BAB II
LANDASAN TEORI

2.1 Landasan Teori

Dalam menunjang penelitian ini, maka diperlukan teori-teori yang mendasar. Teori-teori yang digunakan dalam penelitian ini adalah:

2.1.1 Sistem

Sistem merupakan suatu bentuk integrasi antara satu komponen dengan komponen lainnya, karena sistem memiliki sasaran yang berbeda untuk setiap kasus yang terjadi yang ada di dalam sistem tersebut[1]. Sistem terdiri dari tiga unsur, yaitu: input (masukan), proses dan output (pengeluaran). Berikut ini sifat dari sistem sebagaimana merujuk kepada La Midjan dan Susanto (2004) yaitu [2]:

1. Tujuan Sistem, merupakan target atau sasaran akhir yang ingin dicapai oleh suatu sistem,
2. Batas Sistem, merupakan garis abstraksi yang memisahkan antara sistem dan lingkungannya,
3. Subsistem, merupakan komponen atau bagian dari suatu sistem, subsistem ini bisa phisik ataupun abstrak,
4. Hubungan dan Hirarki Sistem, merupakan hubungan yang terjadi antar subsistem dengan subsistem lainnya yang setingkat atau antara subsistem dengan yang lebih besar,
5. Input-Proses-Output, yaitu sebagai masukan, diolah untuk menghasilkan berbagai keluaran, dan,
6. Linkungan Sistem, merupakan faktor-faktor di luar sistem yang mempengaruhi sistem.

Berdasarkan pendapat yang dikemukakan di atas, jadi dapat disimpulkan bahwa sistem merupakan kumpulan suatu elemen yang saling terkait untuk mencapai tujuan tertentu. Tujuan dari sistem yakni memproses inputan untuk diolah sehingga menghasilkan output yang diperlukan.

2.1.2 *Monitoring*

Monitoring adalah proses pengumpulan dan analisis informasi berdasarkan indikator yang ditetapkan secara sistematis dan kontinu tentang suatu kegiatan atau program sehingga mampu dilaksanakan tindakan koreksi untuk penyempurnaan kegiatan itu selanjutnya[3].

2.1.3 Sistem *Monitoring*

Sistem *monitoring* adalah suatu sistem yang melakukan proses pemantauan secara terus menerus. Sistem *monitoring* dibutuhkan dalam proses pemantauan keadaan suatu objek yang diamati guna mendapatkan informasi yang tepat waktu[4]. Secara umum *monitoring* bertujuan mendapatkan umpan balik bagi kebutuhan program pembelajaran proses yang sedang berjalan,

dengan mengetahui kebutuhan ini pelaksanaan program akan segera mempersiapkan kebuturan dalam pembelajaran tersebut[5].

2.1.4 Pekerjaan

Pekerjaan adalah berbagai aktivitas yang dilaksanakan, dikerjakan, atau diselesaikan oleh seseorang, serta tanggung jawab dan tugas yang harus dipenuhi. Pekerjaan juga terdiri dari rangkaian tugas dan tanggung jawab yang dialokasikan kepada tenaga kerja, yang akan, sedang, atau telah mereka selesaikan dalam jangka waktu tertentu. Upaya ini tidak hanya mencakup tindakan fisik atau mental tetapi juga hasil akhir dari upaya tersebut, yang berkontribusi pada pencapaian tujuan organisasi atau individu[6].

2.1.5 *Cleaning Service*

Cleaning service adalah memberikan pelayanan kebersihan, kerapihan dan higenisasi dari sebuah gedung atau bangunan baik *indoor* ataupun *outdoor* sehingga tercipta suasana yang *comfortable* dalam menunjang aktifitas sehari-hari sebagai tujuan jangka pendeknya dan sebagai tujuan jangka panjangnya[7].

