

BAB II

LANDASAN TEORI

2.1 Sistem Informasi

Penggunaan sistem informasi dalam pengolahan data dan informasi dalam sebuah organisasi maupun perusahaan akan mempermudah untuk pengolahan data, dan memperoleh informasi yang akurat serta mudah di akses kapanpun dan dimanapun[2].

2.2 Monitoring

Monitoring merupakan proses rutin dalam pemantauan serta pengumpulan data kemajuan suatu objek, baik dari segi proses, kualitas, dan hasil akhir. Langkah paling awal dalam monitoring pelaksanaan suatu proyek, yaitu dengan melakukan monitoring rutin secara harian, mingguan, dan bulanan. Dengan cara ini dapat dihindari kemunduran atau keterlambatan dalam penyelesaian suatu pekerjaan[2].

2.3 Manajemen Proyek

Manajemen proyek adalah suatu proses pengolahan proyek yang meliputi perencanaan, pengorganisasian dan pengaturan tugas-tugas sumber daya untuk mewujudkan tujuan yang ingin dicapai, dengan mempertimbangkan faktor-faktor waktu dan biaya[3].

2.4 Website

Website adalah kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, atau gabungan dari semuanya itu baik yang bersifat statis maupun dinamis di mana masing-masing dihubungkan dengan jaringan-jaringan halaman[4].

2.5 Rekayasa Perangkat Lunak

Rekayasa Perangkat Lunak (RPL) adalah suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu *communication*, *requiremets capturing* (Analisa kebutuhan pengguna), *specification* (menentukan spesifikasi dari kebutuhan pengguna), desain, *coding*, *testing*, sampai *maintenance* (pemeliharaan sistem) setelah digunakan[5].

2.6 PHP

PHP (*Hypertext Preprocessor*) adalah bahasa pemrograman server-side yang dirancang khusus untuk pengembangan web. *PHP* dapat ditanamkan ke dalam kode *HTML* dan dijalankan pada *server* web, menghasilkan konten dinamis yang kemudian dikirim ke *browser* pengguna. *PHP* umumnya digunakan untuk membuat halaman *web* yang dapat berinteraksi dengan basis data, mengelola formulir, menghasilkan konten dinamis, dan melakukan berbagai tugas pemrograman *server-side* lainnya[6].

2.7 Object Oriented Programming

Object Oriented Programming (OOP) adalah suatu cara baru dalam berpikir serta berlogika dalam menghadapi masalah-masalah yang akan dicoba atasi dengan bantuan komputer, dimana setiap objek adalah entitas tunggal yang memiliki kombinasi struktur data dan fungsi tertentu[9]. Sehingga dengan bantuan komputer, pendekatan baru dalam berpikir dan berlogika dalam menangani masalah-masalah dihadirkan, di mana setiap entitas tunggal, baik itu manusia, tempat, benda, atau kejadian, direpresentasikan sebagai objek dengan kombinasi struktur data dan fungsi yang spesifik. Hal ini tercermin dalam berbagai aktivitas seperti pembayaran uang pendidikan, registrasi biodata siswa, membaca buku, dan lain sebagainya[10].

2.8 Basis Data

Basis data atau *database* adalah sebuah system yang di buat untuk mengorganisasi, menyimpan dan menarik data dengan mudah. *Database* terdiri dari kumpulan data yang terorganisir untuk 1 atau lebih penggunaan, dalam bentuk digital[11]. *Database* memungkinkan pengguna untuk menyimpan, mengambil, dan mengelola data dengan efisien, serta memberikan dukungan bagi aplikasi perangkat lunak dalam menjalankan fungsinya. Hal ini membuat database menjadi komponen penting dalam berbagai sistem informasi, mulai dari aplikasi bisnis hingga situs *web* dan aplikasi *mobile*.

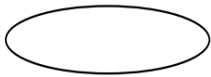

2.9 UML

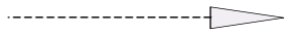
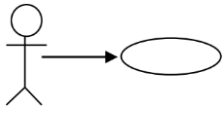
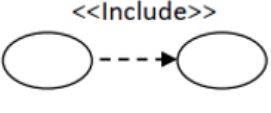
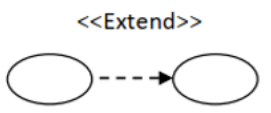
UML (Unified Modeling Language) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan *software* berbasis *OO (Object-Oriented)*. *UML* sendiri juga memberikan standar penulisan sebuah sistem *blue-print*, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem *software*[7]. Salah Satu diagram yang termasuk dalam *UML* yaitu *Use Case*.

