

## **BAB II**

### **TINJAUAN PUSTAKA DAN LANDASAN TEORI**

#### **2.1 Tinjauan Pustaka**

Penelitian sebelumnya tahun 2017 dengan judul Sistem Informasi Pemesanan dan Produksi pada Revolver Konveksi Berbasis Web”. Revolver konveksi dalam proses transaksinya masih menggunakan media kertas untuk menulis semua transaksi pemesanan sehingga membuat proses pengolahan data pesanan membutuhkan waktu yang lama, dari segi penyimpanan data juga masih dicatat dalam buku sehingga memiliki banyak resiko kehilangan data. Dengan sistem informasi pemesanan dan produksi berbasis web diharapkan dapat untuk mempermudah pengelolaan data yang ter integrasi dalam database serta mendukung proses bisnis yang ada[2].

Penelitian sebelumnya tahun 2019 dengan judul “Sistem Informasi Pemesanan pada bidang usaha jasa konveksi berbasis web” . Pada bidang jasa khususnya jasa konveksi, yang masih mengandalkan layanan secara konvensional memiliki keterbatasan dalam hal coverage area layanan dan secara otomatis membatasi jumlah pelanggan yang mengakses layanan jasa[3].

Penelitian sebelumnya tahun 2020 dengan judul “Pengembangan Sistem Informasi Perusahaan Konveksi dan Sablon Berbasis Website Menggunakan Metode Waterfall” . Dalam penelitian tersebut hal yang menjadi salah satu kendala dalam proses pemesanan karena dapat membatasi konsumen yang tertarik pada konveksi dan sablonan yang ditawarkan terutama pada konsumen yang bertempat tinggal jauh dari lokasi perusahaan. Selain itu proses pencatatan setelah pemesanan atau laporan dari hasil pemesanan dan penjualannya yang kurang efektif karena masih menggunakan sistem konvensional[4].

Penelitian saya kali ini yang dilakukan adalah membuat sebuah sistem informasi pemesanan dan penjualan di Toko Cahaya Agung berbasis *website*. Perbedaan penelitian yang dilakukan dengan penelitian

sebelumnya adalah adanya fitur *custom* produk dimana pelanggan dapat melakukan pemesanan produk sesuai keinginan dengan memasukkan data pemesanan dan mengunggah desain gambar yang ingin dipesan di dalam sistem tersebut.

## **2.2 Landasan Teori**

Landasan teori berisi hal-hal atau teori-teori yang berkaitan dengan permasalahan dan ruang lingkup permasalahan sebagai landasan dalam pembuatan laporan ini.

### **2.2.1 Sistem Informasi**

Sistem informasi merupakan suatu kombinasi teratur dari orang-orang, hardware, software, jaringan komunikasi dan sumber daya data yang mengumpulkan, mengubah, dan menyebarkan informasi di dalam sebuah organisasi[5]. Fungsi Sistem Informasi :

1. Memperbaiki produktivitas dari suatu aplikasi pengembangan dan pemeliharaan dari sistem.
2. Menjamin ketersediaan kualitas dan keterampilan dalam memanfaatkan sistem informasi secara kritis.
3. Mengidentifikasi kebutuhan mengenai keterampilan dari pendukung suatu sistem informasi.

Sistem Informasi terdiri dari beberapa komponen yang masing-masing saling berhubungan satu sama lain dan membentuk satu kesatuan untuk mencapai tujuan. Komponen-komponen sistem informasi antara lain :

1. Komponen *Input*, merupakan data yang masuk ke dalam suatu sistem informasi
2. Komponen model, merupakan kombinasi prosedur, logika dan suatu model matematika yang memproses data yang tersimpan.
3. Komponen *output*, merupakan suatu hasil informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.
4. Komponen teknologi, merupakan alat di dalam suatu sistem informasi, teknologi digunakan untuk menerima *input*,

menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan *output* dan memantau pengendalian sistem.

