BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Algoritma *Knuth Morris Pratt* telah banyak diterapkan dalam penelitian untuk pencarian dan pencocokan *string*. Beberapa penelitian yang pernah dilakukan untuk menyelesaikan permasalahan pencocokan *string* yaitu pada hasil penelitian Wistiani Astuti (2017) Analisis *String Matching* pada judul skripsi dengan algoritma *Knuth Morris Pratt.* Endang Rismawati, dkk (2016) Implementasi algoritma *Knuth Morris Pratt* dalam pencarian nomor dan lirik lagu dalam kidung jemaat berbahasa Nias. Firman Matondang, dkk (2016) perancangan aplikasi text editor dengan menerapkan algoritma *Knuth Morris Pratt.* Rivalri Kristianto Hondro, dkk (2016) Implementasi algoritma *Knuth Morris Pratt* pada aplikasi penerjemah bahasa Mandailing ke Indonesia. Rio Alamanda, dkk (2016) Aplikasi pendekteksi plagiat terhadap karya tulis berbasis web menggunakan *Natural Language Processing* dan algoritma *Knuth Morris Pratt*.

II.2. Landasan Teori

II.2.1. Algoritma *Knuth Morris Pratt*

Algoritma *Knuth Morris Pratt* adalah salah satu algoritma pencarian *string*, dikembangkan secara terpisah oleh Donald E. Knuth pada tahun 1967 dan James H. Morris bersama Vaughan R. Pratt pada tahun 1966, namun keduanya

mempublikasikan secara bersamaan pada tahun 1977. Jika kita melihat algoritma *brute force* lebih mendalam, kita mengetahui bahwa dengan mengingat beberapa perbandingan yang dilakukan sebelumnya kita dapat meningkatkan besar pergeseran yang dilakukan. Hal ini akan menghemat perbandingan, yang selanjutnya akan meningkatkan kecepatan pencarian.

Pada algoritma KMP, kita memelihara informasi yang digunakan untuk melakukan jumlah pergeseran. Algoritma menggunakan informasi tersebut untuk membuat pergeseran yang lebih jauh, tidak hanya satu karakter. Pada algoritma Knuth Morris Pratt (KMP) informasi ketidakcocokan pattern dengan teks disimpan untuk menentukan jumlah pergeseran. Algoritma KMP melakukan pergeseran lebih jauh sesuai informasi yang disimpan yang menyebabkan waktu pencarian dapat dikurangi secara signifikan. (Wistiani Astuti, 2017).

Secara sistematis, langkah-langkah yang dilakukan algoritma *Knuth Morris Pratt* pada saat mencocokan *pattern* adalah:

1. Algoritma *Knuth Morris Pratt* mulai mencocokkan *pattern* pada awal teks.
2. Dari kiri ke kanan, algoritma ini akan mencocokkan karakter per karakter *pattern* dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
    a. Karakter di *pattern* dan di teks yang dibandingkan tidak cocok (*mismatch*).
    b. Semua karakter di *pattern* cocok, kemudian algoritma akan memberitahukan penemuan di posisi ini.
3. Algoritma kemudian menggeser *pattern* berdasarkan tabel next, lalu mengulangi langkah 2 sampai *pattern* berada di ujung teks.

II.2.2. Aplikasi

Berasal dari bahasa Inggris *application* yang berarti penerapan, lamaran ataupun penggunaan. Sedangkan secara umum, pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanakan suatu fungsi bagi pengguna jasa aplikasi serta jasa pengguna aplikasi lain yang dapat digunakan oleh pengguna yang akan dituju**.** (Rivalri Kristianto Hondro, dkk, 2016).

II.2.3. Penerjemah

Penerjemahan adalah proses pengalihan bahasa, kata demi kata dari satu bahasa ke bahasa lain. Teori lain menyatakan bahwa penerjemahan sebagai kegiatan mengganti materi teks/ujaran/tuturan dalam bahasa sumber ke materi teks/ujaran/tuturan yang sepadan ke dalam bahasa sasaran (Catford, 1989) dikutip oleh (Friyanto, Fanji 2017).

II.2.4. Bahasa Jawa

Bahasa Jawa adalah bahasa yang digunakan penduduk suku bangsa Jawa di Jawa Tengah, Yogyakarta dan Jawa Timur. Selain itu, bahasa Jawa juga digunakan oleh penduduk yang tinggal di beberapa daerah lain seperti di Banten terutama kota Serang. Kabupaten Serang, kota Cilegon dan kabupaten Tanggerang, Jawa Barat khususnya kawasan pantai utara terbentang dari pesisir utara Karawang, Subang, Indramayu, kota Cirebon dan kabupaten Cirebon. Penyebaran bahasa Jawa di karenakan penduduk Jawa yang merantau, membuat

bahsa Jawa bisa ditemukan di bebagai daerah maupun luar negeri. Penyebaran bahasa Jawa di berbagai daerah seperti Lampung, Sumatra Utara, Jambi, Sumatra Selatan, bahkan kebudayaan Jawa sampai ke negara tetangga seperti malaysia (Sumber : Wikipedia).

