**PENERAPAN *TEXT MINING* UNTUK PENGELOLAAN IDENTIFIKASI**

**KELUHAN PENGGUNA SECARA OTOMATIS DI**

**POLITEKNIK NEGERI INDRAMAYU**

# Tugas Akhir

diajukan untuk memenuhi sebagian persyaratan mencapai gelar Ahli Madya jenjang Diploma III

Jurusan Teknik Informatika



Oleh :

**ELBA AYU KURNIA**

**NIM. 1703056**

**JURUSAN TEKNIK INFORMATIKA POLITEKNIK NEGERI INDRAMAYU 2020**

# HALAMAN PENGESAHAN

Tugas Akhir ini diajukan oleh :

Nama : Elba Ayu Kurnia

NIM : 1703056

Program Studi : D3 Teknik Infotmatika

Judul : Penerapan *Text Mining* Untuk Pengelolaan Identifikasi

Keluhan Pengguna di Politeknik Negeri Indramayu

Pembimbing : 1. Iryanto, S.Si.,M.Si. ..................................

2. Alifia Puspaningrum, S.Pd., M.Kom.  ..................................

Telah berhasil dipertahankan dihadapan dewan penguji pada tangal 16 Agustus 2020 dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Ahli Madya Program Studi Teknik Informatika, Jurusan Teknik Informatika, Politeknik Negeri Indramayu.

DEWAN PENGUJI

Nama Jabatan Tandatangan Tanggal

1. Ketua Penguji ................ ................

2. Sekretaris Penguji ................. ................

3. Anggota ................. ................

Indramayu, Agustus 2020

Ketua Jurusan Teknik Informatika

**Iryanto, S.Si.,M.Si**

NIP. 199008012019031014

# MOTTO

“W*hen you started it, then you have to finished it*. Karena itu adalah salah satu candradimuka untuk membentukmu menjadi pribadi yang bertanggung jawab pada apa yang kamu pilih untuk dirimu sendiri, terhadap orang lain juga, dan apa yang sudah dilakukan orang tua terhadap kalian.”

(Hanum Salsabiela Rais)

# PERNYATAAN KEASLIAN

Saya menyatakan dengan sesungguhnya bahwa Tugas Akhir ini adalah asli hasil karya saya sendiri, dan sepanjang pengetahuan saya tidak terdapat karya atau pendapat yang pernah ditulis atau dipublikasikan oleh orang lain, kecuali yang secara tertulis disebutkan sumbernya dalam naskah dan dalam daftar pustaka.

Indramayu, 16 Agustus 2020 Yang Menyatakan,

ELBA AYU KURNIA

NIM. 1703056

# ABSTRAK

Dalam sebuah perusahaan atau instansi, kepuasan pelanggan adalah salah satu hal yang dibutuhkan untuk meningkatkan kualitas perusahaan atau instansi itu sendiri.. Penanganan komplain saat ini dinilai masih kurang karena bersifat manual. Bukan hal sulit jika proses tersebut dilakukan oleh manusia, hanya saja memerlukan waktu yang tidak sedikit. Oleh karena itu, dengan menggunakan *text mining* dalam tugas akhir ini diharapkan dapat membantu dalam memilah atau mengetahui kategori dari sebuah komplain. Dengan *Text mining* dapat dicari kata-kata yang dapat mewakili isi dari komplain, lalu dianalisis apakah komplain tersebut masuk dalam kategori suatu unit kerja atau tidak, untuk selanjutnya di *broadcast* agar keluhan dapat segera ditinjau. Agar hasil pengukuran tingkat *similaritas* dokumen dengan *keyword* mendapatkan hasil yang optimal maka digunakanlah algoritma *CountVectorizer* yang dapat mengubah fitur teks menjadi sebuah representasi vector*.* Dari data teks yang kita miliki (atau biasa disebut korpus) kita dapat menyusun sebuah *vocabulary*, kita kemudian menjadikan setiap data menjadi representasi *vector* masing - masing untuk setiap kata. Tiap elemen dari *vector* menunjukkan jumlah fitur kata yang ada pada data. Sedangkan untuk pengklasifikasian kategori digunakanlah metode Support Vector Machine (SVM) digunakan untuk mencari hyperplane terbaik dengan memaksimalkan jarak antar kelas. Hyperplane adalah sebuah fungsi yang dapat digunakan untuk pemisah antar kelas. Dalam 2-D fungsi yang digunakan untuk klasifikasi antar kelas disebut sebagai line whereas, fungsi yang digunakan untuk klasifikasi antas kelas dalam 3-D disebut plane similarly, sedangan fungsi yang digunakan untuk klasifikasi di dalam ruang kelas dimensi yang lebih tinggi di sebut hyperplane. Dalam pemodelan klasifikasi, SVM memiliki konsep yang lebih matang dan lebih jelas secara matematis dibandingkan dengan teknik-teknik klasifikasi lainnya. SVM juga dapat mengatasi masalah klasifikasi dan regresi dengan linear maupun non linear.

Kata kunci : *text mining*, komplain, support vector machine*, count vectorizer.*

## ABSTRACT

In a company or agency, customer satisfaction is one thing that is needed to improve the quality of the company or agency itself. Handling complaints is currently considered lacking because it is manual. It's not difficult if the process is done by humans, it's just that it takes a lot of time. Therefore, using text mining in this thesis is expected to help in sorting or knowing the category of a complaint. Text mining can search for words that can represent the contents of the complaint, then analyze whether the complaint is included in the category of a work unit or not, for further broadcast so that complaints can be immediately reviewed. So that the results of the measurement of the level of similarity of documents with keywords get optimal results, the CountVectorizer algorithm is used that can convert text features into a vector representation. From the text data that we have (or commonly called a corpus) we can compile a vocabulary, we then make each data into a vector representation each for each word. Each element of the vector indicates the number of word features present in the data. As for the classification of categories, the Support Vector Machine (SVM) method is used to find the best hyperplane by maximizing the distance between classes. Hyperplane is a function that can be used for separating between classes. In 2-D functions used for class classifications are referred to as lines whereas, functions used for class classifications in 3-D are called plane similarly, whereas functions used for classifications in classrooms of higher dimensions are called hyperplanes. In classification modeling, SVM has a more mature and more mathematically clear concept compared to other classification techniques. SVM can also overcome the problem of classification and regression with linear or non linear.

Keywords: text mining, complaints, support vector machine, count vectorizer

# KATA PENGANTAR

*Alhamdulillahi Rabbil ’Alamin*, puji syukur saya panjatkan kepada Allah *Subhanahu wa Ta’ala* atas rahmat dan karunia-Nya sehingga saya dapat menyelesaikan Tugas Akhir ini dengan tepat waktu.

Tugas akhir ini merupakan salah satu persyaratan memperoleh gelar Ahli Madya pada jenjang D3 Jurusan Teknik Informatika Politeknik Negeri Indramayu. Tugas Akhir ini berjudul ‘Penerapan *Text Mining* Untuk Pengelolaan Identifikasi Keluhan Pengguna Secara Otomatis di Politeknik Negeri Indramayu’.

Tugas Akhir yang dibuat oleh penulis bertujuan untuk mempermudah unit kerja Pusat Pengembangan Pembelajaran dan Penjaminan Mutu Pendidikan (P4MP) Politeknik Negeri Indramayu dalam mengelola komplain, guna meningkatkan mutu atau kualitas Politeknik Negeri Indramayu.

Dalam penyusunan laporan Tugas Akhir ini penulis telah banyak mendapatkan dukungan dari berbagai pihak, oleh sebab itu penulis ingin mengungkapkan rasa terima kasih kepada :

1. Kedua orang tua penulis yang selalu mendoakan dan memberikan semangat kepada penulis agar dapat menyelesaikan Tugas Akhir dengan lancar dan tepat waktu .
2. Bapak Casiman Sukardi, S.T., M.T selaku Direktur Politeknik Negeri Indramayu.
3. Bapak Iryanto,S.Si.,M.Si.,M.Sc selaku ketua jurusan Teknik Informatika Politeknik Negeri Indramayu sekaligus selaku Dosen Pembimbing I .
4. Ibu Alifia Puspaningrum, S.Pd., M.Kom. selaku Dosen Pembimbing II.

