BAB II
LANDASAN TEORI

1.1. Landasan Teori

2.1.1 Sistem Informasi

Sistem informasi secara umum adalah sistem yang menggabungkan aktivitas manusia dan penggunaan teknologi untuk mendukung aktivitas manajemen dan operasional. Di dalamnya mengacu pada hubungan yang dibuat atas dasar interaksi manusia, data, informasi, teknologi, dan algoritma[3]. Secara singkat Sistem informasi merupakan kumpulan dari sistem yang saling bertukar data dan saling mendukung satu sama lain untuk menyelesaikan sebuah pekerjaan dan menghasilkan informasi baru[2].

2.1.2 Kejuaraan Pencak Silat

Kejuaraan pencak silat merupakan kompetisi pada bidang olahraga prestasi yang dipertandingkan baik di tingkat daerah sampai internasional. Pencak silat sendiri merupakan seni bela diri warisan budaya bangsa yang merupakan hasil karya secara turun menurun dari budaya bangsa Indonesia yang pada masa itu digunakan untuk membela atau mempertahankan diri. Dalam perkembangannya hingga saat ini, pencak silat tidak hanya digunakan untuk membela atau mempertahankan diri saja tetapi sudah menjadi sebuah olahraga prestasi sehingga dapat dilaksanakan kejuaraan pencak silat[4].

2.1.3 Pemrograman Beriorientasi Objek (PBO)

Pemrograman berorientasi objek atau *object oriented programming* merupakan pendekatan pemrograman yang menggunakan *object* dan *class*. Objek dalam PBO adalah unit terkecil pemrograman yang masih memiliki data (sifat karakteristik) dan fungsi. *Class* adalah wadah berisikan pemodelan suatu objek, mendeskripsikan karakteristik dan fungsi objek tersebut[5]. Setiap objek memiliki kemampuan untuk menerima pesan, memproses data, mengirim, menyimpan dan memanipulasi informasi. Sekumpulan objek-objek ini saling berinteraksi untuk bertukar informasi(0). Dengan demikian, dalam pemrograman berorientasi objek (PBO) memiliki karakteristik utama, yaitu :

a. Kelas (*class*)

Kelas meripakan kategori yang mengelompokkan objek dengan karakteristik serupa. Kelas mendefinisikan atribut dan operasi objek serta memungkinkan pembuatan banyak objek dengan struktur serupa.

b. Objek (*Object*)

Objek adalah entitas dalam lingkungan yang memiliki *attribute* dan *method* yang dimiliki oleh kelasnya. Objek memiliki kemampuan untuk melakukan tindakan dan bisa berinteraksi dengan objek lain dalam sistem yang lebih besar. Contoh dari objek adalah motor, kereta, truk. Objek merupakan entitas yang memiliki identitas (nama) dan biasanya menyimpan informasi tentang dirinya sendiri dan objek lain.

c. Metode (*method*)

Metode adalah prosedur atau fungsi yang didefinisikan oleh seorang programmer di dalam sebuah kelas. Dalam metode memiliki izin akses seperti *private*, *public*, dan *protected*. Sebuah kelas dapat memiliki beberapa *method* dengan nama yang berbeda maupun sama, asalkan memiliki parameter masukan yang berbeda. Hal ini memungkinkan kompiler atau interpreter untuk mengenali *method* mana yang akan dipanggil.

d. Atribut (*attribute*)

Atribut merupakan informasi terkait objek. Atribut merupakan karakteristik atau property objek yang mempengaruhi perilaku dan keadaan objek. Dalam pemrograman atribut, atribut merupakan *variable* yang dideklarasikan di dalam kelas dan setiap objek memiliki salinan atribut yang unik. Atribut juga memiliki hak akses seperti *method* yaitu *private*, *public*, dan *protected*.

e. Enkapsulasi (*encapsulation*)

Enkapsulasi merupakan proses membatasi ruang lingkup program terhadap data yang sedang diproses. Konsep pada pengkapsulan adalah menyembunyikan informasi. Data dan prosedur dikemas secara bersamaan dalam bungkus suatu objek sehingga data atau prosedur dari luar tidak dapat mengaksesnya.

