

**SKRIPSI**

**STEGANOGRAFI BERBASIS SUKU KATA BAHASA  
INDONESIA**



**MICHAEL THEODORE PANGESTU**

**NPM: 2012730048**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS  
UNIVERSITAS KATOLIK PARAHYANGAN  
2015**



**UNDERGRADUATE THESIS**

**INDONESIAN SYLLABLE-BASED STEGANOGRAPHY**



**MICHAEL THEODORE PANGESTU**

**NPM: 2012730048**

**DEPARTMENT OF INFORMATICS  
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES  
PARAHYANGAN CATHOLIC UNIVERSITY  
2015**



**LEMBAR PENGESAHAN**

**STEGANOGRAFI BERBASIS SUKU KATA BAHASA  
INDONESIA**

**MICHAEL THEODORE PANGESTU**

**NPM: 2012730048**

**Bandung, «tanggal» «bulan» 2015**

**Menyetujui,**

**Pembimbing Utama**

**Pembimbing Pendamping**

**Thomas Anung Basuki, Ph.D.  
Ketua Tim Penguji**

**«pembimbing pendamping/2»  
Anggota Tim Penguji**

**«penguji 1»**

**«penguji 2»**

**Mengetahui,**

**Ketua Program Studi**

**Thomas Anung Basuki, Ph.D.**



## PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

### **STEGANOGRAFI BERBASIS SUKU KATA BAHASA INDONESIA**

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,  
Tanggal «tanggal» «bulan» 2015

Meterai

Michael Theodore Pangestu  
NPM: 2012730048





## ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia» Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetur.

**Kata-kata kunci:** «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Indonesia»



## ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris» Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetur.

**Keywords:** «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Inggris»



*«kepada siapa anda mempersembahkan skripsi ini...?»*



## KATA PENGANTAR

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Bandung, «bulan» 2015

Penulis





## DAFTAR ISI

|  |             |
|--|-------------|
| <b>KATA PENGANTAR</b>                      | <b>xv</b>   |
| <b>DAFTAR ISI</b>                          | <b>xvii</b> |
| <b>DAFTAR GAMBAR</b>                       | <b>xix</b>  |
| <b>DAFTAR TABEL</b>                        | <b>xx</b>   |
| <b>1 PENDAHULUAN</b>                       | <b>3</b>    |
| 1.1 Latar Belakang                         | 3           |
| 1.2 Rumusan Masalah                        | 4           |
| 1.3 Tujuan                                 | 4           |
| 1.4 Deskripsi Perangkat Lunak              | 4           |
| 1.5 Metode Penelitian                      | 5           |
| <b>2 LANDASAN TEORI</b>                    | <b>7</b>    |
| 2.1 Bahasa Indonesia [1]                   | 7           |
| 2.1.1 Pemenggalan kata                     | 7           |
| 2.2 Pengolahan Teks                        | 8           |
| 2.2.1 Natural Language Processing[2]       | 8           |
| 2.2.2 Information Retrieval[3]             | 9           |
| 2.2.3 Teori Automata[4]                    | 10          |
| 2.2.4 Teori Finite State Automata (FSA)[4] | 11          |
| 2.2.5 Teori Deterministic FSA (DFSA)[4]    | 11          |
| 2.3 Steganografi                           | 12          |
| 2.3.1 Format Based Steganography[5]        | 12          |
| 2.3.2 Linguistic Steganography             | 14          |
| <b>3 ANALISIS</b>                          | <b>15</b>   |
| 3.1 Gambaran Algoritma Umum                | 15          |
| 3.1.1 Pemenggalan Kata                     | 16          |
| 3.1.2 <i>Stego-cover</i>                   | 21          |
| 3.1.3 Kamus sinonim                        | 22          |
| 3.1.4 <i>Embedding</i>                     | 22          |
| 3.1.5 <i>Extracting Information</i>        | 24          |
| 3.2 Analisis Use Case                      | 25          |
| 3.2.1 Skenario Use Case                    | 26          |
| 3.2.2 Analisis Diagram Kelas               | 28          |
| <b>4 PERANCANGAN</b>                       | <b>31</b>   |
| 4.1 Perancangan Antarmuka                  | 31          |
| 4.1.1 <i>Tab Embed</i>                     | 32          |
| 4.1.2 <i>Tab Extract</i>                   | 32          |

