SISTEM MANAJEMEN KASIR PADA COFFE SHOP HANASHI DAN STOK BARANG

PROPOSAL TUGAS AKHIR



DISUSUN OLEH:

[NAMA MHS]

[NIM MHS]

PROGRAM STUDI S1-TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

FAKULTAS TEKNOLOGI INFORMASI DAN KOMUNIKASI

UNIVERSITAS SEMARANG

2021

PERSETUJUAN

PROPOSAL TUGAS AHIR

DENGAN JUDUL

SISTEM MANAJEMEN KASIR PADA COFFE SHOP HANASHI DAN STOK BARANG

NAMA : [NAMA MHS]

NIM : [NIM MHS]

TELAH DISETUJUI

TANGGAL XXX

GANJIL Thn AKADEMIK 2021/2022

KOORDINATOR TUGAS AKHIR PEMBIMBING TUGAS AKHIR

S-1 TEKNIK INFORMATIKA

Whisnumurti Adhiwibowo, M.Kom April Firman Daru, S.Kom., M.Kom

NIS. 06557003102137 NIS. 06557003102133

KETUA PROGRAM STUDI

S-1 TEKNIK INFORMATIKA

Khoirudin, S.Kom, M.Eng

NIS. 06557003102173

**ABSTRAK**

Coffe shop hanashi adalah sebuah usaha dagang yang bergerak dalam bidang cafe. Pada saat ini, Coffe shop hanashi masih menggunakan sistem manual seperti pencatatan penjualan dan pencatatan persediaan barang sehingga hal ini menunjukkan bahwa sistem di Coffe shop hanashi kurang efektif karena disinyalir sering terjadi kesalahan pencatatan penjualan dan pencatatan persediaan barang yang mengakibatkan kesalahan informasi dan laporan keuangan.

Berdasarkan permasalahan tersebut, maka dibutuhkan solusi untuk menangani permasalahan atau kendala yang ada. Solusi yang ditawarkan adalah pembuatan Aplikasi Sistem menejemen kasir Café dan pengelolaan stok barang, hal ini di lakukan untuk pencatatan menjadi lebih efekttif, efisien dan lebih akurat.

Hasil penelitian aplikasi ini adalah sistem dapat mengelola penjualan, dan stok barang. Aplikasi ini juga dapat membuat laporan secara akurat sehingga memudahkan Owner Cafe dalam mengelola Cafe.

**Keyword :** *Sistem management kasir, sistem stok barang.*

PEMBIMBING TUGAS AKHIR

April Firman Daru, S.Kom., M.Kom

NIS. 06557003102133

1. **Latar Belakang**

Perkembangan teknologi yang semakin pesat, khususnya teknologi informasi, dapat mempermudah perusahaan untuk melakukan aktivitas sebagai sarana pendukung produktivitas. Proses ini mengubah sistem yang semula masih manual menjadi terkomputerisasi salah satunya adalah sistem manajament kasir dan stok barang pada *Coffe shop* hanashi.

*Coffe shop* hanashi adalah sebuah usaha dagang yang bergerak dalam bidang cafe. Pada saat ini, *Coffe shop* hanashi masih menggunakan sistem manual seperti pencatatan penjualan dan pencatatan barang sehingga hal ini menunjukkan bahwa sistem di *Coffe shop* hanashi kurang efektif karena disinyalir sering terjadi kesalahan pencatatan penjualan dan menurunkan daya saing dalam industri pasar.

Masalah tidak efisiennya sistem manual, mungkin bisa terselesaikan dengan adanya sistem managemen berbasis web yang menjadi pilihan untuk dirancang dan dibuat dalam menyelesaikan masalah *Coffe shop* hanashi. Dibentuklah “SISTEM MANAJEMEN KASIR PADA COFFE SHOP HANASHI DAN STOK BARANG” adalah sistem yang akan dibangun dengan fungsi untuk mengelola penjualan dan stok barang di *coffe* *shop* hanshi secara komputerisasi, akurat, efisien.

* 1. **Perumusan Masalah**

Berdasarkan uraian latar belakang permasalahan diatas, maka dapat dirumuskan permasalahan yang ada, yaitu bagaimana membangun Sistem menejemen kasir dan stok barang pada *coffe* *shop* hanashi Berbasis *Website*.