Dan semua pihak yang telah membatu penyelesaian Tugas Akhir ini. Semoga Allah SWT senantiasa memudahkan setiap langkah-langkah kita menuju kebaikan dan selalu menganugerahkan kasih sayang-Nya untuk kita semua. Amin.

Demikian apa yang dapat penulis sampaikan. Semoga laporan Tugas Akhir ini dapat bermanfaat bagi kita semua.

Indramayu, 15 Agustus 2020

Penulis

# DAFTAR ISI

# DAFTAR TABEL

# DAFTAR GAMBAR

# DAFTAR LAMPIRAN

# BAB I PENDAHULUAN

# Latar Belakang Masalah

Dalam sebuah perusahaan maupun instansi, kepuasan pelanggan adalah salah satu hal yang dibutuhkan untuk meningkatkan kualitas perusahaan atau instansi itu sendiri. Penanganan komplain saat ini dinilai masih kurang karena bersifat manual. Hal ini menyebabkan pelanggan mengeluh karena komplain mereka tidak diproses secara cepat oleh pihak yang bersangkutan. Bukan hal sulit memang jika proses tersebut dilakukan oleh manusia, hanya saja memerlukan waktu yang tidak sedikit.

Namun, dengan berkembangnya teknologi proses tersebut dapat dilakukan oleh komputer. Akan tetapi ketika dilakukan oleh komputer tentu saja akan terdapat masalah baru, yaitu dapatkah komputer menentukan kategori komplain. (Rudhi, 2018).

Oleh karena itu, dengan menggunakan *text mining* dalam tugas akhir ini diharapkan dapat membantu dalam memilah atau mengetahui kategori dari sebuah komplain. Dengan *Text mining* dapat dicari kata-kata yang dapat mewakili isi dari komplain, lalu dianalisis apakah komplain tersebut masuk dalam kategori suatu unit kerja tertentu atau tidak. Oleh karena itu, dengan menggunakan *text mining* dalam tugas akhir ini diharapkan dapat membantu dalam memilah atau mengetahui kategori dari sebuah komplain tersebut.

Sedangkan pengukuran tingkat *similaritas*  antar dokumen dilakukan dengan membandingkan suatu *keyword*  dengan dokumen yang sudah dibuat sbelumnya atau bisa disebut *datasets*. Agar hasil pengukuran tingkat *similaritas* dokumen dengan *keyword* mendapatkan hasil yang optimal maka digunakanlah algoritma *CountVectorizer* yang dapat mengubah fitur teks menjadi sebuah representasi vector*.*

Dari data teks yang kita miliki (atau biasa disebut korpus) kita dapat menyusun sebuah *vocabulary*, kita kemudian menjadikan setiap data menjadi representasi *vector* masing - masing untuk setiap kata. Tiap elemen dari *vector* menunjukkan jumlah fitur kata yang ada pada data. Sedangkan untuk pengklasifikasian kategori digunakanlah metode Support Vector Machine (SVM) digunakan untuk mencari hyperplane terbaik dengan memaksimalkan jarak antar kelas.

Hyperplane adalah sebuah fungsi yang dapat digunakan untuk pemisah antar kelas. Dalam 2-D fungsi yang digunakan untuk klasifikasi antar kelas disebut sebagai line whereas, fungsi yang digunakan untuk klasifikasi antas kelas dalam 3-D disebut plane similarly, sedangan fungsi yang digunakan untuk klasifikasi di dalam ruang kelas dimensi yang lebih tinggi di sebut hyperplane. Dalam pemodelan klasifikasi, SVM memiliki konsep yang lebih matang dan lebih jelas secara matematis dibandingkan dengan teknik-teknik klasifikasi lainnya. SVM juga dapat mengatasi masalah klasifikasi dan regresi dengan linear maupun non linear.

# Rumusan Masalah

Berdasarkan latar belakang yang dijabarkan, maka diperoleh suatu rumusan permasalahan yang menjadi dasar pembuatan sistem tersebut, yakni sebagai berikut :

1. Bagaimana membuat sebuah aplikasi komplain yang dapat mengelompokkan komplain berdasarkan kategori atau *class* nya?
2. Bagaimana algoritma CountVectorizer dapat digunakan dalam proses penentuan tingkat *similiritas* yang sesuai dengan cara mengukur tingkat *similiritas* antar dokumen dengan membandingkan suatu *keyword* dengan *template* komplain yang sudah dibuat sebelumnya ?
3. Bagaimana metode *Support Vector Mechine* (SVM) dapat mengkasifikasi text keluahan ke dalam label atau kategori yang telah dibuat ?

# Batasan Masalah

Untuk memfokuskan pembahasan, dapat diperoleh beberapa batasan masalah,

diantaranya:

1. Data latih di dapat dari responden yang merupakan mahasiswa di Politeknik Negeri Indramayu.
2. Sistem tidak dapat mengidentifikasi data dukung (gambar) yang di inputkan oleh user apabila ternyata hasil identifikasi keluhannya memiliki 2 kategori.
3. *Delimiters* digunakan pada *system* ini untuk memisahkan teks keluhan karena ada kemungkinan dalam 1 inputan keluhan ditujukan pada lebih dari satu kategori unit kerja. Adapun *delimiters* yang digunakan hanya 3 yaitu :
4. Titik (.)
5. Koma (,)
6. Dan
7. Apabila user menginputkan keluhan yang tidak sesuai dengan keempat kategori keluhan (unit kerja) yang telah dibuat, maka kaluhan tersebut masuk kedalam kategori secara acak (*random*) sesuai dengan prediksi dari model yang telah dibuat. Namun permasalahan ini dapat di *handle* dengan adanya fitur *Report /* Laporkan.

# Tujuan Penelitian

Tujuan dari pembuatan sistem yang dibuat meliputi :

1. Merancang dan membangun aplikasi *Automatic Complaint Management System*, untuk mempermudah pencatatan komplain dan mengidentifikasi kategori komplain.
2. Mengimplementasikan algoritma *CountVectorizer* dan *Support Vector Mechine* (SVM) untuk memenuhi syarat utama pembuatan model *text mining* yaitu *feature extraction* dan *classifier*.

# Manfaat Penelitian

* + 1. Manfaat bagi penulis :

Menerapkan ilmu pengetahuan yang didapat dari perkuliahan, melatih kreatifitas, menambah wawasan dan pengalaman untuk membuat suatu sistem.

* + 1. Manfaat bagi pengguna :

Mendapatkan keluhan sesuai *role* atau unit kerja yang dipegang, dan keluan yang sudah dianalisis tadi dapat dimonitor atau ditinjau dengan mudah melalui system yang dibuat.

* + 1. Manfaat bagi pembaca :

Sebagai referensi pembaca untuk membuat suatu apikasi yang berkaitan dengan *text mining,* *CountVectorizer* dan *Support Vector Mechine* (SVM).

# Sistematika Penulisan Laporan

Untuk memahami lebih jelas laporan ini, maka materi-materi yang tertera pada laporan tugas akhir ini dikelompokkan menjadi beberapa bab dengan sistematika penyampaian sebagai berikut :

# BAB I PENDAHULUAN

Pada bab ini berisi pemaparan latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan laporan.

# BAB II LANDASAN TEORI

Bab ini berisi teori dasar yang menguraikan tentang kecerdasan buatan, *text mining*, metode *CountVectorizer* dan *Support Vector Mechine* (SVM), *UML* dan teknik pengujian perangkat lunak.

