BAB II
LANDASAN TEORI

2.1 Sistem Manajemen

Sistem adalah entitas yang terdiri dari komponen-komponen saling terhubung yang bekerja bersama untuk mencapai hasil tertentu. Manajemen adalah disiplin yang mencakup perencanaan, pengorganisasian, pengarahan, dan pengawasan sumber daya manusia untuk mencapai tujuan yang telah ditetapkan. Sistem manajemen mengintegrasikan konsep sistem dan manajemen, bertujuan untuk mencapai hasil yang diinginkan melalui kerangka kerja yang melibatkan perencanaan, pengorganisasian, pengarahan, dan pengawasan. Sistem manajemen memastikan efisiensi dan efektivitas dalam mencapai tujuan organisasi. Komponen dalam sistem manajemen yaitu[3]:

1. Perangkat keras, merujuk pada peralatan fisik dan alat-alat yang terkait dengan komputer.
2. Perangkat lunak adalah kumpulan instruksi yang memungkinkan perangkat keras untuk memproses data.
3. Basis data melibatkan tabel, hubungan, dan unsur lain yang terkait penyimpanan data.
4. Prosedur pengoperasian adalah serangkaian aturan atau petunjuk untuk menggunakan sistem informasi berbasis komputer.
5. Personel pengoperasian mencakup ahli komputer, manajer, pengguna, analis, programmer, serta jabatan-jabatan terkait dengan pemanfaatan sistem informasi berbasis komputer.

2.2 Barang Habis Pakai

Barang habis pakai atau BHP adalah barang yang hanya dapat digunakan satu kali pemakaian saja dan setelah itu fungsi barang tersebut akan habis. Contoh barang habis pakai mencakup kebutuhan rumah tangga seperti tisu, sikat gigi, sabun, dan sampo. Barang-barang tersebut menjadi sarana penunjang kegiatan operasional dalam suatu asrama di lembaga pendidikan. Dalam konteks ini, peran barang habis pakai sangat penting untuk menjaga kebersihan, kesehatan, dan kelancaran operasional asrama[1].

2.3 *Website*

Website merupakan sebuah kumpulan informasi atau halaman yang dapat diakses melalui jaringan internet oleh semua orang selama terhubung ke internet. Ini mencakup halaman *web* dan *file* pendukungnya, seperti gambar, video, dan berbagai *file* digital yang disimpan di *server web*[4]. Secara sederhana, website merupakan kombinasi folder dan *file* yang berisi perintah dan fungsi khusus yang memungkinkan pengguna mengakses informasi dan berinteraksi dengan fitur[5].

2.4 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak adalah penerapan prinsip teknik untuk menciptakan perangkat lunak yang ekonomis, andal, dan efisien. Proses ini membentuk dasar keseluruhan proses perangkat lunak dengan serangkaian aktivitas kerangka kerja yang berlaku untuk semua proyek, tanpa memandang ukuran atau kompleksitasnya. Tujuan utamanya adalah mengembangkan metode yang meningkatkan sistem skala besar dengan konsistensi, menghasilkan perangkat lunak berkualitas tinggi sambil menjaga biaya rendah dan menghemat waktu. Secara keseluruhan, rekayasa perangkat lunak membantu pengembang menghasilkan perangkat yang berkualitas[6].

2.5 Metode Pengembangan Sistem

Sistem ini dibangun dengan menggunakan metode *Waterfall*, yaitu metode ini dilakukan secara terstruktur dan berurutan, jika tahap pertama belum selesai, maka tahap kedua tidak bisa berjalan dan tidak dapat mengulang ke tahap sebelumnya jika tahap pertama sudah selesai karena saling berkaitan untuk mengurangi terjadinya kesalahan[2]. Tahapan-tahapan pengembanga sistem yaitu:

