

BAB II

TINJAUAN PUSTAKA

2.1 Teori Terkait

Pengembangan sistem informasi manajemen proyek konstruksi berbasis *web* sangat dipengaruhi oleh berbagai teori dan temuan dari penelitian sebelumnya.

Penelitian yang dilakukan oleh Choi & Ha (2021) dalam jurnal *Journal of Asian Architecture and Building Engineering* yang berjudul "*Validation of Project Management Information Systems for Industrial Construction Projects*". Sebuah penelitian eksperimental ini membuktikan bahwa sistem informasi manajemen proyek berbasis *web* mampu meningkatkan akurasi pelaporan progres konstruksi hingga 35% dan mengurangi waktu pembuatan laporan dari 8 jam menjadi 2 jam per minggu melalui otomatisasi data [5].

Penelitian yang dilakukan oleh Adnan et al. (2020) dalam jurnal *International Journal of Advanced Computer Science and Applications* yang berjudul "*Web-Based Project Monitoring System Using Laravel Framework*". Penelitian ini mengembangkan sistem manajemen proyek menggunakan Laravel dan menemukan bahwa *framework* tersebut mampu mengurangi waktu pengembangan modul *CRUD* hingga 50% berkat fitur *Eloquent ORM* dan migrasi *database* terstruktur [6].

Penelitian yang dilakukan oleh Abdullah Muhammad Farouk Hakim dan Nurudin Santoso (2022) dalam jurnal *Jurnal Pengembangan Teknologi*

Informasi dan Ilmu Komputer dengan judul "Pengembangan Aplikasi Manajemen Proyek Perusahaan Jasa Konstruksi Kecil dan Menengah berbasis *web* (Studi Kasus: AMF-Haq Engineering and Consultant)". Penelitian ini menunjukkan bahwa digitalisasi manajemen proyek menggunakan aplikasi berbasis *web* dapat mengurangi risiko kesalahan pencatatan informasi alat dan material, mempercepat pembagian tugas pekerja, serta memperkecil kemungkinan kehilangan berkas proyek. Dengan aplikasi ini, pengelolaan proyek pada perusahaan konstruksi kecil menjadi lebih terorganisir, efisien, dan mampu mendorong peningkatan jumlah proyek yang dapat ditangani secara bersamaan [7].

Penelitian yang dilakukan oleh Kineber et al. (2023) dalam jurnal *International Journal of Construction Management* yang berjudul "*BIM-Based Risk Management in Construction Projects: A Case Study in High-Rise Buildings*". Penelitian ini menunjukkan bahwa integrasi sistem informasi dengan BIM mengurangi risiko keterlambatan proyek sebesar 22% melalui deteksi dini konflik desain dan alokasi sumber daya yang terukur [8].

Penelitian yang dilakukan oleh Mandiyo Priyo dan Noor Adi Wibowo (2008) dalam Jurnal Ilmiah Semesta Teknika berjudul "Konsep Earned Value dalam Aplikasi Pengelolaan Proyek Konstruksi". Studi ini membahas penggunaan konsep earned value untuk mengintegrasikan monitoring biaya dan waktu proyek. Hasil penelitian menunjukkan bahwa dengan pendekatan earned value, penyimpangan biaya dan jadwal proyek dapat terdeteksi lebih awal, memungkinkan perbaikan strategi eksekusi proyek untuk menghindari

pembengkakan biaya dan keterlambatan penyelesaian proyek konstruksi secara signifikan [9].

Penelitian yang dilakukan oleh Ucok Jimmy dan Agus Munandar (2025) dalam jurnal Jurnal ANC berjudul "Sistem Informasi Manajemen Pengelolaan Persediaan Material Proyek Konstruksi Berbasis Web dan Mobile (Studi Kasus PT XYZ)". Penelitian ini menemukan bahwa penggunaan sistem informasi berbasis *web* dan *mobile* untuk pengelolaan persediaan material dapat meningkatkan efisiensi waktu, menjaga kualitas material sesuai spesifikasi, serta mengoptimalkan pengendalian biaya proyek dengan meminimalkan risiko kekurangan dan kelebihan material pada proyek konstruksi [10].

2.2 Landasan Teori

2.2.1 Teori Pengambilan Keputusan

Menurut Simon, pengambilan keputusan adalah proses memilih suatu tindakan dari beberapa alternatif berdasarkan informasi yang tersedia. Dalam konteks manajemen proyek, sistem informasi manajemen menyediakan data yang akurat dan *real-time* untuk mendukung manajer dalam membuat keputusan yang cepat dan tepat [11].

