**BAB II**

**TINJAUAN PUSTAKA**

* 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.DPRD di daerah sering di sebut sebagai fungsi representatif karena bertugas menyuarakan aspirasi masyarakat dan bertindak atas nama rakyat (*representatif government*) di bidang legislatif.

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

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.

Diharapkan dengan penerapan aplikasi aspirasi masyarakat berbasis mobile 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. **Kerangka Teori**
     1. **Definisi Aplikasi**

Aplikasi adalah program siap pakai yang digunakan untuk menjalankan perintah-perintah dari pengguna aplikasi tersebut dengan tujuan untuk mendapatkan hasil yang yang lebih akurat sesuai dengan tujuan pembuatan aplikasi tersebut, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu tehnik pemprosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang di inginkan atau diharapkan maupun pemprosesan data yang diharapkan.

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.

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 *text mining* dan data mining terletak pada sumber data yang digunakan. Pada data mining, pola diekstrak dari basis data yang terstruktur, sedangkan di *text 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:

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. ***Cosine Similarity***

Menurut Manning, Raghavan, dan Schutze, (2008): “cosine similarity digunakan untuk mengukur kedekatan antara dua vektor.”

Fahmi Arif Dewoputro, Imam Much Ibnu Subroto, Sam Farisa Chaerul Haviana (2016: 39) : “Cosine similarity adalah ukuran kesamaan yang lebih umum digunakan dalam information retrieval dan merupakan ukuran sudut antara vector gejala (titik (A)) dan (titik (B))”.

Tiap vektor tersebut merepresentasikan setiap gejala dalam setiap penyakit yang dibandingkan, sehingga dapat diterapkan hukum kosinus untuk menyatakan bahwa:

Dimana

A . B = Vektor *dot product* dari A dan B di hitung dengan ∑ 𝐴. 𝐵

Dan

||A|| = Panjang dari vektor A, di hitung dengan

||B|| = Panjang dari vektor B, di hitung dengan

Maka dapat di rumuskan seperti berikut :

=

*Cosine Similarity* dapat diimplementasikan untuk menghitung nilai kemiripan antar kalimat dan menjadi salah satu teknik untuk mengukur kemiripan teks yang popular. Contoh penggunaan Cosine Similarity dalam menguji kemiripan dua buah kalimat adalah sebagai berikut:

Diberikan dua buah kalimat yaitu kalimat A dan B, yaitu:

A : Julie loves me more than Linda loves me

B : Jane likes me more than Julie loves me

Tabel 2. Uji kemiripan teks

|  |  |  |  |
| --- | --- | --- | --- |
| **indek** | **Daftar Kata** | **Jumlah Kemucunlan kata** | |
| A | **B** |
| 1 | Julie | 1 | 1 |
| 2 | Loves | 2 | 1 |
| 3 | Me | 2 | 2 |
| 4 | More | 1 | 1 |
| 5 | Than | 1 | 1 |
| 6 | Linda | 1 | 0 |
| 7 | Jane | 0 | 1 |
| 8 | Likes | 0 | 1 |

Berdasarkan rumus tersebut di atas dilakukan penghitungan seperti di bawah ini.

Tingkat kemiripan teks =

=

= 0.821584

Setelah dilakukan perhitungan dan menghasilkan nilai dari tingkat kemiripan teks, nilai tertinggi dari hasil perhitungan cosine similarity akan menjadi range atau nilai maksimal. Nilai yang diatas 75% dari nilai tertinggi dikatakan sebagai multi komisi atau dapat masuk lebih dari 1 komisi.

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

Betha Sidik (2012:4) *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. ***World 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. i 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:

Tabel 2. 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”.

Tabel 2. 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. 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 2.5 berikut.

Tabel 2. 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)

* + 1. **Pengujian *Black Box***

Menurut Gianty.(2016:27):

Pengujian *black box* testing berfokus pada persyaratan fungsional perangkat lunak. Dengan demikian, pengujian *black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Pengujian *black-box* bukan merupakan alternatif dari teknik *white-box*, tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan daripada metode *white-box*.

