluruh aturan bisnis, validasi domain, dan orkestrasi proses dijalankan pada lapisan *service*. *Service* kemudian berinteraksi dengan *repository* untuk mengambil atau menyimpan data ke *PostgreSQL* melalui *Bun ORM*. Hasil pemrosesan dikembalikan dari *repository* ke *service*, diolah jika perlu, kemudian dikemas oleh *controller* sebagai *response* yang dikirimkan ke *frontend*. Data transfer antara lapisan menggunakan *DTO* sehingga



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

struktur data yang dikembalikan ke *frontend* terpisah dari struktur *domain/model* internal.

Backend Prolytics terdiri dari beberapa *service* utama yang masing-masing menangani domain fungsional tertentu. *Auth* dan *User Service* mengelola proses *autentikasi*, profil pengguna, *role*, dan status akun. *Project* dan *Phase Service* bertanggung jawab atas pengelolaan *project*, pembuatan dan pengeditan *phase*, pengaturan anggota, serta perubahan status *project* dari *draft* hingga aktif. Tugas *Service* mengelola tugas teknis yang mencakup judul, deskripsi, durasi estimasi, *assignee*, status pengerjaan, status *approval*, dan relasi *dependency* antar tugas. *Approval Service* mengimplementasikan *workflow submit, approve, reject, revise, bulk submit*, dan *bulk approve*, serta menghasilkan *audit approval* yang dapat ditelusuri.

Basis data Prolytics menggunakan *PostgreSQL* sebagai sistem manajemen *basis data relasional*. Pemilihan *basis data relasional* didasarkan pada kebutuhan sistem untuk menjaga integritas referensial antar entitas data yang saling bergantung, yaitu *project, phase, tugas, task_dependencies, task_approvals, project_cpm_snapshots, users, project_repositories, activities, task_activities*, dan *project_logs*. Relasi antar tabel ini menjadi fondasi yang penting terutama bagi *CPM Engine*, karena perhitungan CPM membutuhkan data tugas dan *dependency* yang valid, lengkap, dan bebas dari siklus referensial.

Snapshot CPM disimpan pada tabel *project_cpm_snapshots* setiap kali terjadi perubahan yang memengaruhi hasil perhitungan authoritative pada *project* aktif. Penyimpanan *snapshot* ini memungkinkan audit terhadap perubahan *jalur kritis* dari waktu ke waktu serta pelacakan kondisi *tugas kritis* pada setiap titik perhitungan. Aktivitas GitHub yang diterima melalui *webhook* disimpan pada tabel *activities* dan *task_activities*, namun bersifat *monitoring-only* sehingga tidak mengubah atribut tugas maupun mempengaruhi hasil perhitungan *CPM* secara langsung. Akses ke basis data dilakukan melalui *Bun ORM* pada lapisan *repository*, sehingga *query* ke *PostgreSQL* terstruktur dan tidak tersebar pada lapisan lain.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

```
// pkg/config/database.go
package config

import "time"

type Database struct {
    Host      string    `env:"DB_HOST"`
    Port      string    `env:"DB_PORT" envDefault:"5432"`
    User      string    `env:"DB_USER"`
    Password  string    `env:"DB_PASSWORD"`
    Name      string    `env:"DB_NAME"`
    SSLMode   string    `env:"DB_SSLMODE" envDefault:"disable"`
    Timezone  string    `env:"DB_TIMEZONE"`
    MaxOpenConns int       `env:"DB_MAX_OPEN_CONNS" envDefault:"20"`
    MaxIdleConns int      `env:"DB_MAX_IDLE_CONNS" envDefault:"5"`
    ConnMaxLifetime time.Duration `env:"DB_CONN_MAX_LIFETIME" envDefault:"30m"`
}

func (d Database) Enabled() bool {
    return d.Host != "" && d.User != "" && d.Name != ""
}
```

Gambar 4. 28 Potongan Kode Konfigurasi *Database* (pkg/config/database.go)

Implementasi koneksi *database* pada *backend* Prolytics terbagi dalam dua bagian yang saling melengkapi. Bagian pertama ditunjukkan pada Gambar 4.28 yang memperlihatkan struktur konfigurasi *database* pada file pkg/config/database.go. Struct *Database* mendefinisikan seluruh parameter koneksi meliputi *host*, *port*, nama *database*, *kredensial*, *SSLMode*, *timezone*, serta parameter *pool koneksi* seperti *MaxOpenConns*, *MaxIdleConns*, dan *ConnMaxLifetime* yang masing-masing dibaca dari variabel lingkungan. Pola ini memisahkan konfigurasi dari *logika inisialisasi* sehingga parameter koneksi dapat disesuaikan per lingkungan *deployment* tanpa mengubah kode aplikasi.

Bagian kedua ditunjukkan pada Gambar 4.29 yang memperlihatkan implementasi fungsi *Init* pada file pkg/database/postgres.go. Fungsi ini menerima *struct* konfigurasi yang telah didefinisikan sebelumnya, membangun *Data Source Name* (DSN) melalui fungsi *buildDSN*, membuka koneksi ke *PostgreSQL* menggunakan driver standar, mengonfigurasi parameter *pool koneksi*, melakukan verifikasi koneksi melalui *PingContext*, kemudian menginisialisasi instance *Bun ORM* yang akan dipakai bersama oleh seluruh *repository* pada *backend*.



Hak Cipta:

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

```
// pkg/database/postgres.go
func Init(cfg config.Database) (*bun.DB, error) {
    log := logger.Log()

    if !cfg.Enabled() {
        log.Warn("database not configured, running without DB",
            zap.String("hint", "set DB_HOST, DB_USER, DB_NAME to enable database"),
        )
        return nil, nil
    }

    sqldb, err := sql.Open("postgres", buildDSN(cfg))
    if err != nil {
        return nil, fmt.Errorf("open postgres: %w", err)
    }

    sqldb.SetMaxOpenConns(cfg.MaxOpenConns)
    sqldb.SetMaxIdleConns(cfg.MaxIdleConns)
    sqldb.SetConnMaxLifetime(cfg.ConnMaxLifetime)

    ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
    defer cancel()
    if err := sqldb.PingContext(ctx); err != nil {
        _ = sqldb.Close()
        return nil, fmt.Errorf("ping postgres: %w", err)
    }

    db := bun.NewDB(sqldb, pgdialect.New())
    log.Info("database connected", zap.String("host", cfg.Host), zap.String("name",
        cfg.Database)) db, nil
}

func buildDSN(c config.Database) string {
    q := url.Values{}
    q.Set("sslmode", c.SSLMode)
    if c.Timezone != "" {
        q.Set("timezone", c.Timezone)
    }

    u := &url.URL{
        Scheme: "postgres",
        User:   url.UserPassword(c.User, c.Password),
        Host:   fmt.Sprintf("%s:%s", c.Host, c.Port),
        Path:   "/" + c.Name,
        RawQuery: q.Encode(),
    }
    return u.String()
}
```

Gambar 4. 29 Potongan Kode Fungsi Init Koneksi Database
(pkg/database/postgres.go)

Gambar 4.29 menunjukkan bahwa fungsi *Init* pada *pkg/database/postgres.go* bertanggung jawab atas seluruh siklus pembukaan koneksi database. Fungsi *buildDSN* membangun URL koneksi secara terprogram berdasarkan parameter konfigurasi, sedangkan parameter *pool koneksi* seperti *MaxOpenConns*, *MaxIdleConns*, dan *ConnMaxLifetime* dikonfigurasi setelah koneksi berhasil dibuka. Verifikasi koneksi dilakukan menggunakan *PingContext* dengan batas waktu lima detik sebelum *instance Bun ORM* dibuat. Pola ini memastikan bahwa



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

backend tidak akan berjalan dengan koneksi *database* yang tidak valid, serta bahwa seluruh *repository* menggunakan satu *instance* koneksi yang sama secara konsisten.

4.3.4 Implementasi Modul Utama

Modul utama Prolytics diimplementasikan berdasarkan kebutuhan proses *monitoring* proyek perangkat lunak. Setiap modul memiliki fungsi yang saling melengkapi, mulai dari pengelolaan *project* dan *phase*, *breakdown requirement* menjadi technical tugas, pengaturan *dependency*, perhitungan CPM, *approval workflow*, integrasi GitHub, hingga bantuan penyusunan *requirement* menggunakan LLM.

1. Implementasi Hak Akses Pengguna

Implementasi hak akses pengguna dilakukan dengan menggabungkan autentikasi berbasis JWT dan validasi kontekstual pada *backend*. JWT digunakan untuk membawa identitas pengguna dan *role*, sedangkan validasi tambahan dilakukan berdasarkan *project* membership, *phase* membership, ownership tugas, dan status *project*. Dengan cara ini, hak akses tidak hanya ditentukan oleh *role* global, tetapi juga oleh hubungan pengguna terhadap *project* atau tugas tertentu.

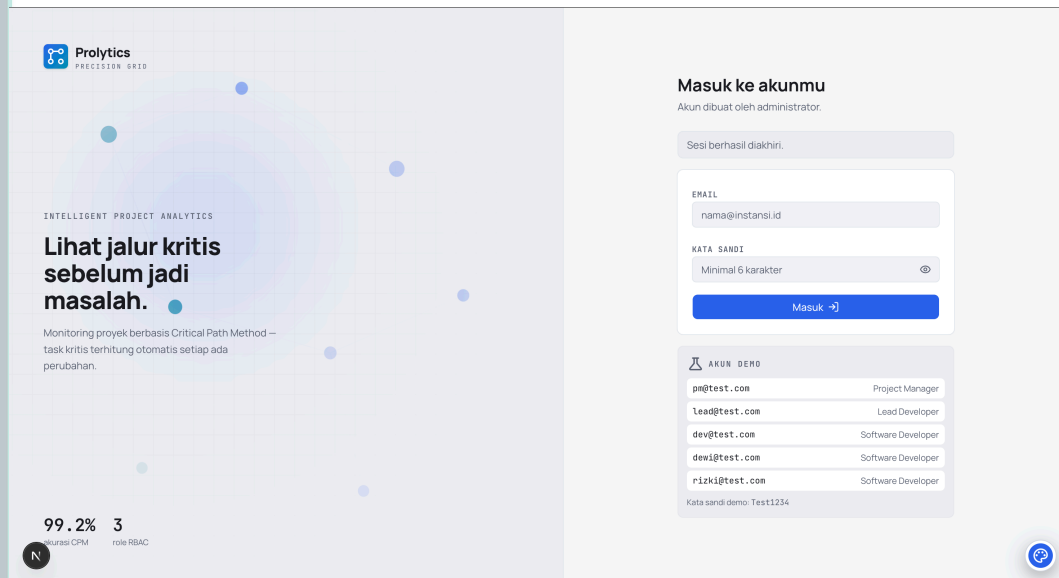
Pada sisi *frontend*, sistem menyesuaikan menu dan aksi yang tampil berdasarkan *role* pengguna. Namun, penyembunyian menu pada *frontend* hanya berfungsi sebagai bantuan pengalaman pengguna, sedangkan proteksi utama tetap dilakukan oleh *backend* melalui *middleware* dan service policy.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 4. 30 Tampilan *Login Sistem Prolytics*

Gambar 4.30 memperlihatkan titik awal autentikasi sebelum pengguna masuk ke sistem. Setelah proses *login* berhasil, *backend* mengembalikan *token* yang digunakan *frontend* untuk memuat identitas pengguna dan menampilkan navigasi sesuai *role*. Validasi akses tetap diperiksa kembali pada setiap *request API* yang membutuhkan kewenangan tertentu.

Project Manager memiliki akses untuk mengelola *project*, *user*, *phase*, *approval*, dan informasi *monitoring*. *Lead Developer* memiliki akses untuk menyusun tugas teknis, mengatur *dependency*, mengajukan *approval*, serta memantau CPM pada *project* yang menjadi tanggung jawabnya. *Software Developer* memiliki akses yang lebih terbatas, yaitu melihat tugas yang ditugaskan, memperbarui status pekerjaan, dan menautkan aktivitas teknis yang relevan.

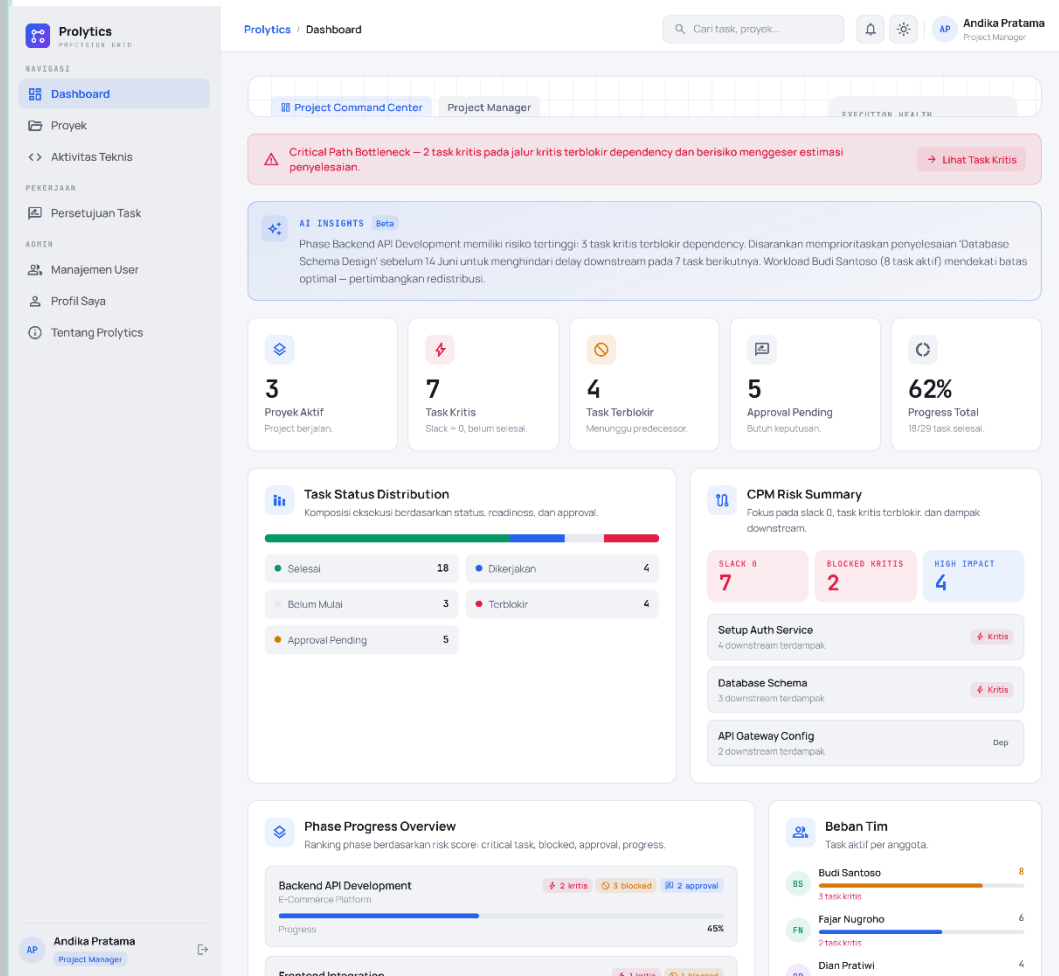
2. Implementasi Modul Manajemen *Project* dan *Phase*

Modul manajemen *project* dan *phase* digunakan untuk mengelola ruang kerja utama proyek. *Project Manager* dapat membuat *project*, mengubah informasi *project*, mengatur anggota, membuat *phase/termin*, mengubah jadwal *phase*, serta mengaktifkan *project* setelah tugas memenuhi syarat. *Lead Developer* dan *Software Developer* mengakses *project* berdasarkan membership yang diberikan.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Tampilan *dashboard* digunakan sebagai pintu masuk untuk memantau ringkasan proyek dan aktivitas pengguna. Informasi yang disajikan membantu *Project Manager* melihat kondisi umum sebelum masuk ke detail *project*.



Gambar 4. 31 *Dashboard Project Manager*

Gambar 4.31 menampilkan ringkasan *project* dan indikator pekerjaan yang relevan bagi *Project Manager*. Tampilan ini membantu pengguna memahami kondisi umum sistem tanpa harus membuka setiap *project* satu per satu.