2.2.2 Metode *Prototyping* dalam Pengembangan Perangkat Lunak

Metode *Prototyping* adalah pendekatan dalam pengembangan perangkat lunak yang berfokus pada pembuatan model awal

(prototipe) dari sistem yang akan dikembangkan. Prototipe ini berfungsi sebagai representasi fungsional dari sistem akhir, yang memungkinkan pengguna memberikan umpan balik langsung. Berdasarkan umpan balik tersebut, prototipe diperbaiki dan dikembangkan lebih lanjut, hingga sistem yang lebih lengkap dapat dibangun. Metode ini sangat efektif digunakan dalam pengembangan sistem yang memerlukan interaksi langsung dengan pengguna dan pengujian fungsionalitas yang cepat.

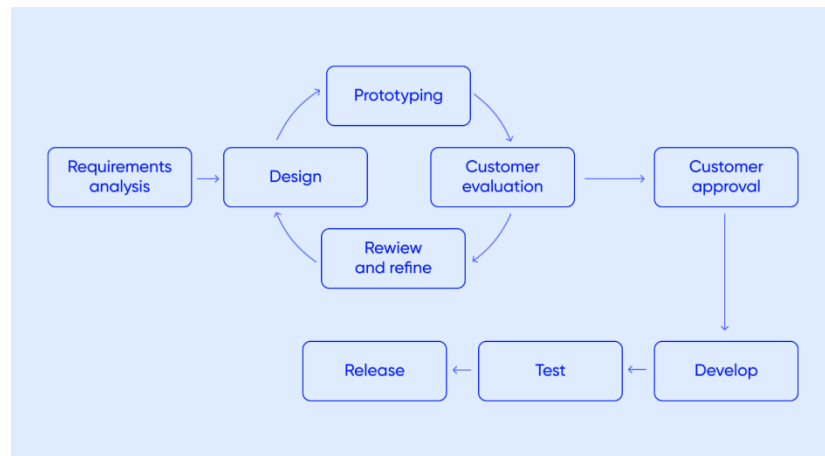
Tahapan dalam metode *Prototyping* terdiri dari:

1. Pengumpulan Kebutuhan Awal: Pada tahap pertama, pengembang dan klien (manajer proyek) mengidentifikasi kebutuhan dasar sistem yang diperlukan. Dalam proyek konstruksi, ini bisa mencakup kebutuhan fitur untuk manajemen jadwal, pengelolaan anggaran, dan pemantauan progres proyek.
2. Pengumpulan Kebutuhan Awal: Pada tahap pertama, pengembang dan klien (manajer proyek) mengidentifikasi kebutuhan dasar sistem yang diperlukan. Dalam proyek konstruksi, ini bisa mencakup kebutuhan fitur untuk manajemen jadwal, pengelolaan anggaran, dan pemantauan progres proyek.
3. Perancangan Prototipe: Berdasarkan kebutuhan awal, tim pengembang membuat prototipe sederhana dari aplikasi yang akan digunakan untuk mengelola proyek konstruksi. Prototipe ini tidak harus memiliki semua fitur lengkap, tetapi cukup untuk

memberikan gambaran umum tentang bagaimana sistem akan bekerja, dengan fokus pada fungsi-fungsi dasar yang esensial.

4. Uji Coba dan Evaluasi: Setelah prototipe dikembangkan, pengguna (misalnya manajer proyek atau tim lapangan) mulai mengujinya. Pada tahap ini, pengguna memberikan umpan balik mengenai fungsionalitas, kegunaan, dan fitur yang dibutuhkan atau kurang sesuai.
5. Perbaikan dan Pengembangan Lebih Lanjut: Berdasarkan umpan balik dari pengguna, prototipe diperbaiki dan ditingkatkan. Fitur-fitur baru dapat ditambahkan, dan yang kurang efektif diperbaiki.
6. Pengujian dan Evaluasi Kembali: Setelah perbaikan dilakukan, maka *prototype* diuji kembali untuk memastikan aplikasi semakin mendekati kebutuhan pengguna dan siap untuk diterapkan dalam proyek konstruksi yang sesungguhnya.

Metode *Prototyping* sangat berguna dalam pengembangan sistem informasi manajemen proyek, karena memungkinkan perubahan dan penyesuaian yang cepat sesuai dengan umpan balik pengguna. Pendekatan ini juga sangat membantu dalam sistem yang memerlukan adaptasi seiring waktu, terutama dalam proyek-proyek konstruksi yang sering kali berubah kebutuhan dan kondisi di lapangan [12].

Gambar 2.1 Metode *Prototyping*

2.2.3 Laravel

Laravel adalah *framework PHP* berbasis konsep *Model-View-Controller (MVC)* yang dirancang untuk memudahkan pengembangan aplikasi *web*. Laravel menawarkan sintaks yang ekspresif dan elegan, serta menyediakan berbagai fitur seperti *routing*, *session management*, *authentication*, *ORM (Eloquent)*, dan sistem *migrasi database*. Dengan Laravel, pengembang dapat membangun aplikasi yang lebih terstruktur, efisien, dan aman [13].