**C. Batasan Masalah**

Adapun batasan masalah dalam Sistem ini adalah:

1. Aplikasi berbasi website, yang artinya aplikasi ini hanya berjalan di web browser seperti *edge*, *firefox* dan *chrome*.
2. Penyimpanan data menggunakan MySQL.
3. Aplikasi ini tidak mempunyai sistem antrian pelanggan.
4. Aplikasi ini tidak melayani return atau batal pemesanan.
5. Aplikasi ini hanya berfokus pada bagian pengelolaan kasir dan persediaan brang.

**D. Tujuan Tugas Akhir**

Tujuan dari pembangunan Sistem ini adalah sebagai berikut :

1. Menganalisis kebutuhan yang diperlukan untuk Sistem *Management* kasir dan persediaan barang berbasis *website*
2. Merancang dan membangun Sistem Management kasir dan persediaan barang berbasis *website* yang mampu mempermudah proses pengelolaan secara efektif dan akurat.
3. Melakukan pengujian dan mengimplementasikan Sistem kasir dan persediaan barang berbasis *website* pada admin dan pemilik cafe.

**E. Manfaat Tugas Akhir**

1. Bagi Penulis
2. Penulis dapat menerapkan dan mengembangkan apa yang telah penulis terima tentang teknologi informasi selama belajar di Universitas Semarang.
3. Penulis menjadi mampu untuk membuat suatu sistem yang interaktif.
4. Bagi coffe shop hanashi
5. Membantu dalam proses pengolahan data yang lebih cepat, tepat dan efisien waktu.
6. Sebagai masukan dan solusi yang dapat digunakan dalam pembangunan sistem di kemudian hari.
7. Bagi Akademik
8. Sebagai tolak ukur kemampuan mahasiswa dalam menguasai materi yang diajarkan selama perkuliahan.
9. Dapat digunakan sebagai kajian dan referensi pada perpustakaan serta dapat digunakan oleh pihak-pihak tertentu.

**F. Tinjauan Pustaka**

1. **Pengertian Sistem**

Secara etimologi, Sistem berasal dari bahasa Latin (systcma) dan bahasa Yunani (sustcma) yang berarti suatu kesatuan yang terdiri komponen atau elemen yang dihubungkan bersama untuk memudahkan aliran informasi, materi atau energi untuk mencapai tujuan (A. S. Nugroho, 2017).

(Jogiyanto, 2018) mendefinisikan sistem sebagai gabungan dari berbagai elemen yang berhubungan dan berinteraksi untuk mencapai tujuan tertentu. system ini menggambarkan kejadian-kejadian dan kesatuan objek nyata, misalnya tempat, benda, dan orang-orang yang benar-benar ada dan nyata. Sehingga secara umum sistem diartikan sebagai suatu kumpulan objek atau unsur-unsur atau bagian-bagian yang memiliki arti berbeda-beda yang saling memiliki hubungan, saling berkerjasama dan saling mempengaruhi satu sama lain serta memiliki keterikatan pada rencana yang sama dalam mencapai suatu tujuan tertentu pada lingkungan yang kompleks.

1. **Pengertian Managemen**

Managemen disebut sebagai seni untuk merealisasikan pekerjaan melalui orang lain dalam kaitannya dengan organisasi bahwa manager dalam mencapai tujuan organisasi dengan cara pengaturan orang lain (pegawai). Manajemen mencakup fungsi perencanaan, pengorganisasian, penyusunan personalia, pengarahan dan pengawasan (I Heryanto, 2016).

Menurut George R. Terry (Ahmad, 2018) manajemen merupakan proses yang khas yang terdiri dari tindakan-tindakan perencanaan, pengorganisasian, penggerakkan, dan pengawasan untuk mencapai tujuan yang telah ditetapkan memalui pemanfaatan sumber daya manusia dan sumber daya lainnya Definisi lain Stoner AF (Ahmad, 2018) menyatakan bahwa manajemen merupakan proses perencanaan, pengorganisasian dan pengawasan antar anggota organisasi dengan menggunakan seluruh sumber

1. **Pengertian Sistem Managemen**

Definisi Sistem Managemen menurut (Ahmad, 2018) sistem managemen adalah jaringan prosedur pengolahan data yang dikembangkan dalam organisasi dan disatukan apabila di pandang perlu, dengan maksud memberikan data kepada manajemen setiap waktu diperlukan, baik data yang bersifat intern maupun yang bersifat ekstern, untuk dasar pengambilan keputusan dalam rangka mencapai tujuan organisasi.

