BAB II

LANDASAN TEORI

2.1. Konsep Dasar Sistem

A. Sistem

Sistem berasal dari Bahasa latin (*systema*) dan Bahasa Yunani (*sustema*) adalah suatu kesatuan yang tediri komponen atau elemen yang dihubungkan bersama untuk memudahkan aliran informasi, materi atau energi untuk mencapai suatu tujuan. Dalam pengertian yang paling umum, sebuah sistem adalah sekumpulan benda yang memiliki hubungan di antara mereka.

"Sistem adalah kumpulan atau himpunan dari unsur atau variabel–variabel yang saling berkait, saling berintraksi, dan saling tergantung satu sama lainuntuk mencapai tujuan" (Tohari, 2017).

Selain itu, sistem juga dapat didefinisikan sebagai sekumpulan objek – objek yang saling berelasi dan berinteraksi, serta hubungan antara objek bisa di lihat sebagai satu kesatuan yang dirancang untuk mencapai satu tujuan yang telah di tetapkan.

"Pemrograman Berorientasi Objek atau Object Oriented Programming (OOP) adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya"(Shalahuddin, 2017).

B. Website

"Website terhubung dengan suatu jaringan internet yang akan membawa pengguna kesuatu tujuan yang diinginkan oleh pengguna dengan cara mengklik link yang berupa teks, gambar"(Endra & Aprilita, 2018).

> "Situs Web (*Website*) awalnya merupakan suatu layanan sajian informasi yang menggunakan konsep hiperlink yang memudahkan surfer (sebutan bagi pemakai komputer yang melakukan penyelusuran informasi di Internet) untuk mendapatkan informasi dengan cukup mengklik suatu link berupa teks atau gambar maka informasi dari teks atau gambar akan ditampilkan secara lebih terperinci (*detail*)"(Sukmaindrayana & Sidik, 2017).

"*Website* adalah Kumpulan halaman yang saling terhubung yang di dalamnya terdapat beberapa item seperti dokumen dan gambar yang tersimpan di dalam web server" (Friyadie, 2016).

Berdasarkan penjelasan diatas, penulis dapat menyimpulkan bahwa *Website* adalah aplikasi yang berisikan dokumen-dokumen multimedia teks, gambar, suara, animasi, video dan bisa diakses seluruh dunia melalui jaringan internet.

C. Basis Data

"Basis data merupakan data yang saling terhubung dan berkaitan dengan subjek tertentu pada tujuan tertentu pula. Hubungan antardata ini dapat dilihat oleh adanya *field* ataupun kolom" (Saputra, 2017)

Berdasarkan dari kutipan diatas dapat disimpulkan bahwa *database* atau biasa disebut basis data merupakan kumpulan data yang saling terhubung dan berkaitan dengan subjek tertentu pada tujuan tertentu pula.

D. Model Pengembangan Perangkat Lunak

Metode pengembangan perangkat lunak yang digunakan dalam membangun sistem informasi penggajian karyawan berbasis web ini menggunakan model *Waterfall*.

"Model air terjun (*Waterfall*) menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis,desain pengodean,pengujian dan tahap pendukung (*support*)".(Rosa & & Shalahuddin, 2018)

1. Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dubutuhkan oleh user.

2. Desain

Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program Perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosuder pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

3. Pembuatan Kode Program

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai desain yang telah dibuat pada tahap desain.

4. Penguji

Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (error)dan memastikan keluaran yang Dihasilkan sesuai dengan yang diinginkan

5. Pendukung (*support*) atau pemeliharaan

Pendukung (*support*) Atau pemeliharaan (*maintenance*) Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke user. Perubahan bisa terjadi karena adanya kesalahn yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses

pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

2.2. Teori Pendukung

Teori pendukung dalam penyusunan laporan tugas akhir ini sangat diperlukan karena sebagai referensi untuk menunjang atau memperdalam pemahaman terhadap informasi-informasi yang disajikan.