Gambar 2.2 Logo Laravel

2.2.4 MySQL

MySQL adalah sistem manajemen basis data relasional (RDBMS) yang menggunakan *Structured Query Language* (SQL) sebagai bahasa utama untuk mengelola dan memanipulasi data. MySQL dikenal karena keandalannya, kecepatan pemrosesan, dan kemampuannya menangani volume data yang besar. Dalam pengembangan aplikasi *web*, MySQL sering dipilih sebagai basis data karena integrasinya yang mudah dengan berbagai platform, termasuk PHP dan Laravel [14].



Gambar 2.3 Logo MySQL

2.2.5 Filament

Filament adalah pustaka open-source berbasis Laravel yang memudahkan pembuatan panel admin, manajemen konten, dan *CRUD* (*Create, Read, Update, Delete*) *interface* yang modern. Filament dirancang untuk bekerja cepat dengan Laravel dan menawarkan komponen antarmuka yang interaktif serta terintegrasi langsung dengan *Eloquent ORM*. Dengan Filament, pengembang dapat

mempercepat pembuatan *dashboard* administrasi tanpa harus membangun dari nol [15].



Gambar 2.4 Logo Filament PHP

2.2.6 Tailwind CSS

Tailwind CSS adalah *framework* CSS *utility-first* yang memberikan kelas-kelas siap pakai untuk membangun antarmuka pengguna secara cepat dan konsisten. Dengan pendekatan *utility-first*, pengembang dapat mengontrol langsung tampilan elemen tanpa perlu membuat banyak *file* CSS khusus. Tailwind CSS mendukung desain responsif secara *default* dan sangat fleksibel untuk berbagai kebutuhan desain *front-end* [16].



Gambar 2.5 Logo Tailwind CSS

2.2.7 Unified Modeling Language (UML)

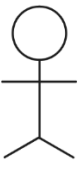
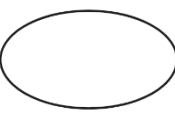
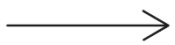
UML adalah sebuah bahasa visual standar yang digunakan untuk memodelkan, mendokumentasikan, dan merancang sistem perangkat lunak. UML memungkinkan pengembang untuk merepresentasikan berbagai aspek dari sistem secara grafis, sehingga memudahkan dalam memahami struktur, perilaku, serta interaksi antar komponen sistem yang akan dibangun.

Dalam pengembangan sistem informasi manajemen proyek konstruksi berbasis *web* ini, UML digunakan sebagai alat bantu dalam proses perancangan sistem. Penggunaan UML membantu memvisualisasikan proses bisnis, alur interaksi pengguna, serta hubungan antar entitas dalam sistem. Adapun jenis diagram UML yang digunakan dalam proyek ini meliputi:

1. Diagram *Use Case*

Diagram ini digunakan untuk menggambarkan fungsionalitas utama sistem dari sudut pandang pengguna (aktor). *Use case* menunjukkan interaksi antara aktor seperti admin proyek, kontraktor, subkontraktor, serta fungsi-fungsi yang dapat mereka akses, seperti manajemen progres, penjadwalan, dan pengelolaan dokumen proyek. Simbol-simbol yang digunakan dalam diagram ini, termasuk aktor, sistem, dan hubungan antar elemen, dijelaskan secara detail pada Tabel 2.1.

Tabel 2.1 Simbol Diagram Use Case

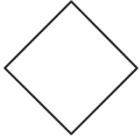
No	Simbol	Keterangan
1		<i>Actor</i> : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan <i>use case</i> .
2		<i>Use Case</i> : Abstraksi dan interaksi antara sistem dan <i>Actor</i> .
3		<i>Association</i> : Abstraksi dari penghubung antara <i>Actor</i> dengan <i>Use Case</i> .
4		Menunjukkan bahwa suatu <i>Use Case</i> seluruhnya merupakan fungsionalitas dari <i>Use Case</i> lainnya.
5		Menunjukkan bahwa suatu <i>Use Case</i> merupakan tambahan fungsionalitas dari <i>Use Case</i> lainnya jika suatu kondisi terpenuhi.

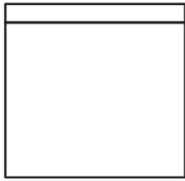
2. Diagram Activity

Diagram Activity digunakan untuk memodelkan alur kerja (*workflow*) atau proses bisnis dalam suatu sistem. Diagram ini membantu menggambarkan langkah-langkah yang dilakukan oleh pengguna (*user*) atau sistem dalam melaksanakan suatu

fungsi tertentu, seperti pembuatan laporan progres, pengunggahan dokumen proyek, atau proses lainnya yang relevan dengan sistem. Dengan memvisualisasikan langkah-langkah tersebut, diagram aktivitas memudahkan pemahaman terhadap proses yang terjadi, mendukung analisis kebutuhan sistem, dan menjadi acuan dalam implementasi pengembangan perangkat lunak. Informasi lengkap mengenai simbol-simbol aktivitas dalam diagram ini disajikan pada Tabel 2.2.

