**BAB II**

**TINJAUAN UMUM DAN LANDASAN TEORI**

* 1. **Kerangka Pikir**

Untuk lebih memperjelas kerangka berpikir yang akan disajikan, maka akan digambarkan pada Gambar 2.1., sebagai berikut:

Dewan Perwakilan Rakyat Daerah (DPRD) Kota Makassar merupakan salah satu institusi pemerintahan yang berperan penting dalam lembaga legislatif yang berkedudukan di Kota Makassar.

Permasalahan dalam penelitian ini adalah setiap komisi masih belum memiliki akses langsung ke sistem, artinya untuk meneruskan aspirasi masyarakat ke komisi terkait masih membutuhkan bantuan bagian Hubungan Masyarakat (Humas) dalam proses penyeleksiannya, dimana Humas akan menerima aspirasi masyarakat dan melakukan proses seleksi untuk menentukan ke komisi mana aspirasi tersebut akan diteruskan, hal ini tentunya akan memakan waktu yang lama dalam proses penyeleksiannya.

Tujuan dalam penelitian ini adalah untuk membangun aplikasi yang dapat menampung aspirasi masyarakat yang telah dikategorikan berdasarkan komisi terkait secara otomatis serta dapat langsung diteruskan ke komisi terkait.

Diharapkan dengan penerapan aplikasi aspirasi masyarakat berbasis web ini dapat mempermudah dan menghemat waktu dalam pengolahan data aspirasi dari masyarakat Kota Makassar. Sehingga aspirasi masyarakat dapat tersampaikan dengan cepat dan tepat sasaran sesuai dengan komisi yang bersangkutan.

**Gambar 2.1 Diagram Kerangka Berpikir**

* 1. **Tinjauan Umum**
     1. **Definisi Aplikasi**

Menurut Hengky W. Pramana sumber (2012:3): “Aplikasi adalah satu unit perangkat lunak yang sengaja dibuat untuk memenuhi kebutuhan akan berbagai aktivitas ataupun pekerjaan, seperti aktivitas perniasgaan, periklanan, pelayanan masyarakat, game, dan berbagai aktivitas lainnya yang dilakukan oleh manusia”.

Sedangkan menurut Jogiyanto (2005:22): “aplikasi adalah penggunaan atau penerapan suatu konsep yang menjadi pokok pembahasan. Aplikasi dapat diartikan juga sebagai program computer yang dibuat untuk menolong manusia dalam melaksanakan tugas tertentu”.

* + 1. **Text Mining**

Menurut Miller (2005:7) : menyatakan bahwa Text Mining merupakan proses otomatis atau sebagian proses otomatis untuk teks. Ini melibatkan pembentukan *text* yang lebih terstruktur dan penggalian informasi yang relevan dari teks.

Sedangkan menurut Clara Bridge (2011:7) : menyatakan bahwa text mining adalah bidang interdisipliner yang mengacu pada pencarian informasi, pertambangan data, pembelajaran mesin, statistic, dan komputasi linguistik.

Dari kedua kutipan diatas dapat diambil kesimpulan bahwa Text Mining adalah proses pencarian informasi, pertambangan data yang dilakukan dengan sistem komputerisasi dan menghasilkan informasi baru yang lebih terstruktur.

Perbedaan mendasar antara teks mining dan data mining terletak pada sumber data yang digunakan. Pada data mining, pola diekstrak dari basis data yang terstruktur, sedangkan di teks mining pola diekstrak dari data tekstual (natural language). Secara umum, basis data didesain untuk program dengan tujuan melakukan pemrosesan secara otomatis, sedangkan teks ditulis untuk dibaca langsung oleh manusia.

* + 1. **Tahapan dalam *Text Mining***

Tahapan dalam text mining yang dilakukan secara umum adalah:

Folding

Tokenizing

Filtering

Analyzing

Gambar 2.2 Tahapan Text Mining

1. *Folding*

Tahap *folding* adalah mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya huruf “a” sampai dengan “z” yang diterima

1. *Tokenizing*

Tahap *tokenizing* adalah tahap pemotongan string input berdasarkan tiap kata yang menyusunnya.

1. *Filtering*

Tahap *filtering* adalah tahap mengambil kata-kata penting dari hasil token.

1. *Analyzing*

Tahap *analyzing* merupakan tahap penentuan seberapa jauh keterhubungan antar kata-kata antar dokumen yang ada.

