**BAB II**

**LANDASAN TEORI**

* 1. **Aplikasi**

**Aplikasi berasal dari kata *Aplication* yang menurut kamus Komputer Eksekutif (1993:9), *Aplication* adalah masalah yang memakai teknik pemrosesan data aplikasi biasanya mengacu pada komputasi yang diinginkan, atau pemrosesan data. Pengertian aplikasi menurut Yan Tirtobisono (1992:21) adalah istilah yang digunakan untuk pengguna komputer bagi pemecahan masalah. Sedangkan menurut Kamus Besar Bahasa Indonesia (1998:52), “aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu.**

**Aplikasi adalah suatu program komputer yang dibuat untuk mengejakan dan melaksanakan tugas khusus dari pengguna. Aplikasi merupakan program siap pakai yang dapat digunakan untuk menjalankan perintah-perintah dari pengguna aplikasi tersebut.**

* 1. **Manajemen Artefak Pembangun Perangkat Lunak**

Manajemen telah banyak disebut sebagai seni “seni untuk merealisasikan pekerjaan melalui orang lain”. Terjemahan bebasnya kira-kira seperti ini: seorang yang lain mancapai tujuan tertentu, namun untuk mencapai tujuan tersebut dia menggunakan “tangan” orang lain. Dalam kaitannya dengan organisasi, bisa berari bahwa para manajer dalam mencapai tujuan organisasi dengan cara peraturan orang lain ( misalkan staf atau karyawan) untuk melakukan berbagai pekerjaan sesuai dengan tujuan organisasi, tanpa harus melakukan dengan “tangan” sendiri. Manajemen lebih bersifat arsitektur atau konseptual (perancangan-perancangan strategis), bukan tehnikal maupun operasional. Manajemen mencakup fungsi perencanaan ( Penetapan apa yang tidak akan dilakukan ), penorganisasian ( perancangan dan penugasan kelompok kerja ), penyusunan personalia ( Penarikan, seleksi, pengembangan dan penilaian prestasi kerja), pengarahan ( motivasi, kepemimpinan, integritas dan pengolahan konflik) dan pengawasan.

Manajemen Proyek merupakan suatu ilmu pengetahuan, keahlian dan juga keterampilan, cara teknis yang terbaik serta dengan sumber daya yang terbatas untuk mencapai sasaran atau tujuan yang sudah ditentukan agar mendapat hasil yang optimal dalam hal kinerja, waktu dan keselamatan kerja. Tujuan utama dari manajemen proyek adalah agar proyek dapat dilaksanakan dengan efisien, tepat waktu, dan mencapai hasil yang diinginkan. Sering terjadi pada sebuah proyek yang berlarut pada pekerjaannya sehingga pada akhirnya harus mengalami penjadwalan ulang. Oleh karena itu, peran perencanaan dalam suatu proyek sangat penting, segala sesuatu harus dimulai dari rencana dan harus disepakati bersama antara para *stakeholder* yang terlibat pada proyek. *Stakeholder* yang dimaksud didalam proyek adalah pemilik proyek (*project owner*), komite pengarah (*steering committee*), pengguna hasil proyek dan pelaksana proyek (Heryanto, 2015).

Artefak merupakan salah satu dari banyak jenis produk sampingan nyata yang dihasilkan selama pengembangan perangkat lunak. beberapa artefak (misalnya, kasus penggunaan, diagram kelas, dan model Unified Modelling Language ( UML ) Lainnya, persyaratan dan dokumen desain) membantu menjelaskan fungsi, arsitektur, dan desain perangkat lunak. artefak lain berkaitan dengan proses pengembangan itu sendiri seperti rancangan proyek, kasus bisnis, dan penilaian risiko. Namun dalam lingkungan pengembangan perangkat lunak cepat terdapat berbagai macam artefak yang terdapat dalam satu dokumentasi, hal ini mencerminkan fakta bahwa dalam model pengembangan perangkat lunak cepat setiap artefak dapat digunakan sebagai dokumentasi.

Perangkat lunak (*Software*) adalah program komputer yang terasosiasi dengan dokumentasi preangkat lunak seperti dokementasi kebutuhan, model sistem, dan cara penggunaannya. Rekayasa perangkat lunak (RPL) merupakan pembangunan sebuah perangkat lunak dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomis yang di percaya dan bekerja secara efisien menggunakan mesin. RPL lebih fokus pada praktek pengembangan perangkat lunak yang bermanfaat bagi pelanggan (*User*) dengan memenuhi kriteria sebagai berikut :

1. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring  
   berkembangnya teknologi dan lingkungan (*Maintainability*).
2. Dapat diandalkan dengan proses bisnis yang dijalankan dan perubahan yang terjadi (*Dependability dan Robust*).
3. Efisiensi dari segi sumber daya dan penggunaan.
4. Kemampuan untuk dipakai sesuai dengan kebutuhan (*Usability*)  
   Jadi perangkat lunak yang baik adalah perangkat lunak yang fokus kepada pengguna atau pelanggan.
   1. **Pengertian Search Engine**

Search Engine (Mesin Pencari) adalah salah satu program komputer yang dirancang untuk menemukan atau mencari file - file yang disimpan dalam computer. Mesin pencari memungkinkan kita untuk menemukan file sesuai dengan kriteria yang spesifik yang mengandung prase atau kata kunci yang di inginkan .