# BAB III METODOLOGI PENELITIAN

Bagian ini berisi penjelasan tentang metode penelitian, metode pengumpulan data, analisa data, analisa kebutuhan sistem, *flowchart*, perancangan sistem, perancangan *ERD ,* perancangan *database,* perancangan desain antarmuka dan perancangan *text mining*.

# BAB IV HASIL DAN PEMBAHASAN

Pada bab ini berisi hasil dan pembahasan tentang implementasi *database*, implementasi desain antarmuka dan pengujian sistem.

# BAB V PENUTUP

Bab ini berisi kesimpulan dan saran penulis.

# BAB II LANDASAN TEORI

# Kecerdasan Buatan

Pada dasarnya AI adalah suatu pengetahuan yang membuat komputer dapat meniru kecerdasan manusia sehingga komputer dapat melakukan hal-hal yang dikerjakan manusia dimana membutuhkan suatu kecerdasan; misalkan melakukan analisa penalaran untuk mengambil suatu kesimpulan atau keputusan atau penerjemahan dari satu bahasa ke bahasa lain. Kecerdasan buatan merupakan suatu cabang dalam bidang sains komputer Sains yang mengkaji tentang bagaimana untuk melengkapi sebuah komputer dengan kemampuan atau kepintaran seperti manusia. Sebagai contoh, bagaimana komputer bisa belajar sendiri dari pengalaman dan data-data yang telah dikumpulkannya, bagaimana komputer mampu berkomunikasi dan mengucapkan kata demi kata. Dengan kemampuan ini, diharapkan komputer mampu mengambil keputusan sendiri untuk berbagai kasus yang ditemuinya.

(Wicaksono T dkk, 2011)

# *Text Mining*

Text mining adalah salah satu bidang khusus dalam data mining yang memiliki definisi menambang data berupa teks dimana sumber data biasanya didapatkan dari dokumen dan tujuannya adalah mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisa keterhubungan antar dokumen.

Text mining dapat menganalisadokumen, mengelompokkan dokumen berdasarkan kata-kata yang terkandung di dalamnya, serta menentukan kesamaan di antara dokumen untuk mengetahui bagaimana mereka berhubungan dengan variabel lainnya. Penerapan yang paling umum dilakukan text mining saat ini misalnya penyaringan spam, analisa sentimen, mengukur preferensi pelanggan, meringkas dokumen, pengelompokan topik penelitian, dan banyak lainnya (Rona, 2018).

# Algoritma *CountVectorizer*

# *CountVectorizer* adalah metode untuk mengubah susuan kata dalam sebuah kalimat menjadi vektor dengan menghitung frekuensi kemunculan kata dalam sebuah kalimat. *CountVectorizer* dapat mengubah fitur teks menjadi sebuah representasi vector*.* Dari data teks yang kita miliki (atau biasa disebut korpus) kita dapat menyusun sebuah *vocabulary*, kita kemudian menjadikan setiap data menjadi representasi *vector* masing - masing untuk setiap kata. Tiap elemen dari *vector* menunjukkan jumlah fitur kata yang ada pada data.

# 

# Gambar xx Proses Transformasi Kalimat Menjadi Vector

# dengan *CountVectorizer*

# *Unified Modeling Languange (UML)*

*UML (Unified Modeling Language)* adalah ‘bahasa’ pemodelan untuk sistem atau perangkat lunak yang berparadigma ‘berorientasi objek”. Pemodelan *(modeling)* sesungguhnya digunakan untuk penyederhanaan permasalahan- permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipahami. (Nugroho,2009)

Berikut penjelasan dari beberapa diagram yang digunakan pada penelitian ini.

* + 1. **Diagram *Use-Case (Usecase Diagram)***

*Usecase diagram* menggambarkan fungsionalitas dari sebuah sistem. Sebuah *usecase* merepresentasikan sebuah interaksi antara *actor* dengan sistem. A*ctor* adalah sebuah entitas manusia yang berinteraksi dengan sistem untuk melakukan pekerjaan tertentu. Simbol *Usecase Diagram* dapat dilihat pada Tabel xx

Tabel xx Simbol *Usecase Diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 1 |  | *Actor* | Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case* |
| 2 |  | *Depedency* | Hubungan dimana perubahan yang terjadi pada satu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri  (*independent*) |

Sumber : (Abdullah, 2016)

Tabel xx Simbol *Usecase Diagram* (Lanjutan)

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 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 secraa eksplisit |
| 5 |  | *Extend* | Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik  yang diberikan |
| 6 |  | *Association* | Apa yang menghubungjkan antara objek satu dnegan  objek lainnya |
| 7 | System | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas |
| 8 |  | *Use Case* | Deskripsi dari urutan aksi- aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terstruktur  bagi suatu aktor |

Sumber : (Abdullah, 2016)

Tabel 2.4 Simbol *Usecase Diagram* (Lanjutan)

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 9 |  | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-  elemennya (sinergi) |
| 10 | Note | *Note* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu  sumber daya komputasi |

Sumber : (Abdullah, 2016)

* + 1. **Diagram Kelas (*Class Diagram*)**

*Class* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. *Class* memiliki tiga area pokok, yaitu nama (dan *stereotype*), atribut, dan metoda. Atribut dan metoda dapat memiliki salah satu sifat berikut (Abdullah, 2016):

1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan
2. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak- anak yang mewarisinya
3. *Public*, dapat dipanggil oleh siapa saja Simbol *Class Diagram* dapat dilihat pada Tabel xx.

Tabel 2.5 Simbol *Class Diagram*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Gambar** | | | **Nama** | **Keterangan** |
| 1 |  | | | *Generalization* | Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk  (*ancestor)* |
| 2 |  | | | *Nary Association* | Upaya untuk menghindari asosiasi dengan lebih dari 2 objek. |
| 3 |  | Class Name |  | *Class* | Himpunan dari objek-  objek yang berbagi atribut serta operasi yang sama. |
| Attributes |
| Operations |
|  | | |
| 4 |  | | | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-  elemennya (sinergi) |

Sumber : (Abdullah, 2016)

Tabel 2.6 Simbol *Class Diagram* (Lanjutan)

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 5 |  | *Extend* | Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada  suatu titik yang diberikan |
| 6 |  | Association | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri |
| 7 |  | *Association* | Apa yang  menghubungjkan antara objek satu dnegan objek  lainnya |

Sumber : (Abdullah, 2016)

* + 1. **Diagram Aktivitas (*Activity Diagram*)**

*Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu. Simbol *Activity Diagram* dapat dilihat pada Tabel xx.

Tabel 2.7 Simbol *Activity Diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 1 |  | *Activity* | 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 |  | *Activity Final Node* | Bagaimana objek  dibentuk dan  dihancurkan |
| 5 |  | *Fork Node* | Satu aliran yang pada  tahap tertentu berubah menjadi beberapa aliran |

Sumber : (Abdullah, 2016)

# Teknik Pengujian Perangkat Lunak

Pengujian pada *text mining* untuk mengidentifikasi kategori keluhan ini terpadat dua teknik pengujian yaitu pengujian *Black-box* dan pengujian kuisioner. Pengujian *Black-box* merupakan pengujian untuk mengetahui apakah semua fungsi perangkat lunak telah berjalan semestinya sesuai dengan kebutuhan fungsional yang telah didefinisikan. Kasus ini bertujuan untuk menunjukkan fungsi perangkat lunak tentang cara beroperasinya. Teknik pengujian ini yaitu melakukan kasus uji dengan mempartisi domain *input* dan *output* program. Metode *Black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua persyaratan

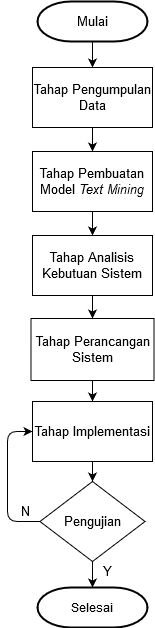
fungsional untuk suatu program. Pengujian ini berusaha menemukan kesalahan dalam kategori fungsi-fungsi yang tidak benar atau hilang, kesalahan *interface*, kesalahan kinerja, dan inisialisasi dan kesalahan terminal (Pressman, 2010). Sedangkan pengujian kuisioner adalah pengujian yang dilakukan dengan memberikan pertanyaan kepada beberapa penguji/responden mengenai sistem untuk menentukan apakah sudah cukup baik digunakan.