2.1.6 PT Provice Group

PT Provice Group adalah perusahaan pengelolaan aset yang didirikan pada tahun 2007, dengan awal bisnis di kawasan Rasuna Epicentrum. Lima tahun terakhir, Provice Group memperluas jangkauan pengelolaan properti di seluruh nusantara, meningkatkan layanan untuk memenuhi tuntutan dan standar klien. Provice Group fleksibel mengembangkan strategi untuk memenuhi kebutuhan klien, baik dalam bentuk layanan tunggal maupun terpadu, termasuk nasehat, penilaian biaya, pengoperasian bangunan, hubungan antar penyewa, pengelolaan parkir, pemeliharaan, keamanan, staf administrasi, hingga manajemen keuangan. Provice Group berkomitmen mengoptimalkan nilai, menurunkan biaya, dan meningkatkan keuntungan aset klien, dengan fokus pada investor, pengembang, penghuni, dan ritel, serta mencakup kebutuhan properti seperti kantor, ritel, industri, perumahan, dan perhotelan. Dengan pengalaman luas di bidang pengelolaan properti, Provice Group menjaga harga pasar yang bersaing dan meningkatkan keuntungan. Perusahaan mengutamakan etika "*People First*" dan bekerja sebagai satu kesatuan yang selaras, memastikan operasi yang terkoordinasi dengan baik.

2.1.7 Website

Website adalah salah satu aplikasi yang berisikan dokumen–dokumen multimedia (teks, gambar, suara, animasi, video) di dalamnya yang mengunakan protokol HTTP (*Hypertext Transfer*

Protokol) dan untuk mengakses menggunakan perangkat lunak yang disebut *browser*. Biasanya, sebuah situs web terdiri dari beberapa halaman web yang saling berhubungan[8].

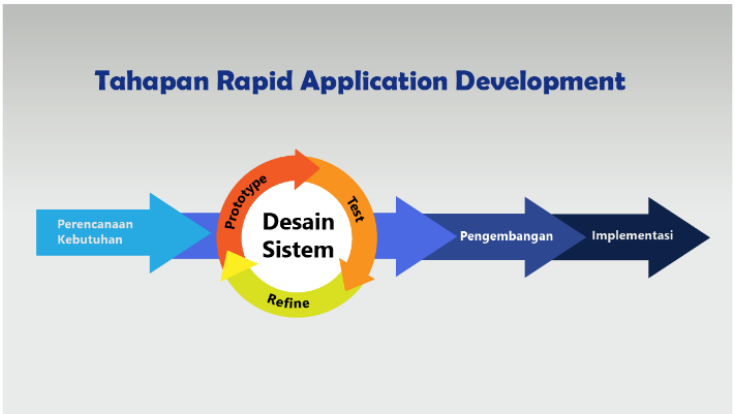
2.1.8 *Framework Laravel*

Framework laravel adalah kumpulan intruksi-intruksi yang dikumpulkan dalam *class* dan *function-function* dengan fungsi masing-masing untuk memudahkan *develope*r dalam memanggilnya tanpa harus menuliskan *syntax* program yang sama berulang-ulang serta dapat menghemat waktu. *Laravel* merupakan sebuah *Framework* PHP *(Hypertext Preprocessor)* yang dirilis di bawah lisensi MIT, dibangun dengan konsep MVC (*Model, View, Controller*). *Laravel* adalah *Framework* PHP yang dibuat oleh Taylor Otwell dan dirilis pertama kali pada tahun 2011. *Laravel* memiliki banyak fitur modern yang membantu dalam proses pengembangan website seperti *artisan, blade template engine, database migration, pagination, dan eloquent* ORM *(Object Relation Mapping)*. Dengan menggunakan *laravel* proses pengembangan aplikasi menjadi lebih cepat dan *powerfull*. Dalam pengembangan *website* menggunakan *laravel*, ada 2 *tools* yang akan sering kita pakai. yaitu *composer* dan *artisan*[9].