|          |  |           |
|----------|--|-----------|
| 4.1.3    | <i>Tab</i> Tambah <i>Cover</i> . . . . . | 33        |
| 4.1.4    | <i>Tab</i> Tambah Sinonim . . . . .      | 34        |
| 4.1.5    | <i>Tab</i> Petunjuk . . . . .            | 35        |
| 4.2      | Diagram Kelas Lengkap . . . . .          | 35        |
| 4.3      | Analisis Diagram Aktivitas . . . . .     | 38        |
| <b>5</b> | <b>IMPLEMENTASI DAN PENGUJIAN</b>        | <b>41</b> |
| 5.1      | Implementasi . . . . .                   | 41        |
| 5.2      | Pengujian . . . . .                      | 44        |
| 5.2.1    | Pengujian Fungsional . . . . .           | 44        |
| 5.2.2    | Pengujian Eksperimental . . . . .        | 44        |
|          | <b>DAFTAR REFERENSI</b>                  | <b>47</b> |

## DAFTAR GAMBAR

|     |   |    |
|-----|---|----|
| 2.1 | Kalimat awal (S0)   | 13 |
| 2.2 | Kalimat setelah melalui proses <i>line-shift coding</i> (S1)        | 13 |
| 2.3 | Perbedaan S0 dan S1   | 13 |
| 2.4 | Hasil penumpukkan kalimat S0 dan S1                                 | 13 |
| 3.1 | Diagram Transisi DFSA tahap 1[6]                                    | 17 |
| 3.2 | Diagram Transisi DFSA tahap 2[6]                                    | 19 |
| 3.3 | Diagram Transisi DFSA tahap 3[6]                                    | 20 |
| 3.4 | Format kamus sinonim  | 22 |
| 3.5 | <i>Data Flow Diagram Embedding</i>                                  | 23 |
| 3.6 | <i>Data Flow Diagram Extract</i>                                    | 25 |
| 3.7 | Use Case Diagram Perangkat Lunak Steganografi                       | 25 |
| 3.8 | Class Diagram perangkat lunak steganografi                          | 28 |
| 4.1 | Tampilan <i>tab</i>   | 31 |
| 4.2 | Tampilan <i>tab Embed</i>   | 32 |
| 4.3 | Tampilan <i>tab Extract</i>   | 32 |
| 4.4 | Tampilan <i>tab</i> Tambah <i>Cover</i>                             | 33 |
| 4.5 | Tampilan <i>tab</i> Tambah Sinonim                                  | 34 |
| 4.6 | Tampilan <i>tab</i> Petunjuk  | 35 |
| 4.7 | Diagram kelas lengkap   | 36 |
| 4.8 | Activity Diagram perangkat lunak steganografi saat menyisipkan      | 39 |
| 4.9 | Activity Diagram perangkat lunak steganografi saat proses ekstraksi | 39 |
| 5.1 | Tampilan <i>tab Embed</i>   | 41 |
| 5.2 | Tampilan <i>tab Extract</i>   | 42 |
| 5.3 | Tampilan <i>tab</i> Tambah <i>Cover</i>                             | 42 |
| 5.4 | Tampilan <i>tab</i> Tambah Sinonim                                  | 43 |
| 5.5 | Tampilan <i>tab</i> Petunjuk  | 43 |

## DAFTAR TABEL

|     |   |    |
|-----|---|----|
| 5.1 | Tabel Pengujian Fungsional . . . . .    | 44 |
| 5.2 | Tabel Pengujian Eksperimental . . . . . | 45 |





# BAB 1

## PENDAHULUAN

### 1.1 Latar Belakang

Tidak dapat dipungkiri bahwa teknologi membawa perubahan besar pada seluruh umat manusia di dunia. Dengan adanya internet yang merupakan salah satu produk dari kemajuan teknologi, jarak dan waktu bukanlah menjadi suatu masalah. Komunikasi antar benua, bahkan antar belahan dunia pun bukanlah masalah. Internet juga memudahkan pencarian segala informasi yang mulanya sulit ditemukan.