5. Komponen basis data, merupakan kumpulan dari data yang saling berhubungan yang tersimpan di dalam komputer dengan menggunakan aplikasi *database*.
6. Komponen kontrol, merupakan komponen yang mengendalikan atau mengontrol apabila ada gangguan terhadap sistem.

### 2.2.2 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak adalah sebuah profesi yang dilakukan oleh seorang perancang perangkat lunak yang berkaitan dengan pembuatan dan pemeliharaan aplikasi perangkat lunak dengan menerapkan teknologi dan praktik dari ilmu computer, manajemen proyek dan bidang-bidang lainnya[6].

Pada rekayasa perangkat lunak ada beberapa model yang digunakan dalam pengembangan sistem. Salah satu model yang digunakan dalam pengembangan sistem adalah model waterfall. Model waterfall terdiri dari 5 tahapan, yaitu Analisa Kebutuhan, Desain Sistem, Penulisan Kode Program, Pengujian Program, Penerapan Program dan Pemeliharaan.

Berikut penjelasan tahapan-tahapan dari model *waterfall*[7] :

#### 1. Analisis Kebutuhan

Pada tahap ini merupakan proses melakukan analisa terhadap kebutuhan sistem dengan cara pengumpulan data. Pengumpulan data pada tahap ini dapat dilakukan dengan melakukan sebuah wawancara, atau *study* literatur. Dalam tahapan tersebut seorang Sistem analis akan menggali informasi sebanyak-banyaknya dari *user* mengenai sistem yang diinginkan yang akan menghasilkan dokumen *user requirement*.

#### 2. Desain Sistem

Proses desain sistem akan menerjemahkan syarat kebutuhan ke sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat *coding*. Tahapan dalam pembuatan desain sistem adalah dengan menerjemahkan data dalam bentuk UML

yang meliputi di antaranya *Use Case*, *Sequence Diagram*, *Entity Relationship Diagram*.serta dengan membuat desain sistem terinci yaitu Rancangan Antarmuka.

3. Penulisan Kode Program

*Coding* merupakan penerjemah *design* dalam bahasa yang bisa dikenali oleh komputer. Tahapan ini dilakukan dengan menuliskan kode program/ *script* ke dalam bahasa pemrograman PHP dan database MySQL.

4. Pengujian Sistem

Tahapan ini bisa dikatakan *final* dalam pembuatan sebuah sistem. Setelah melakukan analisa, *design* dan pengkodean maka akan dilakukan pengujian, ada 3 jenis pengujian yang akan dilakukan meliputi pengujian *black-box*, pengujian fungsionalitas fitur sistem, pengujian kuesioner.

5. Penerapan Sistem dan Pemeliharaan






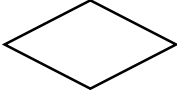

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Pada tahap ini dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

Tahapan rancangan Rekayasa Perangkat Lunak menggunakan Tools sebagai berikut :

**A. Flowchart**

*Flowchart* adalah bagan-bagan yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah. *Flowchart* merupakan cara penyajian dari suatu algoritma[8]. Simbol *flowchart* dapat dilihat pada Tabel 2.1

Tabel 2. 1 Simbol *Flowchart*


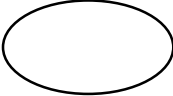
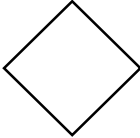

SIMBOL	NAMA DAN FUNGSI
	<i>Terminator</i> digunakan pada permulaan/akhir program
	<i>Flow Line</i> atau Garis alir digunakan untuk menunjukkan arah aliran program
	<i>Process</i> digunakan pada proses perhitungan/proses pengolahan data
	<i>Input/output data</i> ,digunakan pada proses memasukan/mengeluarkan data parameter,Informasi
	<i>Manual Operation</i> digunakan untuk Operasi Manual
	<i>Decision</i> sebagai Perbandingan pernyataan, penyeleksian data yang memberikan pilihan untuk langkah selanjutnya
	<i>Document</i> digunakan untuk hasil <i>Input/Output</i> dalam format yang dicetak

### B. Entity Relational Diagram (ERD)

*Entity Relational Diagram* merupakan tools yang digunakan untuk memodelkan struktur data dengan menggambarkan entitas (*relationship*) secara abstrak (konseptual) [9].