# BAB III METODOLOGI PENELITIAN

Pada bab metodologi penelitian ini berisikan langkah-langkah yang digunakan dalam pembuatan *text mining* untuk mengidentifikasi keluhan pengguna sebagai tugas akhir. Dengan adanya metodologi ini proses pembuatan aplikasi dapat dipahami oleh pembaca.

* 1. **Metode Penelitian**

Berdasarkan rumusan masalah dan tujuan penelitian yang ada pada bab I, pembuatan sistem *text mining* untuk mengidentifikasi kategori keluhan ini menggunakan metode penelitian seperti yang terdapat pada Gambar xx berikut.



Metode penelitian pada *text mining* untuk identifikasi keluhanini dimulai dari pengumpulan data, kemudian membuat model *text mining* yang siap pakai, selanjutnya dilakukan analisis kebutuhan sistem, berikutnya perancangan lalu implementasi, dan tahap terakhir adalah pengujian.

* 1. **Pengumpulan Data**

Salah satu komponen yang penting dalam penelitian adalah proses peneliti dalam pengumpulan data. Kesalahan yang dilakukan dalam proses pengumpulan data akan membuat proses analisis menjadi sulit. Selain itu hasil dan kesimpulan yang akan didapat pun akan menjadi rancu apabila pengumpulan data dilakukan tidak dengan benar.

Pengumpulan data dilakukan untuk memperoleh informasi yang dibutuhkan dalam rangka mencapai tujuan penelitian. Pengumpulan data *text mining* untuk identifikasi keluhanini dibuat dalam bentuk angket (kuisioner).

* + 1. **Angket (Kuesioner)**

Kuesioner merupakan metode pengumpulan data yang dilakukan dengan cara memberi seperangkat pertanyaan atau pernyataan tertulis kepada responden untuk dijawab. Kuesioner merupakan metode pengumpulan data yang lebih efisien bila peneliti telah mengetahui dengan pasti variabel yag akan diukur dan tahu apa yang diharapkan dari responden. Selain itu kuesioner juga cocok digunakan bila jumlah responden cukup besar dan tersebar di wilayah yang luas.

Berdasarkan bentuk pertanyaannya, kuesioner dapat dikategorikan dalam dua jenis, yakni kuesioner terbuka dan kuesioner tertutup. Kuesioner terbuka adalah kuesioner yang memberikan kebebasan kepada objek penelitian untuk menjawab. Sementara itu, kuesioner tertutup adalah kuesioner yang telah menyediakan pilihan jawaban untuk dipilih oleh objek penelitian. Seiring dengan perkembangan, beberapa penelitian saat ini juga menerapkan metode kuesioner yang memiliki bentuk semi terbuka. Dalam bentuk ini, pilihan jawaban telah diberikan oleh peneliti, namun objek penelitian tetap diberi kesempatan untuk menjawab sesuai dengan kemauan mereka.

* 1. **Analisis Kebutuhan Sistem**

Dalam pembuatan *text mining* identifikasi keluhan ini terdapat kebutuhan *hardware* dan kebutuhan *software.* Berikut adalah kebutuhan-kebutuhan yang diperlukan :

* + 1. **Kebutuhan *Hardware***

Kebutuhan hardware yang diperlukan pada pembuatan sistem *text mining* untuk mengidentifikasi kategori keluhan ini terdapat pada Tabel xx.

Tabel xx Tabel Kebutuhan *Hardware*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Jenis *Hardware*** | **Kebutuhan *Hardware*** | **Keterangan** |
| 1 | *Processor* | AMD A8 | Untuk mendukung berjalannya aplikasi  pembuat sistem |
| 2 | *RAM* | 4 GB atau lebih | Minimal |

* + 1. **Kebutuhan *Software***

Adapun kebutuhan *software* dalam pembuatan aplikasi sistem pakar diagnosa penyakit padi ini terdapat pada Tabel xx.

Tabel xx Tabel Kebutuhan *Software*

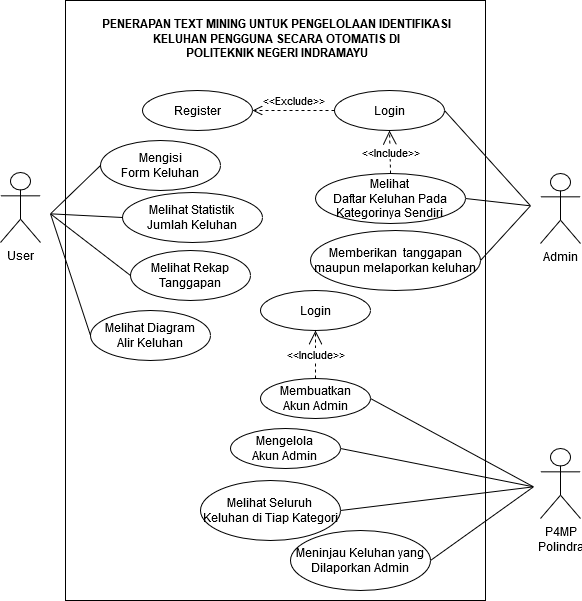
|  |  |  |
| --- | --- | --- |
| **No** | **Jenis Software** | **Kebutuhan Software** |
| 1 | Bahasa *Scripting* | HTML 5, CSS, JavaScript, Python |
| 2 | *Software* Pengolah | VsCode, JupyterLab, Anaconda |
| 3 | Penyimpan Data | Database (MySQL) |
| 4 | Web Browser | Chrome, Firefox, dan lainnya |

* 1. **Perancangan Sistem**

Untuk mendapatkan gambaran mengenai sistem yang dibuat, maka dimodelkan dengan menggunakan *functional modelling*. Proses dan data dari sistem dimodelkan dengan *use case diagram, activity diagram,* dan *class diagram.*

* + 1. ***Use Case* Diagram**

Pada Gambar xx merupakan *use case* diagram dari sistem *text mining* untuk mengidentifikasi kategori keluhan secara otomatis*.*



*.*

Gambar xx *Use Case* Diagram

Adapun penjelasan dari gambar xx *Use Case Diagram* adalah pada Tabel xx berikut ini :

Tabel xx Penjelasan *Use Case* Diagram

|  |  |  |
| --- | --- | --- |
| ***Actor*** | **Nama *Use Case*** | **Keterangan** |
| Admin | *Register* | Sebelum dapat *login* ke system, admin harus registrasi akun terlebih dahulu. Dimana akun tersebut akan dibuatkan oleh *superadmin.* |
| *Login* | Admin dapat masuk ke sistem menggunakan akun yang telah dibuatkan oleh *superadmin* atau P4MP Polindra. |
| Melihat Daftar Keluhan Pada Kategorinya Sendiri | Admin dapat melihat seluru daftar keluhan yang masuk dan teridentifikasi sebagai kategori unit kerja nya. |
| Memberikan Tanggapan Maupun Melaporkan Keluhan | Admin dapat memberikan tanggapannya terhadap keluhan yang masuk. Dan juga dapat melaporkan keluhan ke P4MP apabila isi keluhan tidak sesuai (*spam*). |