*Black box testing* mengasumsikan kode menjadi sebuah *blackbox* yang merespon berbagai inputan. Pengujian berfokus pada *output* dari berbagai jenis inputan. Pengujian ini juga berfokus pada tes validasi, batas masalah, tes kinerja, dan pengujian yang berhubungan dengan keamanan.

*Black box testing* melibatkan pengujian *interface* untuk memastikan bahwa kode tersebut memenuhi persyaratan fungsional dan berfungsi. Beberapa alasan penggunaan pengujian *blackbox* pada penelitian ini adalah:

1. Lebih mudah dilakukan dibandingkan dengan metode pengujian lainnya.
2. Pengujian terhadap sistem dapat dilakukan tanpa harus menelusuri kedalam kode program.

Dengan mengamati *output* sistem berdasarkan *input* yang diberikan dan *requirement*, mudah untuk mengetahui adanya kesalahan dan kekeliruan pada suatu sistem.

* + 1. ***White Box***

Menurut Rizky (2011:261), “White box testing secara umum merupakan jenis testing yang lebih berkonsentrasi terhadap isi dari perangkat lunak itu sendiri”. Jenis ini lebih banyak berkonsentrasi kepada *source code* dari perangkat lunak yang dibuat sehingga membutuhkan proses testing yang jauh lebih lama dan lebih “mahal” dikarenakan membutuhkan ketelitian dari para tester serta kemampuan teknis pemrograman bagi paratesternya.

* + 1. **Dewan Perwakilan Rakyat Daerah (DPRD)**

Pasal 1 ayat (4) Undang-Undang Nomor 32 Tahun 2004 tentang Pemerintahan Daerah menyebutkan bahwa pengertian Dewan Perwakilan Rakyat Daerah yang selanjutnya disebut DPRD adalah lembaga perwakilan rakyat daerah sebagai unsur penyelenggara pemerintah daerah. Miriam Budiarjo dalam Baskoro (2005;30) menyebutkan DPRD adalah lembaga legislate atau membuat peraturan, peraturan perundang-undangan yang dibuatnya mencerminkan kebijakan-kebijakan itu.

Pada DPRD Kota Makassar terdapat beberapa komisi yang masing – masing mempunyai tugas yang berbeda yaitu Komisi A Bidang Hukum, Pemerintahan, Dan Aset, Komisi B bertugas pada Bidang Keuangan Dan Ekonomi, Komisi C Bidang Pembangunan, dan Komisi D bertugas pada Bidang Pendidikan Dan Kesejahteraan Masyarakat

Dalam sekretariat DPRD Kota makassar dibentuk salah satu sub bagian yaitu Hubungan Masyarakat (Humas). Lembaga DPRD Kota Makassar telah lama berdiri dan Humas DPRD Kota Makassar terbentuk menjadi satu sub bagian pada tahun 2000 serta ditambahkan ruang aspirasi dalam Humas pada tahun 2007.

Aktivitas Humas DPRD Kota Makassar banyak berhubungan dengan masyarakat, mediator antara masyarakat dengan lembaga antara lain humas menjembatani aspirasi masyarakat ke komisi terkait. Humas turut serta dengan anggota DPRD untuk melakukan reses dan mempublikasi kegiatan antara lain dokumentasi, mengkliping koran yang memuat segala berita yang berhubungan dengan DPRD Kota Makassar.

* + 1. **Java**

Java adalh bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Aplikasi-aplikasi berbasis Java umumnya dikompilasi ke dalam p-code (*bytecode*) dan dapat dijalankan pada berbagai *Mesin Virtual Java* (JVM). Java bersifat umum/ non-spesifik (*general purpose*), dan secara khusus didesain untuk memanfaatkan ketergantungan implementasi seminimal mungkin. Dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi aplikasi berbasis web.

* + 1. **Web Service**

Menurut World Wide Web Consortium (W3C) selaku badan penemu dan pengembangan service, web service adalah salah satu bentuk sistem perangkat lunak yang didesain mendukung interaksi mesin ke mesin melalui jaringan. Sistem web service memungkinkan sebuah fungsi didalam web service dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detail pemrograman yang terdapat di dalamnya.