2.1.9 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak merupakan perihal kegiatan yang kreatif dan sistematis berdasar suatu disiplin ilmu yang membangun suatu perangkat lunak berdasar suatu aspek masalah tertentu. Proses perangkat lunak sebagai sebuah kerangka kerja untuk tugas-tugas yang dibutuhkan dalam membangun perangkat lunak dengan kualitas yang baik. Proses perangkat lunak menentukan pendekatan yang digunakan pada perangkat lunak yang dikembangkan. Pengembangan perangkat lunak juga meliputi teknologi yang mempopulasikan metode-metode, teknis, alat-alat bantu otomatis, dan prosedur-prosedur atau sering disebut dengan proses.

Sistem ini dikembangkan menggunakan metode RAD atau *Rapid Application Development.* RAD merupakan model pengembangan perangkat lunak yang berorientasi pada objek. RAD ditujukan untuk mengurangi pemakaian waktu yang semula membutuhkan waktu yang lama, dengan RAD waktu yang digunakan lebih efektif. RAD menerapkan metode berulang (iteratif) untuk mengembangkan sistemnya secara bertahap dengan menetapkan sesuai dengan kebutuhan user. RAD muncul dari gabungan beberapa teknik diantaranya terstruktur dan *prototyping,* dan *joint application*[10]. Tahapan metode RAD dapat dilihat pada Gambar 2. 1.



Gambar 2. 1 Tahapan Metode RAD

Metode RAD memiliki model proses pengembangan yang singkat, sehingga aplikasi dapat dibuat secara cepat. Ada 4 tahapan dalam metode RAD, yaitu:

1. Perencanaan Kebutuhan.

   Pada tahap ini melibatkan perencanaan kebutuhan yang diperlukan dalam pengembangan sistem. Kebutuhan tersebut didapatkan dari hasil wawancara dan observasi dengan PT Provice Group dan studi literatur yang telah dilakukan.
2. Desain Sistem.

   Pada tahap ini melibatkan proses desain dengan *Flowchart* dan *Unified Modeling Languange* (UML), dilanjutkan dengan membuat desain *prototype*. Jika tidak sesuai dengan kemauan PT Provice Group, maka desain akan diperbaiki.
3. Pengembangan.

   Pada tahap ini desain yang telah di rencanakan, dimasukan sebagai perangkat lunak dengan *Framework Laravel*. Pengujian juga dilakukan untuk memastikan minim kesalahan dan memenuhi kebutuhan sistem.
4. Implementasi.

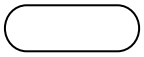
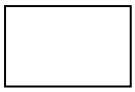
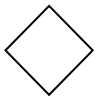
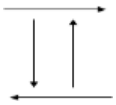
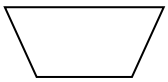
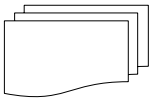
   Tahap terakhir yaitu mengimplementasikan sistem yang telah di kembangkan untuk mendapatkan *respons* terhadap sistem dari *stackholder* terkait.

Adapun struktur data merupakan cara menyimpan atau mempresentasikan data didalam komputer agar bisa dipakai secara efisien. Berikut bagian-bagian yang ada pada struktur data:

A. *Flowchart*

Flowchart merupakan suatu diagram yang menampilkan langkah-langkah dan keputusan untuk melakukan sebuah proses dari suatu program [11]. Pada Tabel 2.1 merupakan simbol-simbol *flowchart*:

Tabel 2. 1 Simbol-simbol *flowchart*

No	Simbol	Nama	Keterangan
1.		*Terminator Symbol*	Simbol yang menyatakan awal atau akhir suatu program.
2.		*Input-Output*	Memasukkan data maupun menunjukkan hasil dari suatu proses tanpa tergantung dengan jenis pendataannya.
3.		*Processing Symbol*	Menyatakan suatu proses yang dilakukan oleh komputer.
4.		*Decision symbol*	Menunjukkan kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban, yaitu ya atau tidak.
5.		*Flow symbol*	Menghubungkan antara simbol satu dengan simbol yang lain atau menyatakan jalannya arus dalam suatu proses. Simbol arus ini sering disebut juga dengan *connecting line*.
6.		*Manual symbol*	Menyatakan suatu proses yang tidak dilakukan komputer.
7.		*Document Symbol*	Merupakan simbol untuk data yang terbentuk informasi.
8.		*Display*	Menunjukkan output dari suatu proses atau aktivitas yang akan ditampilkan kepada pengguna.
9.		*Multidocument*	Menunjukkan bahwa output dari suatu proses atau aktivitas terdiri dari beberapa dokumen

B. *Unified Modeling Languange* (UML)

Salah satu standar bahasa yang sering digunakan di dunia industri untuk mendefinisikan kebutuhan, membuat analisis dan desain, serta menggambarkan arsitektur adalah *Unified Modeling Language* (UML). UML memudahkan pemahaman terhadap masalah-masalah yang kompleks dengan menyederhanakannya sehingga dapat dipelajari dengan lebih mudah. UML dapat menunjukkan hubungan antar kelas dengan menggunakan garis lurus. Model UML yang diterapkan dalam pengembangan sistem ini adalah *use case* dan *class* diagram[12].

1. *Use Case*

Use case diagram adalah permodelan yang digunakan untuk merancang sistem informasi yang akan dibuat. Diagram ini mendeskripsikan interaksi antara satu atau lebih aktor dengan sistem informasi yang sedang dikembangkan. *Use case* diagram digunakan untuk mengidentifikasi fungsi-fungsi yang ada dalam sistem serta siapa saja yang berhak menggunakan fungsi-fungsi tersebut[13]. Tabel 2.2 menunjukkan simbol-simbol yang digunakan dalam *use case* diagram.

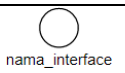
Tabel 2. 2 Simbol-simbol *use case*

No	Simbol	Nama	Keterangan
1		*Use case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*.
2		*Actor*	Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*.
3		*Assosiation*	Menghubungkan antara objek satu dengan objek lainnya.
4		*Include*	Menspesifikan bahwa *use case* sumber segala eksplisit.
5		*Extend*	Menspesifikan bahwa *use case* target memperluas perilaku dari *use case* sumber suatu titik yang diberikan.
6		*System*	Menspesifikan paket yang menampilkan sistem secara terbatas.
7		*Generaliztion*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada diatasnya objek induk.

2. *Class Diagram*

Class diagram merupakan gambaran struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. *Class diagram* terdiri dari atribut dan operasi dengan tujuan pembuat program dapat membuat hubungan antara dokumentasi perancangan dan perangkat lunak sesuai. Simbol-simbol *class diagram* dapat dilihat di Tabel 2.3

Tabel 2. 3 Simbol-simbol *class diagram*

No	Simbol	Nama	Keterangan
1		*Class*	Kelas pada struktur sistem.
2		*Interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3		*Association*	Relasi anatar kelas dengan makna umum.
4.		Coordinated assosiactes	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas lain.
4		*Generalization*	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum khusus).
5		*Dependency*	Relasi antarkelas dengan makna kebergantungan antarkelas.
6		*Aggregation*	Relasi antarkelas dengan makna semua-bagian *(whole-part)*.