1. **Café**

Cafe atau *Coffee Shop* (kedai kopi) adalah suatu tempat (kedai) yang menyajikan olahan kopiespreso dan kudapan kecil. Seiring perkembangan jaman coffe shop menyediakan makan kecil dan makanan berat. (Sumber: Kamus Besar Bahasa Indonesia, Departement Pendidikan dan Kebudayaan,Jakarta: Balai Pustaka, 1988)

1. **Pengertian *PHP***

Pada awalnya PHP merupakan kependekan dari *Personal Home Page*. PHP pertama kali dibua oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya masih berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari web. Selanjutnya Rasmus merilis kode sumber untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP. PHP memiliki 8 (delapan) tipe data, yaitu (Abdulloh, 2018) :

1. *Integer*

Tipe ini meliputi semua bilangan bulat dengan range -2,147,483,648 sampai +2,147,483,647 pada *platform* 32bit. PHP juga akan mengkonversi secara otomatis bila suatu bilangan berada diluar *range* tersebut ke dalam tipe data *floating point*. Tipe ini juga dapat dinyatakan dalam bentuk oktal (berbasis 8), desimal (berbasis 10), heksadesimal (berbasis 16).

1. *Floating Point*

Tipe ini biasa digunakan dalam bilangan pecahan namun bisa juga bilangan desimal. Tipe ini memiliki range 1.7E-308 sampai 1.7E+308. Dapat dinyatakan dalam bentuk bilangan desimal atau dalam bentuk pangkat.

1. *String*

Tipe data string dinyatakan dengan mengapitnya menggunakan tanda petik tunggal (‘ ‘) atau tanda petik ganda (” “). Perbedaan dari penggunaan keduanya adalah dengan tanda petik tunggal kita tidak dapat menggunakan variable dan *escape sequence handling* bersama dalam suatu kalimat.

1. Boolean

Tipe data boolean digunakan untuk menyimpan nilai *true* atau *false.* Biasanya tipe data ini mayoritas digunakan untuk melakukan pengecekan kondisi pada php.

1. Null

Tipe data yang tidak memuat apapun. Setiap Variabel yang diset menjadi Tipe Data NULL ini akan menjadikan Variabel tersebut kosong.

1. *Array*

Tipe ini dapat mengandung satu atau lebih data juga dapat diindeks berdasarkan numerik atau string. Ia juga mendukung multiarray dimensi dan membolehkan semua datanya berbeda tipe data.

1. Object

Tipe data object dapat berupa bilangan, variable atau fungsi. Object dibuat dengan tujuan agar para programmer terbiasa dengan OOP, meski fasilitas ini masih minim.

1. *Resource*

Tipe Data Spesial yang satu ini di khususkan untuk menyimpan resource, sumber atau alamat. Variabel tersebut hanya dapat diciptakan oleh suatu fungsi khusus yang mengembalikan nilai berupa resource seperti penggunaan fungsi fopen, opendir, mysql\_connect, mysql\_query dan semacamnya.

Beberapa kelebihan PHP dari bahasa pemrograman *web*, antara lain :

1. Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaannya.
2. *Web* Server yang mendukung PHP dapat ditemukan dimana-mana, mulai dari apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis-milis dan developer yang siap membantu dalam pengembangan.
4. Dalam sisi pemahaman, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
5. PHP adalah bahasa *open source* yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.
6. ***MySQL***

MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersil. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basis data, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. MySQL memiliki beberapa keistimewaan, antara lain (B. Nugroho, 2019):

* + 1. *Portabilitas.*

MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Max Os X Server, Solaris, Amiga, dan masih banyak lagi.

* + 1. Perangkat lunak sumber terbuka.

MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.

* + 1. *Multi-user*.

MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah.

* + 1. *Performance-tuning*.