2.1.4 Framework

Framework merupakan komponen pemrograman yang siap digunakan ulang kapan saja, sehingga *programmer* tidak harus membuat skrip yang sama untuk tugas yang sama[6].

2.1.5 Laravel

Laravel merupakan sebuah *framework* web berbasis PHP yang terbuka dan tidak berbayar, diciptakan oleh Taylor Otwell dan digunakan untuk pengembangan aplikasi berbasis web yang menggunakan pola MVC (*Model View Controller*). *Laravel* terdapat *routing* yang menjembatani antara *request* dari *user* dan *controller*, sehingga *controller* tidak langsung menerima *request* tersebut[6].

2.1.6 PHP

PHP merupakan singkatan dari *Hypertext Preprocessor* yang pada awalnya dikembangkan oleh Rasmus Lerdorf. PHP adalah bahasa pemrograman dengan sisi server yang digunakan untuk membuat aplikasi web dinamis dan interaktif. Hal itu berarti bahwa semua konfigurasi yang ditulis dalam bahasa ini dijalankan di server dan hasilnya kemudian dikirimkan ke *browser* pengguna sebagai halaman web statis yang dapat dilihat dan digunakan. Dalam hal menampilkan data dan mengelola situs web skrip PHP harus dikombinasikan dengan HTML (*Hypertext Markup Language*) dan CSS (*Cascading Style Sheets*) yang merupakan bahasa pemrograman yang digunakan dalam desain web[7].

2.1.7 Prototyping

Prototyping merupakan metode pengembangan perangkat lunak yang berupa model fisik kerja sistem dan berfungsi sebagai versi awal dari sistem. Metode *prototyping* ini akan menghasilkan *prototype* sistem sebagai perantara pengembang dan pengguna agar dapat berinteraksi dalam proses kegiatan pengembangan sistem informasi. Agar proses pembuatan *prototype* ini berhasil dengan baik maka perlu untuk mendefinisikan aturan-aturan pada tahap awal, yaitu pengembang dan pengguna harus satu pemahaman bahwa *prototype* akan dihilangkan atau ditambahkan pada bagiannya sehingga sesuai dengan perencanaan dan analisis yang dilakukan oleh pengembang sampai dengan ujicoba dilakukan secara simultan seiring dengan proses pengembangan[8].

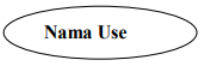
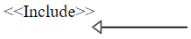
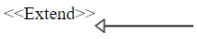
2.1.8 *Unified Modelling Language* (UML)

UML singkatan dari *Unified Modelling Language* sebagai bahasa, berarti UML memiliki sintaks dan semantik. Saat kita membuat suatu model menggunakan konsep UML ada berbagai aturan yang harus diikuti. Bagaimana elemen-elemn pada model-model yang dibuat berhubungan satu dengan yang lainnya harus sesuai dengan standar[2]. UML merupakan bahasa

visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. Beberapa pemodelan yang termasuk kedalam pemodelan UML seperti *use case diagram*, *class diagram*, *activity diagram*, dan *sequence diagram*[9].

Salah satu pemodelan yang digunakan dalam penelitian ini adalah *use cae* diagram. *Use case diagram* merupakan pemodelan untuk kelakuan sistem informasi yang akan dibuat yang mendeskripsikan interaksi antara *actor* dengan sistem[9]. Di bawah ini adalah simbol yang digunakan untuk membuat *use case diagram*, antara lain :