* + 1. **Pengertian *PHP* dan *MySql***

Betha Sidik (2012:4) mendefinisikan “*PHP* adalah bahasa pemrograman script – script yang membuat dokumen *HTML* secara on the fly yang dieksekusi di server web”

*PHP* seperti kita ketahui adalah bahasa pemrograman berbasis web. Bahasa ini mempunyai kelebihan yaitu kompabilitasnya dengan berbagai macam jenis *database* dan dukungan dengan berbagai macam jenis operasi.

*PHP* tidak terbatas hanya menghasilkan keluaran *HTML*. Ia juga bisa digunakan untuk menghasilkan gambar GIF, atau bahkan sumber gambar GIF yang dinamis.

* + 1. **Web Dinamis**

Pada awalnya web dibangun hanya dengan menggunakan bahasa disebut *HTML (Hypertext Markup Language)* dan protocol yang digunakan dinamakan *HTTP (Hypertext Transfer Protocol)*.

Menurut Ardhana (2012:3), Web adalah suatu layanan sajian informasi yang menggunakan konsep hyperlink, yang memudahkan *surfer* (sebutan para pemakai computer yang melakukan *browsing* atau penelusuran informasi melalui internet)

* + 1. **Worl Wide Web**

Pada awalnya internet adalah sebuah proyek yang dimaksudkan untuk menghubungkan para ilmuan dan peneliti di Amerika, namun saat ini telah tumbuh menjadi media komunikasi yang dipakai oleh seluruh umat manusia di muka bumi ini. Kemudian orang mulai berpikir membuat sesuatu yang lebih baik, popularitas internet semakin berkembang pesat setelah standar baru *HTTP* dan *HTML* diperkenalkan pada masyarakat.

*HTTP (Hypertext Transfer Protocol)* membuat pengaksesan informasi melalui protokol TCP/IP menjadi lebih mudah dari sebelumnya. *HTML* (*Hyperterxt Markup Language)* memungkinkan orang menyajikan informasi yang secara visual (tampil) lebih menarik. Pemunculan *HTTP* dan *HTML* yang memungkinkan orang mengenai istilah baru dalam internet yang sekarang menjadi sangat popular, bahkan sedemikian populernya sehingga identic dengan internet itu sendiri, yaitu *World Wide Web (WWW)*.

* + 1. ***Hypertext Markup Language (HTML)***

Menurut Imzen Sitorus (2012:7) “*HTML (Hypertext Markup Language)* adalah bahasa pemrograman dasar yang dipakai untuk menampilkan informasi pada halaman web”. Sesungguhnya *HTML* justru tidak dibuat untuk mempublikasikan informasi di *web,* namun oleh karena kesederhanaan penggunaannya, *HTML* kemudian dipilih orang untuk mendistribusikan informasi di *web.* Perintah – perintah *HTML* diletakkan dalam file berekstensi \*.html dan ditandai dengan mempergunakan *tag*(tanda).

* + 1. ***Browser* dan *Server Web***

Dalam dunia *web,* perangkat lunak *client* yaitu *browser web* mempunyai tugas yang sama yaitu menerjemahkan informasi yang diterima dari *server web* dan menampilkannya pada layar computer pengguna. Oleh karena itu *HTTP* memungkinkan *server web* mengirimkan beragam data, seperti teks atau gambar, *browser* harus bisa mengenali berbagai macam data yang akan diterimanya, dan selanjutnya harus tahu cara untuk menampilkannya dengan benar. Teks harus ditampilkan sebagai teks dan gambar harus ditampilkan sebagai gambar.

Umumnya *browser web* menerima data dalam bentuk *HTML.* File *HTML* sebenarnya adalah file teks biasa yang selain berisi informasi yang hendak ditampilkan kepada pengguna, juga mempunyai perintah – perintah untuk mengatur tampilan data tersebut. *Browser*lah yang memiliki kuasa penuh dalam menerjemahkan data – data tadi. Meskipun sudah dibuat consensus untuk menstandarkan format dan elemen-elemen *HTML*, setiap jenis *browser* bisa menerjemahkan file *HTML* yang sama secara berbeda.

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

Menurut Nugroho (2013:6), ”UML *(Unified Modeling Language)* adalah bahasa pemodelan untuk system atau perangkat lunak yang berparadigma (berorientasi objek).” Pemodelan *(modeling)* sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami.