MySQL memiliki kecepatan yang menakjubkan dalam menangani *query* sederhana, dengan kata lain dapat memproses lebih banyak SQL persatuan waktu.

* + 1. Ragam tipe data.

MySQL memiliki ragam tipe data yang sangat kaya, seperti signed/unsigned integer, float, double, char, text, date, timestamp, dan lain-lain.

* + 1. Perintah dan fungsi.

MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah *select* dan *where* dalam perintah (*query*).

* + 1. Keamanan.

MySQL memiliki beberapa lapisan keamanan seperti level *subnetmask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail serta sandi terenskripsi.

* + 1. Skalabilitas dan pembatasan.

MySQL mampu menangani basis data secara besar, dengan jumlah rekaman (*records*) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.

* + 1. Konektivitas.

MySQL dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, UNIX soket (UNIX), atau Named Pipes (NT).

* + 1. Lokalisasi.

MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meskipun demikian, Bahasa Indonesia belum termasuk di dalamnya.

* + 1. Antar muka.

MySQL memiliki antar muka (*interface*) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (*Application Programming Interface*).

* + 1. Klien dan peralatan

MySQL dilengkapi dengan berbagai peralatan (*tool*) yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk *online*.

* + 1. Struktur tabel

MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan dengan basis data lainnya semacam PostgreSQL ataupun Oracle.

1. ***XAMPP***

Menurut (Riyanto, 2014), *XAMPP* adalah sebuah *software* yang berfungsi untuk menjalankan *website* berbasis *PHP* dan menggunakan pengolah data *MySQL* di komputer secara local. *XAMPP* berperan sebagai *website* *server* pada komputer. *XAMPP* juga dapat disebut sebuah CPanel *server* *virtual*, yang dapat membantu anda melakukan preview sehingga dapat memodifikasi *website* tanpa harus online atau ter akses dengan internet.

*Software* *XAMPP* bersifat *open sources* yang dapat diperoleh secara gratis dari situs [www.apachefriends.org](http://www.apachefriends.org). *XAMPP* adalah perangkat lunak yang mendukung banyak sistem operasi dan merupakan komplikasi dari beberapa program. Fungsinya adalah sebagai *server* yang berdiri sendiri dan terdiri atas *Apache, MySQL*, dan bahasa pemrograman *PHP*.

1. ***Unified Modeling Language* (*UML*)**

Pada perkembangan teknik pemograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemograman berorientasi objek yaitu *Unified Modelling Language*(UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung (Muslihudin, 2016).

UML hanya berfungsi sebagai untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek.

1. ***Use case Diagram***

*Use case diagram* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu (Muslihudin, 2016).

*Use case diagram* dapat sangat membantu bila sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem. *Sebuah* *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain. Berikut adalah simbol - simbol *Use Case Diagram*.

Tabel 1 Simbol *Use case Diagram* (Muslihudin, 2016)

|  |  |  |
| --- | --- | --- |
| **SIMBOL** | **NAMA** | **KETERANGAN** |
|  | *Actor* | Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*. |
|  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*). |
|  | *Generalization* | Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
|  | *Include* | Menspesifikasikan bahwa *use case* sumber secara *eksplisit*. |
|  | *Extend* | Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan. |
|  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
|  | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas. |
|  | *Use Case* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor |
|  | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi). |

(Lanjutan) Tabel 1 Simbol *Use case Diagram* (Muslihudin, 2016)

|  |  |  |
| --- | --- | --- |
| **SIMBOL** | **NAMA** | **KETERANGAN** |
|  | *Note* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi |

1. ***Activity Diagram***

*Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi (Muslihudin, 2016).

*Activity diagram* merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. Berikut adalah simbol - simbol *Activity Diagram*.