1. *Requirement*
   Pada tahap perencanaan, peneliti mengumpulkan data-data terkait sistem yang akan dibangun melalui wawancara dan observasi di Akademi Maritim Nusantara Cilacap. Data yang diperoleh meliputi kebutuhan perangkat lunak, hambatan yang dihadapi, dan tujuan pembuatan sistem.
2. *Design*
   Tahap ini berfokus pada pembuatan program perangkat lunak, termasuk struktur data, arsitektur perangkat lunak, perancangan antarmuka, dan prosedur pengodean. Dalam konteks pembangunan sistem manajemen barang habis pakai, tahap ini melibatkan pembuatan alur kerja yang dijelaskan dalam *flowchart*, desain antarmuka, dan pengembangan algoritma sistem.
3. *Implementation*
   Tahap ini merupakan tahapan pembuatan sistem dengan menggunakan kode-kode bahasa pemrograman tertentu yang dapat dikenali komputer dan sesuai dengan kebutuhan pengguna. Pengembang mengimplementasikan semua komponen dan modul sistem berdasarkan desain yang telah dibuat.
4. *Verification*
   Pada tahap ini, semua kode yang telah dibuat dievaluasi melalui untuk memastikan bahwa sistem yang dikembangkan sudah sesuai dengan keinginan pengguna dan mengidentifikasi kegagalan atau kesalahan sistem.
5. *Maintenance*
   Pada tahap akhir ini, perangkat lunak yang telah selesai dikembangkan akan diimplementasikan dan dilakukan pemeliharaan. Tindakan pemeliharaan meliputi perbaikan kesalahan yang mungkin tidak terdeteksi pada tahap sebelumnya.

2.6 PBO (Pemprograman Berbasis Objek)

Pemrograman Berorientasi Objek (PBO) adalah suatu metode pemrograman yang memanfaatkan objek dan kelas. Pendekatan ini menggambarkan pemecahan masalah dalam program sebagai interaksi antar objek. Dalam PBO, objek terdiri dari sejumlah *field* atau atribut dan metode terkait dengan objek tersebut. Bentuk objek atau variabel tersebut bersifat tidak tetap, bergantung pada program yang sedang dibuat[7].

Berikut kelebihan dari PBO yaitu:

1. *Reusable*, kemampuan untuk menggunakan kembali kode yang telah dibuat sebelumnya.
2. *Extensibility*, seorang pemrogram memiliki kemampuan untuk membuat metode baru atau mengubah yang sudah ada sesuai kebutuhan tanpa perlu membuat kode dari awal.
3. *Maintainability*, kode yang telah dibuat menjadi lebih mudah dikelola, terutama dalam skala besar aplikasi, yang meminimalkan risiko kesalahan selama pengembangan. Konsep modularitas dalam pemrograman berorientasi objek (*OOP*) membantu mengatasi potensi masalah tersebut.

2.6.1 *Framework*

Framework adalah kumpulan komponen pemrograman yang telah siap pakai untuk digunakan ulang, yang membantu programmer menghindari menulis skrip yang serupa untuk tugas-tugas yang sama. Ini terdiri dari instruksi program terorganisir dalam class dan fungsi, memudahkan pengembang dalam penggunaannya tanpa perlu menulis ulang sintaks program yang serupa. Penggunaan *framework* memberikan efisiensi waktu dengan mengurangi penulisan kode yang repetitif, sehingga menghasilkan *source code* yang lebih bersih dan terstruktur [13].

1. Laravel

Laravel merupakan suatu kerangka kerja web berbasis *PHP* yang bersifat *open-source* dan gratis, dirancang oleh Taylor Otwell untuk memfasilitasi pengembangan aplikasi web yang mengadopsi pola *Model-View-Controller (MVC)*. Ini berarti aplikasi yang dikembangkan dengan *Laravel* memiliki struktur yang terorganisir dengan jelas, memisahkan logika aplikasi dari presentasi tampilan. Dalam *Laravel*, terdapat fitur *routing* yang berfungsi sebagai penghubung antara permintaan (*request*) dari pengguna dan kontroler. Dengan kata lain, kontroler tidak secara langsung menerima permintaan tersebut. Proses *routing* ini memungkinkan pengembang untuk mendefinisikan secara jelas bagaimana aplikasi *web* menanggapi setiap jenis permintaan HTTP (seperti *GET, POST, PUT, DELETE*) yang diterima oleh server. Laravel menjadi pilihan utama bagi pengembang dalam membangun aplikasi *web* yang stabil, efisien, dan mudah dikelola[13].