* 1. **Information Retrival System (IRS)**

Penelusuran informasi adalah mencari kembali informasi yang pernah ditulis orang mengenai topik tertentu, informasi tersebut terdapat dalam publikasi yang diterbitkan baik dalam maupun luar negeri (Djatin, 1996: 3). Penelusuran informasi sebagaimana dijelaskan oleh Marchioni (1995) dalam Large (2001: 27) yaitu sebuah kebutuhan hidup yang digunakan untuk merencanakan, mengambil tindakan dan melakukan apa yang dibutuhkan untuk mendapatkan sebuah pemahaman yang dapat diterima oleh akal. .

Information Retrieval System (IRS) atau sistem temu kembali informasi merupakan bagian dari ilmu komputer yang berhubungan dengan pengambilan informasi dari dokumen-dokumen yang didasarkan pada isi dan konteks dari dokumen-dokumen itu sendiri. Proses dalam Information Retrieval System (IRS) dapat digambarkan sebagai sebuah proses untuk mendapatkan dokumen yang relevan dari koleksi dokumen melalaui pencarian query yang diinputkan user. Maksud dan tujuan Information Retrieval System adalah untuk memanggil dokumen-dokumen atau informasi masyarakat pengguna (Hasugian, 2001: 6). Sedangkan menurut Salton Information Retrieval System bertujuan untuk menjembatani kebutuhan informasi user dengan sumber informasi yang tersedia:Salton:1989.

Perkembangan Information Retrieval atau Temu Kembali Informasi dari sisi user task ada 2 jenis yaitu:

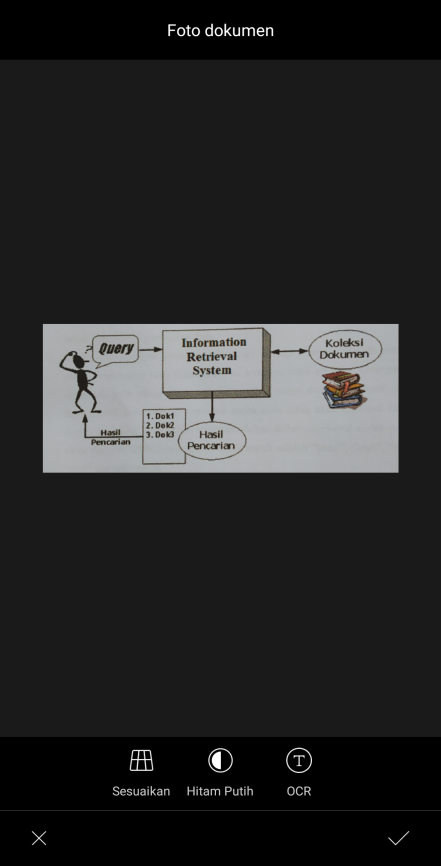
**ModeI Klasik.**

1. Model Boolean : merupakan model sistem temu kembali informasi sederhana yang berdasarkan atas teori himpunan dan aljabar boolean .
2. Model Vector Space : merupakan model sistem temu kembali informasi yang merepresentasikan dokumen dan query dalam bentuk vektor dimensional
3. Model Probabilistic . merupakan model sistem temu kembali informasi yang menggunakan framework probabilistik

**Model Terstruktur**

1. Non Overlapping List: Sistem yang menggunakan model ini akan membagi-bagi dokumen sebagai "wilayah teks" tertentu.
2. Proximal Nodes: model ini menggunakan struktur indeks yang memiliki hirarkiindependen (non-flet) terhadap sebuah dokumen.
   * 1. **Komponen Information Retrieval System (IRS)**

Secara umum Information Retrieval System atau Sistem Temu Kembali Informasi terdiri dari tiga komponen utama, yaitu masukan (input), pemroses (processor) dan keluaran Output). Komponen-komponen ini digambarkan pada Gambar berikut :



**Gambar 2.1** Skema umum information retrieval system(IRS)

Sedangkan Menurut Hasugian ( 2007 : 3 ) ada lima komponen IRS yaitu : pengguna. query. dokumen, indeks dokumen dan pencocokan.