Tabel 2. 1 Simbol-simbol *Use Case*

No	Simbol	Nama	Keterangan
1		*Actor*	Merupakan pengguna dari sistem. Penamaan aktor menggunakan kata benda
2		*Use Case*	Merupakan pekerjaan yang dilakukan oleh aktor. Penamaan *use case* dengan kata kerja
3		*Association* / Asosiasi	Hubungan antara aktor dengan *use case*
4		*Include*	Hubungan antara *use case* dengan *use case*. *Include* menyatakan bahwa sebelum pekerjaan dilakukan harus mengerjakan pekerjaan lain terlebih dahulu
5		*Extends*	Hubungan antara *use case* dengan *use case*. *Extends* menyatakan bahwa jika pekerjaan yang dilakukan tidak sesuai atau terdapat kondisi khusus, maka lakukan pekerjaan itu

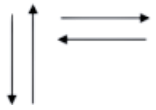
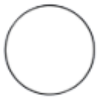
2.1.9 Flowchart

Flowchart adalah suatu bagan dengan simbol-simbol tertentu yang menggambarkan urutan proses secara detail dan hubungan antara suatu proses (intruksi) dengan proses lainnya [9]. Pada dasarnya, dalam merancang *flowchart* tidak ada ketentuan mutlak yang harus dipenuhi. Hal tersebut dikarenakan *flowchart* dibuat berdasarkan pemikiran untuk menganalisa suatu masalah. Hanya saja, flowchart dapat dirancang dengan menggunakan simbol-simbol standar yang umum digunakan dalam proses pembuatannya. Berikut penjelasan mengenai simbol-simbol *flowchart* yang dibagi dalam 3 kategori, diantaranya [10] :

(a) Simbol Arus (*Flow Direction Symbols*)

Simbol yang termasuk dalam kategori ini dapat digunakan sebagai simbol penghubung[11]. Beberapa simbol yang termasuk ke dalam kategori ini, yaitu :

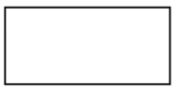
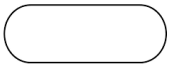
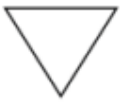
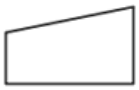
Tabel 2. 2 Simbol Arus

No	Simbol	Nama	Keterangan
1		*Flow Direction / Connecting Li*ne	Berfungsi untuk menghubungkan simbol yang satu dengan yang lainnya, menyatakan arus suatu proses
2		*Communication Link*	Berfungsi untuk transmisi data dari satu lokasi ke lokasi lain
3		*Connector*	Digunakan untuk menyatakan sambungan dari proses yang satu ke proses berikutnya di halaman yang sama
4		*Offline Connector*	Digunakan untuk menyatakan sambungan dari proses yang satu ke proses berikutnya di halaman yang berbeda

(b) Simbol Proses (*Processing Symbols*)

Simbol proses digunakan untuk menyatakan simbol yang berkaitan dengan serangkaian proses yang dilakukan[11]. Berikut beberapa simbol yang termasuk kedalam bagian proses, yaitu :

Tabel 2. 3 Simbol Proses

No	Simbol	Nama	Keterangan
1		*Processing*	Digunakan untuk menunjukkan pengolahan yang akan dilakukan dalam komputer
2		*Manual Operation*	Digunakan untuk menunjukkan pengolahan yang tidak dilakukan oleh komputer
3		*Decision*	Digunakan untuk memilih proses yang akan dilakukan berdasarkan kondisi tertentu
4		*Predefined Process*	Digunakan untuk mempersiapkan penyimpanan yang sedang/akan digunakan dengan memberikan harga awal
5		Terminal	Digunakan untuk memulai atau mengakhiri program
6		*Offline Storage*	Berfungsi untuk menunjukkan bahwa data akan disimpan ke media tertentu
7		*Manual Input Symbol*	Digunakan untuk menginputkan data secara manual dengan *keyboard*

(c) Simbol I/O (*Input/Output*)

simbol yang termasuk ke dalam bagian *input*/*output* berkaitan dengan masukan dan keluaran [11]. Berikut beberapa simbol yang termasuk, yaitu :