2.1.10 Pemrograman Berorientasi Objek (PBO)

Pemrograman berorientasi objek adalah jenis pemrograman yang berfokus pada objek, di mana semua data dan fungsi dikemas dalam kelas-kelas atau objek-objek[14]. Agar sebuah bahasa pemrograman dapat digolongkan sebagai berorientasi objek, bahasa tersebut harus memiliki fitur-fitur yang memungkinkan implementasi dari keempat konsep utama pemrograman berorientasi objek, yaitu :

a) *Abstraksi* adalah sebuah prinsip yang digunakan untuk merepresentasikan dunia nyata yang kompleks ini menjadi sebuah model yang sederhana dengan menghiraukan aspek - aspek lainnya yang tidak sesuai dengan permasalahan.

b) *Encapsulation* adalah bagaimana membungkus data dan *method* yang menyusun *class* hingga *class* dipandang sebagai suatu modul.

c) *Polymorphison* adalah sesuatu yang memiliki banyak bentuk, diartikan sebagai modul yang memiliki kesamaan nama, namun *behaviour* (tingkah laku yang berbeda) sehingga *listing* kode implementasinya berbeda.

d) *Inheritance* (Penurunan sifat) adalah proses pewarisan data dan method dari suatu *class* kepada *class* yang lain.

Metode berorientasi objek adalah sebuah sistem yang komponennya dibungkus menjadi kelompok data dan fungsi. Sistem berorientasi objek merupakan sebuah sistem yang dibangun dengan berdasarkan metode berorientasi objek. Metode berorientasi objek adalah sebuah sistem yang komponennya dibungkus menjadi data dan fungsi. Sehingga, objek dapat memiliki kelakuan yang berbeda, apabila diperbarui implementasi dari objek tersebut sesuai dengan karakteristiknya.

2.1.11 Design Antarmuka Pengguna

Design antarmuka pengguna adalah istilah yang merujuk pada tampilan dari mesin atau komputer yang berinteraksi langsung dengan pengguna. *User Interface* (UI) atau tampilan antarmuka pengguna merupakan bagian penting dalam sebuah sistem atau aplikasi. UI adalah bagian dari sistem yang digunakan untuk berinteraksi langsung dengan pengguna. Desain dan penyusunan tampilan antarmuka perlu diperhatikan untuk menghasilkan tampilan yang baik. Dengan merancang tampilan antarmuka yang mudah digunakan, pengembang aplikasi dapat menciptakan sistem yang menarik bagi pengguna. Dari perspektif pengguna, diharapkan semua aplikasi ke depannya menggunakan desain tampilan antarmuka yang ramah pengguna sehingga mereka tidak merasa terganggu ketika menggunakan sistem informasi[12].

2.1.12 Basis Data

Basis data merupakan koleksi dari data yang terorganisasi dengan cara sedemikian rupa sehingga data tersebut mudah disimpan dan dimanipulasi. Disamping berisi atau menyimpan data, setiap basis data juga mengandung atau menyimpan definisi struktur[15].

a) *Database Management System* (DBMS)

Database Management System (DBMS) atau dalam Bahasa Indonesia sering disebut sebagai sistem manajemen basis data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola dan menampilkan data. Dalam penggunaan DBMS dibutuhkan komponen-komponen antara lain :

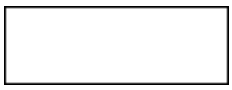
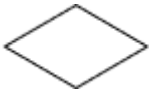
(1) *Query processor,* komponen yang mengubah bentuk *query* dalam bentuk instruksi ke dalam *database manager*.

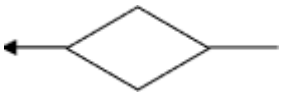
(2) *Database manager*, menerima *query*, menguji eksternal dan konseptual untuk menentukan apakah *record-record* tersebut dibutuhkan untuk memenuhi permintaan kemudian hari dari *database manager* dengan memanggil *file manager* untuk menyelesaikan permintaan.
(3) *File manager*, memanipulasi penyimpanan file dan mengatur alokasi ruang penyimpanan *disk.*
(4) *Data manipulation language processor*, modul yang mengubah perintah DML yang ditempelkan ke dalam program aplikasi dalam bentuk fungsi-fungsi.
(5) *Data definition language compiler*, mengubah statement DDL menjadi kumpulan tabel atau *file* yang berisi *data dictionary* atau meta data.
(6) *Dictionary manager*, mengatur akses dan memelihara *data dictionary*

Salah satu *software* yang tergolong ke dalam DBMS adalah MySQL. MySQL merupakan salah satu *Relational Database Management System* (RDBMS) yang saat ini sedang banyak diminati.