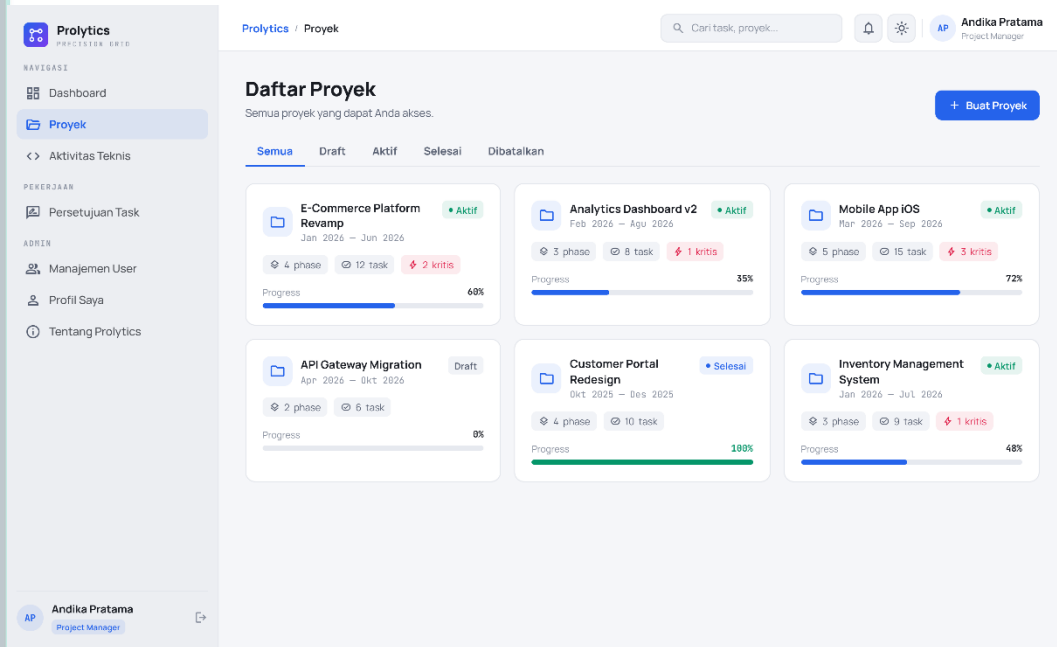
Pada halaman detail *project*, sistem menampilkan metadata *project*, daftar *phase*, komentar perencanaan, serta kondisi CPM proyek. Setelah fitur prediksi jadwal dihapus dari *scope*, informasi yang ditampilkan berfokus pada kondisi CPM seperti jumlah tugas, *tugas kritis* aktif, tugas terblokir, tugas tanpa assignee, dan mode CPM.



Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



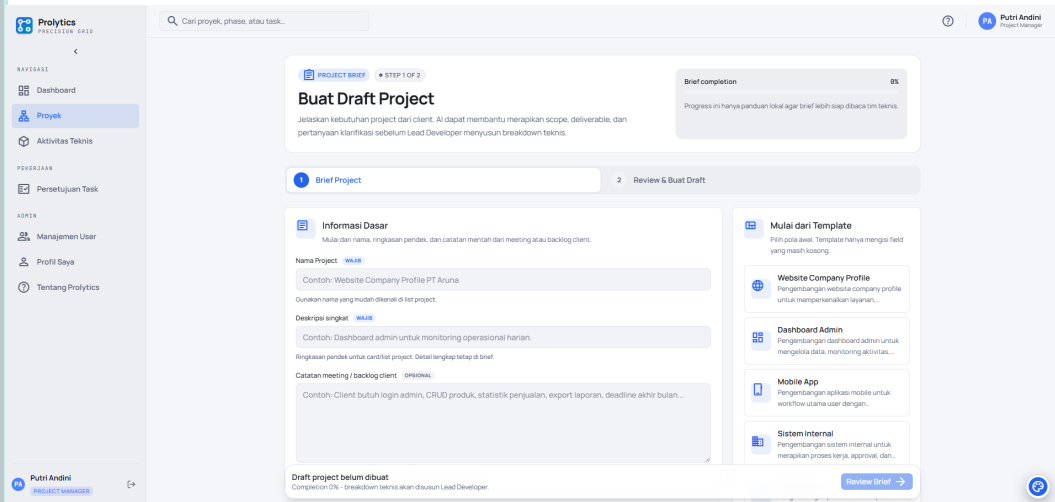
Gambar 4. 32 Detail *Project* dan Kondisi CPM Proyek

Gambar 4.32 menunjukkan bahwa halaman detail *project* berfungsi sebagai pusat pemantauan *project*. *Project Manager* dapat melihat status *project*, *phase* yang tersedia, readiness untuk aktivasi, dan ringkasan kondisi CPM tanpa menampilkan klaim prediksi tanggal selesai.

Selain halaman detail, modul ini juga menyediakan halaman pembuatan *project* baru. Pada halaman ini, *Project Manager* mengisi informasi dasar *project* seperti nama *project*, deskripsi, dan tanggal mulai. Setelah *project* berhasil dibuat, *Project Manager* dapat langsung menambahkan anggota tim, membuat *phase*, dan menyusun tugas melalui navigasi yang tersedia. Gambar 4.24 menampilkan halaman pembuatan *project* baru pada sistem Prolytics.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 4. 33 Halaman Create *Project* Prolytics

Gambar 4.33 menampilkan form pembuatan *project* pada sistem Prolytics. *Project Manager* mengisi nama *project*, deskripsi singkat, dan tanggal mulai, kemudian menyimpan data tersebut melalui tombol yang tersedia. Sistem akan membuat *project* baru dengan status draft sehingga pengelolaan *phase* dan tugas dapat segera dimulai sebelum *project* diaktifkan.

3. Implementasi Modul *Requirement* dan *Technical* Tugas

Modul *requirement* dan *technical* tugas digunakan untuk mengubah kebutuhan *project* menjadi daftar pekerjaan teknis yang dapat dieksekusi oleh tim pengembang. Pada implementasinya, tugas disimpan dengan atribut seperti judul, deskripsi, durasi estimasi, *phase*, assignee, status pengerjaan, status *approval*, dan daftar *dependency*.

Lead Developer berperan sebagai pengguna utama yang membuat dan mengubah *technical* tugas. *Software Developer* berperan memperbarui status tugas yang sudah disetujui dan ditugaskan kepadanya. Sistem juga menyediakan filter, halaman detail tugas, assignment, dan validasi perubahan agar tugas yang sudah selesai tidak berubah secara struktural sehingga riwayat pekerjaan tetap stabil.



© Hak Cipta milik Politeknik Negeri Jakarta

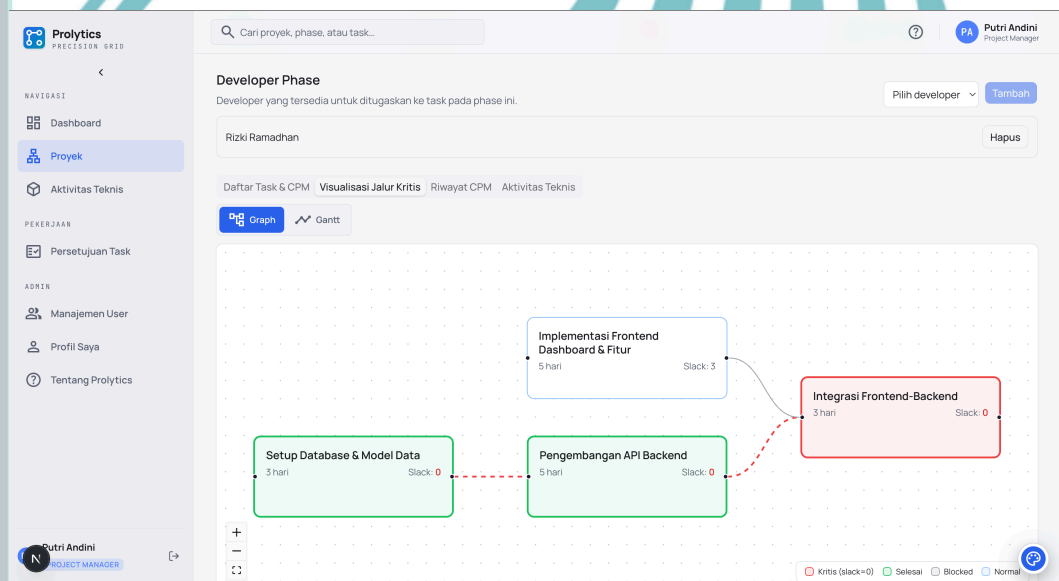
Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4. Implementasi Modul *Dependency* Tugas

Dependency tugas diterapkan dengan pola predecessor dan successor. Sebuah tugas dapat memiliki satu atau lebih predecessor, dan successor baru dianggap siap dikerjakan ketika predecessor yang dibutuhkan telah selesai. Data *dependency* disimpan sebagai relasi antar tugas dalam *satu project* agar dapat dipakai oleh **engine** CPM.

Sistem menampilkan relasi *dependency* dalam bentuk visualisasi *jalur kritis*. Tampilan ini membantu pengguna memahami urutan pengerjaan tugas, tugas yang saling bergantung, dan tugas yang masuk *jalur kritis*.



Gambar 4. 34 Visualisasi *Dependency* dan *Jalur Kritis*

Gambar 4.34 menunjukkan hubungan antar tugas yang dipakai sebagai dasar perhitungan CPM. Validasi *dependency* dilakukan untuk mencegah *circular dependency* dan *dependency* yang tidak valid, karena *dependency* yang membentuk siklus akan membuat perhitungan CPM tidak dapat dilakukan secara benar.

5. Implementasi Modul *Critical Path Method*

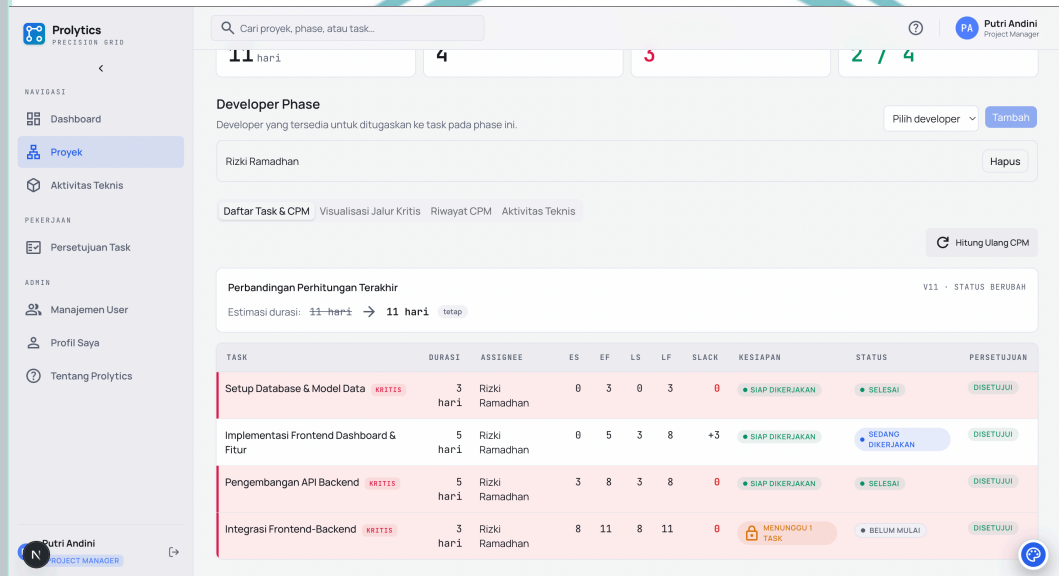
Modul *Critical Path Method* merupakan inti sistem Prolytics. Perhitungan CPM dilakukan pada level *project* dengan *input* berupa *project_start_date*, daftar *approved* tugas, *estimated_duration*, dan *dependency Finish-to-Start* dalam satu

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

project. Pada *project draft*, sistem menghasilkan *preview* CPM secara on-demand tanpa menyimpan *snapshot*. Pada *project* aktif, hasil authoritative dihitung ulang dan disimpan sebagai *snapshot* ketika terjadi perubahan yang relevan.

Hasil CPM ditampilkan dalam tabel agar pengguna dapat melihat nilai Early Start (ES), Early Finish (EF), Late Start (LS), Late Finish (LF), *slack/float*, readiness, status pengerjaan, dan status *approval* pada setiap tugas.



The screenshot shows the 'Developer Phase' section of the Prolytics application. It displays a table of tasks with their durations, assignees, and CPM values. The table includes columns for Task, Duration, Assignee, ES, EF, LS, LF, Slack, Readiness, Status, and Approval. Tasks are color-coded: red for critical (Kritis), green for ready to start (Siap Dikerjakan), blue for in progress (Sedang Dikerjakan), and orange for pending (Menunggu Task).

TASK	DURASI	ASSIGNEE	ES	EF	LS	LF	SLACK	KESIAPAN	STATUS	PERSetujuan
Setup Database & Model Data	3 har1	Rizki Ramadhan	0	3	0	3	0	SIAP DIKERJAKAN	SELESAI	DISETUJUI
Implementasi Frontend Dashboard & Fitur	5 har1	Rizki Ramadhan	0	5	3	8	+3	SIAP DIKERJAKAN	SEDANG DIKERJAKAN	DISETUJUI
Pengembangan API Backend	5 har1	Rizki Ramadhan	3	8	3	8	0	SIAP DIKERJAKAN	SELESAI	DISETUJUI
Integrasi Frontend-Backend	3 har1	Rizki Ramadhan	8	11	8	11	0	MENUNGGU 1 TASK	BELUM MULAI	DISETUJUI

Gambar 4. 35 Tabel Hasil Perhitungan CPM Project-Wide

Gambar 4.35 memperlihatkan bahwa tugas dengan *slack* bernilai nol ditandai sebagai *tugas kritis*. Informasi readiness juga membantu pengguna mengetahui tugas yang dapat segera dikerjakan dan tugas yang masih menunggu penyelesaian predecessor.

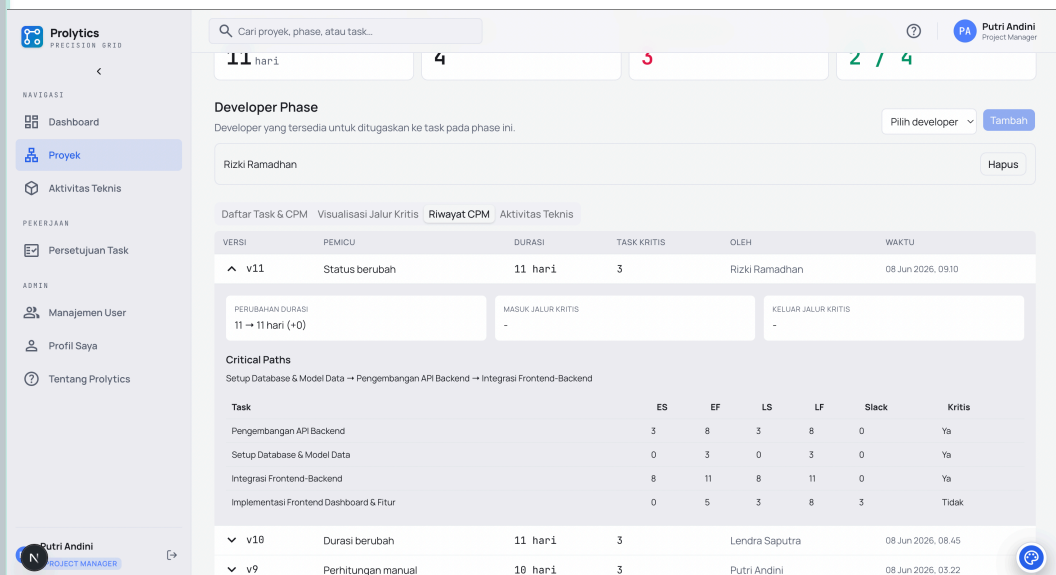
Selain tabel hasil saat ini, sistem menyimpan riwayat *snapshot* CPM pada *project* aktif. Riwayat ini digunakan untuk melihat perubahan jalur kritis project, perubahan tugas yang masuk atau keluar dari jalur kritis, serta kondisi CPM pada setiap titik waktu.



Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 4. 36 Riwayat *Snapshot* CPM

Gambar 4.36 menunjukkan riwayat CPM dalam bentuk accordion. Setiap versi menyimpan informasi pemicu perhitungan, *project calculated finish date*, jumlah tugas kritis, aktor, waktu, *critical paths*, dan hasil perhitungan tugas pada versi tersebut.

Implementasi algoritma *Critical Path Method* (CPM) pada Prolytics ditulis dalam bahasa Go dan diorganisasikan sebagai modul murni tanpa ketergantungan basis data sehingga dapat diuji secara independen. Modul ini terdiri dari beberapa bagian: definisi struktur data, inisialisasi graf *dependency*, *forward pass* dan *backward pass*, perhitungan *slack* dan identifikasi *tugas kritis*, serta pembentukan *critical path*. Struktur data yang digunakan oleh *engine* CPM ditunjukkan pada Gambar 4.37.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

```

// Package cpm implements the Critical Path Method engine as pure, in-memory
// functions (no database) so the algorithm is independently testable (PRD §5,
// §12.5). The CPMSERVICE orchestrates loading, persistence, and snapshots around
// these functions.
package cpm

import (
    "sort"

    "github.com/google/uuid"

    "prolytics-be/internal/constants"
)

// TaskInput is the per-task data the engine needs. Status drives readiness.
type TaskInput struct {
    ID        uuid.UUID
    Duration  int
    Status    int // constants.TaskStatus*
}

// DependencyEdge is a Finish-to-Start edge: Predecessor must finish before
// Successor starts.
type DependencyEdge struct {
    Predecessor uuid.UUID
    Successor   uuid.UUID
}

// TaskResult is the computed CPM outcome for one task.
type TaskResult struct {
    TaskID        uuid.UUID `json:"taskId"`
    EarlyStart    int      `json:"earlyStart"`
    EarlyFinish   int      `json:"earlyFinish"`
    LateStart     int      `json:"lateStart"`
    LateFinish    int      `json:"lateFinish"`
    Slack         int      `json:"slack"`
    IsCritical    bool     `json:"isCritical"`
    Readiness     int      `json:"-` // constants.TaskReadiness*
}

// Result is the full CPM outcome for a phase.
type Result struct {
    Duration      int
    Tasks         map[uuid.UUID]TaskResult
    CriticalTasks []uuid.UUID
    CriticalPaths [][]uuid.UUID
}

// ErrCycle is returned when the dependency graph is not a DAG.
type ErrCycle struct{}

func (ErrCycle) Error() string { return "dependency graph contains a cycle" }

```

Gambar 4. 37 Struktur Data Engine CPM

Gambar 4.37 menunjukkan tiga *struct* utama yang menjadi fondasi *engine* CPM, yaitu TaskInput sebagai representasi satu *tugas* beserta durasi dan statusnya,



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DependencyEdge yang mendefinisikan relasi *Finish-to-Start* antara *predecessor* dan *successor*, serta TaskResult yang menyimpan seluruh nilai hasil perhitungan per-tugas meliputi *Early Start* (ES), *Early Finish* (EF), *Late Start* (LS), *Late Finish* (LF), *slack*, dan *readiness*. Terdapat pula *struct* Result sebagai wadah hasil keseluruhan *phase* dan ErrCycle yang dikembalikan apabila graf mengandung *circular dependency*. Proses inisialisasi data dan pembentukan *graf dependency* dari struktur tersebut ditunjukkan pada Gambar 4.38.

```
// Calculate runs the forward/backward pass, computes slack/critical tasks,
// derives all critical paths, and computes readiness. Edges that reference a task
// outside the provided task set are ignored (the caller is expected to pass the
// approved task set and same-set edges). Returns ErrCycle if a cycle is detected.
func Calculate(tasks []TaskInput, deps []DependencyEdge) (Result, error) {
    res := Result{
        Tasks:         make(map[uuid.UUID]TaskResult, len(tasks)),
        CriticalTasks:  []uuid.UUID{},
        CriticalPaths:  [][]uuid.UUID{},
    }

    if len(tasks) == 0 {
        return res, nil
    }

    inSet := make(map[uuid.UUID]bool, len(tasks))
    duration := make(map[uuid.UUID]int, len(tasks))
    status := make(map[uuid.UUID]int, len(tasks))
    for _, t := range tasks {
        inSet[t.ID] = true
        duration[t.ID] = t.Duration
        status[t.ID] = t.Status
    }

    preds := make(map[uuid.UUID][]uuid.UUID)
    succs := make(map[uuid.UUID][]uuid.UUID)
    indegree := make(map[uuid.UUID]int, len(tasks))
    for _, id := range orderedIDs(inSet) {
        indegree[id] = 0
    }

    seen := make(map[[2]uuid.UUID]bool)
    for _, e := range deps {
        if !inSet[e.Predecessor] || !inSet[e.Successor] || e.Predecessor == e.Successor {
            continue
        }

        key := [2]uuid.UUID{e.Predecessor, e.Successor}
        if seen[key] {
            continue
        }

        seen[key] = true
        succs[e.Predecessor] = append(succs[e.Predecessor], e.Successor)
        preds[e.Successor] = append(preds[e.Successor], e.Predecessor)
        indegree[e.Successor]++
    }

    order, ok := topoSort(orderedIDs(inSet), succs, indegree)
    if !ok {
        return Result{}, ErrCycle{}
    }
}
```

Gambar 4. 38 Inisialisasi Data dan Pembentukan Graph *Dependency*

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.38 memperlihatkan fungsi Calculate sebagai titik masuk *engine* CPM. Sistem menginisialisasi *map* untuk *inSet*, *duration*, dan status setiap *tugas*, kemudian membangun graf *dependency* berupa peta *preds*, *succs*, dan *indegree* dengan mengabaikan *dependency* yang tidak valid atau merujuk ke *tugas* di luar *set*. Setelah graf terbentuk, sistem menjalankan *topological sort* untuk memvalidasi bahwa graf merupakan *Directed Acyclic Graph* (DAG). Apabila ditemukan siklus, fungsi mengembalikan *ErrCycle*. Proses *forward pass* dan *backward pass* yang dijalankan setelah validasi graf ditunjukkan pada Gambar 4.39.

```
// Forward pass.
es := make(map[uuid.UUID]int, len(tasks))
ef := make(map[uuid.UUID]int, len(tasks))
phaseDuration := 0
for _, id := range order {
    start := 0
    for _, p := range preds[id] {
        if ef[p] > start {
            start = ef[p]
        }
    }

    es[id] = start
    ef[id] = start + duration[id]
    if ef[id] > phaseDuration {
        phaseDuration = ef[id]
    }
}

// Backward pass (reverse topological order).
ls := make(map[uuid.UUID]int, len(tasks))
lf := make(map[uuid.UUID]int, len(tasks))
for i := len(order) - 1; i >= 0; i-- {
    id := order[i]
    if len(succs[id]) == 0 {
        lf[id] = phaseDuration
    } else {
        finish := -1
        for _, s := range succs[id] {
            if finish == -1 || ls[s] < finish {
                finish = ls[s]
            }
        }
        lf[id] = finish
    }

    ls[id] = lf[id] - duration[id]
}
```

Gambar 4. 39 *Forward Pass* dan *Backward Pass*

Gambar 4.39 memperlihatkan implementasi *forward pass* dan *backward pass*. Pada *forward pass*, sistem mengiterasi *tugas* sesuai urutan topologis. Nilai *Early Start*



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

(ES) ditentukan dari nilai *Early Finish* (EF) terbesar milik *predecessor*, sedangkan EF diperoleh dari ES ditambah durasi *tugas*. Nilai EF terbesar dari seluruh *tugas* sekaligus menjadi *project calculated finish date*. Pada *backward pass*, sistem mengiterasi *tugas* dalam urutan terbalik. *Tugas* tanpa *successor* memperoleh *Late Finish* (LF) sebesar *project calculated finish date*, sedangkan *tugas* yang memiliki *successor* memperoleh LF dari nilai *Late Start* (LS) terkecil milik *successor*-nya. Selanjutnya, LS dihitung dari LF dikurangi durasi *tugas*. Hasil kedua proses ini digunakan untuk perhitungan *slack* dan identifikasi *tugas* kritis yang ditunjukkan pada

Gambar

4.40.

```
// Slack, criticality, readiness.
for _, id := range order {
    slack := ls[id] - es[id]
    critical := slack == 0
    res.Tasks[id] = TaskResult{
        TaskID:    id,
        EarlyStart: es[id],
        EarlyFinish: ef[id],
        LateStart:   ls[id],
        LateFinish:  lf[id],
        Slack:       slack,
        IsCritical:  critical,
        Readiness:   readiness(preds[id], status),
    }
}

res.Duration = phaseDuration
res.CriticalTasks = collectCriticalTasks(order, res.Tasks)
res.CriticalPaths = buildCriticalPaths(order, succs, es, ef, res.Tasks)
return res, nil
}

func readiness(predecessors []uuid.UUID, status map[uuid.UUID]int) int {
    for _, p := range predecessors {
        if status[p] != constants.TaskStatusDone {
            return constants.TaskReadinessBlocked
        }
    }
    return constants.TaskReadinessReady
}

func collectCriticalTasks(order []uuid.UUID, results map[uuid.UUID]TaskResult) []uuid.UUID {
    out := []uuid.UUID{}
    for _, id := range order {
        if results[id].IsCritical {
            out = append(out, id)
        }
    }
    return out
}
```

Gambar 4. 40 Perhitungan *Slack*, *Tugas Kritis*, dan *Readiness*

Gambar 4.40 menunjukkan bagian kode yang menghitung nilai *slack* setiap *tugas* dari selisih LS dan ES, kemudian menentukan status kritis berdasarkan kondisi



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

slack bernilai nol. Setiap *tugas* juga dinilai kesiapannya melalui fungsi *readiness* yang memeriksa status seluruh *predecessor*. Apabila terdapat *predecessor* yang belum selesai, *tugas* dikembalikan sebagai *blocked*. Bagian ini juga memanggil fungsi *collectCriticalTasks* untuk mengumpulkan daftar *tugas* kritis dan *buildCriticalPaths* untuk menyusun jalur kritis lengkap sebelum hasilnya dikembalikan sebagai *Result*. Implementasi fungsi *buildCriticalPaths* dan *topological sort* yang mendukung seluruh proses tersebut ditunjukkan pada Gambar 4.40.





© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

```
// buildCriticalPaths derives every critical path. An edge belongs to a critical
// path when both endpoints are critical and the link is tight (EF(pred) ==
// ES(succ)), i.e. there is no slack between them.
func buildCriticalPaths(
  order []uuid.UUID,
  succs map[uuid.UUID][]uuid.UUID,
  es, ef map[uuid.UUID]int,
  results map[uuid.UUID]TaskResult,
) [][]uuid.UUID {
  critSucc := make(map[uuid.UUID][]uuid.UUID)
  hasCritPred := make(map[uuid.UUID]bool)

  for _, id := range order {
    if !results[id].IsCritical {
      continue
    }

    children := append([]uuid.UUID(nil), succs[id]...)
    sort.Slice(children, func(i, j int) bool {
      return children[i].String() < children[j].String()
    })

    for _, s := range children {
      if results[s].IsCritical && ef[id] == es[s] {
        critSucc[id] = append(critSucc[id], s)
        hasCritPred[s] = true
      }
    }
  }

  paths := [][]uuid.UUID{}
  var dfs func(node uuid.UUID, path []uuid.UUID)
  dfs = func(node uuid.UUID, path []uuid.UUID) {
    path = append(path, node)
    if len(critSucc[node]) == 0 {
      cp := make([]uuid.UUID, len(path))
      copy(cp, path)
      paths = append(paths, cp)
      return
    }

    for _, s := range critSucc[node] {
      dfs(s, path)
    }
  }

  // Sources: critical tasks with no critical predecessor.
  for _, id := range order {
    if results[id].IsCritical && !hasCritPred[id] {
      dfs(id, nil)
    }
  }

  return paths
}

func topoSort(
  ids []uuid.UUID,
  succs map[uuid.UUID][]uuid.UUID,
  indegreeIn map[uuid.UUID]int,
) ([]uuid.UUID, bool) {
  indegree := make(map[uuid.UUID]int, len(indegreeIn))
  for k, v := range indegreeIn {
    indegree[k] = v
  }

  queue := make([]uuid.UUID, 0, len(ids))
  for _, id := range ids {
    if indegree[id] == 0 {
      queue = append(queue, id)
    }
  }

  order := make([]uuid.UUID, 0, len(ids))
  for len(queue) > 0 {
    // Deterministic: pop the smallest id among current zero-indegree nodes.
    sort.Slice(queue, func(i, j int) bool {
      return queue[i].String() < queue[j].String()
    })

    node := queue[0]
    queue = queue[1:]
    order = append(order, node)

    children := append([]uuid.UUID(nil), succs[node]...)
    sort.Slice(children, func(i, j int) bool {
      return children[i].String() < children[j].String()
    })

    for _, s := range children {
      indegree[s]--
      if indegree[s] == 0 {
        queue = append(queue, s)
      }
    }
  }

  if len(order) != len(ids) {
    return nil, false
  }

  return order, true
}

func orderedIDs(set map[uuid.UUID]bool) []uuid.UUID {
  ids := make([]uuid.UUID, 0, len(set))
  for id := range set {
    ids = append(ids, id)
  }

  sort.Slice(ids, func(i, j int) bool {
    return ids[i].String() < ids[j].String()
  })

  return ids
}
```

Gambar 4. 41 Pembentukan *Critical Path* dan Topological Sort

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.41 menunjukkan dua fungsi pendukung utama *engine* CPM. Fungsi `buildCriticalPaths` menelusuri *tugas kritis* menggunakan *Depth-First Search* (DFS) untuk menemukan semua *critical path* sebuah *edge* dianggap masuk ke *jalur kritis* apabila kedua titik ujungnya kritis dan nilai EF *predecessor* sama dengan ES *successor*. Fungsi `topoSort` mengimplementasikan *topological sort* berbasis *indegree* yang bersifat deterministik karena tugas dengan *indegree* nol diurutkan secara leksikografis sebelum dimasukkan ke antrian, sehingga urutan pemrosesan konsisten meski graf memiliki banyak percabangan paralel.

Pada tahap awal, sistem membentuk struktur graf dari *dependency* tugas. *Dependency* yang tidak valid atau mengarah ke tugas di luar daftar tugas yang dihitung akan diabaikan. Setelah itu, sistem menjalankan proses topological sorting untuk memastikan bahwa *dependency* membentuk *Directed Acyclic Graph* (DAG). Jika ditemukan siklus *dependency*, sistem mengembalikan error karena CPM tidak dapat dihitung pada graf yang memiliki *circular dependency*.

Setelah graf valid, sistem menjalankan *forward pass* untuk menghitung nilai Early Start (ES) dan Early Finish (EF). Nilai ES sebuah tugas ditentukan dari nilai EF terbesar milik *predecessor*-nya. Nilai EF kemudian dihitung dengan menjumlahkan ES dan durasi tugas. Dari proses ini, sistem juga memperoleh durasi internal *phase* berdasarkan nilai EF terbesar dari seluruh tugas.

Selanjutnya, sistem menjalankan *backward pass* untuk menghitung Late Finish (LF) dan Late Start (LS). Tugas yang tidak memiliki *successor* diberi LF sebesar durasi *phase*. Untuk tugas yang memiliki *successor*, LF dihitung dari nilai LS terkecil milik *successor*. Nilai LS kemudian diperoleh dari LF dikurangi durasi tugas.