1. Pengguna. yaitu orang yang menggunakan dan memanfaatkan IRS untuk kegiatan pencarian informasi dan pengelolaan informasi.
2. Query. yaitu format bahasa yang digunakan oleh pengguna kedalam IRS
3. Dokumen, yaitu istilah yang digunakan untuk bahan pustaka berupa e- book dan e- journal
4. Indeks dokumen, yaitu istilah atau kata yang disimpan kedalam database yang berfungsi sebagai representasi sebuah dokumen.
5. Pencocokan, yaitu istilah yang dimasukkan oleh pengguna, dan proses pencocokan mesin komputerlah yang berperan.
   * 1. **Tahapan Information Retrieval system (IRS)**

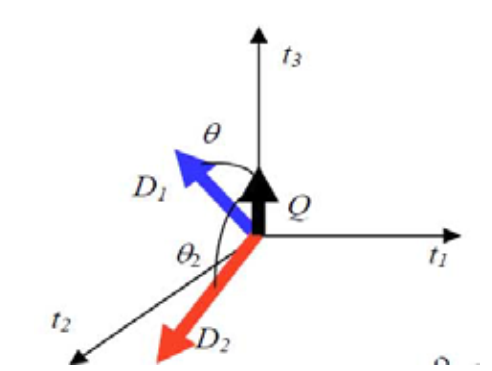
Tahapan Inormation Retrieval system (IRS) terdiri dari : modul pengumpulan dokumen, modul tokenisasi (tokenizing), modul pembuangan stopword (filtering), modul Pengubahan kata dasar (stemming), modul Pengindekaan kata (indexing), dan modul Vector Space Model (term similarity).

1. Modul Pengumpulan Dokumen. Proses pengumpulan dokumen-dokumen yang dipilih untuk disimpan dalam korpus. Dokumen-dokumen yang dipilih adalah dokumen berbasis Teks.
2. Modul Tokenisasi (tokenizing). Tokenizing adalah proses pemotongan string input berdasarkan tiap kata yang menyusunnya. Umumnya setiap kata yang teridentifikasi atau terpisahkan dengan kata yang lain oleh karakter spasi. sehingga proses ini menggunakan spasi pada dokumen untuk melakukan pemisahan kata,
3. Modul Pembuangan stopword (filtering). Tahap filtering adalah proses pembuangan term yang tidak memiliki arti atau tidak relevan. Term yang diperoleh pada tahap tokenisasi dicek dalam suatu daftar stopword, jika term masuk dalam daftar stopword maka term tidak akan diproses lebih lanjut. tapi jika term tidak termasuk dalam daftar stopword maka term akan diproses lebih lanjut, Contoh stopwords adalah "yang", "dan", "di","dari" dan seterusnya.
4. Modul Pengubahan Kata Dasar (Stemming). Proses stemming adalah tahap mencari kata dasar (root) dari tiap kata hasil filtering. Pada tahap ini dilakukan proses pengembalian berbagai bentukan kata ke dalam bentuk kata dasar. Stemming digunakan untuk mereduksi bentuk term untuk menghindari ketidak cocokan sehingga dapat mengurangi recall.
5. Indexing, Proses Indexing adalah tahap pengindeksan kata dari koleksi teks yang digunakan untuk mempercepat proses pencarian. Setelah kata telah dikembalikan dalam bentuk kata dasar, kemudian disimpan dalam tabel basis data. Proses Indexing Menghasilkan Database Index.
6. Modul Vector Space Model. Hasil indexing selanjutnya dihitung tingkat kemiripannya dengan query menggunakan metode vector space model,
   1. **Stopword**

Stopwords adalah kata umum (common words) yang biasanya muncul dalam.jumlah besar dan dianggap tidak memiliki makna Stop words umumnya dimanfaatkan dalam task information retrieval. termasuk oleh Google . Contoh stop words untuk bahasa Inggris diantaranya "of', "the". Sedangkan untuk bahasa Indonesia diantaranya "yang", "di", "ke"

* 1. ***Vector Space Model***

*Vector Space Model* (VSM) adalah metode melihat tingkat kedekatan atau kesamaan (*smilarity) term* dengan cara pembobotan *term.* Dokumen pidanpang sebagai sebuah vektor yang memiliki *magnitude* (jarak) dan *direction* (arah). Pada *Vector Space Model,*  sebuah istilah direpresentasikan dengan sebuah dimensi dari ruang cektor. Relevansi sebuah dokumen ke sebuah *query* didasarkan pada similaritas diantara vektor dokumen dan cektor *query.*