Tabel 2 Simbol *Activity Diagram* (Muslihudin, 2016)

|  |  |  |
| --- | --- | --- |
| **SIMBOL** | **NAMA** | **KETERANGAN** |
|  | State | Kondisi yang mungkin dialami oleh suatu obyek. |
|  | Note | Note digunakan untuk memberikan keterangan ataukomentar |
|  | Aktivitas | Perilaku obyek yang dilakukan saat obyek berada dalam *state* tertentu. |
|  | Start State | *Start state* digunakan untuk memulai diagram *statechart.* |
|  | End State | End start digunakan untuk mengakhiri diagram. |
|  | Decision | *Decision* digunakan sebagai pilihan untuk pengambilan keputusan. |
|  | Penggabungan / Join | Digunakan untuk split dan join.pada saat diagram akanmembagi 2, bar ini akanditambahkan. dan sebelum diagram digabung menjadi satu, sebagai join. |
|  | Asosiasi *(association)* | Apa yang menghubungkan antara objek satu dengan objek yang lainnya. |

1. ***Class Diagram***

*Class* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain (Muslihudin, 2016). *Class* memiliki tiga area pokok :

1. Nama (dan stereotype)
2. Atribut
3. Metoda

*Class* dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*. Hubunganantar *class* yaitu:

* 1. Asosiasi

Hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah *query* antar class.

* 1. Agregasi

Hubungan yang menyatakan bagian (“terdiri atas..”).

* 1. Pewarisan

Hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.

* 1. Hubungan Dinamis

Rangkaian pesan (*message*) yang di-*passing* dari satu *class* kepada *class* lain.

Tabel 3. Simbol *Class Diagram* (Muslihudin, 2016)

|  |  |  |
| --- | --- | --- |
| **SIMBOL** | **NAMA** | **KETERANGAN** |
|  | *Generalization* | Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
|  | *Nary Association* | Upaya untuk menghindari asosiasi dengan > 2 objek. |
|  | *Class* | Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama. |
|  | *Collaboration* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor |
|  | *Realization* | Operasi yang benar-benar dilakukan oleh suatu objek. |
|  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri |
|  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya |

1. ***Sequence Diagram***

*Sequence diagram* menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan (Muslihudin, 2016).

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*. Berikut adalah simbol - simbol *Sequence Diagram*.

Tabel 4. Simbol *Sequence Diagram* (Muslihudin, 2016)

|  |  |  |
| --- | --- | --- |
| **SIMBOL** | **NAMA** | **KETERANGAN** |
|  | *LifeLine* | Menggambarkan objek *entity*, antarmuka yang saling berinteraksi. |
|  | *Message* | Menggambarkan pengiriman pesan |
|  | *Actor* | Menggambarkan orang yang sedang berinteraksi dengan sistem |
|  | *Entity Class* | Menggambarkan hubungan kegiatan yang akan dilakukan |
|  | *Boundary Class* | Menggambarkan sebuah penggambaran dari form |
|  | *Control Class* | Menggambarkan penghubung antara *boundary* dengan tabel |

**G. Metodologi Penelitia**

Untuk melaksanakan penelitian diperlukan metode penelitian yang tepat. Adapun metodologi yang diterapkan memiliki beberapa tahap berikut:

1. Metode Pengumpulan Data
2. Pengamatan (Observasi)

Pengamatan langsung terhadap alur pengelolan yang dilakukan admin dan pemilik cafe untuk memperoleh informasi yang nantinya akan diolah kedalam sistem management.

1. Wawancara (*Interview*)

Penulis mengumpulkan data dengan cara melakukan wawancara dengan owner coffe shop.

1. Studi Pustaka

Mengumpulkan data dan informasi dengan cara mencari, menganalisa dan menyimpulkan dari berbagai buku bacaan lain yang berhubungan dengan objek penelitian.

1. Sumber Data

Sumber data yang digunakan antara lain:

1. Data Primer

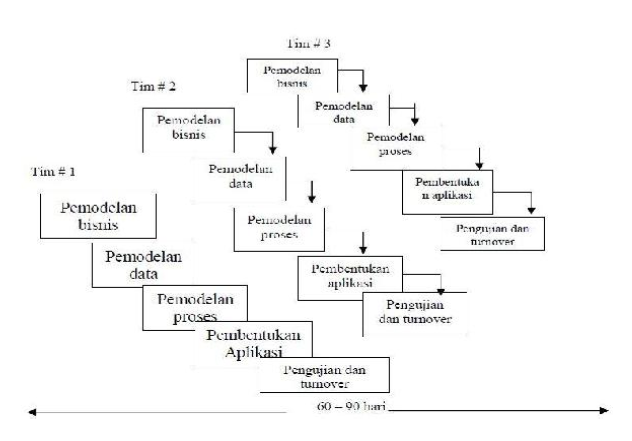
Data Primer adalah data-data yang didapat secara langsung dari sumber yang menjadi objek penelitian. Data Primer diperoleh dengan cara observasi dan wawancara langsung terhadap pemilik cafe.