2.9.1 Use Case Diagram

Use Case Diagram adalah deskripsi dari sebuah sistem dari perspektif pengguna. *Use Case Diagram* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai[8].

Tabel 2. 1 Simbol *Use Case Diagram*

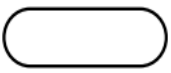

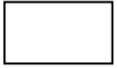



No.	Gambar Simbol	Nama	Keterangan
1.		<i>Use Case</i>	Menggambarkan bagaimana seseorang akan menggunakan atau memanfaatkan sistem.
2.		Aktor	Seseorang atau suatu yang berinteraksi dengan sistem yang sedang kita kembangkan.

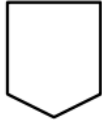
3.		Relasi	Sebagai penghubung antara actor-usecase, usecase-usecase dan lain-lain.
4.		Relasi Asosiasi	Relasi terjadi antara faktor dengan <i>usecase</i> biasanya berupa garis lurus dengan kepala panah disalah satu ujungnya.
5.		Relasi Cakupan	<i>Include Relationship</i> (relasi cakupan): Memungkinkan suatu <i>usecase</i> untuk menggunakan fungsionalitas yang disediakan oleh <i>usecase</i> yang lainnya.
6.		<i>Extand Relationship</i>	<i>Extand Relationship</i> : Memungkinkan <i>usecase</i> memiliki kemungkinan untuk memperluas fungsionalitas yang disediakan oleh <i>usecase</i> yang lainnya.

2.9.2 Flowchart

Berdasarkan definisi-definisi di atas dapat disimpulkan bahwa *flowchart* merupakan sebuah bagan yang digunakan untuk menggambarkan suatu proses dalam program dengan menggunakan simbol-simbol tertentu[12].

Tabel 2. 2 Simbol *Flowchart*





No.	Gambar Simbol	Nama	Keterangan
1.		<i>Terminator</i>	Simbol yang digunakan untuk menunjukkan awal atau akhir program.
2.		Garis Alir	Simbol yang digunakan untuk menunjukkan alur atau aliran program.
3.		Proses	Simbol yang digunakan untuk proses pengolahan data.
4.		<i>Input / Output Data</i>	Simbol yang digunakan untuk memasukkan atau mengeluarkan data.
5.		<i>Decision</i>	Simbol yang digunakan untuk memberikan pilihan.
6.		<i>On Page Connector</i>	Simbol yang digunakan untuk menghubungkan bagian-bagian

			<i>flowchart</i> dalam halaman yang sama.
7.		<i>Off Page Connector</i>	Simbol yang digunakan untuk menghubungkan bagian-bagian <i>flowchart</i> dalam halaman yang berbeda.

2.9.3 ERD (Entity Relationship Diagram)

Entity Relationship Diagram merupakan suatu diagram yang dibangun dengan menggunakan simbol-simbol yang menggambarkan hubungan antar entitas beserta relasinya yang saling terhubung dalam sebuah sistem[12].

Tabel 2. 3 Simbol *ERD*

No.	Gambar Simbol	Nama	Keterangan
1.		Entitas	Merupakan data inti yang akan disimpan.
2.		Atribut	Field atau kolom data yang butuh disimpan dalam suatu entitas.
3.		Relasi	Relasi yang menghubungkan antar entitas.
4.		Asosiasi	Penghubung antara relasi dan entitas di mana di kedua ujungnya memiliki <i>multiplicity</i> kemungkinan jumlah pemakaian.

2.9.4 Framework & Laravel

Framework adalah kumpulan intruksi-intruksi yang dikumpulkan dalam *class* dan *function-function* dengan fungsi masing-masing untuk memudahkan *developer* dalam memanggilnya tanpa harus menuliskan *syntax* program yang sama berulang-ulang serta dapat menghemat waktu[13]. *Laravel* adalah sebuah *framework PHP* yang dirilis di bawah lisensi MIT, dibangun dengan konsep *MVC (model view controller)*. *Laravel* adalah pengembangan website berbasis *MVC* yang ditulis dalam *PHP* yang dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan, dan untuk meningkatkan pengalaman bekerja dengan aplikasi dengan menyediakan sintaks yang ekspresif, jelas dan menghemat waktu[14]. *Framework* yang akan digunakan dalam pembuatan sistem ini yaitu *Framework Laravel* versi 10.