**Gambar 2.2** Ilustrasi Vector Space Model

Dimana

ti = kata di database

Di = Dokumen

Q = kata kunci

Cara kerja dari *vector space model* adalah dengan menghitung nilai *cosines* sudut dari dua vektor, yaitu vektor kata kunci terhadap vektor tiap dokumen dengan rumus :

Dimana

Q= query (kata kunci)

Di= dokumen ke-i

Maka :

Dimana

Di = dokumen ke-i

Q = query (kata kunci)

J = kata diseluruh dokumen

Dimana

Di = dokumen ke-i

Q = query (kata kunci)

|Q| = Vektor Q

|Di| = Vektor Di

**2.6.1 Modul Vector Space Model (Similarity Analysis)**

Tahapan proses analisa VSM adalah sebagai berikut; Menghitung bobot dokumen dengan tf-idf, Menghitung jarak tiap *query* dan dokumen, Menghitung dot produk, Meng hitung *similarity,* dan Membuat rangking.

**2.6.1.1 Menghitung bobot dokumen dengan tf-idf**

Langkah pertama adala menghitung bobot dengan *tf-idf.*  Setelah hasl perhitungan tf didapatkan, langkah slanjutnya dilakukan perhitungan *inverse document frequency* (idf) tiap tokenuntuk menghitung bobot token, seperti yang ditunjukkan pada persamaan (2.1) :

*Idf = log (D/df) .............................*Persamaan 2.1

Dimana D = jumlah dokumen dalam korpus

Selanjutnya, setelah nilai *tf*  dan *idf*  telah didapatkan, kemudian dimasukkan dalam perhitungan *tf-idf weighting* untung menghiung bobot hubungan suatu token di dalam dokumen [Robertson,2005]. Seperti yang ditunjukkan pada persamaan (2.2) :

.............................Persamaan 2.2

Dimana

d = dokumen ke-d

t =kata ke-t dari kata kunci

W = bobot dokumen ke-d terhadap kata ke-t

**2.6.1.2 Menghitung jarak tiap dokumen dan query**

Selanjutnya setelah pembobolan dengan *tf-idf* didapatkan, kemudian melakukan perhitungan jarak tiap dokumen dan *query.*  Seperti yang ditunjukkan pada persamaan (2.3) dan Persamaan (2.4) :

.............................Persamaan 2.3

Perhitungan *vector* dari *query*

sqrt (Q) = sqrt ()

dimana

j = kata di database

...................................Persamaan 2.4

Perhitungan *vector* dari dokumen

sqrt (D) = sqrt

keterangan

j =kata di database

**2.6.1.3 Menghitung dot produk**

Selanjutnya setelah jarak dari dokumen dan query didapatkan, dilakukan perhitungan dot produk. Seperti yang ditunjukkan pada Persamaan (2.5) :

...............Persamaan 2.5

Keterangan

j = kata di database

**2.6.1.4 Menghitung similaritas**

Langkah selanjutnya adalah menghitung nilai Cosinus sudut antara *vector query* dengan tiap dokumen seperti yang ditunjukkan pada persamaan (2.6) :

.............................Persamaan 2.6

**2.6.1.5 Membuat rangking**

Setelah menghitung nilai cosinus lalu di buat perangkingan dari dokumen-dokumen tersebut.

* 1. **Scrum**

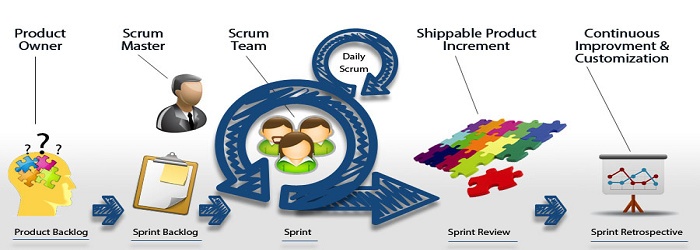
Scrum sebuah kerangka kerja di mana orang-orang dapat menyelesaikan permasalahan kompleks yang senantiasa berubah, di mana pada saat bersamaan menghasilkan produk dengan nilai setinggi mungkin secara kreatif dan produktif.

Scrum Sifat :

* Ringan
* Mudah dipahami
* Sulit dikuasai

Scrum adalah kerangka kerja proses yang telah digunakan untuk mengelola pengembangan produk kompleks semenjak awal 1990-an. Scrum bukanlah sebuah proses ataupun teknik untuk mengembangkan produk; daripada itu, ini adalah sebuah kerangka kerja di mana di dalamnya anda dapat memasukkan beragam proses dan teknik. scrum akan mengekspos pererakan efektufutas manajemen produk dan praktik pengembangan yang sedan anda jalani, dengan begitu anda dapat melakukan peningkatan.