Namun segala sesuatu yang memiliki sisi positif, pasti ada sisi negatifnya. Informasi yang berserakan di internet dapat dilihat oleh siapa pun. Semua orang yang beraktivitas di internet dapat mengklaim dirinya sebagai siapa pun yang dikehendaki. Tidak terkecuali juga informasi pribadi yang seharusnya tidak diketahui sembarang orang, akhirnya dapat dibaca di internet. Hal ini memungkinkan orang yang tidak dikenali, dapat mengetahui nama lengkap, nama-nama kerabat, sampai alamat rumah yang seharusnya menjadi data pribadi dan tidak diketahui oleh sembarang orang. Semua informasi yang ada di internet merupakan informasi yang sengaja disebarkan atau pun yang secara tidak sadar ikut tersebar di internet.

Kriptografi merupakan salah satu cara yang dapat digunakan untuk menjaga kerahasiaan agar tidak sembarang orang dapat mengetahui suatu informasi. Dalam kriptografi ada dua istilah yang dikenal dengan enkripsi dan dekripsi, yang berarti menyandikan pesan dan memunculkan pesan asli. Proses enkripsi merupakan konversi data ke dalam bentuk lain yang disebut *ciphertext*<sup>1</sup>. *Ciphertext* tidak dapat dimengerti dengan mudah oleh semua orang, kecuali pihak yang berwenang. Dekripsi merupakan proses pemunculan kembali pesan asli dari *ciphertext*. Sebagian besar *ciphertext* merupakan tulisan yang tidak memiliki arti, sehingga hal ini akan dengan mudah menimbulkan kecurigaan pihak lain bahwa ada suatu informasi yang sebenarnya disembunyikan di dalam *ciphertext* tersebut.

Selain kriptografi, steganografi juga dapat digunakan untuk menjaga kerahasiaan informasi. Steganografi mengatasi permasalahan timbulnya kecurigaan pihak tidak berkepentingan pada saat membaca *ciphertext* yang biasanya merupakan kumpulan huruf dan angka yang tidak memiliki arti. Untuk orang awam yang melihat *ciphertext* mungkin tidak berarti apa-apa. Berbeda halnya jika yang melihat adalah orang yang mengerti kriptografi, dengan sendirinya akan diketahui bahwa ada informasi yang sengaja disembunyikan. Steganografi lebih berfokus pada penyembunyian informasi/pesan rahasia. Dengan menyembunyikan pesan rahasia di dalam pesan lain yang tidak bersifat

---

<sup>1</sup><http://searchsecurity.techtarget.com/definition/encryption>

rahasia, kecurigaan dari pihak tidak berkepentingan pun dapat diatasi.

Ada banyak alasan mengapa steganografi dibutuhkan. Orang-orang yang ingin konseling tentang masalah yang sangat pribadi akan merasa aman. Polisi dapat berkomunikasi dengan *undercover agents* untuk menyergap komplotan orang jahat. Informasi pribadi terjaga dari eksploitasi teroris dan orang jahat.[7]

Steganografi dapat menggunakan berbagai media sebagai media penyembunyian atau yang dikenal dengan istilah *stego-cover*, seperti contohnya media berupa gambar, video, suara, atau teks. Kebanyakan orang memakai gambar sebagai media, karena lebih mudah dalam teknik penyembunyiannya. Media teks masih belum banyak penggunaannya, karena harus memperhatikan cara penulisan, arti dari kalimat, dan lain-lain. Walaupun masih belum banyak yang menggunakannya, media teks merupakan media yang baik untuk menyembunyikan pesan karena kapasitasnya yang cukup besar. Ada berbagai cara menyembunyikan pesan di dalam teks, termasuk memanfaatkan bentuk dari teks itu sendiri (*format-based*) dan memanfaatkan susunan kalimat (*syntax-based*), tetapi belum banyak yang memanfaatkan suku kata.