2.9.5 Leaflet

Leaflet merupakan pustaka *JavaScript* yang digunakan untuk mengembangkan peta interaktif dengan kinerja tinggi, yang diperkenalkan oleh *Vladimir Agafonkin* pada tahun 2011. *Leaflet* menawarkan dukungan untuk berbagai jenis peta dan ekosistem *plugin* yang luas, menjadikannya pilihan utama dalam implementasi aplikasi peta interaktif. Arsitektur modular

Leaflet memastikan performa yang optimal, sementara integrasi melalui *HTML* dan *API* intuitif menyederhanakan proses pengembangan.

2.9.6 E-Charts

E-Charts adalah pustaka *JavaScript* yang dikembangkan oleh *Baidu* pada tahun 2013 untuk visualisasi data interaktif. Pustaka ini menonjol dengan kemampuannya dalam menghasilkan berbagai jenis grafik seperti garis, batang, pie, dan heatmap, yang dikonfigurasi melalui objek *'option'* dan dapat dengan mudah diintegrasikan dengan *framework* seperti *React* dan *Angular*. Arsitektur modular *ECharts* mendukung performa tinggi dalam pengolahan data yang besar, serta didukung oleh komunitas pengembang yang aktif dan dokumentasi yang lengkap.

2.9.7 Metode Pengembangan

Pengembangan sistem ini, menggunakan metode *System Development Life Cycle (SDLC)* dengan model *Waterfall*. *SDLC* atau siklus hidup pengembangan sistem dalam rekayasa sistem dan rekayasa perangkat lunak adalah proses pembuatan dan perubahan sistem serta model dan metodologi yang digunakan untuk mengembangkan sistem-sistem tersebut. *SDLC* juga merupakan pola untuk mengembangkan sistem perangkat lunak yang terdiri dari tahapan perencanaan dan analisis (*requirements*) desain (*design*), implementasi (*implementation*), uji coba (*testing/verification*) dan pengelolaan (*maintenance*)[1]. Tahapan-tahapan pengembangan sistem yaitu:

1. *Requirements*

Tahap ini adalah tahap perencanaan yang melibatkan pengumpulan data terkait sistem yang akan dibangun, yang dapat dilakukan melalui wawancara atau observasi. Pengumpulan data dilakukan secara langsung di Dinas Pekerjaan Umum dan Penataan Ruang, sebagai tempat penelitian. Kegiatan ini bertujuan untuk mengumpulkan informasi yang menjadi kebutuhan dasar dari sistem yang akan dikembangkan.

2. *Design*

Fokus desain perangkat lunak terletak pada pengembangan program, termasuk struktur data, arsitektur perangkat lunak, rancangan antar muka, dan metode pengodean. Pada tahap pembuatan sistem informasi monitoring progres pekerjaan jalan dan jembatan, tahapan ini melibatkan penyusunan alur kerja yang dijelaskan dalam *flowchart*, perancangan antarmuka atau tampilan, dan pembuatan algoritma sistem.

3. *Implementation*

Pada tahap ini, sistem sistem pertama kali dikembangkan di program kecil yang disebut unit, yang terintegrasi dalam tahap selanjutnya. Setiap unit dikembangkan dan diuji untuk fungsionalitas yang disebut sebagai unit testing.

4. *Verification*

Pada tahap ini, sistem mengalami verifikasi dan pengujian untuk memastikan apakah sistem tersebut memenuhi persyaratan secara keseluruhan atau sebagian. Pengujian dapat dibagi menjadi unit testing (dilakukan pada modul kode tertentu), sistem pengujian (untuk mengamati respons sistem saat semua modul terintegrasi), dan pengujian penerimaan (dilakukan dengan atau atas nama pelanggan untuk memastikan kepuasan terhadap semua kebutuhan pelanggan).

5. *Maintenance*

Ini merupakan tahap akhir dari metode *Waterfall* di mana perangkat lunak yang telah selesai dikembangkan diimplementasikan dan menjalankan proses pemeliharaan. Pemeliharaan melibatkan perbaikan kesalahan yang mungkin tidak terdeteksi pada tahap sebelumnya.