1. Data Sekunder

Data Sekunder adalah data yang diperoleh secara tidak langsung dari sumber yang dijadikan objek penelitian dan digunakan sebagai pendukung data primer.

1. Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan adalah metode RAD *(Rapid Application Development).*



Gambar 1 Model RAD (Rosa A.S, & Shalahudin, 2018)

Berdasarkan Gambar 1 di atas, berikut penjelasan dari fase dan tahapan dari metode RAD , terdapat lima fase dalam RAD yang melibatkan penganalisis dan pengguna dalam tahap penilaian, perancangan, dan penerapan. Adapun kelima fase tersebut adalah Pemodelan bisnis, Pemodelan data, Pemodelan proses, Pembuatan aplikasi,Pengujian dan pergantian.Sesuai dengan metodologi RAD, berikut ini adalah tahap-tahap pengembangan aplikasi dari tiap-tiap fase pengembangan aplikasi.

1. Pemodelan Bisnis

Pemodelan yang di lakukan untuk memodelkan fungsi bisnis untuk mengetahui informasi apa yang terkait dengan proses bisnis, informasi apa yang harus dibuat, siapa yang harus membuat informasi itu, bagaimana alur informasi itu, proses apa saja yang terkait informasi itu.

1. Pemodelan Data

Memodelkan data apa saja yang dibutuhkan berdasarkan pemodelan bisnis dan mendefinisikan atribut-atributnya beserta relasinya dengan data-data yang lain.

1. Pemodelan Proses

Mengimplementasikan fungsi bisnis yang sudah didefinisikan terkait dengan pendefinisian data. Pada tahap ini, penulis melakukan gambaran sebuah rancangan sistem serta desain sistem guna mendapatkan persetujuan user untuk menilai apakah masih ada kekurangan yang masih belum sesuai kebutuhan perusahaan dengan bantuan Diagram UML.

1. Pembuatan Aplikasi

Mengimplementasikan pemodelan proses dan data menjadi program.

1. Pengujian dan Pergantian

Menguji komponen-komponen yang dibuat. Jika sudah teruji maka tim pengembang komponen dapat mengembangkan komponen berikutnya. Pada tahap ini, setelah penulis melakukan coding dan sistem telah selesai dibuat maka di lakukan pengujian. Pengujian sistem yang di lakukan adalah *black box* dan *white box*.

**DAFTAR PUSTAKA**

Abdulloh, R. (2018). *7 in 1 Pemrograman Web Untuk Pemula*. Elex Media Komputindo.

Ahmad, L. (2018). *Sistem Informasi Manajemen: Buku Referensi: Sistem Informasi Manajemen* (Vol. 1). KITA Publisher.

I Heryanto, T. T. (2016). *Manajemen Proyek Berbasis Teknologi Informasi*. Informatika Bandung.

Jogiyanto, H. . (2018). *Sistem Informasi Manajemen (Edisi 2)*. Universitas Terbuka.

Muslihudin, M. (2016). *Analisis Dan Perancangan Sistem Informasi Menggunakan Model Terstruktur Dan UML*. Penerbit Andi.

Nugroho, A. S. (2017). *Analisis dan Perancangan Sistem Informasi*. Trans Tekno. http://103.255.15.77/detail-opac?id=300171

Nugroho, B. (2019). *Aplikasi Pemrograman WEB Dinamis dengan PHP dan MySQL ( Studi Kasus Membuat Sistem Informasi Pengola*. Gava Media. https://www.gavamedia.net/produk-462-aplikasi-pemrograman-web-dinamis-dengan-php-dan-mysql--studi-kasus-membuat-sistem-informasi-pengola.html

Riyanto. (2014). *Membuat Aplikasi Mini Market Integrasi Barcode Reader dengan PHP & MySQL*. Gava Media.

Rosa A.S, & Shalahudin, M. (2018). *Rekayasa Perangkat Lunak (Terstruktur dan Beorientasi Objek)*. Informatika Bandung.