Berikut adalah simbol dari ERD dan fungsinya dapat dilihat pada Tabel 2.2 .

**Tabel 2. 2** Simbol ERD

No	Simbol	Nama dan Fungsinya
1.		<i>Entity</i> , digunakan untuk menggambarkan objek yang diidentifikasi ke dalam lingkungan
2.		<i>Atribut</i> , digunakan untuk menggambarkan elemen-elemen dari suatu entity, yang menggambarkan karakter <i>entity</i>
3.		Hubungan ( <i>relasi</i> ), <i>Entity</i> dapat berhubungan satu sama lain. Hubungan ini disebut juga dengan <i>relationship</i>
4.		Garis ( <i>line</i> ), digunakan untuk menghubungkan <i>entity</i> dengan <i>relasi</i> /hubungan, maupun <i>entity</i> dengan atribut

### 2.2.3 Pemrograman Berorientasi Objek

Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis.

Adapun keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut :

1. Meningkatkan produktifitas.  
Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai diulang kembali untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).
2. Kecepatan pengembangan  
Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean.
3. Kemudahan pemeliharaan  
Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola mungkin sering berubah ubah.
4. Adanya konsistensi  
Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.
5. Meningkatkan kualitas perangkat lunak  
Karena pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Berikut ini adalah beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek [10]:

1. Kelas (*class*)  
Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama yang mungkin lahir atau diciptakan dari kelas tersebut..

2. **Objek (*object*)**

Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, suatu organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan.
3. **Metode (*method*)**

Operasi atau metode atau *method* pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri. Operasi atau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek.
4. **Atribut (*attribute*)**

Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya. Atribut sebaliknya bersifat privat untuk menjaga konsep enkapsulasi.
5. **Abstraksi (*abstraktion*)**

Prinsip untuk mempresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.
6. **Enkapsulasi (*encapsulation*)**

Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.
7. **Pewarisan (*inheritance*)**

Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.



8. *Antarmuka (interface)*  
Antarmuka atau *interface* sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah *interface* dapat diimplementasikan oleh kelas lain.
9. *Reusability*  
Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.
10. *Generalisasi dan Spesifikasi*  
Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus.
11. *Komunikasi Antar objek*  
Komunikasi antar objek dilakukan lewat pesan (*message*) yang dikirimkan dan satu objek ke objek lainnya.
12. *Polimorfisme (polymorphism)*  
Kemampuan suatu objek yang digunakan di banyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.
13. *Package*  
*Package* adalah sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga bernama sama disimpan dalam *package* yang berbeda.

Pada pemrograman berorientasi objek, UML digunakan untuk pemodelan sistem. UML adalah bahasa untuk menspesifikasi, memvisualisasi, membangun dan mendokumentasikan artifacts (bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, artifact tersebut dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya [10]. Macam-macam dari *Unified Modeling Language* (UML) antara lain: *use case diagram*, *sequence diagram*.

### a. Use Case Diagram

*Use Case* adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use Case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem tersebut *scenario* sedangkan pengguna disebut *actor*. *Actor* adalah sebuah peran yang biasa dimainkan oleh pengguna dalam interaksinya dengan sistem. Model *use case* adalah bagian dari model *requirement*. Definisi lain *use case* adalah abstraksi dari interaksi antara sistem dan *actor*. *Use case* dibuat berdasarkan keperluan *actor*[11].


Berdasarkan definisi diatas maka dapat disimpulkan bahwa *Use Case* adalah kontruks untuk mendeskripsikan bagaimana sistem akan terlihat dimata pengguna potensial yang terdiri dari sekumpulan *scenario* dan *actor*. Sedangkan *use case* diagram memfasilitasi komunikasi diantara analis dan pengguna serta analis dan klien.