II.2.5. Bahasa Indonesia

Sebagai lambang identitas nasional, bahasa Indonesia merupakan "lambang" bangsa Indonesia. Dalam hal ini, bahasa Indonesia dapat dikatakan memiliki kedudukan yang setara dan serasi dengan lambang kebangsaan yang lain, seperti bendera merah putih, garuda Pancasila, dan lagu kebangsaan Indonesia Raya. Ini berarti, dengan bahasa Indonesia, bangsa Indonesia menyatakan jati dirinya, menyatakan sifat, perangai, dan wataknya sebagai bangsa Indonesia.

Fungsi bahasa Indonesia sebagai lambing kebanggaan dan identitas nasional berkaitan erat dengan fungsinya yang ketiga, yaitu sebagai alat yang memungkinkan terlaksananya penyatuan berbagai suku bangsa yang mempunyai latar belakang sosial, budaya, dan bahasa daerah yang berbeda-beda ke dalam satu kesatuan kebangsaan yang bulat, bersatu dalam cita-cita dan rasa nasib yang sama. Dalam hubungan dengan hal ini, bahasa Indonesia memungkinkan berbagai suku bangsa itu mencapai keserasian hidup sebagai bangsa yang bersatu dengan tidak perlu meninggalkan identitas kesukuan dan kesetiaan kepada nilai-nilai sosial, budaya, dan latar belakang bahasa daerah yang bersangkutan (Sudiana et al. 2013) dikutip oleh (Friyanto, Fanji 2017).

II.2.6. *Android*

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencangkup sistem operasi *middleware* dan aplikasi. *Android* menyediakan *platform* terbuka bagi para pegembang untuk menciptakan aplikasi. Awalnya *Google inc* membeli *Android Inc* yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel (Endang Rismawati, dkk, 2016).

II.2.7. *Java*

Java merupakan teknologi dimana ternologi tersebut mencangkup java sebagai bahasa pemograman yang memiliki sintaks dan aturan pemograman tersendiri, juga memiliki virtual machine dan library yang diperlukan untuk menulis dan menjalankan program yang ditulis dengan bahasa pemograman java,

Java merupakan bahasa pemograman yang berorientasi objek yang diciptakan oleh Sun Microsystem pada tahn 1995. Java dapat membiat seluruh bentuk aplikasi, desktop, web dan lainnya sebagaimana dibuat dengan menggunakan bahasa pemograman yang lain (Fery Wongso, 2015).

II.2.8. *SQLite Database*

SQLite merupakan proyek yang bersifat public domain yang dikerjakan oleh D. Ricard Hipp. SQLite memiliki fitur relasional database, hampir sama dengan SQL pada desktop hanya saja SQLite membutuhkan memori sedikit. Pada prinsipnya SQLite merupakan library C yang diimplementasikan secara embeddable (tertanam) sebagai SQL database engine. Kemasan praktis inilah yang

memberikan banyak sekali keuntungan dimana tidak perlu melakukan manajemen database server terpisah. Selain itu, ukuran kecil yang sengaja didesain pada SQLite memungkinkan kita untuk membuta aplikasi yang ringan dengan kinerja Tinggi (Tuji Rochma Wati dan Heri Sismoro, 2014).

II.2.9. *Eclipse*

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform (*platform-independent*). Berikut ini adalah sifat dari *Eclipse Multi-platform*, *Multi-language*, *Multi-role*. *Eclipse* pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemograman perangkat lunak ini. Selain itu, kelebihan dari *Eclipse* yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in* (Syifa Nur Rakhmah, 2016).

II.3 *Unified Modeling Language* (UML)

Menurut Windu Gata, Grace (2013:4), *Unified Modeling Language (UML)* adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membanngun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem, dikutip oleh (Ade Hendini, 2016)

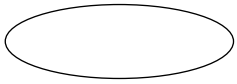
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut:

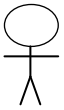
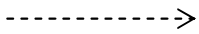
1. *Use Case Diagram*

Use case diagram merupakan pemodelan untuk kelakuakn (*behavior*) sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *Use Case Diagram* yaitu:

Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.1. Simbol *Use Case*

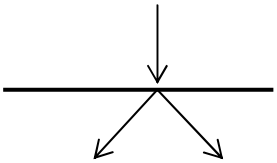
Gambar	**Keterangan**
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.

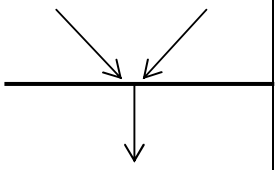
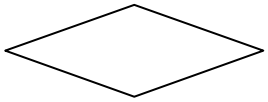
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	*Include,* merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

2. Diagram Aktivitas (*Activity Diagram*)

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

Tabel II.2. Simbol *Activity Diagram*

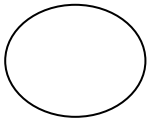
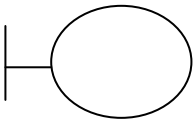
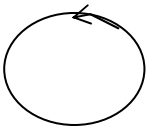
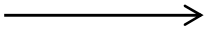
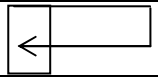
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.

	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points,* menggambarkan pilihan untuk pengambilan keputusan, *true, false.*
New Swimline	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

3.Diagram Urutan (*Sequence Diagram*)

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

Tabel II.3. Simbol *Sequence Diagram*

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

	dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility,* tingkat akses objek eksternal kepada suatu

operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.4. ***Multiplicity Class Diagram***

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