A. *Entity Relationship Diagram* (ERD)

”ERD (*Entity Relationship Diagram*) merupakan suatu model data yang dikembangkan berdasarkan objek”(Sutanta, 2017).

”Entity Relationship Diagram (ERD) merupakan teknik yang digunakan untuk memodelkan kebutuhan data dari suatu organisasi, biasanya oleh System Analys dalam tahap analisis persyaratan proyek pengembangan system”(Sutanta, 2017).

ERD menggunakan sejumlah notasi dan symbol untuk menggambarkan struktur dan hubungan antar data, pada dasarnya ada tiga macam komponen yang digunakan yaitu :

1. **Entitas** adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai, sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat. Sebagai contoh pelanggan, pekerja dan lain-lain.
2. ***Attribute*** berfungsi mendeskripsikan karakter entiti. Misalnya atribut nama pekerja dari entiti pekerja. Setiap entiti bisa terdapat lebih dari satu atribut.
3. **Hubungan** atau ***(Relationship)*** sebagaimana halnya entiti maka dalam hubungan pun harus dibedakan antara hubungan atau bentuk hubungan antar entiti dengan isi dari hubungan itu sendiri. Misalnya dalam kasus hubungan antara entiti

siswa dan entiti mata kuliah adalah mengikuti, sedangkan isi hubungannya dapat berupa nilai ujian. *Relationship* disimbolkan dalam bentuk intan/*diamonds*.

4. **Garis Penghubung** adalah merangkai keterkaitan antara notasi-notasi yang digunakan seperti entitas, *relationship*, dan *attribute*.

B. *Logical Record Structure* (LRS)

"LRS adalah representasi dari struktur record – record pada table – table yang terbentuk dari hasil relasi antar himpunan entitas" (Friyadie, 2016).

""LRS adalah digambarkan kotak persegi panjang dan dengan nama yang unik. File record pada LRS ditempatkan dalam kotak. LRS terdiri dari link diantara tipe record lainnya, banyaknya link dari LRS yang diberi nama oleh filed-filed yang kelihatan pada kedua link tipe record" (Jacob & Sumaryana, 2018).

"*Logical Record Structure* dibentuk dengan nomor tipe *record*, beberapa tipe record digambarkan oleh kotak empat persegi Panjang dan dengan nama yang unik" (No et al., 2019).

C. *Unified Modelling Language* (UML)

> "Pengertian *Unified Modelling Language* (UML) adalah merupakan sistem arsitektur yang bekerja dalam OOAD (*Object-Oriented Analysis Design*) dengan satu bahasa yang konsisten untuk menentukan, visualisasi, mengkontruksi, dan mendokumentasi *artifact* (sepotong informasi yang digunakan atau dihasilkan dalam suatu proses rekayasa *software*, dapat berupa model, deskripsi, atau *software*) yang terdapat dalam sistem *software"*(Wahyu, 2017).

"*Unified Modelling Language* (UML) adalah Bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek" (Nugroho, 2018).

"UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem"(Hendini, 2017).

"Tehnik pengembangan sistem yang menggunakan Bahasa grafis sebagai alat untuk mendokumentasi dan melakukan spesifikasi pada sistem"(Mulyani, 201:48).

Dari pendapat para ahli disimpulkan "bahasa grafis dan desain serta menggambarkan arsitektur dalam atau perangkat lunak yang berparadigma berorientasi objek.

UML diaplikasikan untuk maksud tertentu sebagai berikut :

a. Merancang perangkat lunak

b. Sarana komunikasi antara perangkat lunak dengan proses bisnis

c. Menjabarkan sistem secara rinci untuk Analisa dan mencari apa yang diperlukan sistem

d. Mendokumentasikan sistem yang ada proses-proses dan organisasinya

1. Activity Diagram

"*Activity Diagram* menggambarkan aktivitas utama dari *user* pada sistem informasi yang dibuat (Irmayani & Susyatih, 2017).