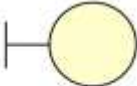



### b. Sequence Diagram

*Sequence Diagram* digunakan untuk menggambarkan perilaku pada sebuah *scenario*. Diagram ini menunjukkan sejumlah obyek dan *message* yang diletakan antara obyek-obyek didalam *use case*.

Komponen utama *sequence diagram* terdiri atas objek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical*. *Sequence Diagram* menambahkan dimensi waktu pada interaksi diantara obyek[11]. Simbol-simbol yang dipakai dalam pembuatan *sequence diagram* dapat dilihat pada Tabel 2.3

**Tabel 2. 3** Simbol *Sequence Diagram*

No	Simbol	Keterangan
1.		<i>Actor</i> , Menggambarkan orang yang berinteraksi dengan sistem

No.	Simbol	Keterangan
2.		<i>Entity Class</i> , Menggambarkan hubungan yang akan dilakukan
3.		<i>Boundary Class</i> , Menggambarkan sebuah gambaran dari sebuah <i>form</i>
4.		<i>Control</i> ,Menggambarkan penghubung antara <i>boundary</i> dengan tabel
5.		<i>Lifeline</i> , Menggambarkan tempat mulai dan berakhirnya <i>message</i>
6.		<i>Message entry</i> ,Menggambarkan pengiriman pesan/hubungan objek yang menunjukkan urutan yang terjadi

#### 2.2.4 Basis Data

Basis data adalah kumpulan file-file yang mempunyai kaitan antara satu file dengan file lain sehingga membentuk satu bangun data untuk menginformasikan suatu perusahaan instansi, dalam bahasan tertentu [12].

Salah satu komponen di dalam basis data adalah *Database Management System*(DBMS). DBMS adalah perangkat lunak yang

digunakan untuk mengelola dan mengontrol pengaksesan *database*, bahasa *database* umumnya terdiri dari berbagai macam instruksi yang diformulasikan sehingga instruksi tersebut dapat diproses oleh DBMS.

Dengan menggunakan perangkat lunak ini pengelolaan data menjadi lebih mudah dilakukan. Selain itu perangkat lunak ini juga menyediakan berbagai piranti yang berguna. Misalnya piranti yang memudahkan dalam membuat berbagai bentuk laporan.

Perintah atau instruksi tersebut umumnya ditentukan oleh user, adapun bahasa yang digunakan dibagi kedalam dua macam diantaranya sebagaimana di bawah ini:

a. *Data Definition Language (DDL)*

DDL atau kepanjangannya *Data Definition Language*, yaitu dipakai untuk menggambarkan desain dari basis data secara menyeluruh. Ini dipakai untuk membuat tabel baru, memuat indeks, maupun mengubah tabel. Hasil dari kompilasi DDL akan disimpan di kamus data. Itulah definisi dari DDL.

b. *Data Manipulation Language (DML)*

DML atau kepanjangannya *Data Manipulation Language*, yaitu dipakai untuk memanipulasi dan pengambilan data pada suatu basis data, misalnya seperti penambahan data yang baru ke dalam suatu basis data, menghapus data pada suatu basis data dan mengubah data pada suatu basis data.

### **2.2.5 Blackbox Testing**

Pengujian *black-box* merupakan pengujian yang berfokus pada persyaratan fungsional dari suatu perangkat lunak. Dengan demikian, pengujian *black-box* memungkinkan perekayasa perangkat lunak dengan mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program atau sistem. Pengujian *black-box* bukan merupakan alternatif dari teknik *white-box*.<sup>[5]</sup>

Pengujian *black-box* berusaha menemukan kesalahan dalam kategori sebagai berikut:

1. Fungsi-fungsi yang tidak benar atau hilang.
2. Kesalahan interface.
3. Kesalahan dalam struktur data atau akses database eksternal.

4. Kesalahan kinerja.
5. Inisialisasi dan kesalahan terminasi

Tidak seperti pengujian *white-box*, yang dilakukan pada saat awal proses pengujian, pengujian *black-box* cenderung diaplikasikan selama tahap akhir pengujian. Karena pengujian *black-box* memperhatikan struktur kontrol, maka perhatian berfokus pada domain informasi.