2.7 Basis Data

Basis data adalah suatu wadah di mana data yang saling terkait terkumpul dalam suatu entitas seperti perusahaan atau organisasi. Basis data terdiri dari kumpulan tabel yang terhubung tanpa duplikasi, diorganisir berdasarkan atribut kunci dan hubungan antar tabel. Tujuannya adalah menghasilkan informasi yang akurat, tepat waktu, dan terorganisir dengan baik, sekaligus mengurangi duplikasi data.[8]. Perangkat lunak yang digunakan untuk mengelola basis data adalah *Database Management System* (*DBMS*). *DBMS* adalah suatu sistem perangkat lunak yang memungkinkan pengguna untuk menentukan, membuat, merawat, dan memberikan akses terkontrol terhadap data. Terdapat dua jenis bahasa yang digunakan dalam *DBMS*:

1. *Data Definision Language (DDL*), digunakan oleh administrator database saat membuat atau menyiapkan *file database*.
2. *Data Manipulation Language (DML)*, merupakan sebagian dari bahasa *DBMS* yang berfungsi untuk mengelola data.

2.8 UML

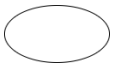
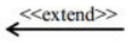
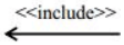
Unified Modelling Language (UML) merupakan suatu metode permodelan yang disajikan secara visual sebagai alat untuk merancang sistem berorientasi objek. *UML* berperan sebagai bahasa umum yang digunakan untuk visualisasi, perancangan, dan dokumentasi sistem perangkat lunak. Penerapan *UML* tidak hanya terbatas pada pemodelan bisnis, tetapi juga digunakan dalam pemodelan sistem non-perangkat lunak lainnya. Selain itu, *UML* menjadi bahasa pemodelan yang menerapkan konsep orientasi objek [9].

1. *Use Case*

Diagram *use case* merupakan salah satu bentuk diagram *UML* (*Unified Modeling Language*) yang memvisualisasikan interaksi antara sistem dan aktor. *Use case* dapat menjelaskan jenis interaksi yang terjadi antara pengguna sistem dan sistem itu sendiri. Diagram ini digunakan untuk

mengidentifikasi fungsi yang terdapat dalam sebuah sistem informasi dan menentukan siapa yang berhak menggunakan fungsi tersebut[10]. Interaksi antara sistem dan aktor dijelaskan dalam diagram *use case* dengan menggunakan simbol tertentu yang dapat ditemukan dalam Tabel 2.1

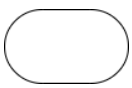
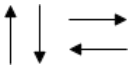
Tabel 2. 1 Simbol *Use Case*

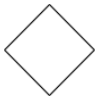
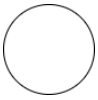
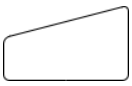
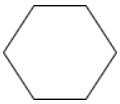
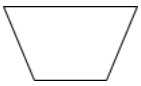
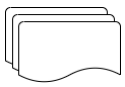
No.	Gambar Simbol	Nama	Keterangan
1.		*Use case*	Deskripsi urutan aksi yang dilakukan sistem untuk berinteraksi dengan seorang aktor.
2.		*Actor*	Entitas yang berinteraksi dengan sistem, berupa pengguna atau sistem lain, yang berhubungan dengan *use case*.
3.		*Extend*	Menunjukan bahwa *use case* target merupakan tambahan fungsionalitas dari *use case* lainnya.
4.		*Association*	Menghubungkan antara aktor dan *use case* yang menunjukkan interaksi aktor dengan use case.
5.		*Include*	Menunjukan bahwa sebuah *use case* memerlukan *use case* lain untuk menjalankan fungsinya.
6.		*Generalisasi*	Menunjukan hubungan hierarkis di mana aktor atau *use case* anak mewarisi sifat dan perilaku aktor atau use case induk.