Bagian akhir kode menghitung nilai *slack/float*, menentukan apakah tugas termasuk kritis, dan menyusun readiness tugas. *Slack* dihitung dari selisih LS dan ES. Tugas dengan *slack* bernilai nol ditandai sebagai *tugas kritis* karena keterlambatan pada tugas tersebut dapat memengaruhi penyelesaian project secara keseluruhan. Informasi readiness digunakan untuk menunjukkan apakah tugas sudah siap dikerjakan berdasarkan status *predecessor*-nya.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

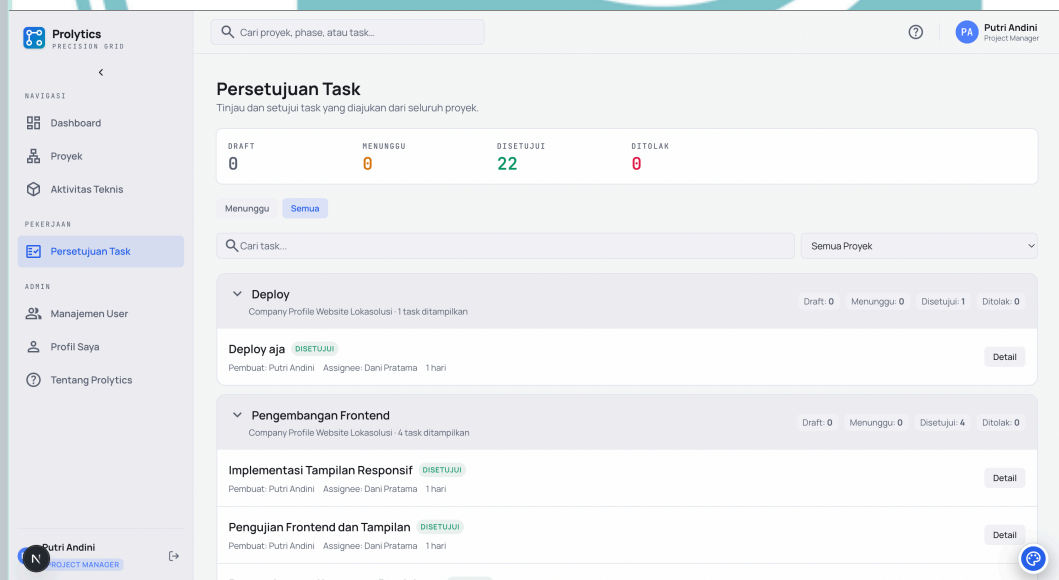
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Secara keseluruhan, potongan kode ini memperlihatkan bahwa perhitungan CPM pada Prolytics dilakukan secara sistematis melalui validasi graf, *forward pass*, *backward pass*, perhitungan *slack*, identifikasi *tugas kritis*, dan penentuan kesiapan tugas. Hasil perhitungan tersebut kemudian digunakan oleh sistem untuk menampilkan ES, EF, LS, LF (dalam tanggal kalender), slack, critical tugas, dan critical path pada halaman monitoring project.

6. Implementasi Modul *Approval Workflow*

Approval workflow digunakan untuk memastikan technical tugas yang disusun oleh *Lead Developer* telah ditinjau oleh *Project Manager* sebelum menjadi tugas resmi pada *project* aktif. Alur ini mendukung status draft, pending, approved, rejected, dan revision sehingga perubahan tugas dapat ditelusuri melalui status persetujuan.

Halaman *approval* menampilkan antrean persetujuan yang dikelompokkan berdasarkan *phase*. Pengelompokan ini membantu *Lead Developer* dan *Project Manager* meninjau tugas berdasarkan konteks *phase*, bukan hanya daftar tugas yang panjang.



Gambar 4. 42 Halaman *Approval Workflow* Berdasarkan *Phase*

Gambar 4.42 memperlihatkan tugas dalam satu *phase* beserta status *approval*, assignee, durasi, dan tombol aksi. *Lead Developer* dapat mengajukan tugas eligible,



Hak Cipta :

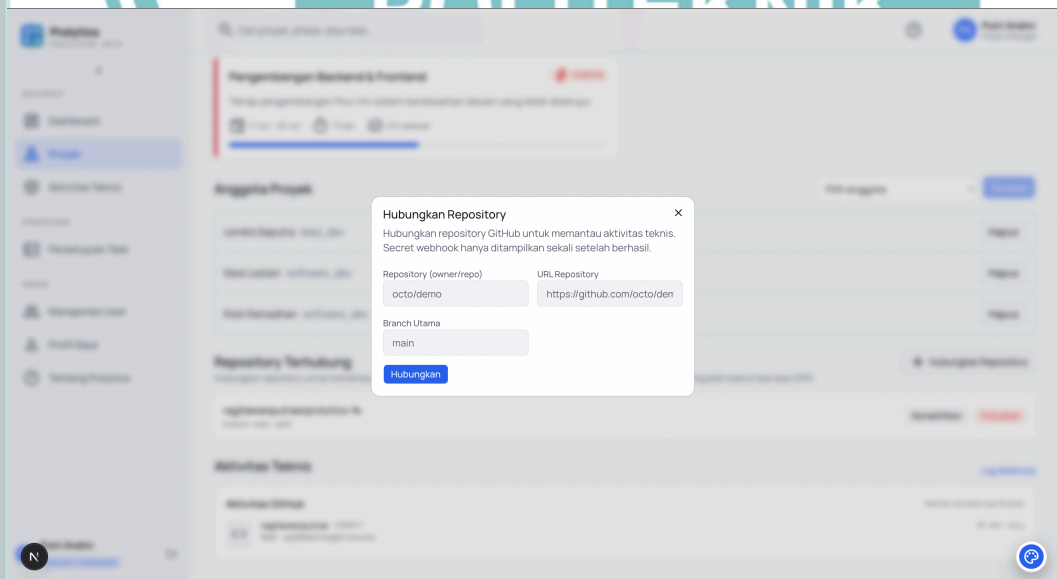
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

sedangkan *Project Manager* dapat menyetujui, menolak, atau meminta revisi melalui *endpoint approval* yang tersedia.

Pada *backend*, *action approval* dipisahkan menjadi submit, approve, reject, revise, bulk submit, dan bulk approve. Setiap action menghasilkan audit *approval* dan dapat memicu perhitungan CPM authoritative jika perubahan tersebut memengaruhi tugas pada *project* aktif.

7. Implementasi Modul Integrasi GitHub

Integrasi GitHub diimplementasikan sebagai mekanisme *monitoring* aktivitas teknis yang bersifat informatif. *Project Manager* dan *Lead Developer* dapat menghubungkan *repository* ke *project*, kemudian sistem menerima aktivitas melalui *webhook* GitHub. Aktivitas seperti *commit*, *pull request*, *issue*, dan *review* dinormalisasi menjadi activity feed yang dapat dilihat pada level *project*, *phase*, atau tugas apabila ditautkan. Dengan demikian, progres yang dilaporkan pada sistem dapat dikaitkan dengan bukti aktivitas pengembangan yang nyata di *repository*. Gambar 4.39 menunjukkan tampilan *repository* yang terhubung beserta daftar aktivitas teknis, sedangkan Gambar 4.40 menampilkan detail aktivitas GitHub yang ditautkan pada tugas tertentu.



Gambar 4. 43 Tampilan *Repository* Terhubung dan Activity Feed GitHub
Gambar 4.43 menunjukkan halaman integrasi GitHub pada sistem Prolytics. Pada halaman ini, *Project Manager* dan *Lead Developer* dapat melihat *repository* yang

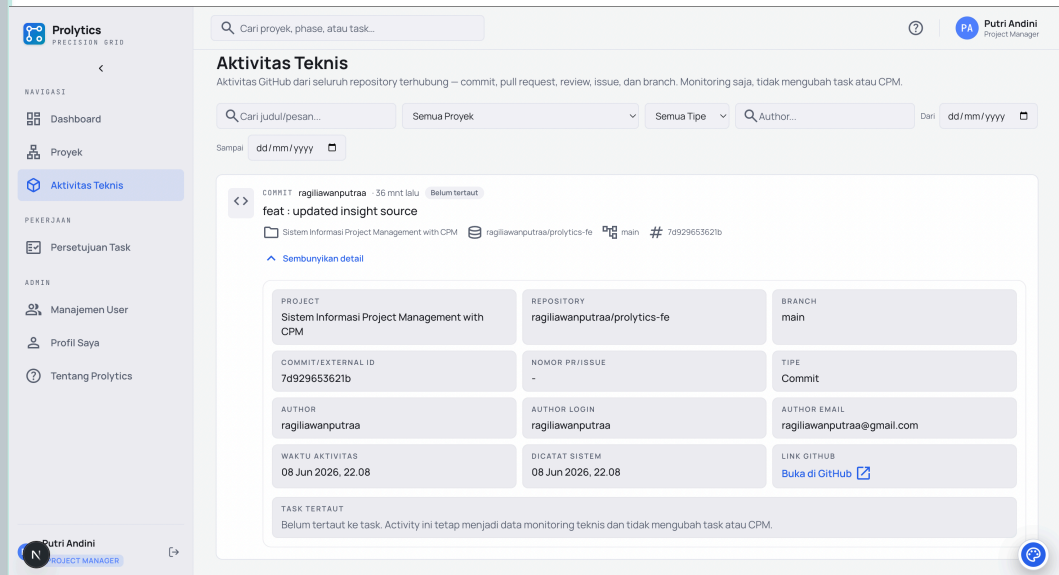


© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

telah dihubungkan ke *project* beserta daftar aktivitas teknis yang masuk melalui *webhook*. Setiap aktivitas ditampilkan dengan informasi tipe, waktu, dan aktor yang melakukan aksi, sehingga kondisi pekerjaan teknis di *repository* dapat dipantau tanpa meninggalkan sistem.



Gambar 4. 44 Detail Aktivitas GitHub yang Ditautkan pada Tugas

Gambar 4.44 memperlihatkan detail aktivitas GitHub yang telah ditautkan pada tugas tertentu dalam sistem Prolytics. Pengguna dapat menautkan satu atau lebih aktivitas teknis seperti *commit* atau *pull request* ke tugas yang sedang dikerjakan, sehingga keterlacakan antara pekerjaan yang tercatat di sistem dan aktivitas nyata di *repository* terjaga. Aktivitas GitHub tidak mengubah hasil perhitungan *Critical Path Method* secara langsung, tetapi berfungsi sebagai bukti pendukung progres teknis yang dilaporkan oleh *Software Developer*.

8. Implementasi Modul Asisten *Requirement* Berbasis LLM

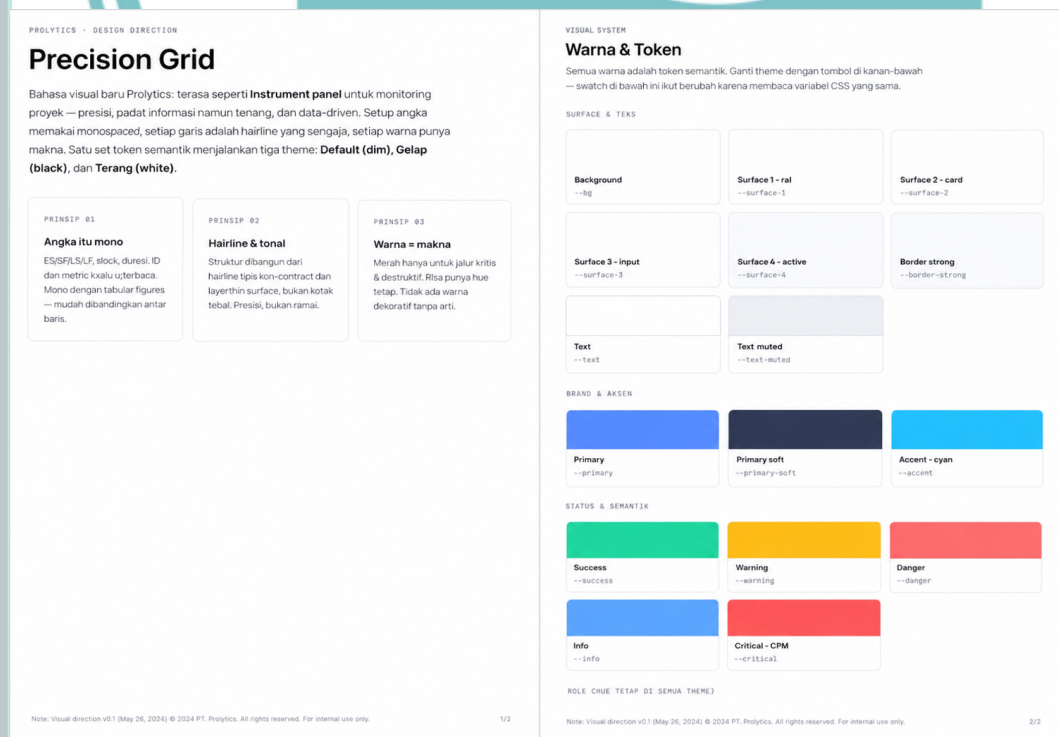
Modul asisten *requirement* berbasis LLM digunakan sebagai fitur pendukung pada tahap perencanaan *project*. Pengguna dapat memberikan konteks kebutuhan *project*, kemudian sistem menghasilkan usulan *phase* dan tugas *planning* yang dapat ditinjau sebelum diterapkan sebagai draft.

Hasil dari LLM tidak dianggap sebagai keputusan otomatis final. *Project Manager* tetap perlu memeriksa, mengubah, dan memvalidasi hasil usulan sebelum data disimpan ke *project*. Dengan demikian, LLM berperan sebagai alat bantu penyusunan awal, sedangkan keputusan akhir tetap berada pada pengguna.

4.3.5 Implementasi Tampilan Visual dan Responsivitas Antarmuka

Secara umum, antarmuka Prolytics menggunakan layout berbasis sidebar, header, card, tabel, *dialog*, badge status, dan tombol aksi yang konsisten. Tampilan disesuaikan dengan *role* pengguna agar menu dan aksi yang tidak relevan tidak mengganggu proses kerja. Detail tampilan fitur telah dijelaskan pada masing-masing modul utama sehingga bagian ini hanya menegaskan konsistensi visual dan responsivitas antarmuka secara umum.

Sistem warna dan tema diterapkan untuk menjaga konsistensi visual sekaligus memberikan pilihan tampilan yang nyaman bagi pengguna. Prolytics menyediakan mode terang dan mode gelap yang dapat dipilih melalui fitur pergantian tema, dengan palet warna, kontras, dan komponen antarmuka yang tetap konsisten pada kedua mode tersebut.



Gambar 4. 45 Sistem Warna dan Tema



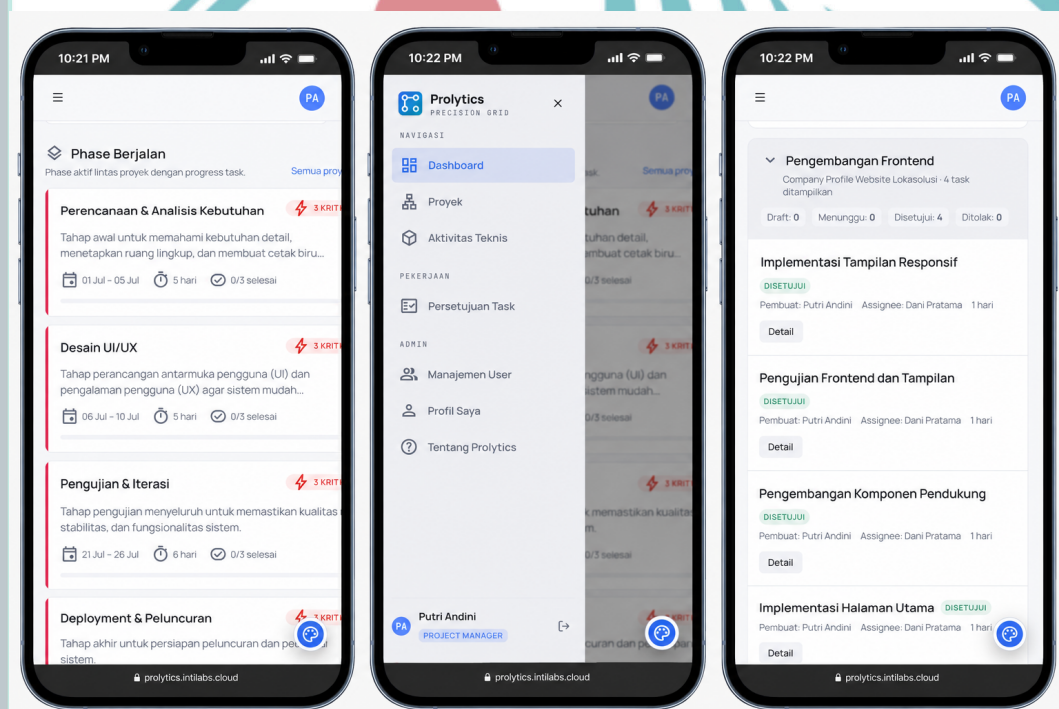
© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.45 memperlihatkan penerapan sistem warna dan tema pada antarmuka Prolytics. Penggunaan palet warna yang konsisten membantu pengguna mengenali status, aksi, dan komponen penting, sedangkan ketersediaan mode terang dan gelap memberikan fleksibilitas tampilan sesuai preferensi dan kondisi lingkungan kerja pengguna.