Kerangka kerja Scrum terdiri dari tim Scrum, serta peran-peran mereka di dalamamnya, acara-acara, artefak-artefak, dan aturan-aturan. setiap komponen di dalam kerangka kerja memiliki maksud tertentu dan peran penting demi keberhasilan pengguna scrum.



**Gambar 2.3** Ilustrasi Metode Scrum

* + 1. Product Owner

Product Owner bertanggung-jawab untuk memaksimalkan nilai produk dan hasil kerja Tim Pengembang. Cara pelaksanaannya sangat bervariasi antar organisasi, Tim Scrum dan individu. Product Owner merupakan satu-satunya orang yang bertanggung-jawab untuk mengelola Product Backlog.

* + 1. Scrum Master

Scrum Master bertanggung jawab untuk memastikan Scrum telah dipahami dan dilaksanakan. Scrum Master melakukannya dengan memastikan Tim Scrum mengikuti teori, praktik, dan aturan main Scrum.

* + 1. Tim Scrum

mengatur diri mereka sendiri dan berfungsi antar-lintas. Tim yang mengatur dirinya sendiri menentukan cara terbaik untuk menyelesaikan pekerjaannya, daripada diatur oleh pihak lain yang berada di luar anggota tim. Tim yang berfungsi antar-lintas memiliki semua kompetensi yang dibutuhkan untuk menyelesaikan pekerjaan, tanpa mengandalkan pihak lain yang berada di luar anggota tim. Model tim di dalam Scrum dirancang sedemikian rupa untuk mengotimalisasi fleksibilitas, kreatifitas dan produktifitas. Tim Scrum menghantarkan produk secara berkala dan bertahap untuk memperbesar kesempatan mendapatkan masukan. Penghantaran secara bertahap dari sebuah produk yang “Selesai”, memastikan produk yang berpotensi dapat digunakan, selalu siap.

* + 1. Inkremen

Inkremen ( tambahan potongan produk ) adalah gabungan dari semua item Product Backlog yang diselesaikan pada Sprint berjalan dan nilai-nilai dari Inkremen sprint-sprint sebelumnya. Pada akhir Sprint, inkremen terbaru harus “Selesai”, yang artinya berada dalam kondisi yang berfungsi penuh dan memenuhi definisi “Selesai” yang dibuat oleh Tim Scrum. Terlepas apakah Product Owner akan merilis produknya, produk harus selalu berada dalam kondisi yang berfungsi penuh.

* + 1. Product Backlog

Product Backlog adalah daftar terurut, dari setiap hal yang berkemungkinan dibutuhkan di dalam produk, dan juga merupakan sumber utama, dari daftar kebutuhan mengenai semua hal yang perlu dilakukan terhadap produk. Product Owner bertanggung-jawab terhadap Product Backlog, termasuk isinya, ketersediaannya, dan urutannya.

* + 1. Spint Backlog

Sprint Backlog adalah sekumpulan item Product Backlog yang telah dipilih untuk dikerjakan di Sprint, juga di dalamnya rencana untuk mengembangkan potongan tambahan produk dan merealisasikan Sprint Goal. Sprint Backlog adalah perkiraan mengenai fungsionalitas apa yang akan tersedia di Inkremen selanjutnya dan pekerjaan yang perlu dikerjakan untuk menghantarkan fungsionalitas tersebut menjadi potongan tambahan produk yang “Selesai”.

* + 1. Sprint

Jantung dari Scrum adalah Sprint, sebuah batasan waktu selama satu bulan atau kurang, di mana sebuah Inkremen yang “Selesai”, berfungsi, berpotensi untuk dirilis dikembangkan. Sprint biasanya memiliki durasi yang konsisten sepanjang proses pengembangan produk. Sprint yang baru, langsung dimulai setelah Sprint yang sebelumnya berakhir.

* + 1. Sprint Review

Sprint Review diadakan di akhir Sprint untuk meninjau Inkremen dan merubah Product Backlog bila diperlukan. Pada saat Sprint Review, Tim Scrum dan stakeholder berkolaborasi untuk membahas apa yang telah dikerjakan dalam Sprint yang baru usai. Berdasarkan hasil tersebut tersebut dan semua perubahan Product Backlog pada saat Sprint, para hadirin berkolaborasi menentukan apa yang dapat dikerjakan di Sprint berikutnya, untuk mengoptimalisasi nilai produk. Pertemuan ini bersifat informal, bukan merupakan *status meeting*, dan presentasi dari Inkremen diharapkan dapat mengumpulkan masukan dan menumbuhkan semangat kolaborasi.