Tabel xx Penjelasan Use Case Diagram (Lanjutan)

|  |  |  |
| --- | --- | --- |
| ***Actor*** | **Nama *Use Case*** | **Keterangan** |
| P4MP  Polindra | *Login* | *Superadmin* (P4MP) dapat masuk ke sistem dengan akun yang sudah terdaftar di *database.* |
| Membuatkan Akun Admin | *Superadmin* (P4MP) dapat membuatkan akun admin, supaya admin bisa *login*  ke sistem. |
| Mengelola Akun  Admin | *Superadmin* (P4MP) dapat mengelola seluruh akun admin, antara lain mengedit atau menghapus akun admin. |
| Meninjau Keluhan yang di Laporkan Admin | Keluhan yang di laporkan admin dapat ditinjau (*accept/decline)* oleh *superadmin* (P4MP). |
| Melihat Seluruh Keluhan  Di Tiap Kategori | *Superadmin* (P4MP) dapat melihat seluruh keluhan yang masuk dari tiap kategori. |
| User | Mengisi Form Keluhan | *User* dapat memberikan keluhanya melalui form keluhan yang ada di *website E-complaint* Polindra. |
| Melihat Statistik Jumlah Keluhan | *User* juga dapat melihat persentase jumlah keluhan yang masuk di setiap kategori. |

Tabel xx Penjelasan Use Case Diagram (Lanjutan *User*)

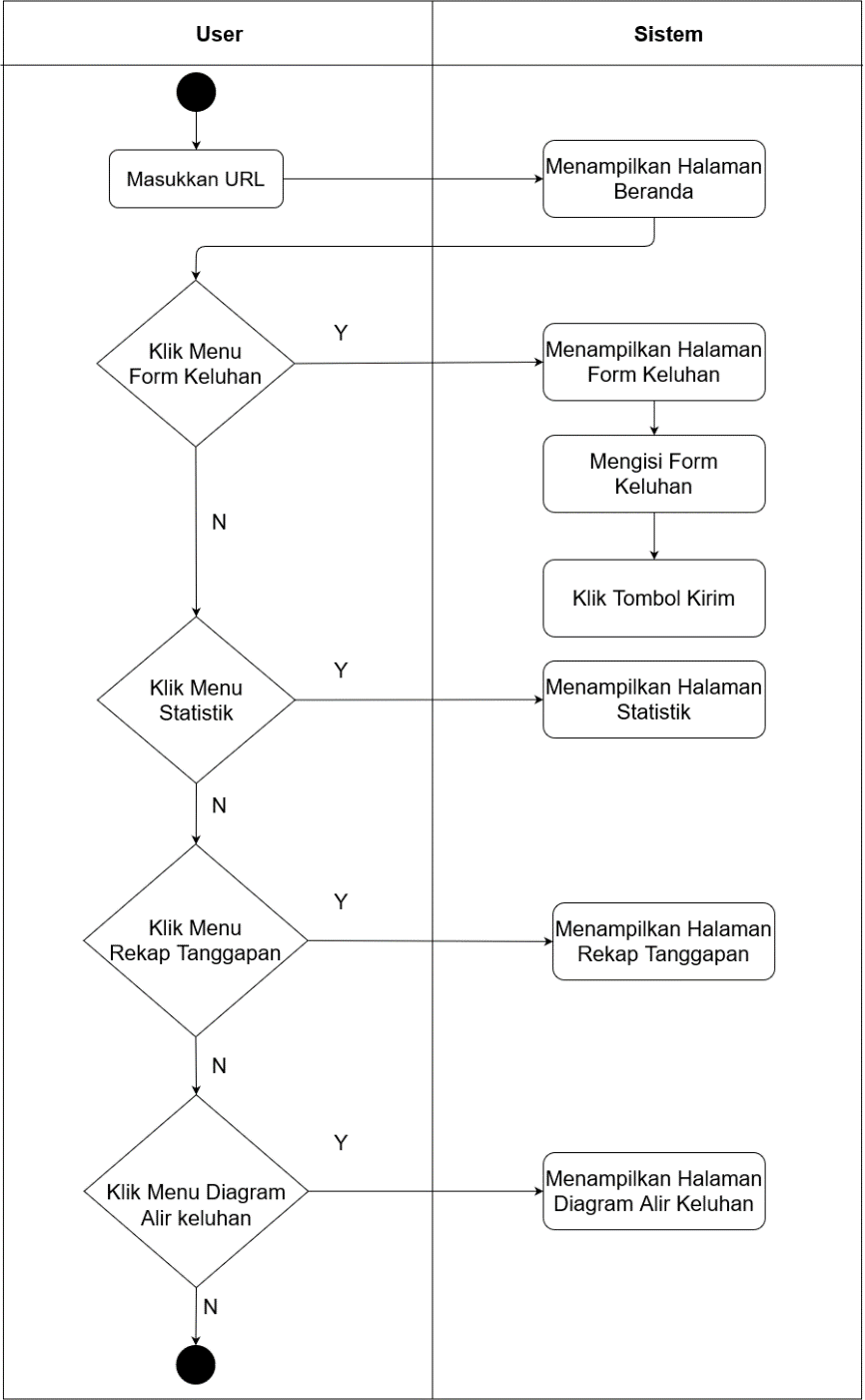
|  |  |  |
| --- | --- | --- |
| ***Actor*** | **Nama *Use Case*** | **Keterangan** |
| *User* | Melihat Rekap  Tanggapan | *User* dapat melihat status tanggapannya di menu Rekap Tanggapan untuk mengetahui apakah tanggapannya sudah di tinjau atau belum. |
| Melihat Diagram  Alir Keluhan | *User* dapat melihat diagram alir *E-complaint* Polindra agar dapat mengetaui bagaimana keluhannya di proses. |

## Activity Diagram

*Activity Diagram* merupakan *state* diagram khusus yang menggambarkan berbagai aliran aktivitas di dalam sistem yang sedang dirancang, bagaimana masing- masing aliran berawal dan berakhir. Bahkan mungkin terjadi *decision* didalamnya. *Activity diagram* juga dapat menggambarkan proses parallel yang mungkin terjadi pada beberapa eksekusi dalam sistem.

* + - 1. ***Activity* Diagram *User***

Pada *activity* diagram *user* ini menggambarkan aliran aktivitas yang dilakukan oleh *user* dalam sistem yang dirancang. Adapun diagram aktivitas *user* terdapat pada Gambar xx.