Selain sistem tema, antarmuka Prolytics juga dirancang responsif agar dapat digunakan pada berbagai ukuran layar. Pada perangkat mobile, tata letak menyesuaikan diri secara otomatis dan navigasi disusun ulang sehingga pengguna tetap dapat mengakses fitur utama dengan nyaman.



Gambar 4. 46 Responsivitas dan Navigasi Mobile

Gambar 4.46 menunjukkan tampilan Prolytics pada perangkat mobile beserta penyesuaian navigasinya. Elemen antarmuka seperti sidebar, tabel, dan tombol aksi disusun ulang agar tetap mudah diakses pada layar berukuran kecil, sehingga proses *monitoring* proyek tetap dapat dilakukan tanpa kehilangan fungsi utama.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.4 Pengujian Sistem

Pengujian sistem dilakukan untuk memastikan fungsi Prolytics berjalan sesuai kebutuhan dan perancangan yang telah dibuat. Pengujian difokuskan pada fungsi utama sistem, validasi hasil perhitungan CPM, serta penerimaan pengguna terhadap alur kerja aplikasi.

4.4.1 Deskripsi Pengujian

Pengujian yang digunakan terdiri dari *black box testing*, validasi perhitungan CPM, dan *User Acceptance Testing* (UAT). *Black box testing* digunakan untuk mengevaluasi fungsi utama sistem dari sisi *input*, aksi pengguna, dan *output* yang diharapkan tanpa melihat kode program secara langsung. Selain *black box testing* dan *User Acceptance Testing* (UAT), pengujian juga mencakup *Usability Testing* menggunakan metode *System Usability Scale* (SUS) untuk mengukur tingkat kemudahan penggunaan sistem Prolytics berdasarkan pengalaman pengguna sesuai perannya masing-masing.

4.4.2 Prosedur Pengujian

Prosedur pengujian disusun agar setiap skenario memiliki langkah yang jelas, data uji yang dapat ditelusuri, dan hasil yang dapat dibandingkan dengan ekspektasi. Prosedur ini menjadi acuan sebelum pengujian resmi dilakukan.

1. Prosedur *Black Box Testing*

Black Box Testing dilakukan dengan menjalankan fitur utama melalui antarmuka aplikasi. Setiap skenario diuji berdasarkan *input* pengguna, aksi yang dilakukan, *output* yang diharapkan, dan status hasil pengujian. Fitur yang diuji mencakup autentikasi, manajemen *project*, manajemen *phase*, *tugas*, *dependency*, *approval*, CPM, *GitHub*, *LLM*, dan profil pengguna.

2. Prosedur *User Acceptance Testing*

User Acceptance Testing dilakukan dengan melibatkan pengguna yang mewakili *role Project Manager*, *Lead Developer*, dan *Software Developer*. Pengguna diminta menjalankan skenario sesuai alur kerja masing-masing, kemudian memberikan



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

penilaian terhadap kemudahan penggunaan, kesesuaian fitur, dan manfaat informasi CPM.

3. Prosedur Usability Testing (SUS)

Usability Testing dilakukan untuk menilai tingkat kemudahan penggunaan antarmuka sistem oleh pengguna menggunakan metode *System Usability Scale* (SUS). Responden mencoba fitur yang relevan dengan tugas dan hak aksesnya, kemudian mengisi kuesioner yang terdiri atas sepuluh pernyataan. Setiap pernyataan dinilai menggunakan skala 1 sampai 5, dengan nilai 1 menunjukkan Sangat Tidak Setuju dan nilai 5 menunjukkan Sangat Setuju. Hasil jawaban dihitung menggunakan rumus SUS, yaitu pernyataan bernomor ganjil memperoleh kontribusi skor dari nilai jawaban dikurangi 1 dan pernyataan bernomor genap memperoleh kontribusi skor dari 5 dikurangi nilai jawaban, kemudian seluruh kontribusi dijumlahkan dan dikalikan 2,5 sehingga menghasilkan skor SUS pada rentang 0 sampai 100. Daftar pernyataan SUS yang digunakan disajikan pada Tabel 16.

Tabel 16 Daftar Pernyataan SUS

No	Pernyataan SUS
1	Saya rasa akan sering menggunakan sistem Prolytics ini.
2	Saya merasa sistem Prolytics ini terlalu rumit untuk digunakan.
3	Saya merasa sistem Prolytics ini mudah digunakan.
4	Saya merasa membutuhkan bantuan orang lain atau teknisi untuk dapat menggunakan sistem Prolytics ini.
5	Saya merasa fitur-fitur dalam sistem Prolytics ini saling terintegrasi dan berjalan dengan baik.
6	Saya merasa terdapat terlalu banyak ketidakkonsistenan dalam sistem Prolytics ini.
7	Saya merasa kebanyakan pengguna akan cepat memahami cara menggunakan sistem Prolytics ini.
8	Saya merasa sistem Prolytics ini membingungkan untuk digunakan.
9	Saya merasa percaya diri ketika menggunakan sistem Prolytics ini.
10	Saya perlu mempelajari banyak hal terlebih dahulu sebelum dapat menggunakan sistem Prolytics ini dengan baik.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.4.3 Hasil Pengujian

Pengujian sistem Prolytics dilaksanakan dalam dua tahap utama. Tahap pertama adalah black box testing yang mencakup pengujian fungsional seluruh fitur sistem dan validasi kebenaran perhitungan CPM. Tahap kedua adalah User Acceptance Testing (UAT) yang melibatkan pengguna langsung dari ketiga role utama. Berikut ini diuraikan hasil dari masing-masing tahap pengujian tersebut.

1. Hasil Black Box Testing

Pengujian *black box testing* dilakukan terhadap 59 skenario yang mencakup seluruh kebutuhan fungsional dan non-fungsional sistem Prolytics beserta validasi perhitungan CPM. Skenario 1 hingga 45 menguji kebutuhan fungsional sistem mulai dari autentikasi, manajemen pengguna, manajemen *project, phase*, tugas, *dependency, approval, change request*, perhitungan CPM, integrasi GitHub, hingga penggunaan LLM sebagai asisten *brief*. Skenario 46 hingga 54 menguji kebutuhan non-fungsional yang bersifat teramati, meliputi pembatasan akses berbasis *role*, keterlacakan, antarmuka, pergantian tema, dokumentasi API, paginasi, pencarian, dan *filtering* data. Skenario 55 hingga 59 merupakan validasi kebenaran perhitungan CPM yang mencakup pola *dependency* berurutan, *dependency* paralel, tugas dengan *slack* lebih dari nol, penolakan *circular dependency*, serta perhitungan ulang CPM pada *project* aktif. Setiap skenario ditelusuri ke kode kebutuhan fungsional (KF) atau non-fungsional (NFR) yang bersesuaian dan dibandingkan dengan hasil yang diharapkan. Hasil pengujian disajikan pada Tabel 17.

Tabel 17 Blackbox Testing

No.	Kode Kebutuhan	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
1	KF-01	Login dengan akun valid	Pengguna berhasil masuk dan diarahkan ke dashboard sesuai role	Pengguna masuk dan diarahkan ke dashboard sesuai role	Berhasil
2	KF-01	Login dengan akun tidak valid	Sistem menolak login dan menampilkan pesan kesalahan	Login ditolak dan pesan kesalahan tampil	Berhasil



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Kode Kebutuhan	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
3	KF-02	Logout dari sistem	Sesi pengguna berakhir dan diarahkan ke halaman login	Pengguna keluar dan diarahkan ke halaman login	Berhasil
4	KF-03	Membuka dashboard sesuai role (Project Manager, Lead Developer, Software Developer)	Dashboard menampilkan ringkasan sesuai role pengguna	Dashboard tampil sesuai role masing-masing pengguna	Berhasil
5	KF-04	Melihat profil pengguna	Data profil pengguna ditampilkan	Data profil tampil sesuai akun	Berhasil
6	KF-05	Memperbarui profil pengguna	Perubahan data profil tersimpan	Perubahan profil tersimpan dan tampil	Berhasil
7	KF-06	Mengunggah avatar pengguna	Avatar tersimpan dan tampil pada profil	Avatar tersimpan dan tampil	Berhasil
8	KF-07	Melihat daftar pengguna	Daftar pengguna ditampilkan	Daftar pengguna tampil	Berhasil
9	KF-08	Membuat pengguna baru	Data pengguna baru tersimpan	Pengguna baru tersimpan dan tampil pada daftar	Berhasil
10	KF-09	Memperbarui data pengguna	Perubahan data pengguna tersimpan	Data pengguna diperbarui	Berhasil
11	KF-10	Menghapus pengguna	Data pengguna terhapus dari sistem	Pengguna terhapus dari daftar	Berhasil
12	KF-11	Project Manager membuat proyek draft dan menentukan anggota	Proyek tersimpan berstatus draft beserta anggotanya	Proyek tersimpan sebagai draft dengan anggota	Berhasil



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Kode Kebutuhan	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
13	KF-11	Mengisi project brief dengan bantuan AI Brief Assistant	Brief tersusun dengan bantuan AI atau fallback saat AI tidak tersedia	Brief tersusun; fallback tampil saat AI tidak tersedia	Berhasil
14	KF-12	Memperbarui informasi proyek	Perubahan informasi proyek tersimpan	Informasi proyek diperbarui	Berhasil
15	KF-14	Mengelola anggota proyek	Perubahan anggota proyek tersimpan	Anggota proyek diperbarui	Berhasil
16	KF-13	Menghapus proyek	Proyek terhapus dari sistem	Proyek terhapus	Berhasil
17	KF-15	Mengaktifkan proyek setelah tugas siap dan disetujui	Status proyek berubah dari draft menjadi active	Proyek menjadi active setelah syarat terpenuhi	Berhasil
18	KF-16	Lead Developer membuat phase pada proyek draft	Phase tersimpan dan tampil pada detail proyek	Phase tersimpan dan tampil	Berhasil
19	KF-17	Memperbarui metadata phase	Perubahan metadata phase tersimpan	Metadata phase diperbarui	Berhasil
20	KF-18	Mengelola anggota phase	Perubahan anggota phase tersimpan	Anggota phase diperbarui	Berhasil
21	KF-19	Membuat technical tugas (durasi, assignee, status, dependency)	Tugas tersimpan lengkap dengan atributnya	Tugas tersimpan dengan durasi, assignee, status, dan dependency	Berhasil
22	KF-20	Memperbarui tugas	Perubahan data tugas tersimpan	Tugas diperbarui	Berhasil
23	KF-21	Menghapus tugas	Tugas terhapus dari phase	Tugas terhapus	Berhasil
24	KF-22	Mengatur dependency valid antar-tugas	Dependency tersimpan dan tampil pada tugas terkait	Dependency tersimpan dan tampil	Berhasil



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Kode Kebutuhan	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
25	KF-23	Lead Developer mengajukan tugas untuk approval	Status tugas berubah menjadi pending	Status tugas menjadi pending	Berhasil
26	KF-24	Project Manager melihat antrean persetujuan tugas	Antrean tugas menunggu persetujuan ditampilkan	Antrean persetujuan tugas tampil	Berhasil
27	KF-25	Project Manager menyetujui tugas	Status tugas berubah menjadi approved	Status tugas menjadi approved	Berhasil
28	KF-26	Project Manager menolak tugas	Status tugas berubah menjadi rejected	Status tugas menjadi rejected	Berhasil
29	KF-27	Project Manager meminta revisi tugas	Status tugas berubah menjadi menunggu revisi	Status tugas menjadi menunggu revisi	Berhasil
30	KF-28	Lead Developer merevisi tugas yang ditolak atau diminta revisi	Tugas dapat diperbaiki dan diajukan kembali	Tugas diperbaiki dan diajukan kembali	Berhasil
31	KF-29	Lead Developer mengajukan perubahan pada tugas aktif	Pengajuan perubahan tugas tersimpan	Pengajuan perubahan tugas tersimpan	Berhasil
32	KF-30	Project Manager melihat antrean persetujuan perubahan tugas	Antrean perubahan tugas ditampilkan	Antrean perubahan tugas tampil	Berhasil
33	KF-31	Project Manager menyetujui perubahan tugas	Perubahan tugas diterapkan dan memicu perhitungan ulang CPM	Perubahan tugas diterapkan dan CPM dihitung ulang	Berhasil
34	KF-32	Project Manager menolak perubahan tugas	Perubahan tugas ditolak dan data tugas tetap	Perubahan tugas ditolak	Berhasil



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Kode Kebutuhan	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
35	KF-33	Software Developer melihat tugas yang ditugaskan pada halaman Tugas Saya	Daftar tugas beserta status readiness ditampilkan	Daftar tugas dan readiness tampil pada Tugas Saya	Berhasil
36	KF-33	Software Developer memulai tugas	Status tugas berubah menjadi in progress	Status tugas menjadi in progress	Berhasil
37	KF-33	Software Developer menyelesaikan tugas	Status tugas menjadi done dan progress selesai	Status tugas menjadi done, progress selesai	Berhasil
38	KF-34	Melihat hasil perhitungan CPM project-wide	Sistem menampilkan nilai ES, EF, LS, LF, dan slack	Nilai ES, EF, LS, LF, dan slack tampil	Berhasil
39	KF-35	Melihat jalur kritis	Tugas dengan slack nol ditandai sebagai tugas kritis	Tugas slack nol ditandai sebagai tugas kritis	Berhasil
40	KF-36	Melihat visualisasi CPM	Graf dependency dan jalur kritis ditampilkan secara visual	Visualisasi dependency dan jalur kritis tampil	Berhasil
41	KF-37	Melihat riwayat snapshot CPM	Riwayat snapshot perhitungan CPM ditampilkan	Riwayat snapshot CPM tampil	Berhasil
42	KF-38	Memicu perhitungan ulang CPM saat data tugas berubah	Sistem menghitung ulang dan menampilkan hasil CPM terbaru	Hasil CPM diperbarui setelah data tugas berubah	Berhasil
43	KF-39	Melihat aktivitas teknis dari repository GitHub yang terhubung	Activity feed GitHub ditampilkan pada proyek	Activity feed GitHub tampil	Berhasil



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Kode Kebutuhan	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
44	KF-40	Menautkan aktivitas teknis GitHub ke tugas	Aktivitas GitHub tertaut pada tugas terkait	Aktivitas GitHub tertaut pada tugas	Berhasil
45	KF-41	Melepas tautan aktivitas GitHub dari tugas	Tautan aktivitas GitHub terlepas dari tugas	Tautan aktivitas GitHub terlepas	Berhasil
46	NFR-01	Akses menu dan aksi dibatasi berdasarkan role	Menu dan aksi hanya tampil sesuai hak akses role	Menu dan aksi tampil sesuai hak akses	Berhasil
47	NFR-02	Menambahkan dependency yang membentuk circular dependency	Sistem menolak dependency yang membentuk siklus	Dependency yang membentuk siklus ditolak sistem	Berhasil
48	NFR-03	Menampilkan riwayat approval dan timeline perubahan	Timeline approval dan log perubahan ditampilkan sesuai aksi	Timeline approval dan log tampil sesuai aksi	Berhasil
49	NFR-05	Menampilkan antarmuka berbasis tabel, badge, dan visualisasi sesuai role	Antarmuka menampilkan badge status, tabel, dan visualisasi sesuai role	Badge, tabel, dan visualisasi tampil sesuai role	Berhasil
50	NFR-07	Mengganti tema tampilan terang dan gelap (switch theme)	Tema berubah sesuai pilihan dan tersimpan	Tema terang/gelap berubah dan tersimpan	Berhasil
51	NFR-08	Mengakses dokumentasi API berbasis Swagger	Dokumentasi endpoint API ditampilkan	Dokumentasi endpoint Swagger tampil	Berhasil
52	NFR-09	Menampilkan daftar data dalam jumlah besar dengan paginasi	Data ditampilkan terbagi menggunakan pagination	Data tampil terpaginasi	Berhasil