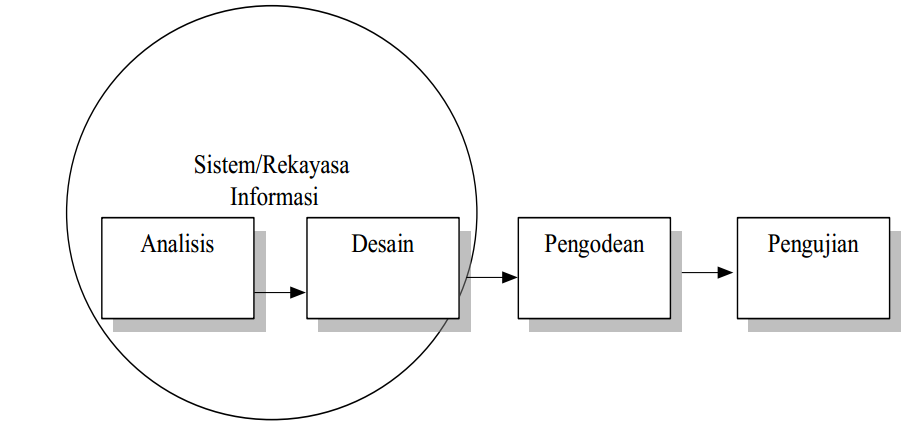
* + 1. Sprint Restrospestive

Sprint Retrospective adalah sebuah kesempatan bagi Tim Scrum untuk meninjau dirinya sendiri dan membuat perencanaan mengenai peningkatan yang akan dilakukan di Sprint berikutnya. Sprint Retrospective dilangsungkan setelah Sprint Review selesai dan sebelum Sprint Planning berikutnya. Ini adalah acara dengan batasan waktu maksimum selama tiga jam untuk Sprint yang berdurasi satu bulan. Untuk Sprint yang lebih pendek, batasan waktunya biasanya lebih singkat. Scrum Master memastikan bahwa acara ini dilaksanakan dan setiap hadirin memahami tujuannya. Scrum Master mengedukasi Tim Scrum untuk melaksanakannya dalam batasan waktu yang telah ditentukan. Scrum Master berpartisipasi sebagai rekan yang bertanggungjawab terhadap proses Scrum.

* 1. **Waterfall**

Model Waterfall merupakan salah satu model pengembangan perangkat lunak yang ada di dalam model SDLC (*Sequencial Developmen Life Cycle*). menurut Sukanto dan Shalahuddin (2013:26) mengemukakan bahwa " SDLC atau *Software Development Life Cycle* atau sering disebut juga *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya, berdasarkan *best practice* atau cara-cara yang sudah teruji baik.

Sedangkan Sukanto dan Shalahuddin (2013:28) di jelaskan bahwa model *waterfall*  sering juga di sebut model sekuensi linear atau alur hidup klasik. pengembangan sistem dikerjakan secara terurut mulai dari analisis, desain, pengkodean, pengujian dan tahap pendukung

**

**Gambar 2.4** Ilustrasi Waterfall

**2.8.1.** Analisis Kebutuhan Perangkat Lunak

Proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

**2.8.2.** 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 prosedur 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.

**2.8.3.** Pembuatan Kode Program

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

**2.8.4.** Pengujian

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 memastikann keluaran yang dihasilkan sesuai dengan yang diinginkan.

**2.8.5.** Pendukung (*Support*) atau Pemeliharaan (*Maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan 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.

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

Menurut Windu Gata (Urva, dkk, 2015) hasil pemodelan pada OOAD terdokumentasikan dalam bentuk Unified Modeling Language (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umumdalam industri perangkat lunak dan pengembangan sistem.

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

Use case Diagram

Use case diagram merupakan pemodelan untuk kelakukan (behavior) sistem informasi yang akan dibuat. Use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbolsimbol yang digunakan dalam use case diagram, yaitu

**Tabel 2.1** Simbol usecase

|  |  |
| --- | --- |
| **Gambar** | **Keterangan** |
|  | Use case menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama use case. |
|  | Aktor adalah abstraction dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan use case, tetapi tidak memiliki control terhadap use case. |
|  | Asosiasi antara aktor dan use case, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data |

**Tabel 2.1** ( lanjutan )

|  |  |
| --- | --- |
| **Gambar** | **Keterangan** |
|  | Asosiasi antara aktor dan use case yang menggunakan panah terbuka untuk mengidinkasikan bila actor berinteraksi secara pasif dengan sistem. |
|  | Include, merupakan di dalam use case lain (required) atau pemanggilan use case oleh use case lain, contohnya adalah pemanggilan sebuah fungsi program. |
|  | Extend, merupakan perluasan dari use case lain jika kondisi atau syarat terpenuhi. |

Diagram Aktivitas *(Activity Diagram*)

Activity Diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam activity diagram yaitu :