UML sebagai sebuah bahasa yang memberikan *vocabulary* dan tatanan penulisan kata-kata dalam ‘*MS Word’* untuk kegunaan komunikasi. Sebuah bahasa model adalah sebuah bahasa yang mempunyai *vocabulary* dan konsep tatanan / aturan penulisan serta secara fisik mempresentasikan dari sebuah sistem. UML adalah sebuah bahasa standard untuk pengembangan sebuah *software* yang dapat menyampaikan bagaimana membuat dan membentuk model-model, tetapi tidak menyampaikan apa dan kapan model yang seharusnya dibuat yang merupakan salah satu proses implementasi pengembangan *software*.

*Unified Modelling Language* (UML) terbagi menjadi sembilan jenis diagram masing-masing memiliki aturan-aturan tertentu dalam penyusunannya. Diagram-diagram tersebut tersusun atas sejumlah elemen grafis saling membentuk satu kesatuan dalam pemodelan *software*. Masing-masing diagram UML mempresentasikan berbagai sudut pandang terhadap sistem dan mendefinisikan apa yang dikerjakan oleh sistem bukan bagaimana cara sistem bekerja.

Menurut Kimmel (2005:30) sembilan diagram UML tersebut dibagi menjadi tiga kelompok berdasarkan fungsinya yaitu :

1. *Diagram* untuk *requirement* dan desain, terdiri dari tujan diagram, *object diagram*, *sequence diagram*, *collaboration diagram*, *state diagram*.
2. Diagram mengenai organisasi umum software, terdiri dari satu diagram yaitu diagram package.
3. Diagram untuk implementasi, terdiri dari satu diagram, yaitu component dan deplopment diagram.

Berikut penjelasan beberapa diagram diantaranya :

1. *Use-Case Diagram*

Sukamto dan Shalahuddin (2013:155), “Use case atau diagram use case merupakan pemodelan untuk kelakuan (behavior) sistem informasi yang akan dibuat. Use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat”. Syarat penamaan pada use case adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada use case yaitu pendefinisian apa yang disebut aktor dan use case. *Use Case* diagram dapat digunakan selama proses analisa untuk menangkap requirement atau permintaan terhadap sistem dan untuk memahami bagaimana sistem tersebut harus bekerja:

Table 2.3 Simbol *Use Case Diagram*

| **NO** | **GAMBAR** | **NAMA** | **KETERANGAN** |
| --- | --- | --- | --- |
| 1. |  | *Actor* | Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*. |
| 2. |  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri*(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*). |
| 3. |  | *Generalization* | Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
| 4. |  | *Include* | Menspesifikasikan bahwa *use case* sumber secara *eksplisit*. |
| 5. |  | *Extend* | Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan. |
| 6. |  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
| 7. |  | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas |
| 8. |  | *Use Case* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor* |
| 9. |  | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi). |
| 10. |  | *Note* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi |

Sumber : Martin Fowler (2009:12)

1. *Activity Diagram*

Sukamto dan Shalahuddin (2013:161), “Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak”.

Table 2.4 Simbol *Activity Diagram*

| **NO** | **GAMBAR** | **NAMA** | **KETERANGAN** |
| --- | --- | --- | --- |
| 1. |  | *Actifity* | Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain |
| 2. |  | *Action* | State dari sistem yang mencerminkan eksekusi dari suatu aksi |
| 3. |  | *Initial Node* | Bagaimana objek dibentuk atau diawali. |
| 4. |  | *Actifity Final Node* | Bagaimana objek dibentuk dan dihancurkan |
| 5. |  | *Fork Node* | Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran |

Sumber : Martin Fowler*,* (2009:21)

1. *Sequence Diagram*

Sukamto dan Shalahuddin (2013:165), “Sequence diagram atau diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirim dan diterima antar objek”.

Tabel 2.5 Simbol *Sequence Diagram*

| **NO** | **GAMBAR** | **NAMA** | **KETERANGAN** |
| --- | --- | --- | --- |
| 1. |  | *LifeLine* | Objek *entity*, antarmuka yang saling berinteraksi. |
| 2. |  | *Message* | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |
| 3. |  | *Message* | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |

Sumber : Martin Fowler*,*(2009:25)

1. *Class Diagram*

Sukamto dan Shalahuddin (2013:141), “Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem”. Kelas memiliki apa yang disebut atribut dan metode atau operasi.”

Dalam kelas, seperti objek, adalah sesuatu yang membungkus (*encapsulate*) informasi (baca : atribut) dan perilaku (baca : operasi) dalam dirinya. Dalam pengembangan sistem tradisional, kita mengadakan pendekatan dengan cara memisahkan informasi-informasi pada sisi basisdata dan perilaku yang mengaksesnya di sisi aplikasi pemasup/pengakses. Simbol untuk *class diagram* dapat dilihat pada Tabel 2.6 berikut.

