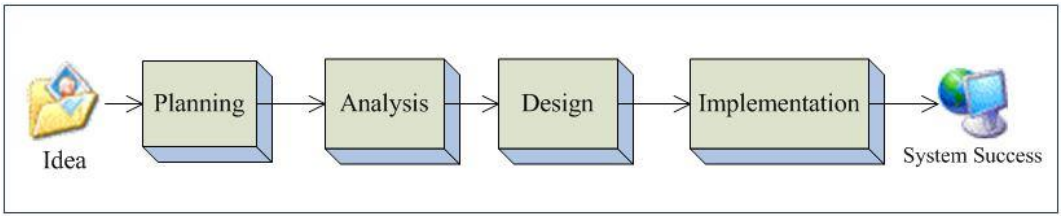
BAB III
METODOLOGI PENELITIAN

Penelitian dengan judul "Desain Web *Online Shop* Sebagai Pendukung Pemasaran Pada Ella *Collection* Menggunakan Metode *Rapid Application Development* (RAD)" ini menggunakan metodologi *System Development Life Cycle* (SDLC). Sedangkan metode yang digunakan adalah *Rapid Application Development* (RAD) dan alat bantu analisa dan desain sistem adalah *Unified Modelling Language* (UML). Berikut ini adalah penjelasan mengenai metodologi penelitian, metode penelitian dan alat bantu penelitian yang digunakan.

3.1 Metodologi *System Development Life Cycle* (SDLC)

System Development Life Cycle adalah sebuah tahapan penting dalam pengembangan sistem informasi yang menentukan bagaimana sistem yang dibangun dapat mendukung kebutuhan bisnis, merancang sistem, membangun sistem dan mengirimkannya kepada pengguna. Membangun sistem informasi menggunakan SDLC mengikuti empat fase, yaitu : perencanaan *(planning)*, analisis *(analysis)*, desain *(design)*, dan implementasi *(implementation)*. SDLC merupakan proses penyempurnaan bertahap. Hasil dari setiap fase digunakan untuk menyempurnakan pekerjaan yang dilakukan sebelumnya[12].
Empat fase tersebut dapat digambarkan sebagai berkut :



Gambar 3. 1 Fase-fase dalam SDLC[13]

Berikut ini adalah uraian dari empat fase dalam *System Development Life Cycle* (SDLC) [12] :

1. Perencanaan (*Planning*)

   Fase perencanaan adalah proses awal untuk memahami mengapa sistem informasi tersebut perlu dibangun dan menentukan bagaimana tim proyek akan bekerja. Pada tahap awal proyek dimulai, dilakukan analisa kelayakan dari beberapa aspek seperti :

   a. Kelayakan teknis (apakah sistem tersebut dapat kita bangun?)
   b. Kelayakan ekonomi (apakah sistem yang dibangun dapat memberikan nilai bisnis?)
   c. Kelayakan ekonomi (setelah sistem dibangun, apakah sistem tersebut akan digunakan?)

   Kemudian hasil dari analisis kelayakan yang memutuskan apakah proyek tersebut harus dijalankan. Setelah proyek disetujui, tahap selanjutnya adalah manajemen proyek. Manajer proyek bertugas membuat rencana kerja, staf proyek dan menerapkan teknik untuk mebantu tim proyek dalam mengendalikan dan mengarahkan proyek menggunakan SDLC.

2. Analisis (*Analysis*)

   Fase analisis digunakan untuk menjawab pertanyaan kebutuhan sistem, siapa yang akan menggunakan sistem, sistem tersebut dapat melakukan apa saja, dan dimana serta kapan sistem tersebut digunakan. Informasi tersebut dapat diperoleh melalui wawancara atau *quesioner* kepada pengguna yang akan menggunakan sistem agar sistem yang dibangun dapat sesuai dengan kebutuhan pengguna. Fase analisis merupakan analisis tingkat tinggi untuk menentukan desain awal sistem.

3. Desain (*Design*)

   Fase desain menetukan bagaimana nantinya sistem akan beroperasi dengan baik. Strategi desain harus ditentukan, ini menjelaskan apakah pengembangan sistem akan dilakukan oleh perusahaan itu sendiri, apakah akan dikembangkan oleh perusahaan lain, atau perusahaan tersebut akan membeli paket perangkat lunak yang telah tersedia. Selain itu *database* dan spesifikasi file perlu dikembangkan

untuk menjelaskan apakah data akan disimpan dan dimana data tersebut akan disimpan. Tim analisis mengembangkan desain program untuk mendefiniskan apa yang akan dilakukan setiap program.

4. Implementasi (*Implementation*)

   Ini adalah fase yang paling banyak mendapat perhatian, karena bagi kebanyakan orang, ini adalah bagian terpanjang dan termahal dari proses pengembangan.