Pemodelan awal basis data yang paling sering digunakan adalah dengan *Entity Relationship Diagram* (ERD). *Entity Relationship Diagram* (ERD) merupakan serangkaian metode atau alat yang digunakan untuk menggambarkan data atau objek-objek yang dibuat berdasarkan dunia nyata, yang disebut entitas, serta hubungan antara entitas-entitas tersebut dengan menggunakan berbagai notasi[15]. Komponen-komponen pembentuk ERD dapat di lihat pada Tabel 2.4 di bawah ini:

Tabel 2. 4 Simbol-simbol ERD

Notasi	Komponen	Keterangan
	Entitas	Individu yang mewakili suatu objek dan dapat dibedakan dengan objek yang lain.
	Atribut	Properti yang dimiliki oleh suatu entitas, dimana dapat mendeskripsikan karakteristik dari entitas tersebut.
	Relasi	Menunjukkan hubungan diantara sejumlah entitas yang berbeda.
	Relasi 1:1	Relasi yang menunjukkan bahwa setiap entitas pada himpunan entitas pertama berhubungan dengan paling banyak satu entitas pada himpunan entitas kedua.

Notasi	Komponen	Keterangan
	Relasi 1:N	Relasi yang menunjukkanbahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak atau sebaliknya. Setiap entitas dapat berelasi dengan banyak entitas pada himpunan entias yang lain.

Metode pemodelan data dengan menggunakan *Entity-Relationship Diagram* (ERD) digunakan untuk mengidentifikasi objek data dan hubungannya dengan menggunakan notasi grafis dalam perekayasaan perangkat lunak. Atribut dari masing-masing objek data digambarkan dengan menggunakan deskripsi objek data. ERD hanya berfokus pada data dengan menunjukkan jaringan data yang ada untuk suatu sistem yang diberikan.

b) *Structural Query Language* (SQL)

Structural Query Language (SQL) adalah Bahasa yang digunakan untuk mengelola data pada RDBMS. SQL awalnya dikembangkan berdasarkan teori aljabar relasional dan kalkulus. Secara umum SQL terdiri dari 2 (dua) bahasa yaitu *Data Definition Language* (DDL) *dan Data Manipulation Language* (DML).

(1) *Data Definition Language* (DDL)

Data Definition Language (DDL) adalah sebuah perintah yang digunakan untuk mendefinisikan atribut-atribut basis data, tabel, serta hubungan antar tabel. DDL lebih berfokus pada *manipulasi struktur database.* DDL digunakan untuk membuat atau menghapus tabel, membuat *key atau indeks,* serta membuat relasi antar tabel. Berikut adalah *sintaks* yang terdapat dalam DDL:

(a) *Create*

Perintah *create* digunakan untuk membuat objek baru, baik berupa *database,* tabel, indeks atau prosedur yang tersimpan.

(b) *Alter*

Perintah *alter* digunakan untuk memodifikasi onjek pada *database,* seperti *indeks,* dan lokasi.

(c) *Drop*

Perintah drop digunakan untuk menghilangkan atau menghapus objek pada *database.*

(2) *Data Manipulation Language* (DML)

Data Manipulation Language (DML) merupakan kelompok perintah yang berfungsi untuk melakukan proses *insert*, *update* atau *delete* ke dalam suatu database. Berikut sintaks yang ada didalam DML:

(a) *Select*

Perintah *select* digunakan untuk menampilkan data/isi tabel dari *database*.

(b) *Insert*

Perintah *insert* untuk menambahkan data baru dalam *database*.

(c) *Update*

Perintah *update* digunakan untuk merubah data didalam *database*.

(d) *Delete*

Perintah *delete* digunakan untuk menghapus data dari *database*.