2.9 *Flowchart*

Flowchart adalah alat pemetaan yang sederhana, menampilkan urutan tindakan dalam suatu proses dengan bentuk yang mudah dibaca dan dikomunikasikan melalui simbol-simbol khusus. Digunakan untuk menjelaskan alur kerja atau urutan prosedur suatu program kepada orang lain, *flowchart* membantu analis dalam memecahkan masalah segmen yang lebih kecil, terutama masalah yang memerlukan pemahaman dan evaluasi lebih lanjut[11]. Simbol-simbol khusus yang digunakan dalam *flowchart* dapat ditemukan pada Tabel 2.2

Tabel 2. 2 Simbol *Flowchart*

No.	Gambar Simbol	Nama	Keterangan
1.		*Terminal point*	Simbol oval digunakan untuk menyatakan titik awal atau akhir dari suatu proses.
2.		*Flow direction*	Simbol panah digunakan untuk menyatakan alur proses kerja dan dan menghubungkan simbol dalam diagram.

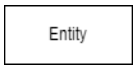
3.		*Process*	Simbol persegi panjang digunakan untuk menyatakan suatu proses yang dilakukan oleh komputer.
4.		*Decision*	Simbol yang menunjukkan titik keputusan dalam alur kerja, di mana kondisi tertentu akan menentukan jalur yang diambil.
5.		*Input output*	Simbol jajaran genjang digunakan untuk menyatakan operasi *input* dan *output* tanpa tergantung pada jenis perangkat.
6.		*Predefined process*	Simbol yang Menunjukan suatu pelaksanaan prosedur yang belum detail dan akan diperinci di tempat lain
7.		*Connector (On-page)*	Simbol yang menghubungkan dengan proses lainnya dalam satu halaman yang sama.
8.		*Connector (Off-page)*	Simbol yang menghubungkan dengan proses lainnya dalam satu halaman yang berbeda.
9.		*Manual input*	Simbol yang menunjukkan input data secara manual menggunakan online *keyboard*.
10.		*Preparation*	Simbol yang menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal.
11.		*Manual operation*	Menunjukan proses atau kegiatan yang dilakukan secara manual tanpa menggunakan komputer.
12.		*Document*	Simbol yang digunakan untuk mencetak keluaran dalam bentuk dokumen (melalui *printer*).
13.		*Multiple document*	Simbol yang digunakan untuk mencetak keluaran dalam bentuk beberapa dokumen (melalui *printer*).
14.		*Display*	Simbol yang digunakan untuk menunjukkan keluaran yang ditampilkan di layar monitor.

2.10 *ERD*

Entity Relationship Diagram (ERD) adalah teknik pendekatan untuk menggambarkan hubungan antara model data. ERD secara jelas mengidentifikasi entitas-entitas (objek data) dan hubungan-hubungan yang terjadi di antara entitas tersebut. Dengan memanfaatkan notasi dan simbol khusus, ERD membantu visualisasi relasi antar entitas, memberikan landasan untuk

memahami struktur data dalam proyek dan hubungan data dalam basis data. Dengan menggunakan *ERD*, dapat dijelaskan dengan jelas dan terperinci mengenai bagaimana data dalam basis data saling terkait berdasarkan objek-objek dasar data yang memiliki hubungan[12]. Simbol-simbol khusus digunakan dalam *ERD*, yang dapat ditemukan pada Tabel 2.3

Tabel 2. 3 Simbol *ERD*

No.	Gambar Simbol	Nama	Keterangan
1.		*Entity*	Simbol yang digunakan untuk menyatakan himpunan entitas yang perlu disimpan dalam *database.*
2.		*Relationship*	Simbol yang digunakan untuk menghubungkan entitas yang ada.
3.		*Atribut*	Simbol yang menyatakan entitas atau relasi yang menyediakan penjelasan detil tentang entitas atau relasi.
4.		*Atribut Multivalue*	Simbol yang menyatakan atribut yang memiliki nilai lebih dari satu setiap baris data.
5.		*Atribut Primary Key*	Simbol yang menyatakan atribut yang menjadi kunci utama dan tidak boleh sama dengan atribut yang lain.
6.		*Association*	Simbol yang menghubungkan atribut dengan entitas dan entitas dengan relasi.