Secara umum, arsitektur web service terdiri dari 3 komponen, yaitu:

1. **Service provider**, merupakan pemilik web service yang berfungsi menyediakan kumpulan operasi dar web service.
2. **Service request**, merupakan aplikasi yang bertindak sebagai klien dari web service yang mencari dan memulai interaksi terhadap layanan yang disediakan.
3. **Service registry**, merupakan tempat dimana *service provider* mempublikasikan layanannya. Pada arsitektur *web service*, *service registry* bersifat optional, teknologi *web service* memungkinkan kita dapat menghubungkan berbagai jenis *software* yang memiliki platform dan sistem operasi yang berbeda.

Ada 2 jenis *web service* yaitu REST dan SOAP (*Simple Oboject Acces Protocol*) merupakan protokol untuk saling bertukar pesan antar aplikasi. Spesifikasi format pesan tersebut didefinisikan seperti amplop berbasis XML yang dikirim beserta aturan-aturan atau cara untuk menerjemahkan representasi data dari XML.

Cara kerja SOAP ialah, aplikasi klien mengirim bentuk XML kepada *provider web service*. *Web service* menerima *request* tersebut, menjalankan *service*, kemudian mengirimkan respon ke aplikasi klien juga dalam bentuk XML. Baik request maupun response keduanya menggunakan protokol SOAP.

REST adalah *web service* yang menerapkan konsep perpindahan antar state dimana dalam bernavigasi REST melalui link HTTP untuk melakukan aktivitas tertentu. Dalam pengaplikasiannya REST banyak digunakan untuk *web service* yang berorientasi pada *resource*. Maksud orientasi pada *resource* adalah orientasi yang menyediakan *resource* sebagai layanannya dan bukan kumpulan dari aktifitas yang mengolah *resource* itu. Response dari web service REST dapat berupa XML atau JSON.

* + 1. **JSON**

JSON(*Javascript Object Notation*) merupakan format yang ringan untuk memasukkan data ke dalam sebuah variabel. Sangat mudah dimengerti dan diimplementasikan oleh manusia, dan mudah juga untuk computer dalam melakukan parsingnya. JSON dibangun dalam dua struktur:

1. Beberapa pasangan dari nama/nilai. Dalam beberapa pemrograman biasa disebut dengan istilah *object, record, struct*, tabel hash, *key list*, atau *associative array.*
2. Nilai-nilai yang tersusun secara ordered list. Biasa disebut dengan *array, vector, list,* atau daftar dalam bahasa pemrograman.

JSON adalah struktur data yang universal, dalam artian bisa digunakan dalam berbagai bahasa pemrograman. Hamper semua bahasa pemrograman mendukung penuh JSON dalam berbagai format. Hal ini memungkinkan format data yang dapat dipertukarkan menggunakan bahasa pemrograman juga menggunakan dasar dari struktur JSON.

* + 1. **Android**

Android adalah sebuah sistem operasi untuk perngkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware,* dan aplikasi. Android menyediakan *platform* yang terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel / *smartphone*. Android merupakan generasi baru *platform mobile* yang memberikan pengembangan untuk melakukan pengembangan sesuai dengan yang diharapkannya. Sistem operasi yang mendasari Android dilisensikan dibawah GNU, *General Public* Lisensi versi 2 (GPLv2), yang sering dikenal dengan istilah “*copyleft”* lisensi dimana setiap perbaikan pihak ketiga harus terus jatuh dibawah terms. Android didistribusikan dibawah lisensi *Apache Software* (ASL/Apache2), yang memungkinkan untuk distribusi kedua dan seterusnya.