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Kode Kebutuhan	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
53	NFR-10	Mencari data menggunakan fitur pencarian	Sistem menampilkan data sesuai kata kunci	Hasil pencarian sesuai kata kunci	Berhasil
54	NFR-11	Menyaring data menggunakan fitur filtering	Sistem menampilkan data sesuai kriteria filter	Data tersaring sesuai kriteria	Berhasil
55	NFR-04	Validasi CPM: pola dependency berurutan (A - B - C)	Nilai ES, EF, LS, LF, dan slack sesuai perhitungan manual	Hasil CPM sesuai perhitungan manual	Berhasil
56	NFR-04	Validasi CPM: pola dependency paralel dua jalur	Jalur kritis sama dengan jalur berdurasi terpanjang	Jalur kritis sesuai jalur terpanjang	Berhasil
57	NFR-04	Validasi CPM: tugas dengan slack lebih dari nol	Tugas non-kritis teridentifikasi sesuai perhitungan manual	Tugas non-kritis sesuai perhitungan manual	Berhasil
58	NFR-02	Validasi CPM: penolakan circular dependency	Perhitungan tidak dijalankan pada graf non-DAG	Circular dependency ditolak, CPM tidak dijalankan	Berhasil
59	NFR-04	Validasi CPM: perhitungan ulang saat perubahan durasi/dependency	Hasil CPM diperbarui sesuai perhitungan manual terbaru	Hasil CPM diperbarui sesuai perhitungan manual	Berhasil

Berdasarkan Tabel 17, seluruh 59 skenario *black box testing* menunjukkan hasil Berhasil. Skenario 1 hingga 45 membuktikan seluruh kebutuhan fungsional sistem Prolytics berjalan sesuai hasil yang diharapkan, mencakup autentikasi, manajemen pengguna, manajemen *project*, *phase*, tugas, *dependency*, *approval*, *change request*, perhitungan CPM, integrasi GitHub, serta *dashboard* untuk setiap *role* pengguna. Skenario 46 hingga 54 membuktikan kebutuhan non-fungsional yang teramati telah terpenuhi, mulai dari pembatasan akses berbasis *role* hingga fitur pencarian dan *filtering* data. Skenario 55 hingga 59 memvalidasi kebenaran perhitungan CPM, yaitu bahwa sistem menghasilkan nilai ES, EF, LS, LF, dan *slack*



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

yang konsisten dengan perhitungan manual, serta mampu mendeteksi dan menolak graf yang tidak membentuk *Directed Acyclic Graph* (DAG). Setiap skenario pada Tabel 17 ditelusuri ke kode kebutuhan fungsional (KF) dan non-fungsional (NFR) sehingga keterlacakan pengujian terhadap kebutuhan sistem dapat ditunjukkan. Selanjutnya dilakukan *User Acceptance Testing* (UAT) untuk memperoleh penilaian langsung dari pengguna terhadap kegunaan dan kesesuaian sistem.

2. Hasil *User Acceptance Testing*

User Acceptance Testing (UAT) dilakukan untuk mengevaluasi apakah sistem memenuhi kebutuhan dan ekspektasi pengguna akhir secara langsung. Pengujian melibatkan responden yang mewakili tiga *role* utama *Project Manager*, *Lead Developer*, dan *Software Developer* yang diminta menjalankan skenario sesuai alur kerja masing-masing. Sebanyak 25 skenario diuji mencakup seluruh alur kerja sistem dari pembuatan *project*, penyusunan *breakdown* teknis, *approval* tugas, eksekusi pekerjaan, hingga pemantauan *dashboard* dan integrasi GitHub. Hasil UAT disajikan pada Tabel 18.

Tabel 18 User Acceptance Testing

No.	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
1	Project Manager login ke sistem	Project Manager berhasil masuk ke dashboard sesuai role	Project Manager masuk ke dashboard sesuai role	Sesuai
2	Project Manager membuat project baru	Project berhasil dibuat sebagai draft	Project baru berhasil dibuat sebagai draft	Sesuai
3	Project Manager menyusun project brief	Brief tersimpan dan dapat dibaca pada detail project	Brief tersimpan dan terbaca pada detail project	Sesuai
4	Project Manager menggunakan bantuan AI untuk brief	AI membantu menyusun brief menjadi lebih terstruktur	Brief tersusun lebih terstruktur dengan bantuan AI	Sesuai

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
5	Project Manager melihat project draft tanpa phase	Sistem menampilkan informasi bahwa breakdown teknis belum dibuat	Informasi bahwa breakdown teknis belum dibuat tampil	Sesuai
6	Lead Developer melihat brief project	Lead Developer dapat memahami kebutuhan project dari brief yang dibuat PM	Lead Developer memahami kebutuhan project dari brief PM	Sesuai
7	Lead Developer membuat breakdown teknis	Phase dan tugas teknis dapat dibuat berdasarkan brief project	Phase dan tugas teknis dibuat berdasarkan brief project	Sesuai
8	Lead Developer membuat dependency tugas	Dependency antar tugas tersimpan dan digunakan dalam perhitungan CPM	Dependency tersimpan dan digunakan dalam perhitungan CPM	Sesuai
9	Lead Developer melihat hasil CPM	Nilai CPM dan tugas kritis dapat ditampilkan dengan jelas	Nilai CPM dan tugas kritis tampil dengan jelas	Sesuai
10	Lead Developer mengajukan tugas ke PM	Tugas masuk ke daftar approval dengan status pending	Tugas masuk ke daftar approval berstatus pending	Sesuai
11	Project Manager menyetujui tugas	Tugas berubah menjadi approved dan siap dikerjakan	Tugas menjadi approved dan siap dikerjakan	Sesuai
12	Project Manager menolak tugas	Tugas berubah menjadi rejected dan dapat diperbaiki oleh Lead Developer	Tugas menjadi rejected dan dapat diperbaiki Lead Developer	Sesuai
13	Project Manager mengaktifkan project	Project berhasil berubah menjadi active setelah tugas valid	Project menjadi active setelah tugas valid	Sesuai
14	Software Developer melihat daftar tugas	Software Developer hanya melihat tugas yang ditugaskan kepadanya	Hanya tugas yang ditugaskan tampil bagi Software Developer	Sesuai



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
15	Software Developer memulai pengerjaan tugas	Status tugas berubah menjadi in progress	Status tugas berubah menjadi in progress	Sesuai
16	Software Developer menyelesaikan tugas	Status tugas berubah menjadi done	Status tugas berubah menjadi done	Sesuai
17	Project Manager memantau dashboard	Dashboard menampilkan informasi project, tugas kritis, approval, bottleneck, dan workload	Dashboard menampilkan project, tugas kritis, approval, bottleneck, dan workload	Sesuai
18	Lead Developer memantau dashboard	Dashboard membantu melihat dependency, blocked tugas, dan tugas tanpa dependency	Dashboard menampilkan dependency, blocked tugas, dan tugas tanpa dependency	Sesuai
19	Software Developer memantau dashboard	Dashboard membantu mengetahui tugas siap dikerjakan dan tugas terblokir	Dashboard menampilkan tugas siap dikerjakan dan tugas terblokir	Sesuai
20	Pengguna melihat detail tugas	Detail tugas menampilkan informasi lengkap mengenai tugas, CPM, dependency, approval, dan aktivitas teknis	Detail tugas tampil lengkap beserta CPM, dependency, approval, dan aktivitas teknis	Sesuai
21	Pengguna melihat riwayat approval	Riwayat approval tampil dalam bentuk timeline	Riwayat approval tampil dalam bentuk timeline	Sesuai
22	Pengguna melihat aktivitas GitHub project	Aktivitas teknis project dapat ditampilkan sebagai bukti monitoring	Aktivitas teknis project tampil sebagai bukti monitoring	Sesuai
23	Pengguna menghubungkan	Aktivitas GitHub berhasil dikaitkan dengan tugas tertentu	Aktivitas GitHub berhasil dikaitkan dengan tugas	Sesuai

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No.	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
	aktivitas GitHub ke tugas			
24	Pengguna membuka halaman project detail	Project detail menampilkan brief, phase, tugas, member, repository, dan aktivitas teknis sesuai akses	Project detail tampil lengkap sesuai hak akses	Sesuai
25	Pengguna logout dari sistem	Pengguna berhasil keluar dari sistem	Pengguna berhasil keluar dari sistem	Sesuai

Berdasarkan Tabel 18, seluruh 25 skenario UAT menunjukkan hasil sesuai. Pengguna dari ketiga *role Project Manager, Lead Developer, dan Software Developer* berhasil menyelesaikan alur kerja sesuai skenario yang telah ditetapkan. Sistem dinilai mampu mendukung kebutuhan operasional masing-masing peran, mulai dari pembuatan dan pengelolaan *project*, penyusunan *breakdown* teknis, proses *approval*, eksekusi dan pemantauan tugas, hingga visualisasi informasi CPM dan integrasi GitHub pada *dashboard*. Dokumentasi pendukung, data responden, dan bukti pelaksanaan UAT akan disertakan pada lampiran penelitian.

3. Hasil Usability Testing (SUS)

Usability Testing dengan metode SUS melibatkan delapan responden yang mewakili ketiga peran pengguna sistem Prolytics, yaitu satu Project Manager, tiga Lead Developer, dan empat Software Developer. Setiap responden mencoba fitur sesuai perannya, kemudian mengisi sepuluh pernyataan SUS. Rincian responden disajikan pada Tabel 19.

Tabel 19 Peran Responden

Kode	Peran	Keterangan
R1	<i>Project Manager</i>	Menguji alur <i>project, approval, dan monitoring</i> CPM.
R2	<i>Lead Developer</i>	Menguji pengelolaan tugas, <i>dependency</i> , dan pengajuan tugas.
R3	<i>Lead Developer</i>	Menguji pengelolaan <i>phase</i> , tugas, dan aktivitas teknis.
R4	<i>Software Developer</i>	Menguji pembaruan status tugas dan <i>dashboard</i> .
R5	<i>Software Developer</i>	Menguji pengerjaan tugas dan informasi <i>readiness</i> .



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode	Peran	Keterangan
R6	<i>Lead Developer</i>	Menguji pengelolaan phase, tugas, dan pengajuan perubahan.
R7	<i>Software Developer</i>	Menguji pembaruan status tugas dan aktivitas teknis.
R8	<i>Software Developer</i>	Menguji pengerjaan tugas dan pemantauan dashboard.

Hasil jawaban responden terhadap sepuluh pernyataan SUS beserta skor masing-masing disajikan pada Tabel 20. Setiap skor dihitung menggunakan rumus SUS.

Tabel 20 Rekapitulasi Jawaban Usability Testing Menggunakan SUS

Responden	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	Skor SUS
R1	5	2	5	1	4	2	4	2	5	2	85,0
R2	5	2	4	2	4	2	4	2	4	2	77,5
R3	5	1	5	1	5	2	4	1	4	2	90,0
R4	4	2	4	1	4	2	4	2	4	2	77,5
R5	5	2	4	2	4	2	4	2	4	1	80,0
R6	5	2	4	2	5	2	5	2	4	3	80,0
R7	5	2	5	1	5	2	5	3	4	1	87,5
R8	5	2	5	2	5	1	4	1	1	1	87,5

Setelah skor SUS masing-masing responden diperoleh, nilai tersebut direkap untuk mengetahui jumlah responden, skor terendah, skor tertinggi, dan rata-rata skor SUS sebagaimana disajikan pada Tabel 21.

Tabel 21 Rekapitulasi Skor SUS

Keterangan	Nilai
Jumlah responden	8
Skor SUS terendah	77,5
Skor SUS tertinggi	90
Rata-rata skor SUS	83,1

Berdasarkan hasil perhitungan SUS, diperoleh skor terendah sebesar 77,5 dan skor tertinggi sebesar 90. Rata-rata skor SUS dari seluruh responden adalah 83,1. Hasil tersebut kemudian dianalisis lebih lanjut menggunakan skala penilaian SUS yang mencakup *Acceptability*, *Grade Scale*, dan *Adjective Rating*.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.4.4 Analisis Data dan Evaluasi Hasil Pengujian

Analisis data pengujian dilakukan untuk mengevaluasi hasil dari tiga teknik pengujian yang digunakan, yaitu *black box testing*, *User Acceptance Testing* (UAT), dan *Usability Testing* menggunakan metode *System Usability Scale* (SUS). Evaluasi bertujuan untuk mengetahui apakah sistem Prolytics telah berjalan sesuai kebutuhan fungsional, diterima oleh pengguna, serta memiliki tingkat kemudahan penggunaan yang memadai. Hasil pengujian dianalisis secara deskriptif berdasarkan jumlah skenario yang berhasil, kesesuaian terhadap kebutuhan pengguna, serta skor *usability* yang diperoleh.

1. Analisis Hasil *Blackbox Testing*

Berdasarkan hasil *black box testing* terhadap 59 skenario yang terdiri atas 45 skenario kebutuhan fungsional, 9 skenario kebutuhan non-fungsional, dan 5 skenario validasi perhitungan CPM, seluruh skenario memperoleh status Berhasil. Dengan demikian, persentase keberhasilan pengujian adalah 100% dan tidak terdapat skenario yang gagal. Setiap skenario telah ditelusuri ke kode kebutuhan fungsional (KF) dan non-fungsional (NFR) pada Tabel 7 dan Tabel 8, sehingga seluruh kebutuhan fungsional dan kebutuhan non-fungsional yang bersifat teramati terbukti telah diuji. Kebutuhan non-fungsional NFR-06 terkait maintainability bersifat internal sehingga diverifikasi melalui struktur kode pada subbab implementasi, bukan melalui *black box testing*. Pada validasi perhitungan CPM, hasil sistem untuk pola *dependency* berurutan, *dependency* paralel, tugas dengan *slack* lebih dari nol, penolakan *circular dependency*, serta perhitungan ulang CPM pada *project* aktif telah sesuai dengan perhitungan manual yang disiapkan.

2. Analisis Hasil *User Acceptance Testing*

Berdasarkan hasil *User Acceptance Testing* terhadap 25 skenario yang melibatkan pengguna dari ketiga peran, yaitu *Project Manager*, *Lead Developer*, dan *Software Developer*, seluruh skenario memperoleh status Sesuai. Hasil ini menunjukkan bahwa alur kerja sistem diterima oleh pengguna dan telah sesuai dengan kebutuhan serta tanggung jawab masing-masing peran, mulai dari penyusunan dan *approval* tugas, pemantauan hasil CPM, hingga pembaruan status pekerjaan.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

3. Analisis Hasil *Usability Testing* (SUS)

Berdasarkan hasil *Usability Testing* menggunakan metode SUS terhadap delapan responden, diperoleh skor terendah 77,5, skor tertinggi 90,0, dan rata-rata 83,1. Berdasarkan rata-rata tersebut, sistem Prolytics berada pada kategori *Acceptable*, memperoleh *Grade A*, dan *Adjective Rating Excellent* karena skor melebihi 80,2. Hasil ini menunjukkan bahwa sistem dinilai mudah digunakan oleh pengguna dari ketiga peran utama.

4. Evaluasi Keseluruhan Pengujian

Berdasarkan hasil ketiga teknik pengujian, sistem Prolytics yang dibangun telah menunjukkan hasil yang baik. *Black box testing* memperoleh keberhasilan 100% dari 59 skenario, *User Acceptance Testing* memperoleh kesesuaian 100% dari 25 skenario, dan *Usability Testing* memperoleh rata-rata skor SUS sebesar 83,1. Ketiga hasil tersebut menunjukkan bahwa sistem telah memenuhi kebutuhan fungsional dan non-fungsional, memenuhi kriteria penerimaan pengguna, serta memiliki tingkat kemudahan penggunaan yang memadai.

Secara fungsional, sistem mampu menjalankan modul utama yang berada dalam ruang lingkup penelitian, yaitu autentikasi dan hak akses berbasis *role*, manajemen pengguna, manajemen *project, phase*, dan tugas, pengaturan *dependency, approval* dan *change request* tugas, perhitungan *Critical Path Method*, integrasi aktivitas GitHub, serta *dashboard* sesuai peran. Secara penerimaan pengguna, sistem telah sesuai dengan kebutuhan kerja *Project Manager, Lead Developer*, dan *Software Developer*. Secara *usability*, sistem memperoleh skor SUS sebesar 83,1 yang berdasarkan interpretasi skala SUS termasuk kategori *Acceptable*, memperoleh *Grade A*, dan *Adjective Rating Excellent*, sehingga dinilai memiliki tingkat kemudahan penggunaan yang sangat baik dan dapat diterima oleh pengguna internal.