Tabel 2. 4 Simbol *Input/Output*

No	Simbol	Nama	Keterangan
1		*Input / Output*	Digunakan untuk menyatakan *input* dan *output* tanpa melihat jenisnya
2		*Punched Card*	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari *card*
3		*Disk Storage*	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari *disk*
4		*Magnetic Tape*	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari pita magnetis
5		*Document*	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari dokumen
6		*Display*	Digunakan untuk menyatakan masukan dan keluaran melalui layar monitor

2.1.10 Basis Data

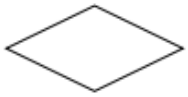
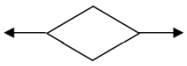
Basis data (*data base*) merupakan salah satu komponen yang penting dalam pembuatan sistem informasi, karena basis data merupakan hal pokok dalam menyediakan informasi tentang data kepada para pengguna khususnya. Pada tahap perancangan basis data, diperlukan tabel-tabel data dan relasinya untuk mengurutkan data atau struktur data [12].

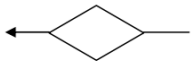
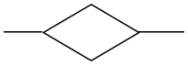
2.1.11 *Entity Relationship Diagram* (ERD)

ERD adalah diagram berbentuk notasi grafis yang berbeda dalam pembuatan *database* yang menghubungkan antara data satu dengan yang lain. Fungsi ERD adalah sebagai alat bantu

dalam pembuatan *database* dan memberikan gambaran bagaimana kerja database yang akan dibuat [13]. Simbol ERD dapat dilihat pada tabel 2.5 .

Table 2. 5 Simbol *Entity Relationship Diagram*

No	Simbol	Nama	Keterangan
1		*Entity*	Simbol yang menyatakan himpunan entitas ini bisa berupa suatu elemen lingkungan, sumber daya, atau transaksi yang begitu pentingnya bagi perusahaan sehingga didokumentasikan dengan data.
2		*Attribute*	Simbol terminal ini untuk menunjukkan namanama attribut yang ada pada suatu *entity*.
3		*Primary Key, Attribute*	Simbol attribut yang digaris bawahi, berfungsi sebagai key (kunci) diantara nama-nama atribut yang ada pada suatu *entity*.
4		*Relationship*	Simbol ini menyatakan relasi ini digunakan untuk menunjukkan hubungan yang ada antara entiti yang satu dengan entiti yang lainnya.
5		*Link*	Simbol berupa garis ini digunakan sebagai penghubung antara himpunan relasi dengan himpunan entitas dan himpunan entitas dengan atributnya.
6		*Relasi 1 : 1*	Simbol relasi yang menunjukkan bahwa setiap entitas pada himpunan entitas pertama berhubungan dengan paling

			banyak satu entitas pada himpunan entitas kedua.
7		*Relasi 1 : N*	Simbol relasi yang menunjukkan bahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak atau sebaliknya. Setiap entitas dapat berelasi dengan banyak entitas pada himpunan entitas yang lain.
8		*Relasi N : N*	Simbol hubungan ini menunjukkan bahwa setiap entitas pada himpunan entitas yang pertama dapat berhubungan dengan banyak entitas pada himpunan entitas yang kedua, demikian juga sebaliknya.

2.1.12 *Black-Box Testing*

Black-Box Testing adalah metode pengujian perangkat lunak yang menitikberatkan pada spesifikasi fungsional yang telah ditetapkan. Dalam proses ini, penguji (tester) bertugas untuk mengidentifikasi berbagai kondisi input yang mungkin terjadi dan kemudian melakukan pengujian berdasarkan spesifikasi fungsional dari program tersebut. Pengujian ini tidak memerlukan pemahaman mendalam tentang struktur internal atau kode sumber dari perangkat lunak yang diuji, sehingga penguji hanya berfokus pada keluaran atau hasil yang diharapkan sesuai dengan kondisi input yang telah ditentukan sebelumnya[14].