* + 1. ***Flowchart***

Siallagan (2009:6), *Flowchart* adalah suatu diagram alir yang mempergunakan simbol-simbol atau tanda untuk menyelesaikan masalah. Gambaran simbol-simbol *Flowchart* sebagai berikut:

Tabel 2. Simbol *Flowchart*

| **No** | **Simbol Flowchart** | **Fungsi Flowchart** |
| --- | --- | --- |
| 1 |  | Terminal menyatakan awal atau akhir dari suatu algoritma |
| 2 |  | Menyatakan proses. |
| 3 |  | Proses yang terdefinisi atau sub program |
| 4 |  | Persiapan yang digunakan untuk memberi nilai awal suatu besaran. |
| 5 |  | Menyatakan masukan dan keluaran (input/output). |
| 6 |  | Menyatakan penyambungan ke symbol lain dalam satu halaman. |
| 7 |  | Menyatakan penyambung ke halaman lainnya |
| 8 |  | Menyatakan percetakan (dokumen) pada kertas. |
| 9 |  | Menyatakan *decision* (keputusan) yang digunakan untuk penyeleksian kondisi di dalam program. |
| 10 |  | Menyatakan media penyimpanan drum mangnetik. |
| 11 |  | Menyatakan input/output menggambarkan disket. |
| 12 |  | Menyatakan operasi yang dilakukan secara manual. |
| 13 |  | Menyatakan input/output dari kartu plong. |
| 14 |  | Menyatakan arah aliran pekerjaan (proses) |
| 15 |  | *Multidocument* (banyak dokumen) |
| 16 |  | Delay (penundaan atau kelambatan) |

(*Sumber: Siallagan, 2009:6-7*)

* 1. **Penelitian Terkait**

Berikut daftar penelitian terkait yang kami lakukan diantaranya :

1. Penelitian oleh Budi Yanto, Jurnal DASI Vol. 14 No. 2 ISSN: 1411-3201 tahun 2013 dengan judul “Perancangan Aplikasi Online ‘JOGJA PEDULI’ Berbasis Mobile untuk Penjaringan Aspirasi Publik Terhadap Infrastruktur Sarana dan Prasarana Jalan Dalam Perkotaaan Daerah Istimewa Yogyakarta”. Pada penelitian ini peneliti menyajikan usulan sistem dari pemerintah untuk masyarakat mencari informasi sarana prasarana serta memberikan kritik untuk meningkatkan layanan serta mempercepat informasi kerusakan sarana dan prasarana sehingga bisa menjadwalkan perbaikan layanan pemerintah.

Perbedaan penelitian diatas dengan penelitian kami yaitu, pada penelitian tersebut hanya menampung aspirasi publik pada bidang infrastruktur jalan di kota Yogyakarta. Sedangkan penelitian yang kami lakukan menampung semua aspirasi publik dari semua bidang yang nantinya akan di kalsifikasikan menggunakan implementasi algoritma *Text Mining* dan *Cosine Similarity.*