Dengan demikian, hasil evaluasi pengujian menunjukkan bahwa sistem Prolytics layak digunakan untuk mendukung monitoring proyek perangkat lunak dengan identifikasi tugas kritis menggunakan *Critical Path Method* di PT Lokasolusi Teknologi Indonesia. Sistem menjawab permasalahan utama pada proses bisnis eksisting, yaitu informasi proyek yang tersebar, kesiapan tugas yang sulit dipantau,



© Hak Cipta milik Politeknik Negeri Jakarta

serta monitoring jalur kritis yang belum terpusat, sekaligus mendukung visibilitas berbasis *role* sehingga setiap pengguna dapat menjalankan tugas sesuai tanggung jawabnya.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB 5 PENUTUP

5.1 Kesimpulan

Penerapan *Critical Path Method* (CPM) pada penelitian ini menghasilkan urutan pengerjaan *tugas* berdasarkan *estimated_duration* dan *dependency* antar *approved tugas*. Melalui proses tersebut, sistem dapat mengidentifikasi *tugas* kritis, nilai *slack* dalam hari kalender, jalur kritis, serta *calculated_start_date* dan *calculated_end_date phase* secara otomatis melalui *backend*. Hasil ini membantu *Project Manager* dan *Lead Developer* dalam memantau prioritas pekerjaan dan mendukung pengambilan keputusan. Selain fitur utama tersebut, sistem juga menyediakan *role-based access control*, *dashboard monitoring* berbasis *role*, integrasi aktivitas teknis GitHub, serta bantuan AI untuk penyusunan *brief*. Berdasarkan hasil *black box testing* terhadap 59 skenario, yang terdiri dari 45 skenario fungsional, 9 skenario nonfungsional, dan 5 skenario validasi perhitungan CPM, seluruh skenario menunjukkan hasil berhasil. Selain itu, *User Acceptance Testing* (UAT) terhadap 25 skenario yang melibatkan *Project Manager*, *Lead Developer*, dan *Software Developer* menunjukkan bahwa alur kerja sistem diterima dan sesuai dengan kebutuhan pengguna.

5.2 Saran

Untuk pengembangan penelitian selanjutnya, sistem Prolytics dapat ditingkatkan dengan beberapa fitur yang lebih mendukung kebutuhan operasional PT Lokasolusi secara luas. Pertama, penambahan *actual_duration* untuk mendukung fitur *baseline vs actual*, *monitoring progress keterlambatan*, dan evaluasi kinerja pengerjaan tugas. Kedua, penerapan *working-day calendar* dan *holiday calendar* agar perhitungan ES, EF, LS, LF, dan *slack* memperhitungkan hari libur dan hari kerja aktual. Ketiga, integrasi *resource availability* dan *workload* atau *resource leveling* agar sistem dapat mendeteksi konflik penugasan developer. Keempat, pengembangan analitik snapshot CPM project dari waktu ke waktu untuk memvisualisasikan evolusi jalur kritis selama project berjalan. Selain itu, integrasi GitHub dan fitur AI juga dapat dikembangkan lebih lanjut agar sistem mampu memberikan insight yang lebih kontekstual berdasarkan data proyek yang berjalan.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR PUSTAKA

- Abraham, J., Ismail, I.E., 2021. Unit Testing dan User Acceptance Testing pada Sistem Informasi Pelayan Kategorial Pelayanan Anak. Repository PNJ 1, 54.
- Agustin, Y.H., Kustiana, R.M., 2025. Rancang Bangun Sistem Manajemen Proyek Perangkat Lunak Berbasis Web Menggunakan Metode Rational Unified Process. Jurnal Algoritma 22, 1282–1292. <https://doi.org/10.33364/algoritma/v.22-2.2577>
- Aji Bayu Prasetyo, D., Alfa Susetyo, Y., 2022. Pengertian PostgreSQL 9, 2407–4322.
- Alfarizi M. R. S., Al-farish, M. Z., Taufiqurrahman, M., Ardiansah, G., Elgar, M., 2023. Penggunaan Python Sebagai Bahasa Pemrograman untuk Machine Learning dan Deep Learning. Karya Ilmiah Mahasiswa Bertauhid (KARIMAH TAUHID) 2, 1–6.
- Angular, 2026. Home • Angular [WWW Document]. URL <https://angular.dev/> (accessed 6.11.26).
- AWS, 2026. Apa itu API RESTful? - Penjelasan tentang API RESTful - AWS [WWW Document]. URL <https://aws.amazon.com/what-is/restful-api/> (accessed 6.11.26).
- Benz, R. Al, Testiana, G., 2025. Penerapan Tailwind CSS Dalam Pengembangan User Interface Website Di Poltekpar Palembang. RIGGS: Journal of Artificial Intelligence and Digital Business 4, 6635–6643. <https://doi.org/10.31004/riggs.v4i4.4431>



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Echo, 2026. Introduction | Echo [WWW Document]. URL <https://echo.labstack.com/docs> (accessed 6.11.26).

Encore, 2026. Gin vs Echo vs Fiber 2026 - Go Web Framework Comparison – Encore [WWW Document]. URL <https://encore.dev/articles/gin-vs-echo-vs-fiber> (accessed 6.11.26).

Erwin Erwin, Purbawati Purbawati, Diyaa Aaisyah Salmaa Putri Atmaja, Ida Rosanti, 2025. Penerapan Metode CPM (Critical Path Method) dalam Manajemen Servis Ganset Caterpillar di PT. Sigma Quantum Insani Balikpapan Kalimantan Timur. Mars: Jurnal Teknik Mesin, Industri, Elektro Dan Ilmu Komputer 3, 21–30. <https://doi.org/10.61132/mars.v3i4.921>

Febriani, R., Kania, R., Maharani, D., Hendra, Y., Rakhmat, E., Komputer, F.I., Jaya, U.B., Serang, C.K., Banten, P., 2025. Perancangan Sistem Manajemen Proyek Menggunakan Critical Path Method (CPM) di PT. Adhi Putra Bangun Perkasa. Jurnal Ilmiah Sains dan Teknologi (Saintek) 9, 254–265.

Fiber, 2026. Fiber [WWW Document]. URL <https://gofiber.io> (accessed 6.11.26).

Firdaus, M., Afwani, R., 2024. Pengembangan Restful Api Untuk Aplikasi Klasifikasi Jenis Tanah Berbasis Mobile Pada Google Cloud. Jurnal Teknologi Informasi, Komputer, dan Aplikasinya (JTika) 6, 275–287. <https://doi.org/10.29303/jtika.v6i1.335>

Gin, 2026. Documentation | Gin Web Framework [WWW Document]. URL <https://gin-gonic.com/en/docs> (accessed 6.11.26).



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Github, n.d. About webhooks - GitHub Docs [WWW Document]. URL

<https://docs.github.com/en/webhooks/about-webhooks>
(accessed 6.9.26).

Go Programming Language, 2026. A Tour of Go [WWW Document]. URL <https://go.dev/tour/concurrency/10>
(accessed 6.11.26).

Gorm, 2026. GORM Guides | GORM - The fantastic ORM library for Golang, aims to be developer friendly. [WWW Document]. URL <https://gorm.io/docs/index.html>
(accessed 6.11.26).

GraphQL, 2026. Learn | GraphQL [WWW Document]. URL <https://graphql.org/learn/> (accessed 6.11.26).

Ikhwan, 2026. Pengembangan RESTFul Web Service Untuk Integrasi Data Pada Sistem Informasi Akademik 5, 3121–3128.

Ismail, I.E., Nalawati, R.E., Putra, A., 2021. System Usability Scale and Net Promoter Score on Donation Application of Toddlers Equipment. Proceedings - 2021 4th International Conference on Computer and Informatics Engineering: IT-Based Digital Industrial Innovation for the Welfare of Society, IC2IE 2021 170–174.
<https://doi.org/10.1109/IC2IE53219.2021.9649186>

Muhammad Arofiq, N., Ferdo Erlangga, R., Irawan, A., Saifudin, A., 2023. OKTAL : Jurnal Ilmu Komputer dan Science Pengujian Fungsional Aplikasi Inventory Barang Kedatangan Dengan Metode Black Box Testing Bagi Pemula. Ilmu Komputer dan Science 2, 1322–1330.

MySQL, 2026. MySQL :: MySQL 9.7 Reference Manual :: 1.2.1 What is MySQL? [WWW Document]. URL



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

https://dev.mysql.com/doc/refman/9.7/en/what-is-mysql.html?utm_source=chatgpt.com (accessed 6.11.26).

Next.js, n.d. Next.js Docs | Next.js [WWW Document]. URL <https://nextjs.org/docs> (accessed 6.11.26).

Node, 2026. Overview of Blocking vs Non-Blocking | Node.js Learn [WWW Document]. URL <https://nodejs.org/learn/asynchronous-work/overview-of-blocking-vs-non-blocking> (accessed 6.11.26).

OpenAPI, 2026. OpenAPI Specification v3.2.0 [WWW Document]. URL <https://spec.openapis.org/oas/v3.2.0.html> (accessed 6.11.26).

PostgreSQL, 2026. PostgreSQL: Documentation: 18: Chapter 8. Data Types [WWW Document]. URL <https://www.postgresql.org/docs/current/datatype.html> (accessed 6.9.26).

Pramudya, B.Z., Raharja, P.A., 2022. Evaluasi Usability Aplikasi Augmented Reality Morphfun Menggunakan System Usability Scale. *Multinetics* 8, 122–130. <https://doi.org/10.32722/multinetics.v8i2.4702>

Puan Maharani, 2025. Pengembangan Website PT. Rantingin Digital Indonesia Menggunakan Framework Next Js dan Tailwind CSS. *Repeater : Publikasi Teknik Informatika dan Jaringan* 3, 129–137. <https://doi.org/10.62951/repeater.v3i1.355>

Raharja, P.A., Pramudya, B.Z., 2022. Evaluasi Usability Aplikasi Augmented Reality Morphfun Menggunakan System Usability Scale. *Multinetics* 8, 122–130. <https://doi.org/10.32722/multinetics.v8i2.4702>



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

React, 2026. React [WWW Document]. URL <https://react.dev/> (accessed 6.11.26).

Ren, Y., Li, J., 2023. Research on Software Project Schedule Planning Technology Based on the Integration of PERT and CPM. *Procedia Comput. Sci.* 228, 253–261. <https://doi.org/10.1016/j.procs.2023.11.029>

Swagger, 2026. Swagger Documentation | Swagger Docs [WWW Document]. URL <https://swagger.io/docs/> (accessed 6.11.26).

TailwindCSS, 2026. Tailwind CSS - Rapidly build modern websites without ever leaving your HTML. [WWW Document]. URL <https://tailwindcss.com/> (accessed 6.11.26).

Umar, M.A., Lano, K., 2024. Advances in automated support for requirements engineering: a systematic literature review, *Requirements Engineering*. Springer London. <https://doi.org/10.1007/s00766-023-00411-0>

Uptrace, 2026. bun package - github.com/uptrace/bun - Go Packages [WWW Document]. URL <https://pkg.go.dev/github.com/uptrace/bun> (accessed 6.9.26).

Vite, 2026. Getting Started | Vite [WWW Document]. URL <https://vite.dev/guide/> (accessed 6.11.26).

Vue.js, 2026. Vue.js - The Progressive JavaScript Framework | Vue.js [WWW Document]. URL <https://vuejs.org/> (accessed 6.11.26).

DAFTAR RIWAYAT HIDUP PENULIS



Ragiliawan Putra Rencana

Lahir di Bekasi pada 21 Januari 2004. Lulus dari SD Islam Al Fajar, SMP Islam Al Fajar, dan SMA Negeri 11 Bekasi. Melanjutkan studi tahun 2022 di CCIT-FTUI dan saat ini sedang menempuh pendidikan Diploma IV pada Program Studi Teknik Informatika Jurusan Teknik Informatika dan Komputer di Politeknik Negeri Jakarta.

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

LAMPIRAN

Lampiran 1 Transkrip Wawancara dengan *Project Manager*

Narasumber : Sita Amira Syarifah

Jabatan : Project Manager

Instansi : PT Lokasolusi Tekno*logi* Indonesia

Tanggal : 09 April 2026

Pewawancara / Narasumber	Keterangan
Ragil (Pewawancara)	Halo semuanya, perkenalkan nama saya Ragil Irawan Putra, mahasiswa tingkat akhir di Politeknik Negeri Jakarta. Di sini saya bersama Kak Sita. Halo Kak Sita, selamat sore.
Sita (Narasumber)	Halo, selamat sore Mas Ragil.
Ragil (Pewawancara)	Boleh dijelaskan peran dan tanggung jawab Kak Sita sebagai <i>Project Manager</i> di PT Loka Solusi?
Sita (Narasumber)	Nama saya Sita Mira Syarifah dan saya berperan sebagai <i>Project Manager</i> di PT Loka Solusi. Tanggung jawab saya adalah memastikan <i>project</i> berjalan sesuai target waktu dan business goal. Saya juga berperan dalam menerima kebutuhan dari <i>stakeholder</i> atau <i>client</i> , menyusun atau memvalidasi <i>backlog</i> awal, membagi koordinasi kerja ke tim, melakukan <i>tracking</i> progress, serta menjadi penghubung antara kebutuhan bisnis dan tim teknis. Selain itu, saya



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

	juga ikut menentukan prioritas pekerjaan ketika ada perubahan kebutuhan atau tingkat urgensi tugas.
Ragil (Pewawancara)	Bagaimana alur <i>project</i> dari awal sampai tugas bisa mulai dikerjakan?
Sita (Narasumber)	Biasanya <i>project</i> dimulai dari meeting atau pembahasan kebutuhan dengan <i>client</i> atau pihak internal. Dari meeting itu saya membuat notulen atau ringkasan kebutuhan <i>project</i> . Hasil notulen tersebut belum langsung menjadi tugas teknis, tetapi biasanya disimpan dulu di media kerja seperti <i>spreadsheet</i> , dokumen, atau tools lain. Setelah itu, dilakukan pembahasan internal dengan tim teknis untuk menerjemahkan hasil meeting menjadi <i>backlog</i> atau tugas yang bisa dikerjakan. Dalam beberapa kondisi, kalau lead programmer tidak tersedia, pembahasan bisa langsung dilakukan dengan developer yang akan menangani <i>project</i> .
Ragil (Pewawancara)	Setelah itu bagaimana tugas dibagikan ke developer?
Sita (Narasumber)	Setelah kebutuhan mulai diterjemahkan menjadi tugas, saya biasanya menyesuaikan dengan kapasitas atau kemampuan masing-masing developer. Kalau ada tugas yang besar, tugas tersebut bisa dibagi atau di-sharing ke developer lain. Jadi



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

	pembagian pekerjaan tetap melihat kebutuhan <i>project</i> dan ketersediaan tim.
Ragil (Pewawancara)	Tools apa saja yang digunakan untuk <i>monitoring</i> progres dan penyusunan tugas?
Sita (Narasumber)	Saat ini kami menggunakan beberapa tools sekaligus. Ada <i>project</i> yang memakai <i>spreadsheet</i> atau dokumen, ada yang memakai <i>Trello</i> , dan komunikasi juga banyak dilakukan melalui chat atau meeting. Selain itu, saya juga perlu melihat aktivitas di GitHub seperti <i>commit</i> atau perubahan lain di <i>repository</i> . Karena belum ada satu platform yang benar-benar dipakai secara konsisten, data masih tersebar dan saya masih harus mengecek beberapa sumber untuk memastikan status aktual suatu pekerjaan.
Ragil (Pewawancara)	Bagaimana penentuan urutan dan ketergantungan antar tugas saat ini?
Sita (Narasumber)	Kami belum memiliki metode formal yang konsisten untuk itu. Dalam praktiknya, urutan pengerjaan masih banyak bergantung pada pengalaman dan koordinasi situasional. Kadang ada tugas yang awalnya terlihat kecil, tetapi ternyata memiliki dampak besar terhadap pekerjaan lain, dan hal itu baru diketahui ketika pengerjaan sudah berjalan.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Ragil (Pewawancara)	Kendala terbesar apa yang dirasakan dalam proses <i>monitoring project</i> saat ini?
Sita (Narasumber)	Kendala utamanya adalah data <i>project</i> belum terintegrasi dan belum <i>real-time</i> . Saya masih harus <i>cross-check</i> manual ke beberapa media untuk memastikan status aktual tugas atau fitur. Saya juga belum punya <i>dashboard</i> yang benar-benar bisa menjadi sumber utama untuk melihat progress, hubungan antar tugas, dan kondisi <i>project</i> secara menyeluruh.
Ragil (Pewawancara)	Fitur seperti apa yang dibutuhkan ke depannya?
Sita (Narasumber)	Saya membutuhkan sistem yang bisa menampilkan tugas dalam satu <i>project</i> , statusnya, progres aktual, keterkaitan antar tugas, serta integrasi dengan <i>repository</i> . Selain itu, saya juga membutuhkan mekanisme yang dapat menggambarkan urutan dan ketergantungan tugas secara objektif agar tugas yang bersifat kritis terhadap penyelesaian <i>phase</i> proyek dapat diidentifikasi, tetapi tetap bisa ditinjau kembali secara manual oleh <i>Project Manager</i> .
Ragil (Pewawancara)	Terima kasih banyak Kak Sita atas waktunya.
Sita (Narasumber)	Sama-sama Mas Ragil, sukses ya skripsinya.