Tabel 2.6 Simbol *Class diagram*

| **NO** | **GAMBAR** | **NAMA** | **KETERANGAN** |
| --- | --- | --- | --- |
| 1. | [Description: http://2.bp.blogspot.com/-ONANWqL3deg/VR5DiCMv1pI/AAAAAAAAATk/yo54xjtEtgE/s1600/Snap%2B2015-04-03%2Bat%2B14.23.03.png](http://2.bp.blogspot.com/-ONANWqL3deg/VR5DiCMv1pI/AAAAAAAAATk/yo54xjtEtgE/s1600/Snap+2015-04-03+at+14.23.03.png) | Kelas | Kelas pada stuktur system |
| 2. | [Description: http://3.bp.blogspot.com/-KrBdyBTW1hc/VR5D8nFNnxI/AAAAAAAAATs/Z2C3ZZSctRE/s1600/Snap%2B2015-04-03%2Bat%2B14.40.20.png](http://3.bp.blogspot.com/-KrBdyBTW1hc/VR5D8nFNnxI/AAAAAAAAATs/Z2C3ZZSctRE/s1600/Snap+2015-04-03+at+14.40.20.png) | Antarmuka (*Interface*) | Sama dengan konsep *interface* dalam pemrograman berorientasi objek |
| 3. | [Description: http://4.bp.blogspot.com/-zHROgHCpdrs/VR4yxlZgtwI/AAAAAAAAAS4/wPElhz9EWeU/s1600/Snap%2B2015-04-03%2Bat%2B13.13.47.png](http://4.bp.blogspot.com/-zHROgHCpdrs/VR4yxlZgtwI/AAAAAAAAAS4/wPElhz9EWeU/s1600/Snap+2015-04-03+at+13.13.47.png) | Asosiasi (*Association*) | Relasi antar kelas dengan makna umum, asosiasi biasanya juga di sertai dengan *multiplicity*. |
| 4. | [Description: http://1.bp.blogspot.com/-GCiv3UbwSrY/VR5EVrJ5qYI/AAAAAAAAAT0/TN8iV3FI4qM/s1600/Snap%2B2015-04-03%2Bat%2B14.41.14.png](http://1.bp.blogspot.com/-GCiv3UbwSrY/VR5EVrJ5qYI/AAAAAAAAAT0/TN8iV3FI4qM/s1600/Snap+2015-04-03+at+14.41.14.png) | Asosiasi berarah (*Directed Association*) | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi berarah biasanya juga disertai dengan *multiplicity* |
| 5. | [Description: http://1.bp.blogspot.com/-cKyffPtNDlo/VR4z51D8JqI/AAAAAAAAATM/sCjErS4-O6g/s1600/Snap%2B2015-04-03%2Bat%2B13.14.52.png](http://1.bp.blogspot.com/-cKyffPtNDlo/VR4z51D8JqI/AAAAAAAAATM/sCjErS4-O6g/s1600/Snap+2015-04-03+at+13.14.52.png) | Generalisasi (*Generalization*) | Relasi antar kelas dengan makna generalisasi-spesialisasi (Umum-khusus) |
| 6. | [Description: http://3.bp.blogspot.com/-erqoC86gu38/VR5EqWRg5-I/AAAAAAAAAT8/PYCgrIvnajU/s1600/Snap%2B2015-04-03%2Bat%2B14.42.51.png](http://3.bp.blogspot.com/-erqoC86gu38/VR5EqWRg5-I/AAAAAAAAAT8/PYCgrIvnajU/s1600/Snap+2015-04-03+at+14.42.51.png) | Kebergantungan (*Dependency*) | Relasi antar kelas dengan makna kebergantungan antar kelas |
| 7. | [Description: http://1.bp.blogspot.com/-ElOpXzQNKiM/VR5EsB4cWaI/AAAAAAAAAUE/DS8j4PXysQg/s1600/Snap%2B2015-04-03%2Bat%2B14.43.20.png](http://1.bp.blogspot.com/-ElOpXzQNKiM/VR5EsB4cWaI/AAAAAAAAAUE/DS8j4PXysQg/s1600/Snap+2015-04-03+at+14.43.20.png) | Agregasi (*Aggregation*) | Relasi antar kelas dengan makna semua-bagian (*Whole-part*) |

Sumber : Munawar*,* (2005:33)