1. Penelitian oleh Imam Adiyana dan Fajriya Hakim ISBN : 978.602.361.002.0 tahun 2015 dengan judul “Implementasi *Text Mining* pada Mesin Pencarian Twitter Untuk Menganalisis Topik-topik Terkait ‘KPK dan JOKOWI’ “. Pada penelitian ini peneliti membahas mengenai penerapan metode text mining untuk data tweet terkait topic KPK dan topic Jokowi, dimana didapatkan beberapa informasi yang bermanfaat seperti keseringan penggunaan kata-kata menurut aturan asosiasi yang menyertai kata KPK adalah kata Polri dan Lapor, serta kata Jokowi adalah kata Widodo, menghadiri, izin, pintu, satu, investor, urus, presiden, nilai, aktif, bahaya, maneuver, menang, mulai, relawan, dan sejumlah. Dalam penelitian ini juga membahas kesamaan topic utama yang membahas mengenai topic KPK dan Jokowi pada tweet yaitu topic utama KPK dan Polri. Perbedaan dengan penelitian yang kami lakukan yaitu pada penelitian ini hanya menggunakan algoritma Text Mining sedangkan pada penelitian kami menggunakan 2 algoritma yaitu *Text Mining* dan *Cosine Similarity* dalam pengklasifikasian aspirasi publik.
2. Penelitian oleh Annisya Aprilia Prasasanti, M. Ali Fauzi, dan M. Tanzil Furqon Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X Vol. 2, No. 2 tahun 2018 dengan judul “Klasifikasi Teks Pengaduan Pada Sambat Online Menggunakan Metode N-Gram dan Neighbor Weighted K-Nearest Neighbor (NW-KNN)”. Pada penelitian ini peneliti membahas tentang penerapan metode NW-KNN pada website SAMBAT online yang disedikan oleh Dinas Komunikasi dan Informatika Kota Malang (Diskominfo Malang). Hasil pengujian yang didapatkan dalam penelitian ini menjukkan bahwa penggunaan metode NW-KNN dengan nilai tetangga *k =* 3 dan Metode N-Gram dengan Unigram memiliki nilai *f-measure* tertinggi sebesar 75.25%. perbedaan pada penelitian kami yaitu pada penelitian ini menggukan banyak metode dan beberapa metode tidak memiliki pengaruh terhadap hasil klasifikasi. Sedangkan pada penelitian kami menggunakan 2 metode yang masing masing memiliki pengaruh dalam pengklasifikasian aspirasi publik.
3. Penelitian oleh Panggasa dan Ruri Suko Basuki, Universitas Dian Nuswantoro Semarang tahun 2015 dengan judul “ Klasifikasi Pengaduan Masyarakat Pada LaporGub Menggunakan Algoritma *Stemming Porter* dan *Naïve Bayes Classifier*”. Pada penelitian ini peneliti mengembangkan sistem dengan metode *Prototype* menggunakan algoritma *Stemming Porter* untuk *text preprocessing* dan *Naïve Bayes Classifier*. Pada penelitian ini sistem klasifikasi aduan berhasil mengklasifikasikan 300 aduan kedalam 17 kategori dengan akurasi optimal 84% dengan dokumen uji dan dokumen latih masing-masing 250 dokumen. Pada penelitian ini menggunakan metode *Stemming Porter* dan *Naïve Bayes Classifier*. Sedangkan pada penelitian kami menerapkan algoritma *Text Mining* dan *Cosine Similarity* dalam pengklasifikasian aspirasi publik.
4. Penelitian oleh Chyntia Megawati, Universitas Indonesia tahun 2015 dengan judul “Analisis Aspirasi dan Pengaduan di Situs LAPOR! Dengan Menggunakan *Text Mining”.* Pada penelitian ini peneliti menggunakan metode *Text Mining* untuk menganalisis data tekstual yang berupa opini atau keluhan dengan mengklasifikasikannya menjadi beberapa kelas kemudian data set setiap kelas akan dikelompokkan lagi menjadi beberapa topik khusus (*cluster*). Hasil penelitian menunjukkan bahwa laporan terkait kemiskinan memiliki jumlah terbanyak dengan topic mayoritas yang dibahas adalah mengenai beberapa jenis bantuan social KPS (Kartu Perlindungan Sosial) dan BLSM (Bantuan Lansung Sementara Masyarakat) yang tidak didistribusikan dengan baik atau tidak tepat sasaran.

Pada penelitian kami lebih berfokus bagaimana menyalurkan aspirasi ke komisi yang tepat dengan menggunakan metode *text mining* dan *cosine simiarity*

1. Penelitian oleh Loura Yasni, Imam Much Ibnu Subroto dan Sam Farisa Chaerul Haviana tahun 2018 dengan judul “Implementasi cosine similarity matching dalam penentuan dosen pembimbing tugas akhir”. Pada penelitian tersebut menerapkan algoritma *cosine similarity* dalam penentuan dosen pembimbing tugas akhir supaya mendapatkan proses bimbingan yang optimal yaitu dengan menghitung nilai tingkat kesamaan antar judul, topik, dan abstrak tugas akhir mahasiswa dibandingkan dengan data dosen pembimbing berupa keahlian dosen pembimbing, tugas akhir yang pernah dibimbing oleh dosen. Kemudian metode *Cosine Similarity* akan memnghitung tingkat kesamaan kedua query tersebut. Nilai kemiripan yang tertinggi akan dimunculkan sebagai dosen pembimbing yang direkomendasikan. Pada penelitian kami akan meghitung nilai tingkat kesamaan antara fungsional setiap komisi dengan aspirasi yang di *input* masyarakat.