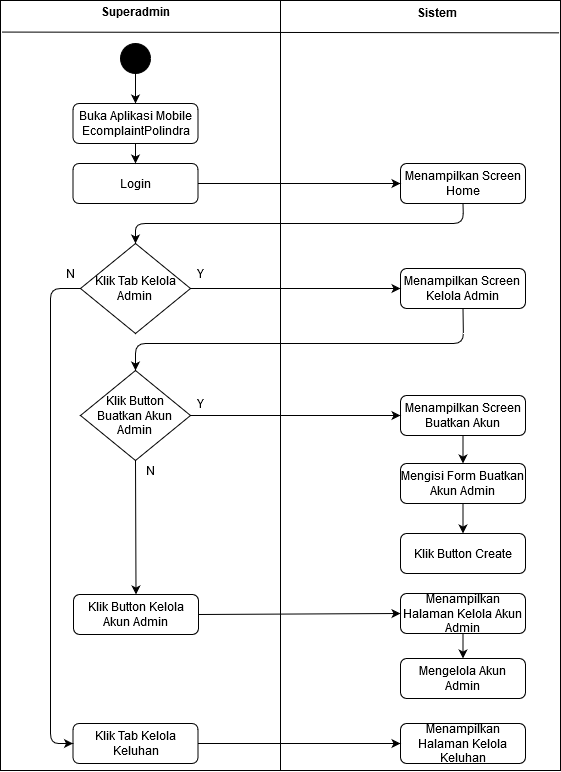


Gambar xx *Activity* Diagram Admin

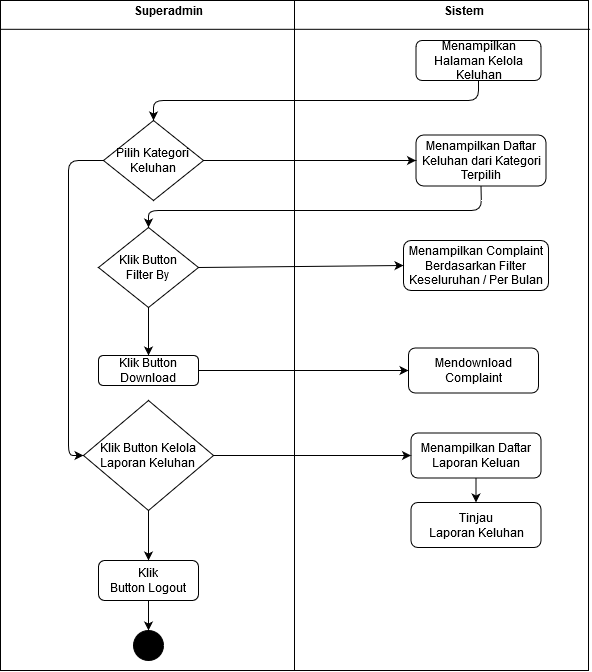
Pada *activity* diagram *user* ini dimulai dengan memasukkan *URL,* lalu *pengguna* akan diarakan pada halaman beranda. User dapat memilih menu *Form Keluhan* untuk menuangkan keluhannya, atau juga dapat memilih menu Statistik untuk dapat mengetahui presentase jumlah keluhan yang masuk di tiap-tiap kategori. Selain itu *user* juga dapat memilih menu Rekap Tanggapan untuk mengetahui tinjauan admin terhadap keluhannya. Dan terakhir user dapat memilih menu Diagram Alir Keluhan untuk dapat mengetahui alur diprosesnya sebuah keluhan yang masuk kedalam system.

* + - 1. ***Activity* Diagram Superadmin**

Pada *activity* diagram superadmin ini menggambarkan aliran aktivitas yang dilakukan oleh superadmin dalam sistem yang dirancang. Adapun *activity* diagram superadmin terdapat pada Gambar xx.



Gambar xx *Activity* Diagram Superadmin



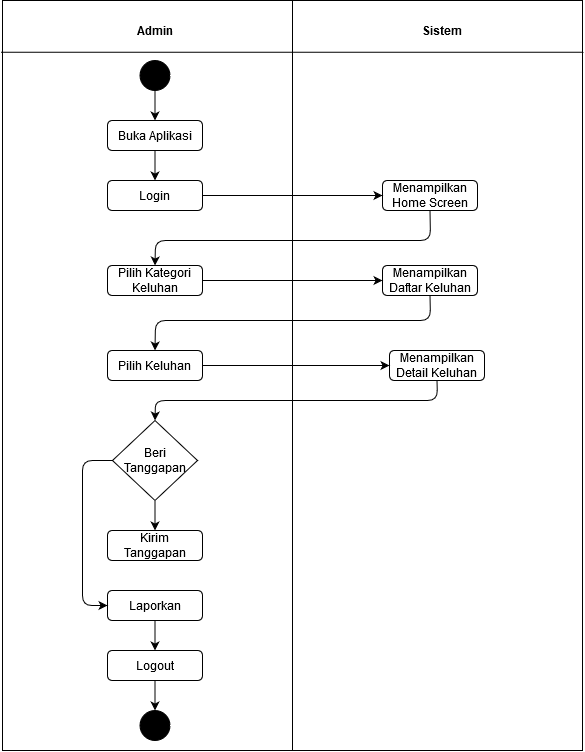
Gambar xx *Activity* Diagram Superadmin (Lanjutan)

Pada *activity* diagram superadmin ini dimulai dengan superadmin membuka aplikasi mobile *E-Complaint* Polindra terlebih dahulu. Lalu superadmin diharuskan login terlebih dahulu untuk masuk ke *system*, kemudian system akan mengarahkan ke halaman *Home.* Pada halaman *Home* ini superadmin dapat memilih *tab* Kelola Admin, pada *tab* ini superadmin dapat memilih menu Buatkan Akun Admin, untuk membuatkan akun admin atau menu Kelola Akun Admin untuk mengelola akun admin seperti *edit / delete* akun admin.

Superadmin juga dapat memilih *tab* Kelola Keluhan yang di dalamnya terdapat 2 submenu lagi yaitu memilih Ketegori Keluhan dimana superadmin dapat melihat daftar keluhan dan bisa mendownload nya ke dalam bentuk *PDF.* Dan submenu lainnya yaitu Kelola Laporan Keluhan dimana super admin dapat meninjau keluhan yang di *report* oleh admin.

* + - 1. ***Activity* Diagram Admin**

Pada *activity* diagram admin ini menggambarkan aliran aktivitas yang dilakukan oleh admin dalam sistem yang dirancang. Adapun *activity* diagram admin terdapat pada Gambar xx.



Gambar xx *Activity* Diagram Superadmin (Lanjutan)

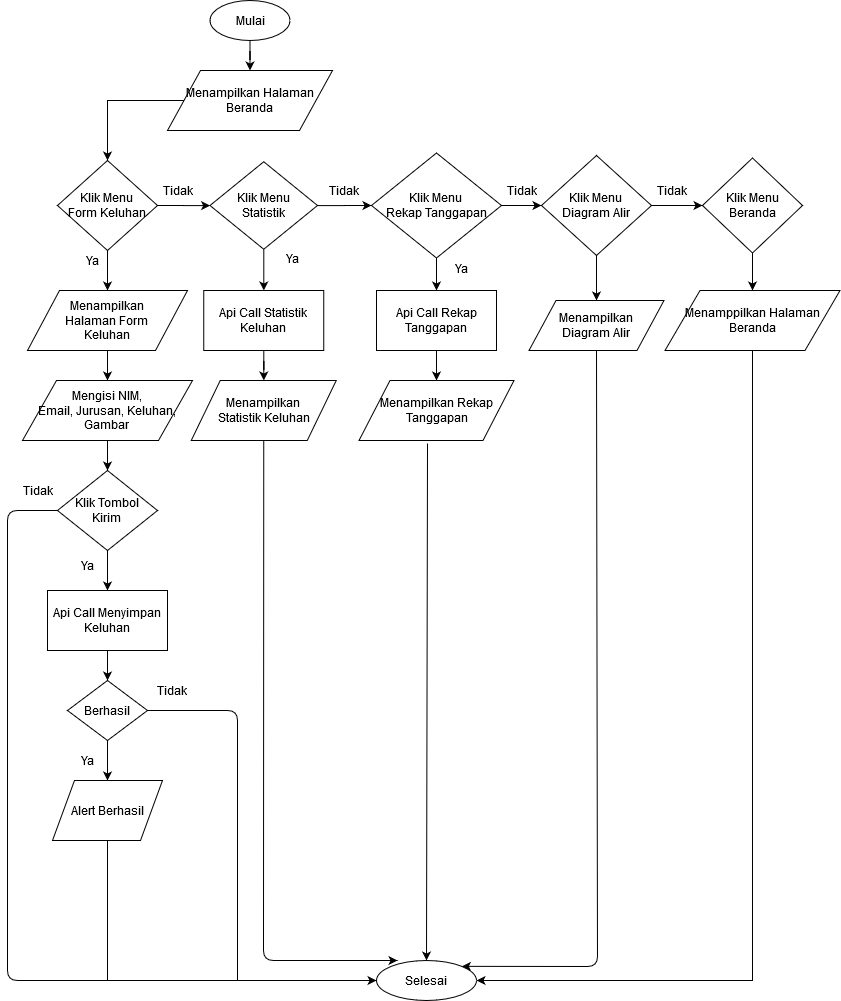
Pada *activity* diagram admin ini dimulai dengan membuka aplikasi mobile *E-complaint* Polindra, lalu admin diharuskan *Login* terlebih dahulu untuk masuk kedalam sistem dengan menggunakan akun yang telah dibuatkan oleh superadmin. Setelah masuk ke halaman *Home*, admin dapat mengklik tombol Kategori Keluhan untuk dapat melihat daftar keluhan. Jika keluhan di klik maka akan muncul *detail* keluhan. Pada *screen* ini *user* dapat memberi tanggapan dengan mengisi *text area* lalu klik tombol Beri Tanggapan, atau admin juga dapat melapurkan keluhan dengan klik tombol Laporkan.