## 1.2 Rumusan Masalah

Berikut beberapa rumusan masalah yang akan dibahas:

- Algoritma penyisipan mana yang tepat untuk diimplementasikan dengan memanfaatkan suku kata?
- Bagaimana cara implementasi algoritma pada perangkat lunak?
- Bagaimana performa yang dihasilkan perangkat lunak?

## 1.3 Tujuan

Berikut merupakan tujuan dari topik skripsi yang saya pilih:

- Menemukan algoritma yang mencakup seluruh/hampir seluruh kemungkinan (optimal).
- Menentukan seperti apa gambaran perangkat lunak yang akan dibuat.
- Mendapatkan performa yang baik dari perangkat lunak.

## 1.4 Deskripsi Perangkat Lunak

Perangkat lunak akhir yang akan dibuat memiliki fitur minimal sebagai berikut:

- Pengguna dapat menyembunyikan pesan rahasianya ke dalam pesan lain yang tidak bersifat rahasia.
- Perangkat lunak akan menghasilkan *stego-object* setelah selesai melakukan proses penyembunyian.
- Perangkat lunak dapat mengekstraksi pesan rahasia yang ada di dalam *stego-object*.



---

## 1.5 Metode Penelitian

Berikut adalah metode penelitian yang digunakan dalam pembuatan skripsi ini:

1. Melakukan studi literatur tentang pengolahan teks.
2. Melakukan studi literatur tentang pemenggalan kata pada Bahasa Indonesia dan jenis-jenisnya.
3. Mengumpulkan teks Bahasa Indonesia untuk dijadikan corpus.
4. Menganalisis kemunculan pola suku kata dari dokumen corpus.
5. Merancang algoritma pembangkitan cover untuk pesan rahasia.
6. Mengimplementasikan rancangan algoritma.
7. Melakukan pengujian.
8. Membuat dokumentasi skripsi.



## BAB 2

### LANDASAN TEORI

Dari topik yang ada, akan dipilah-pilah menjadi beberapa komponen penyusun. Pada judul terdapat suku kata bahasa Indonesia, tentunya akan ada pengantar terlebih dahulu tentang bahasa Indonesia sebelum mengenalkan suku katanya. Begitu pula dengan steganografi, maka pada bab ini akan dibahas mengenai dasar-dasar steganografi, istilah-istilah yang akan digunakan, dan beberapa contoh dari steganografi. Selain unsur-unsur yang nampak jelas pada topik, ada juga beberapa elemen yang tidak nampak tetapi akan dibutuhkan, seperti beberapa teori tentang pemotongan kata, bahasa alami, dan pencarian temu yang tergabung menjadi pengolahan teks.

#### 2.1 Bahasa Indonesia [1]

Bahasa Indonesia merupakan bahasa pemersatu Negara Indonesia. Dalam bahasa Indonesia dikenal bahasa tulisan dan bahasa lisan. Pada bahasa lisan dikenal istilah *fonem* yang merupakan satuan bahasa terkecil yang dapat membedakan arti. Dalam bahasa tulisan, *fonem* dilambangkan dengan huruf (alfabet). *Fonem* dibagi menjadi dua, yaitu vokal dan konsonan. Dalam bahasa Indonesia dikenal 5 vokal yaitu a, i, u, e, o, dan 25 konsonan yaitu b, c, d, f, g, h, j, k, l, m, n, p, q, r, s, t, v, w, x, y, z, kh, ng, ny, sy. Konsonan kh, ng, ny, dan sy merupakan contoh fonem yang terdiri atas dua huruf. Ada pula istilah diftong, yaitu gabungan 2 vokal yang membentuk kesatuan bunyi, di antaranya adalah au, ai, oi.