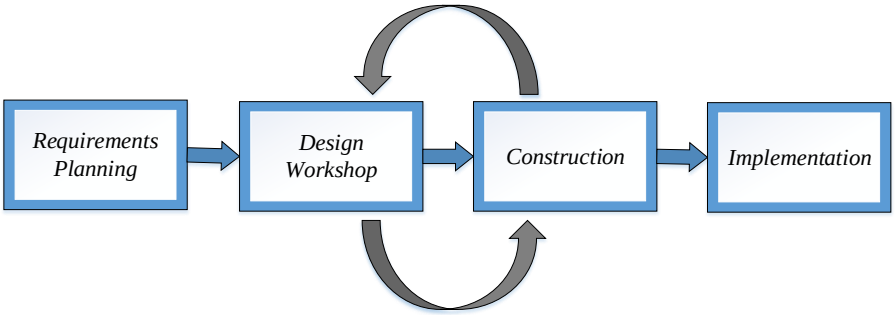
   Fase ini memiliki 3 tahap :

   a. Langkah pertama adalah pembangunan sistem. Sistem dibangun dan diuji untuk memastikan bahwa sistem tersebut berfungsi sesuai seperti yang dirancang. Karena biaya untuk memperbaiki *error* sangat banyak, maka pengujian adalah salah satu langkah penting yang diperlukan dalam fase implementasi. Sebagian besar perusahaan menghabiskan banyak waktu untuk pengujian daripada menulis program.
   b. Pemasangan sistem. Pada tahap ini, sistem yang lama akan dihentikan dan digantikan oleh sistem yang baru.
   c. Tim analisis melakukan tinjauan setelah pemasangan sistem untuk mengidentifikasi kekurangan atau perubahan yang diperlukan oleh sistem agar sistem dapat berjalan sesuai dengan yang diinginkan.

3.2 Metode *Rapid Application Development* (RAD)

Rapid Application Development (RAD) merupakan salah satu metode pengembangan sistem informasi yang ada dalam *System Development Life Cycle* (SDLC). Waktu yang singkat adalah batasan yang penting dalam metode ini. Jika kebutuhan dapat dipahami dengan baik, pengembangan perangkat lunak menggunakan metode RAD memungkinkan tim pengembang dapat menyelesaikannya dalam waktu yang singkat sekitar 60-90 hari[14].

Berikut adalah tahapan-tahapan yang ada didalam *Rapid Application Development* (RAD)[15] :



Gambar 3. 2 Tahapan dalam *Rapid Application Development* (RAD)[15]

1. *Requirements Planning*

   Pada tahap ini, penganalisis dan pengguna melakukan pertemuan untuk mengidentifikasi tujuan dari sistem yang akan dibangun serta informasi dan kebutuhan apa saja yang diperlukan agar dapat mencapai tujuan tersebut agar sistem yang akan dibangun dapat sesuai dengan kebutuhan pengguna.

2. *Design Workshop*

   Tahap ini merupakan pertemuan antara penganalisis dan pemrogram untuk membahas dan merancang sistem yang akan dibangun. Penganalisis dan pemrogram saling bekerja sama dalam membangun sistem dalam bentuk *visual desain* dan pola kerja kepada pengguna sistem. Penganalsis dan pemrogram dapat memperbaiki rancangannya berdasarkan respon pengguna.

3. *Construction*

   Tahap konstruksi merupakan tahap eksekusi dalam bentuk pembuatan *script* program dan merupakan kelanjutan dari tahap desain. Kemudian menghasilkan fungsi baru dan ditunjukan kepada pengguna untuk mendapatkan respon dan revisi. Selanjutnya penganalisis dan pemrogram dapat melakukan perubahan dalam setiap desain berdasarkan instruksi pengguna.

4. *Implementation*

   Pada tahap ini penganalisis berkerja dengan pengguna untuk melihat apakah sistem yang dibangun sudah sesuai dengan kebutuhan pengguna. Jika dirasa masih ada yang kurang, maka analis dan pemrogram akan merancang dan

membangun kekurangannya. Kemudian setelah selesai dan pengguna menyetujui, maka sistem akan dilakukan uji coba dan diterapkan.

3.3 *Tools* Pengembang Sistem

3.3.1 *Unified Modeling Language* (UML)

UML adalah alat bantu yang digunakan dalam pemodelan sistem berorientasi objek dengan tingkat dokumentasi yang sangat tinggi. Oleh karena itu, UML adalah fondasi penting dalam perencanaan dan pengendalian pengembangan perangkat lunak. UML digunakan untuk menggambarkan sistem yang sedang dirancang. [16]. Dalam penelitian ini, penulis menggunakan 6 diagram UML, antara lain *use case diagram*, *class diagram*, *package diagram*, *activity diagram*, *sequence diagram*, dan *deployment diagram*[16].