## Flowchart

*Flowchart* yang dibuat untuk sistem identifikasi keluhan dengan *text mining* ini terdiri dari *flowchart user*, *flowchart* admin, dan *flowchart* superadmin.

1. * 1. ***Flowchart User***

*Flowchart* ini adalah alur berjalannya sistem dari sisi *user. Flowchart* ini terdapat pada Gambar xx

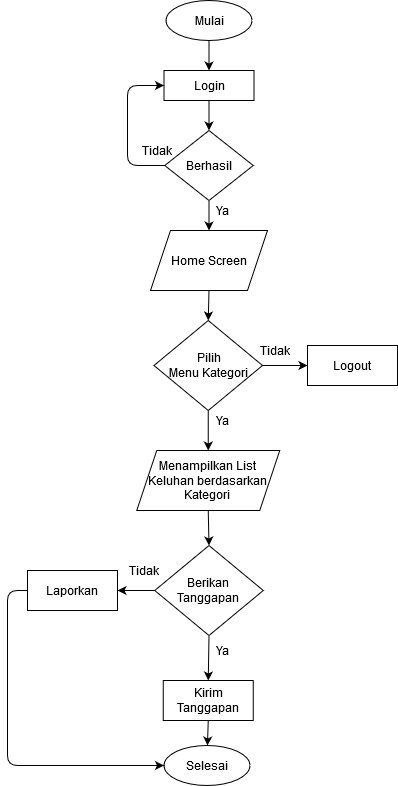


Gambar xx *Flowchart User*

*Flowchart user* dimulai dari menampilkan halaman beranda pada *interface,* apabila user memilih menu *form* keluhan maka user akan diarahkan ke halaman yang berisi *form* untuk menuangkan keluhannya. Ketika *user* memilih menu statistik maka sistem akan menampilkan jumlah presentase keluhan yang masuk di tiap-tiap kategori. Jika *user* memilih menu rekap tanggapan maka *user* dapat melihat tanggapan dari keluhan yang pernah diinputkannya. Terakhir, jika *user* memilih menu diagram alir keluhan *user* dapat melihat bagaimana alur sebuah keluhan diproses.

* + 1. ***Flowchart* Admin**

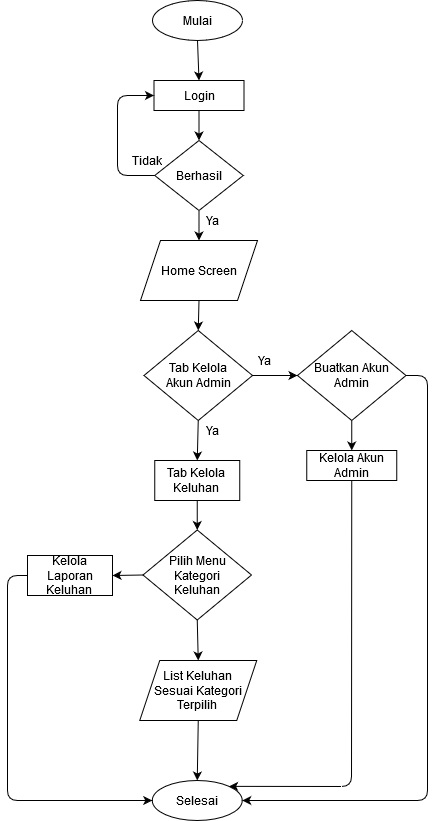
*Flowchart* ini adalah alur berjalannya sistem dari sisi *admin. Flowchart* ini terdapat pada Gambar xx



Gambar xx *Flowchart* Admin

*Flowchart* admin dimulai dengan melakukan *login* menggunakan akun yang telah dibuatkan oleh superadmin, kemudian admin diarahkan ke *home screen.* Selanjutnya *user* dapat memilih menu kategori keluhan, kemudian sistem akan menampilkan daftar keluhan di kategori admin tersebut. Ketika admin mengklik salah satu keluhan maka sistem akan menampilkan halaman detail keluhan. Pada halaman ini admin dapat memberikan tanggapan atau melaporkan keluhan.

* + 1. ***Flowchart* Superadmin**



Gambar xx *Flowchart* Superadmin

.

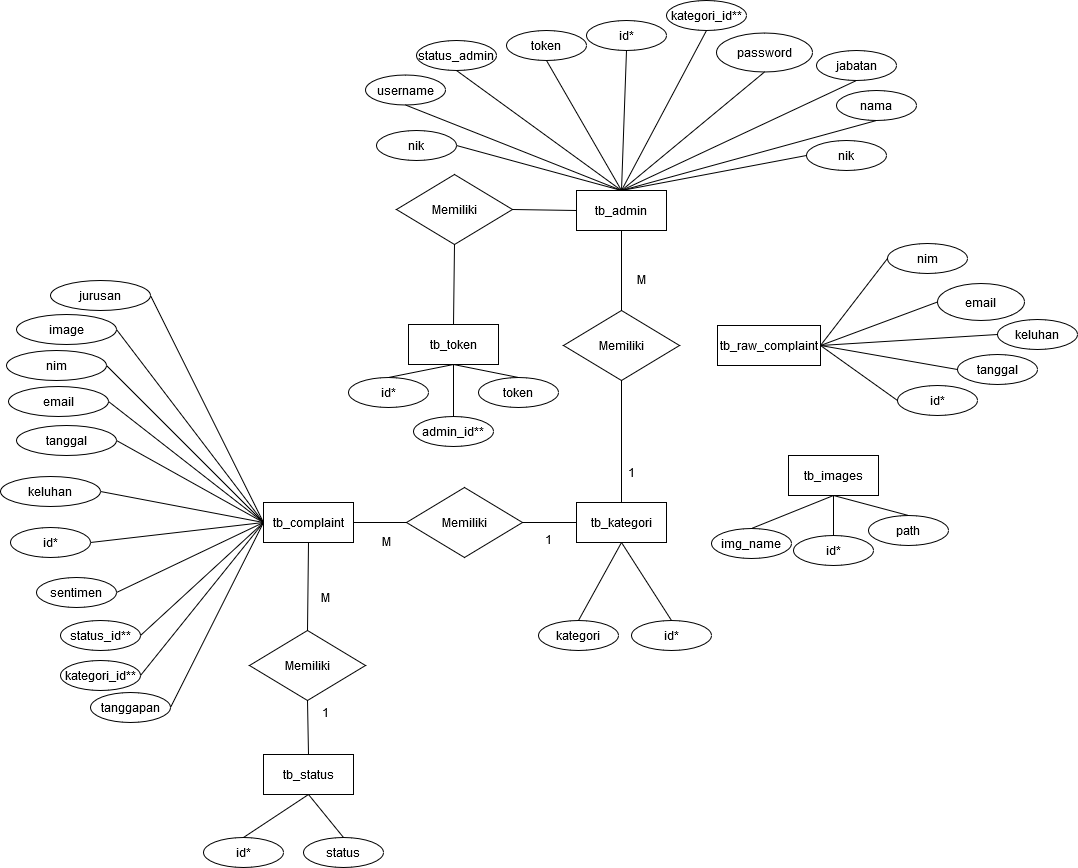
Pada *flowchart* superadmin dimulai dengan *login* menggunakan akun yang dibuat langsung di *database,* kemudian superadmin diarahkan ke halaman *home.* Pada halaman ini terdapat 2 *tab,* yaitu kelola akun dan kelola keluhan. Jika user memilih kelola akun maka user dapat membuatkan akun untuk seorang admin maupun mengelola akun admin yang telah terdaftar, misalnya mengedit atau menghapus akun.