Lampiran 2. Transkrip Wawancara dengan *Lead Developer*

Narasumber : Cahya Nugroho
Jabatan : Lead Developer
Instansi : PT Lokasolusi Teknologi Indonesia
Tanggal : 08 May 2026

Pewawancara / Narasumber	Keterangan
Ragil (Pewawancara)	Halo Kak Cahya, perkenalkan nama saya Ragil Irawan Putra, mahasiswa tingkat akhir di Politeknik Negeri Jakarta. Terima kasih sudah bersedia meluangkan waktu untuk wawancara ini. Selamat sore.
Cahya (Narasumber)	Halo Mas Ragil, selamat sore. Silakan.
Ragil (Pewawancara)	Boleh dijelaskan peran dan tanggung jawab Kak Cahya sebagai <i>Lead Developer</i> di PT Lokasolusi?
Cahya (Narasumber)	Iya, jadi peran saya sebagai <i>Lead Developer</i> di PT Lokasolusi itu mencakup beberapa hal. Yang utama adalah menerjemahkan kebutuhan atau <i>requirement</i> yang sudah dibahas bersama <i>Project Manager</i> menjadi <i>technical tugas</i> yang bisa dikerjakan oleh tim developer. Saya juga bertugas melakukan <i>review code</i> , memantau kualitas pekerjaan teknis, dan memastikan alur pengerjaan di tim berjalan dengan baik. Selain itu, saya juga menjadi penghubung antara <i>Project Manager</i> dan developer kalau

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

	ada hal-hal teknis yang perlu dikomunikasikan.
Ragil (Pewawancara)	Bagaimana cara Kak Cahya memantau progres pengerjaan tugas oleh developer saat ini?
Cahya (Narasumber)	Kalau untuk sekarang, saya biasanya memantau lewat GitHub. Saya lihat <i>pull request</i> yang masuk, saya <i>review changes</i> -nya di sana. Tapi untuk informasi yang lebih umum, kayak tugas mana yang sudah selesai atau masih jalan, biasanya saya dapat dari WhatsApp group tim atau chat langsung sama developernya. Jadi masih tersebar ya, belum ada satu tempat yang bisa saya buka dan langsung tahu kondisi semua tugas sekaligus.
Ragil (Pewawancara)	Apakah hubungan ketergantungan atau <i>dependency</i> antar tugas sudah terdokumentasi dengan baik?
Cahya (Narasumber)	Belum, itu salah satu kendala yang sering saya rasakan. <i>Dependency</i> antar tugas biasanya hanya ada di kepala saya atau dibahas secara lisan saat <i>planning</i> . Tidak ada dokumentasi yang terstruktur soal tugas mana yang harus selesai dulu sebelum tugas lain bisa dimulai. Jadi kadang developer ada yang mulai ngerjain tugas yang sebetulnya masih nunggu tugas lain



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

	selesai, dan itu baru ketahuan pas sudah agak jauh.
Ragil (Pewawancara)	Kendala apa yang paling sering dihadapi dalam koordinasi pekerjaan teknis dengan tim developer?
Cahya (Narasumber)	Koordinasinya masih banyak yang manual. Saya harus aktif tanya ke masing-masing developer untuk tahu progres aktualnya. Kadang juga ada situasi di mana saya baru sadar bahwa suatu tugas belum bisa dikerjakan karena tugas lain belum selesai, tapi developer sudah terlanjur menunggu atau mengerjakan sesuatu yang tidak perlu duluan. Belum ada mekanisme yang jelas untuk melihat tugas mana yang benar-benar siap dikerjakan dan mana yang masih blocked.
Ragil (Pewawancara)	Fitur seperti apa yang menurut Kak Cahya dibutuhkan untuk mendukung pekerjaan sebagai <i>Lead Developer</i> ?
Cahya (Narasumber)	Yang paling saya butuhkan adalah sistem yang bisa menampilkan <i>dependency</i> antar tugas secara jelas, jadi saya bisa tahu tugas mana yang bisa dikerjakan duluan dan mana yang masih harus menunggu. Kalau bisa terintegrasi juga dengan GitHub, misalnya bisa langsung terhubung dengan aktivitas <i>commit</i> atau <i>pull request</i> , itu akan sangat membantu. Selain itu, kalau ada <i>dashboard</i> yang



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

	bisa menampilkan progres keseluruhan tugas dalam satu <i>project</i> tanpa harus tanya satu per satu ke developer, itu ideal sekali buat saya.
Ragil (Pewawancara)	Terima kasih banyak Kak Cahya atas waktunya.
Cahya (Narasumber)	Sama-sama Mas Ragil, semoga skripsinya lancar.

Lampiran 3. Transkrip Wawancara dengan Software Engineer

Narasumber : Gema Hafiz Maulidy
 Jabatan : Software Engineer
 Instansi : PT Lokasolusi Teknologi Indonesia
 Tanggal : 6 May 2026

Pewawancara / Narasumber	Keterangan
Ragil (Pewawancara)	Halo Mas Gema, perkenalkan nama saya Ragil Irawan Putra, mahasiswa tingkat akhir di Politeknik Negeri Jakarta. Terima kasih sudah bersedia meluangkan waktu untuk wawancara ini. Selamat sore.
Gema (Narasumber)	Halo Mas Ragil, selamat sore. Silakan.
Ragil (Pewawancara)	Boleh dijelaskan peran Mas Gema sebagai <i>Software Engineer</i> di PT Lokasolusi?
Gema (Narasumber)	Iya, saya berperan sebagai <i>Software Engineer</i> di PT Lokasolusi. Pekerjaan sehari-hari saya adalah mengerjakan tugas teknis yang sudah dibagikan oleh



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

	<i>Lead Developer</i> atau <i>Project Manager</i> , mengembangkan fitur atau modul sesuai <i>requirement</i> yang sudah ditentukan, dan melaporkan progres pekerjaan secara berkala. Saya juga aktif di GitHub untuk <i>commit</i> pekerjaan dan membuat <i>pull request</i> kalau ada fitur yang sudah selesai dan siap <i>direview</i> .
Ragil (Pewawancara)	Bagaimana cara Mas Gema biasanya menerima dan memahami tugas yang harus dikerjakan?
Gema (Narasumber)	Biasanya saya menerima informasi tugas dari <i>Lead Developer</i> , bisa lewat chat di WhatsApp group tim atau dibahas langsung di meeting. Kadang juga ada yang dicatat di <i>spreadsheet</i> atau dokumen <i>project</i> . Yang jadi tantangan adalah kadang deskripsi tugas tidak terlalu detail, jadi saya harus tanya <i>lagi</i> ke Lead atau ke PM untuk memastikan apa yang harus dikerjakan dan hasilnya seperti apa.
Ragil (Pewawancara)	Apakah Mas Gema mengetahui tugas mana yang harus diselesaikan lebih dahulu sebelum tugas lain bisa dimulai?
Gema (Narasumber)	Tidak selalu. Informasi soal ketergantungan antar tugas biasanya tidak tertulis di mana-mana. Saya tahu ada tugas yang bergantung satu sama lain dari hasil diskusi atau dari



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

	pengalaman. Tapi kadang saya baru sadar bahwa tugas yang sedang saya kerjakan sebetulnya masih menunggu tugas lain selesai, dan itu baru tersampaikan setelah saya sudah mulai duluan.
Ragil (Pewawancara)	Kendala apa yang paling sering Mas Gema rasakan dalam pengerjaan tugas sehari-hari?
Gema (Narasumber)	Kendala yang paling sering saya alami itu ada dua. Pertama, saya tidak selalu tahu apakah tugas yang akan saya kerjakan memang sudah bisa dikerjakan atau masih menunggu sesuatu. Kedua, untuk melaporkan progres, saya masih harus mengirimkan update manual lewat chat atau saat meeting. Tidak ada satu tempat di mana saya bisa update progres dan langsung terlihat oleh Lead atau PM secara <i>real-time</i> .
Ragil (Pewawancara)	Fitur seperti apa yang menurut Mas Gema dibutuhkan untuk membantu pekerjaan sebagai <i>Software Engineer</i> ?
Gema (Narasumber)	Saya butuh sistem yang bisa menampilkan tugas yang harus saya kerjakan secara jelas, lengkap dengan detail deskripsinya dan apakah tugas tersebut sudah siap dikerjakan atau masih perlu menunggu. Kalau bisa juga terhubung dengan GitHub, jadi aktivitas <i>commit</i> saya bisa langsung

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

	terpantau tanpa harus report manual. Yang paling penting buat saya adalah kejelasan tugas apa yang harus dikerjakan, dalam urutan seperti apa, dan apa yang masih blocked sehingga saya tidak salah prioritas.
Ragil (Pewawancara)	Terima kasih banyak Mas Gema atas waktunya.
Gema (Narasumber)	Sama-sama Mas Ragil, semoga skripsinya sukses.

Lampiran 4 Dokumentasi Observasi di PT Lokasolusi Teknologi Indonesia

	Catatan lapangan: dokumentasi kegiatan observasi dan wawancara bersama tim PT Lokasolusi Teknologi Indonesia. Observasi dilakukan untuk memahami proses pengelolaan proyek perangkat lunak, mulai dari penugasan proyek oleh <i>Chief Executive Officer</i> kepada <i>Project Manager</i>, perencanaan dan koordinasi bersama <i>Lead Developer</i> dan <i>Software Developer</i>, hingga pemantauan progres dan kesiapan tugas. Hasil observasi menegaskan bahwa pemantauan jalur kritis proyek selama ini belum terpusat sehingga mendorong kebutuhan sistem <i>monitoring</i> proyek yang dikembangkan pada penelitian ini.
--	---

Lampiran 5 Product Backlog Pengembangan Sistem Prolytics

Rincian product backlog pengembangan sistem Prolytics yang menjadi dasar contoh perhitungan CPM pada subbab 4.2.6. Estimasi dinyatakan dalam satuan hari kerja; total estimasi tiap work package sama dengan durasi kegiatan pada Tabel 11.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No	Kode	Work Package	Tugas	Prioritas	Estimasi	Dependency
1	A	Perancangan dan setup basis data	Perancangan skema basis data (ERD, tabel utama)	P0	2	-
2	A		Setup PostgreSQL, Bun ORM, dan migrasi awal	P0	2	
3	B	Autentikasi dan RBAC	Implementasi login dan JWT	P0	2	A
4	B		Middleware otorisasi berbasis role (RBAC)	P0	2	
5	B		Logout dan manajemen sesi/token	P0	1	
6	C	Manajemen pengguna dan profil	CRUD pengguna oleh Project Manager	P0	2	B
7	C		Melihat dan memperbarui profil	P1	1	
8	C		Mengunggah avatar melalui MinIO	P1	1	
9	D	Manajemen project, phase, dan tugas	CRUD project dan pengelolaan anggota	P0	2	B
10	D		Aktivasi project	P0	1	
11	D		CRUD phase dan anggota phase	P0	2	
12	D		CRUD technical tugas dan pembaruan status	P0	3	
13	E	Pengaturan dependency dan validasi graf	Pengaturan dependency antar-tugas	P0	2	D
14	E		Validasi Directed Acyclic Graph (deteksi circular)	P0	2	
15	E		Perhitungan readiness dan status blocked	P1	1	
16	F	Modul perhitungan CPM	Perhitungan maju (ES dan EF)	P0	2	E
17	F		Perhitungan mundur (LS dan LF)	P0	2	
18	F		Perhitungan slack dan penentuan jalur kritis	P0	2	
19	F		Snapshot dan riwayat CPM	P1	2	



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

No	Kode	Work Package	Tugas	Prioritas	Estimasi	Dependency
20	F		Perhitungan ulang (recalculation)	P1	1	
21	G	Approval workflow dan change request	Pengajuan dan antrean persetujuan tugas	P0	2	D
22	G		Menyetujui, menolak, dan meminta revisi tugas	P0	2	
23	G		Change request pada tugas aktif	P1	2	
24	H	Integrasi GitHub	Koneksi repository dan activity feed	P1	3	D
25	H		Menautkan dan melepas aktivitas ke tugas	P1	2	
26	I	Modul AI Brief	Integrasi LLM untuk penyusunan brief	P1	2	D
27	I		Mekanisme fallback saat AI tidak tersedia	P2	1	
28	J	Antarmuka frontend	Dashboard sesuai role	P0	2	C,F,G
29	J		Halaman tugas, detail, dan Tugas Saya	P0	2	
30	J		Visualisasi CPM dan jalur kritis	P0	2	
31	J		Komponen tabel, badge, paginasi, pencarian, filter	P1	2	
32	K	Integrasi, dokumentasi API, dan pengujian	Dokumentasi API berbasis Swagger	P1	1	H,I,J
33	K		Integrasi antar-modul	P0	2	
34	K		Pengujian (black box dan UAT)	P0	2	

Lampiran 6 Transkrip Wawancara Lanjutan dengan Project Manager

Pewawancara / Narasumber	Keterangan
Ragil (Pewawancara)	Selamat siang, Kak Sita. Terima kasih atas waktunya. Saya ingin menanyakan beberapa hal lanjutan sebagai validasi



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Pewawancara / Narasumber	Keterangan
	terkait pengelolaan proyek di PT Lokasolusi Teknologi Indonesia.
Sita (Narasumber)	Siang, Mas Ragil. Silakan, dengan senang hati.
Ragil (Pewawancara)	Untuk konteks pekerjaan atau proyek yang dikerjakan tim, biasanya proyek tersebut diterima dari mana dan oleh siapa?
Sita (Narasumber)	Proyek umumnya berasal dari klien dan diterima terlebih dahulu oleh <i>Chief Executive Officer</i> . Setelah itu, <i>Chief Executive Officer</i> menugaskan proyek tersebut kepada saya selaku <i>Project Manager</i> . Dari situ saya yang merencanakan proyek, menyusunnya menjadi <i>phase</i> dan tugas, lalu membaginya kepada <i>Lead Developer</i> dan <i>Software Developer</i> sesuai kebutuhan.
Ragil (Pewawancara)	Bagaimana pengelolaan beban proyek untuk <i>Software Developer</i> maupun <i>Lead Developer</i> ? Apakah ada batasan jumlah proyek yang ditangani?
Sita (Narasumber)	Untuk <i>Software Developer</i> , saat ini kami batasi maksimal tiga proyek dalam satu waktu agar fokus dan kualitas pekerjaannya tetap terjaga. Sementara untuk <i>Lead Developer</i> masih bersifat tentatif dan belum ada batasan maksimal beban proyek,



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Pewawancara / Narasumber	Keterangan
	karena perannya lebih banyak pada koordinasi teknis lintas proyek.
Ragil (Pewawancara)	Baik, terima kasih banyak atas penjelasan dan validasinya, Kak Sita.
Sita (Narasumber)	Sama-sama, Mas Ragil. Semoga sistemnya bermanfaat.