#### **2.2.6 Pemesanan Custom**

Pemesanan custom adalah pemesanan yang menyesuaikan dari keinginan, karakter atau fungsi tertentu. *custom* berasal dari kata “*Customized*” yang berarti menyesuaikan.[13]

Pemesanan secara *custom* merupakan suatu kegiatan transaksi jual beli suatu produk yang produknya akan disesuaikan dengan keinginan orang yang akan memesannya. Dalam hal ini pelanggan dapat dengan bebas membuat pesanan suatu produk sesuai keinginan tetapi tetap dengan adanya batasan-batasan yang diberikan oleh penyedia jasa. Batasan-batasan itu ada karena kreativitas dalam hal membuat sebuah produk di manusia itu sendiri tidak terbatas tetapi alat dan sumber daya sangatlah terbatas.

#### **2.2.7 Usaha Konfeksi**

Konfeksi merupakan salah satu bisnis di bidang pakaian yang biasanya memproduksi pakaian dalam partai besar sesuai permintaan[14]. Usaha konfeksi merupakan suatu jenis usaha dalam pembuatan pakaian atau kebutuhan sandang secara massal. Pada usaha konfeksi ini untuk pengerjaannya membutuhkan pekerja yang banyak serta dibutuhkan mesin-mesin yang besar untuk dapat mengerjakannya.

Perbedaan usaha konfeksi dengan usaha garmen ini yaitu menghasilkan produk yang berbeda. Pada usaha konfeksi produk yang dihasilkan dari bahan mentah, setengah jadi, sampai produk jadi. Dalam penelitian tempat yang dijadikan studi kasus yaitu Toko Cahaya Agung merupakan usaha konfeksi yang menghasilkan produk dari bahan mentah hingga menjadi produk jadi, namun selain kegiatan tersebut Toko Cahaya Agung juga memasarkan produk tersebut langsung ke pelanggan.

### **2.2.8 Pembayaran *Down Payment* (DP)**

Pembayaran down payment (DP) berasal dari bahasa Inggris, “*down payment is a prtial payment made at the time of purchase; the balanced to be paid later*” yaitu sebagian pembayaran yang dilakukan pada awal pembelian, sementara sisanya akan di bayar kemudian. Berapa lama waktu pembayaran ditentukan sesuai perjanjian diantara penjual dan pembeli[15].

Di dalam sistem ini pembayaran dapat dilakukan dengan metode pembayaran dp apabila memilih proses transaksi dengan pemesanan *custom* dan membayar uang muka/*down payment* berdasarkan harga yang telah ditentukan yaitu sebesar 50% agar pemesanan dapat diproses oleh pihak toko.

### **2.2.9 PHP Mailer**

PHP Mailer merupakan salah satu library yang disediakan dan bersifat free . Library ini dapat dijalankan di berbagai macam framework PHP, Library ini memungkinkan user dapat memanfaatkan *e-mail* sebagai media interaksi anatar user dengan sistem[16].

PHP Mailer adalah salah satu library PHP open source yang digunakan untuk mengirim *e-mail* dari localhost. PHP Mailer dapat menjalankan fungsinya sebagai *e-mail* jika kita mensupportnya dengan *Simple Mail Transfer Protocol* (SMTP). SMTP adalah suatu protocol yang diperlukan untuk mengirim dan menerima *e-mail*. Karena itulah kita harus menggunakan SMTP sebagai layanan mengirim *e-mail*. Layanan ini dapat digunakan untuk keperluan seperti memverifikasi *e-mail*, contohnya ketika mendaftarkan di Twitter atau Facebook. Setelah mendaftar maka kita diharuskan membuka *e-mail* dan memverifikasinya.

PHP Mailer dapat digunakan untuk verifikasi akun pelanggan yang telah melakukan pendaftaran , dan sebagai notifikasi untuk memberitahu kepada pelanggan informasi bahwa barang yang dipesan sedang dalam proses pengiriman.