1. *Use Case Diagram*

   Use Case Diagram berfungsi untuk membuat diagram yang menunjukkan kasus penggunaan yang lebih detail. *Use Case Diagram* adalah model UML yang menjabarkan kasus penggunaan dan hubungannya dengan pengguna (*actor*). Dalam UML, orang yang menggunakan disebut dengan *actor*. Terkadang *actor* bukanlah orang, *actor* bisa berupa sistem atau perangkat lain[17]. Simbol-simbol yang digunakan dalam *Use Case Diagram* adalah sebagai berikut[18] :

   a. *Use Case*

      Use Case mendeskrpiskan interaksi antara *user* dengan unit tentang bagaimana sebuah sistem dipakai yang dihubungkan menggunakan kata kerja.

   b. *Actor*

      Actor adalah orang atau sistem yang berhubungan dengan proses bisnis. Untuk mengindentifikasi *actor* harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada target sistem.

   c. *Association*

      Abstraksi berupa garis tanpa panah yang menghubungkan antara *actor* dengan *use case* atau *use case* dengan *use case*.

d. *Include*

Didalam *use case* lain *(required)* atau pemanggilan *use case* oleh *use case* lain, contohnya pemanggilan sebuah fungsi program.

e. *Extends*

Merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

2. *Class Diagram*

Konsep *Class Diagram* berasal dari pemodelan data konseptual dan pengembangan perangkat lunak yang berorientasi objek[19]. *Class Diagram* memiliki sejumlah peran penting dalam pengembangan perangkat lunak. Konsep kelas digunakan secara menyeluruh dalam pemodelan dan pemrograman sehingga memungkinkan kita untuk menemukan persyaratan dan kesalahan melalui berbagai aktivitas proyek[16]. *Class diagram* merupakan yang paling penting dan paling banyak digunakan dalam teknik pemodelan UML. Pada umumnya, *class diagram* berasal dari pemodelan entitas yang dipengaruhi oleh diagram aliran data (*data flow diagram*). *Class diagram* menggambarkan seperti apa struktur perangkat lunak dan menghasilkan notasi inti pertama yang akan dibahas dalam pemodelan beorientasi objek[20].

3. *Package Diagram*

Package Diagram merupakan kumpulan kelas-kelas yang dikelompokkan dan diberi nama sesuai dengan isinya[21].

4. *Activity Diagram*

Activity Diagram mendokumentasikan alur proses bisnis yang terjadi antara pelanggan dengan sistem dari awal sampai akhir[22].

Simbol-simbol yang digunakan dalam *activity diagram* adalah sebagai berikut[18] :

a. *Start Point*

Merupakan awal aktivitas yang diletakkan pada pojok kiri atas.

b. *End Point*

Menggambarkan akhir dari aktivitas.

c. *Activities*

Menggambarkan suatu proses atau kegiatan bisnis.

d. *Fork*/Percabangan

Digunakan untuk menunjukan kegiatan yang dilakukan secara paralel.

e. *Join*/Penggabungan

Digunakan untuk menunjukkan adanya penggabungan.

f. *Decision Point*

Menggambarkan pilihan untuk pengambilan keputusan, *true* atau *false.*

g. *Swim Lane*

Pembagian *activity diagram* untuk menunjukkan siapa melakukan apa.

5. *Sequence Diagram*

Sequence diagram digunakan untuk memodelkan antar objek. *Sequence diagram* berfokus pada urutan kronologis pesan yang dilakukan antar mitra interaksi, hal ini berguna untuk membuat model interaksi yang kompleks dan sesuai dengan urutannya[19].

6. *Deployment Diagram*

Deployment Diagram merupakan diagram yang menggambarkan secara detail bagaimana komponen disusun di infrastruktur sistem[18].

3.3.2 *Database*

Database merupakan sekumpulan data yang saling berhubungan. Hubungan tersebut bisa ditunjukan dengan kunci dari setiap data yang ada, atau biasa disebut dengan *primary key*. Data-data tersebut disimpan bersama-sama pada pada suatu media[23]. Pada penelitian ini, penulis akan menjelaskan rancangan *database* menggunakan *tools* sebagai berikut :

1. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram (ERD) merupakan diagram yang digunakan untuk menjelaskan hubungan antar data dalam basis data yang terdiri dari simbol-simbol tertentu[24].

2. Transformasi ERD ke LRS

Transformasi ERD ke LRS merupakan kegiatan penggabungan antara entitas dengan relasi berdasarkan kardinalitas yang ada pada ERD menjadi LRS[25].

3. *Logical Record Structured* (LRS)

   Logical Record Structured (LRS) merupakan penggambaran ERD ke dalam bentuk yang lebih jelas dan mudah untuk dipahami[26].

4. Tabel

   Tabel merupakan tempat yang digunakan untuk menampung data atau sekelompok *record* data yang berisi informasi yang sejenis[27].

5. Spesifikasi Basis Data

   Spesifikasi Basis Data digunakan untuk menjelaskan secara rinci tentang setiap tabel yang berelasi[25].