Tabel 2.2 Simbol Diagram Activity

No	Simbol	Keterangan
1		<i>Start Point</i> : Sebuah diagram yang memiliki sebuah status awal.
2		<i>Activity</i> : Merupakan aktivitas yang dilakukan atau sedang terjadi dalam sistem.
3		<i>Decision</i> : Merupakan suatu titik atau <i>point</i> yang mengindikasikan suatu kondisi yang di mana ada kemungkinan dalam perbedaan transisi.
4		<i>Fork</i> : Digunakan untuk menunjukan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.

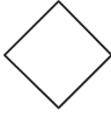
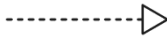
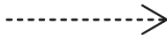
No	Simbol	Keterangan
5		<i>Swimlanes</i> : Memecah aktivitas diagram menjadi kolom dan baris untuk membagi tanggung jawab objek – objek yang melakukan aktivitas.
6		<i>End State</i> : Notasi akhir digunakan untuk menandakan proses tersebut berakhir.

3. Diagram *Class*

Struktur data dalam sistem direpresentasikan melalui diagram ini, yang memuat kelas-kelas utama, atribut, serta relasi antar kelas secara jelas dan sistematis. Diagram ini juga menggambarkan keterkaitan antar elemen penting seperti Proyek, Pengguna, Dokumen, dan Jadwal, sehingga memudahkan pemahaman tentang bagaimana data diorganisasi dan berinteraksi dalam sistem. Deskripsi lebih rinci mengenai kelas-kelas beserta atributnya dapat ditemukan pada Tabel 2.3.

Tabel 2.3 Simbol Diagram Class

No	Gambar	Nama	Keterangan
1		<i>Generation</i>	Hubungan di mana objek anak (<i>descendant</i>) mewarisi perilaku dan struktur data dari objek induk (<i>ancestor</i>).

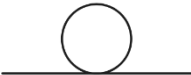
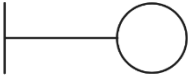
No	Gambar	Nama	Keterangan
2		<i>Nary Association</i>	Relasi yang melibatkan lebih dari dua objek sekaligus dalam satu asosiasi.
3		<i>Class</i>	Kumpulan objek yang memiliki atribut dan operasi yang sama.
4		<i>Collaboration</i>	Deskripsi urutan aksi sistem yang menghasilkan <i>output</i> terukur bagi seorang aktor.
5		<i>Realization</i>	Implementasi nyata dari operasi yang dilakukan oleh suatu objek.
6		<i>Dependency</i>	Hubungan di mana perubahan pada elemen independen memengaruhi elemen yang bergantung padanya.
7		<i>Association</i>	Relasi yang menghubungkan antara satu objek dengan objek lainnya.

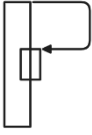
4. Diagram *Sequence*

Diagram *Sequence* digunakan untuk memodelkan urutan pesan atau panggilan yang terjadi antar objek dalam sebuah

sistem secara kronologis. Diagram ini sangat efektif dalam menggambarkan interaksi dinamis antara objek-objek yang terlibat dalam suatu proses atau fungsi tertentu, sehingga memudahkan pemahaman alur komunikasi dan koordinasi antar komponen sistem. Dengan menonjolkan urutan waktu dari pesan yang dikirim dan diterima, diagram ini membantu dalam menganalisis bagaimana berbagai objek berkolaborasi untuk mencapai tujuan bisnis atau teknis yang diinginkan. Untuk keterangan lengkap mengenai simbol dan notasi yang digunakan dalam Diagram *Sequence*, dapat dilihat pada Tabel 2.4.

Tabel 2.4 Simbol Diagram *Sequence*

No	Gambar	Nama	Keterangan
1		<i>Entity Class</i>	Mewakili data atau entitas sistem, seperti pengguna atau proyek.
2		<i>Boundary Class</i>	Antarmuka antara pengguna dan sistem, menangani <i>input/output</i> .
3		<i>Control Class</i>	Mengatur logika proses, menjembatani <i>boundary</i> dan <i>entity</i> .

No	Gambar	Nama	Keterangan
4		<i>Recursive</i>	Pemanggilan fungsi oleh objek itu sendiri (<i>self-call</i>).
5		<i>Activation</i>	Komponen ini menggambarkan waktu yang diperlukan oleh suatu objek untuk menyelesaikan tugasnya
6		<i>Life Line</i>	Komponen yang biasanya digambarkan dengan simbol garis putus – putus.