"Alur kerja, logika, dan hubungan antara aktor dengan alur kerja dalam *use case* digambarkan dalam diagram yang disebut dengan aktivitas atau *activity diagram"*(Mulyani, 2017:55).

"Activity diagram atau diagram aktivitas menggambarkan proses yang terjadi di dalam sebuah sistem dengan bentuk mengalir atau urut dati satu proses ke proses selanjutnya. Diagram ini khususnya menggambarkan aliran *use case* dimana interaksi antara pengguna normal dan alternatif terjadi" (No et al., 2018).

Dari pendapat para ahli dapat disimpulkan *activity* diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem dan ada aliran kerja diperangkat lunak.

2. Use Case Diagram

"*Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat"(Rosa & Shalahuddin, 2018:155).

"Diagram use case menyajikan interaksi antara use case dan aktor. Dimana aktor dapat berupa orang, peralatan atau sistem lain yang berinteraksi dengan sistem yang sedang dibangun. Use case menggambarkan fungsionalitas sistem atau persyaratan yang harus dipenuhi sistem dari pandangan"(Setiawan & Khairuzzaman, 2017).(No et al., 2019)

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

a. Aktor merupakan orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun symbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.

b. Use case merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

3. *Class* Diagram

"*Class Diagram* merupakan model statis yang menggambarkan kelas dan relasi antar kelas yang konstan di dalam sistem dari waktu ke waktu. Class diagram menggambaran kelas, yang mencakup perilaku dan kedudukan dengan hubungan antar kelas" (Allan, 2017).

Class diagram menggambarkan struktur dan deskripsi *class* dan objek hubungan antara lain pewarisan, asosiasi dan lain-lain. *Class diagram* berfungsi untuk menjelaskan tipe dari objek sistem dan hubungannya dengan objek lain. Objek adalah nilai tertentu dari setiap atribut *class entity*. Adapun komponen *class diagram* adalah :

a. ***Object*** merupakan *instance* dari sebuah *class* dan dituliskan tersusun secara *horizontal*. Digambarkan sebagai sebuah class (kotak) dengan nama object di dalamnya yang diawali dengan sebuah titik koma.

b. ***Class*** merupakan blok-blok pembangun pada pemrograman berorientasi objek. Sebuah *class* digambarkan sebagai sebuah kotak yang terbagi dalam atas tiga bagian. Bagian atas adalah bagian nama dari *class*. Bagian tengah mendefinisikan properti atau *atribut class*. Bagian akhir mendefinisikan fungsi atau *method-method* dari sebuah *class*.

c. ***Association*** sebuah asosiasi merupakan sebuah *relationship* paling umum antara dua class dan dilambangkan oleh sebuah garis yang menghubungkan antara dua class. Garis ini bisa melambangkan tipe-tipe *relationship* dan juga dapat menampilkan hukum-hukum multiplisitas pada sebuah relationship. (Contoh: *One-to-one*, *one-to-many*, *many-to-many*).

d. ***Dependency*** kadangkala sebuah *class* menggunakan *class* yang lain. Hal ini disebut dependency. Umumnya penggunaan *dependency* digunakan untuk menunjukkan operasi pada suatu *class* yang menggunakan *class* yang lain. Sebuah *dependency* dilambangkan sebagai sebuah panah bertitik-titik.

e. ***Aggregation*** mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut sebagai relasi.

4. Sequence Diagram

> "Diagram sekuen menggambarkan kelakuan objek pada *user case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case"*(Sukamto & Salahuddin, 2017).

"*Sequence* diagram menggambarkan bagaimana sistem merespon kegiatan *user*. *Sequence* diagram yang dibuat yaitu yang berhubungan langsung dengan kegiatan utama dari sistem informasi anggaran pendapatan dan belanja berbasis objek"(Irmayani & Susyatih, 2017).

Jadi, dari penjabaran diatas dapat disimpulkan bahwa *sequence* diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*.