**Tabel 2.2** Simbol Activity Diagram

|  |  |
| --- | --- |
| **Gambar** | **Keterangan** |
| ACTI1 | Start point, diletakkan pada pojok kiri atas dan merupakan awal aktifitas. |
| a2 | End point, akhir aktifitas. |
| ACTI1 | Activites, menggambarkan suatu proses/kegiatan bisnis. |

**Tabel 2.2** (lanjutan)

|  |  |
| --- | --- |
| **Gambar** | **Keterangan** |
| ACTI1 | Fork (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu. |
| ACTI1 | Join (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi. |
| ACTI1 | Decision Points, menggambarkan pilihan untuk pengambilan keputusan, true, false |
| ACTI1 | Swimlane, pembagian activity diagram untuk menunjukkan siapa melakukan apa. |

Diagram Kelas (*Class Diagram*)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem. Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan. Class diagram secara khas meliputi: Kelas (Class), Relasi, Associations, Generalization dan Aggregation, Atribut (Attributes), Operasi (Operations/Method), Visibility, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan multiplicity atau kardinaliti.

**Tabel 2.3** Multiplicity Class *Diagram*

|  |  |
| --- | --- |
| **Multiplicity** | **Keterangan** |
| 1 | Satu dan hanya satu |
| 0..\* | Boleh tidak ada atau 1 atau lebih |
| 1..\* | 1 atau lebih |
| 0..1 | Boleh tidak ada, maksimal 1 |
| n..n | Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4 |

Diagram Urutan (*Squence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam sequence diagram yaitu :

**Tabel 2.4** Simbol *Sequence* *Diagram*

|  |  |
| --- | --- |
| **Gambar** | **Keterangan** |
|  | EntityClass, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data |
|  | Boundary Class, berisi kumpulan kelas yang menjadi interface atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan form cetak. |
|  | Control class, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek. |
|  | Message, simbol mengirim pesan antar class |

**Tabel 2.4** (lanjutan)

|  |  |
| --- | --- |
| **Gambar** | **Keterangan** |
|  | Activation, activation mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi |
|  | Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang lifeline terdapat activation |

* 1. **Entity Relational Diagram (ERD)**

ERD (Fridayanthie, dkk, 2016) adalah model teknik pendekatan yang menyatakan atau menggambarkan hubungan suatu model. Didalam hubungan ini tersebut dinyatakan yang utama dari ERD adalah menunjukan objek data (Entity) dan hubungan (Relationship), yang ada pada Entity berikutnya. Menurut Simarmata (2010:67), “Entity RelationShip Diagram (ERD) adalah alat pemodelan data utama dan akan mambantu mengorganisasi data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antar entitas”.

Proses memungkinkan analis menghasilkan struktur basis data dapat disimpan dan diambil secara efisien. Simbol-simbol dalam ERD (Entity Relationship Diagram) adalah sebagai berikut:

a. Entitas : suatu yang nyata atau abstrak yang mempunyai karakteristik

dimana. kita akan menyimpan data.

b. Atribut : ciri umum semua atau sebagian besar instansi pada entitas

tertentu.

c. Relasi : hubungan alamiah yang terjadi antara satu atau lebih entitas.

d. Link : garis penghubung atribut dengan kumpulan entitas dan kumpulan

entitas dengan relasi.

* + 1. **Kardinalitas Relasi**

Satu ke satu (One to One) Setiap elemen dari Entitas A berhubungan paling banyak dengan elemen pada Entitas B. Demikian juga sebaliknya setiap elemen B berhubungan paling banyak satu elemen pada Entitas A.

Satu ke banyak (One to Many) Setiap elemen dari Entitas A berhubungan dengan maksimal banyak elemen pada Entitas B. Dan sebaliknya setiap elemen dari Entitas B berhubungan dengan paling banyak satu elemen di Entitas A**.**

Banyak ke satu (Many to One) Setiap elemen dari Entitas A berhubungan paling banyak dengan satu elemen pada Entitas B. Dan sebaliknya setiap elemen dari Entitas B behubungan dengan maksimal banyak elemen di entitas A.

Banyak ke banyak (Many to Many) Setiap elemen dari Entitas A berhubungan maksimal banyak elemen pada Entitas B demikian sebaliknya.

**Tabel 2.5**  Daftar simbol ERD pada aplikasi StarUML

|  |  |  |
| --- | --- | --- |
| **No** | **Simbol** | **Keterangan** |
| 1 |  | *Entity* merupakan suatu objek yang dapat diidentifikasi dalam lingkungan pengguna. |
| 2 |  | *One to One Relationship,* relasi satu ke satu berarti setiap himpunan entitas hanya boleh berhubungan dengan satu entitas lainnya. |
| 3 |  | *One to Many Relationship,* relasi satu ke banyak berarti setiap himpunan entitas boleh berhubungan dengan banyak himpunan entitas lainnya. |
| 4 |  | *Many to Many Relationship,* relasi banyak ke banyak berarti setiap himpunan entitas boleh berhubungan dengan banyak entitas lainnya dan sebaliknya. |

Untuk dokumentasinya bisa diakses langsung melalui situs resmi StarUML di <https://docs.staruml.io>.