* 1. **Perancangan *ERD (Entity Relationship Diagram)***

Pada Gambar xx merupakan gambar *Entity Relationship Diagram (ERD)* yang digunakan untuk proses Analisa perancangan dari sistem identifikasi keluhan dengan *text mining*. Adapun penjelasan dari Gambar xx adalah pada Tabel xx berikut ini :

Tabel xx Penjelasan *ERD*

|  |  |
| --- | --- |
| **Nama Tabel** | **Keterangan** |
| tb\_complaint | Tabel yang digunakan untuk menyimpan data keluhan yang sudah diidentifikasi |
| tb\_raw\_complaint | Tabel yang digunakan untuk menyimpan keluhan yang belum diidentifikasi (untuk keperluan *datasets*) |
| tb\_status | Tabel yang digunakan untuk menyimpan status keluhan |
| tb\_images | Tabel yang digunakan untuk menyimpan data dukung keluhan |
| tb\_token | Tabel yang digunakan untuk menyimpan token di tiap-tiap *device* yang digunakan untuk *login* |
| tb\_kategori | Tabel yang digunakan untuk menyimpan kategori keluhan |
| tb\_admin | Tabel yang digunakan untuk meyimpan data admin dan superadmin |



Gambar xx Perancangan ERD

Pada sistem identifikasi keluhan dengan *text mining* ini terdiri dari 7 tabel yaitu tb\_*complaint*, tb\_*raw*\_*complaint*, tb\_status, tb\_*images*, tb\_token, tb\_kategori, dan tb\_admin*.*

* 1. **Perancangan *Database***

Berikut ini merupakan tabel dari perancangan sistem identifikasi keluhan dengan *text mining*.

# Tabel tb\_admin

Tabel tb\_admin digunakan untuk meyimpan data admin dan superadmin, terdiri dari *field id, username,* nik, jabatan, status\_admin, *password,* kategori\_id, nama, jurusan. Adapun tabel tb\_admin dapat dilihat pada Tabel xx

Tabel xx Tabel tb\_admin.

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Field** | **Tipe Data** | **Keterangan** |
| 1 | id | Int(11) | Untuk menyimpan id admin |
| 2 | username | Varchar(200) | Untuk menyimpan username admin |
| 3 | nik | Varchar(150) | Untuk menyimpan nik admin |
| 4 | jabatan | Varchar(200) | Untuk menyimpan jabatan admin |
| 5 | status\_admin | Varchar(100) | Untuk menyimpan status admin |
| 6 | password | Varchar(200) | Untuk menyimpan password admin |
| 7 | kategori\_id | Int(11) | Untuk menyimpan id kategori admin |
| 8 | nama | Varchar(150) | Untuk menyimpan nama admin |
| 9 | jurusan | Varchar(200) | Untuk menyimpan jurusan admin |

# Tabel tb\_*complaint*

Tabel tb\_*complaint* digunakan untuk menyimpan data keluhan yang sudah diidentifikasi, terdiri dari *field id,* keluhan, sentiment, kategori\_id, status\_id, tanggapan, tanggal, *email*, nim*, images,* jurusan. Adapun tabel tb\_*complaint* dapat dilihat pada Tabel xx.

Tabel xx Tabel tb\_*complaint*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Field** | **Tipe Data** | **Keterangan** |
| 1 | id | Int(11) | Untuk menyimpan id complaint |
| 2 | keluhan | Longtext | Untuk menyimpan keluhan |
| 3 | sentimen | Varchar(100) | Untuk menyimpan sentimen |
| 4 | kategori\_id | Int(11) | Untuk menyimpan kategori dari keluhan |
| 5 | status\_id | Int(11) | Untuk menyimpan status dari keluhan |
| 6 | tanggapan | Longtext | Untuk menyimpan tanggapan |
| 7 | tanggal | Date | Untuk menyimpan tanggal dibuatnya keluhan |
| 8 | email | Varchar(100) | Untuk menyimpan email dari pengguna |
| 9 | nim | Varchar(20) | Untuk menyimpan nim dari pengguna |
| 10 | image | Longtext | Untuk menyimpan image keluhan |
| 11 | jurusan | Varchar(200) | Untuk menyimpan jurusan dari pengguna |

* + 1. **Tabel tb\_*raw\_complaint***

Tabel tb\_*raw\_complaint* digunakan untuk menyimpan keluhan yang belum diidentifikasi (untuk keperluan *datasets*), terdiri dari *field id,* nim, *email,* tanggal, keluhan. Adapun tabel tb\_*raw\_complaint* dapat dilihat pada Tabel xx.

Tabel xx Tabel tb\_*raw\_complaint*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Field** | **Tipe Data** | **Keterangan** |
| 1 | id | Int(11) | Untuk menyimpan id keluhan |
| 2 | nim | Varchar(20) | Untuk menyimpan nim dari pengguna |
| 3 | email | Varchar(100) | Untuk menyimpan email dari pengguna |
| 4 | tanggal | Date | Untuk menyimpan tanggal dibuatnya keluhan |
| 5 | keluhan | longtext | Untuk menyimpan keluhan |

* + 1. **Tabel tb\_status**

Tabel tb\_statusdigunakan untuk menyimpan status keluhan, terdiri dari *field id,* dan status keluhan. Adapun tabel tb\_statusdapat dilihat pada Tabel xx.

Tabel xx Tabel tb\_status

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Field** | **Tipe Data** | **Keterangan** |
| 1 | id | Int(11) | Untuk menyimpan id status |
| 2 | status | Varchar(150) | Untuk menyimpan nama status |

* + 1. **Tabel tb\_kategori**

Tabel tb\_kategoridigunakan untuk menyimpan kategori keluhan, terdiri dari *field id,* dan kategorikeluhan. Adapun tabel tb\_ kategoridapat dilihat pada Tabel xx.

Tabel xx Tabel tb\_ kategori

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Field** | **Tipe Data** | **Keterangan** |
| 1 | id | Int(11) | Untuk menyimpan id kategori |
| 2 | kategori | Varchar(200) | Untuk menyimpan nama kategori |

* + 1. **Tabel tb\_token**

Tabel tb\_tokendigunakan untuk menyimpan token di tiap-tiap *device* yang digunakan untuk *login*, terdiri dari *field id,* token,dan amin\_id. Adapun tabel tb\_ tokendapat dilihat pada Tabel xx.

Tabel xx Tabel tb\_ token

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Field** | **Tipe Data** | **Keterangan** |
| 1 | id | Int(11) | Untuk menyimpan id token |
| 2 | token | Longtext | Untuk menyimpan token device pengguna |
| 3 | admin\_id | Int(11) | Untuk menyimpan id admin |

* + 1. **Tabel tb\_images**

Tabel tb\_imagesdigunakan untuk menyimpan data dukung keluhan, terdiri dari *field id, path*,dan *image\_name*. Adapun tabel tb\_ imagesdapat dilihat pada Tabel xx.

Tabel xx Tabel tb\_ images

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Field** | **Tipe Data** | **Keterangan** |
| 1 | id | Int(11) | Untuk menyimpan id gambar |
| 2 | path | Varchar(100) | Untuk menyimpan link gambar |
| 3 | img\_name | Varchar(150) | Untuk menyimpan nama gambar |

# Perancangan Desain Antarmuka

Ada beberapa rancangan desain antarmuka yang dibuat untuk sistem pakar diagnosa penyakit padi, berikut merupakan rancangannya :