* 1. **Framework**

Menurut Betha Sidik framework adalah : “kumpulan intruksi-intruksi yang dikumpulkan dalam class dan *function-function* dengan fungsi masing-masing untuk memudahkan developer dalam memanggilnya tanpa harus menuliskan syntax program yang sama berulang-ulang serta dapat menghemat waktu”.

* + 1. **Laravel Framework**

Laravel merupkan framework php yang menerapkan konsep MVC (*Model View Controller*)didesain untuk memudahkan programmer dari segi pengembangan dan perawatan serta meningkatkan produktivitas programmer karena didalam framework ini sudah tersedia library yang dapat digunakan secara gratis atau opensource.

Laravel memberikan fasilitas untuk berinteraksi dengan database yang disebut dengan *migration*. Fungsi *migration* adalah memudahkan programmer untuk melakukan modifikasi database melalui CLI (*Command Line Interface*) karena skema dalah database disimpan dalam sebuah class. Migration dapat berjalan di beberapa basis data di antaranya adalah MySQL, PostgreSQL, MSSQL, dan SQLITE).

* + 1. **Angular Framework**

AngularJS adalah kerangka struktural untuk aplikasi web dinamis. AngularJS memungkinkan penggunanya menggunakan HTML sebagai bahasa pemrograman yang dipakai dan memungkinkan pengguna memperluas sintaks HTML untuk mengekspresikan komponen aplikasi yang dibuat oleh pengguna dengan jelas dan ringkas. Data binding AngularJS dan dependency injection AngularJS dapat memperingkas proses coding. Dan semua proses tersebut terjadi dalam browser, sehingga AngularJS mampu menjadi pasangan yang ideal dengan teknologi server. AngularJS adalah salah satu bentuk dari HTML bila ingin digunakan untuk mendesain aplikasi. HTML adalah bahasa yang cukup baik untuk dokumen statis. AngularJS adalah kerangka untuk membuat single page application. Dalam pengaplikasiannya AngularJS dapat menghasilkan halaman dinamis tanpa harus membuat banyak halaman web (Boedijono, dkk, 2015).

**2.12 *Aplication Programming Interface* (API)**

*Aplication programming interface* disingkat API adalah sekumpulan perintah, fungsi, serta protokol yang dapat digunakan oleh programmer saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan sistem operasi. API memiliki empat *method* yaitu:

1. GET : Mendapatkan *resource* yang ada di API.
2. POST : Menambahkan *resource* baru untuk disimpan diserver.
3. PUT : Mengubah atau memodifikasi *resource* yang sudah tersimpan.
4. DELETE: Menghapus *resource* yang tersimpan.

**2.13 Mysql**

Menurut Asrianda, Database adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bisa ditunjukkan dengan kunci dari tiap tabel yang ada. Satu database menunjukkan satu kumpulan data yang dipakai dalam satu lingkup perusahaan atau instansi. Database mempunyai kegunaan dalam mengatasi penyusunan dan penyimpanan data, maka seringkali masalah yang dihadapi adalah:

1. Redundansi dan Inkonsistensi data

2. Kesulitan dalam pengaksesan data

3. Isolasi data untuk standarisasi

4. Multi user

5. Keamanan data

6. Integritas data

7. Kebebasan data

MySQL adalah suatu sistem manajemen basis data relasional (RDBMS-Relational Database Management System) yang mampu bekerja dengan cepat, kokoh, dan mudah digunakan. Contoh RDBMS lain adalah Oracle, Sybase. Basis data memungkinkan anda untuk menyimpan, menelusuri, menurutkan dan mengambil data secara efesien. Server MySQL yang akan membantu melakukan fungsionaliitas tersebut.

Bahasa yang digunakan oleh MySQL tentu saja adalah SQL-standar bahasa basis data relasional di seluruh dunia saat ini.MySQL dikembangkan, dipasarkan dan disokong oleh sebuah perusahaan Swedia bernama MySQL AB. RDBMS ini berada di bawah bendera GNU GPL sehingga termasuk produk Open Source dan sekaligus memiliki lisensi komersial. Apabila menggunakan MySQL sebagai basis data dalam suatu situs Web. Anda tidak perlu membayar, akan tetapi jika ingin membuat produk RDBMS baru dengan basis MySQL dan kemudian mengualnua, anda wajib bertemu mudah dengan lisensi komersial (Urva, dkk, 2015).