Bahasa Indonesia ditulis dengan menggunakan abjad. Abjad dalam bahasa Indonesia ada 52 huruf, yaitu 26 huruf (a sampai z) dan 26 huruf kapital (A sampai Z). Bahasa Indonesia juga memiliki 10 simbol untuk angka yaitu angka 0 sampai 9.

##### 2.1.1 Pemenggalan kata

Pemenggalan kata menghasilkan suku kata. Setiap kata pada Bahasa Indonesia memiliki suku kata yang berbeda-beda. Berikut pengelompokan berdasarkan jumlah suku katanya:

1. Terdiri dari satu suku kata, contoh: cat, bor, bom, dll.
2. Terdiri dari dua suku kata, contoh: pa-gi, ka-mu, dll.
3. Terdiri dari tiga suku kata, contoh: me-re-ka, ke-ma-ri, dll.
4. Terdiri dari empat suku kata, contoh: tu-na-wis-ma, da-sa-war-sa, dll.
5. Terdiri dari lima suku kata, contoh: pra-mu-ni-a-ga, dar-ma-wi-sa-ta, dll.

Bahasa Indonesia juga mengenal beberapa pola umum suku kata yaitu:

1. Pola V (vokal), contoh: **a**-nak, **i**-kan
2. Pola KV (konsonan vokal), contoh: **sa**-ku, **se**-la-ma
3. Pola VK, contoh: **an**-da, **am**-pun
4. Pola KVK, contoh: **lak**-sa-na, pe-**rak**
5. Pola KKV, contoh: **pra**-mu-ga-ri, **sas**-tra
6. Pola KKKV, contoh: ben-**trok**, kon-**trak**
7. Pola VKK, contoh: **ons**, **eks**
8. Pola KVKK, contoh: **pers**, kon-**teks**
9. Pola KKKVKK, contoh: kom-**pleks**
10. Pola KKKV, contoh: in-**stru**-men
11. Pola KKKVK, contoh: **struk**-tur

Untuk melakukan pemenggalan suku kata, telah dibuat aturan pemenggalan atau penyukuan kata:

1. Jika dua vokal berada di tengah kata, maka pemenggalan di antara dua vokal. Terdapat pengecualian untuk huruf diftong. Huruf diftong ai, au, dan oi tidak pernah dipisahkan sehingga pemenggalan kata tidak dilakukan di antara kedua huruf tersebut, contoh: main dipenggal menjadi ma-in.
2. Jika konsonan diapit dua vokal seperti kata anak, barang, maka pemenggalan sebelum huruf konsonan, contoh: a-nak, ba-rang.
3. Jika dua konsonan berurutan di tengah kata, pemenggalan dilakukan di antara huruf konsonan pertama dan kedua, contoh: sanjung menjadi san-jung.
4. Jika di tengah kata terdapat tiga konsonan atau lebih, pemenggalan dilakukan di antara konsonan pertama dan kedua, contoh: bentrok menjadi ben-trok.

## 2.2 Pengolahan Teks

### 2.2.1 Natural Language Processing[2]

*Natural Language Processing* atau disingkat menjadi NLP merupakan cabang ilmu *Artificial Intelligence* yang berfokus pada pengolahan bahasa alami. Bahasa alami merupakan bahasa yang digunakan manusia dalam berkomunikasi terhadap satu sama lain. Bahasa alami tidak dapat langsung diterima oleh komputer, sehingga harus diproses terlebih dahulu agar komputer dapat mengeksekusi perintah sesuai dengan keinginan *user*. Berikut beberapa area utama NLP:

- *Question Answering Systems (QAS)*

Komputer dapat menjawab pertanyaan dari *user*. Berbeda dengan *search engine* seperti *Google* misalnya yang menunggu masukan *keyword* dari *user*, dengan QAS *user* dapat langsung mengetikkan pertanyaannya dalam bahasa alami dalam berbagai bahasa.

- *Summarization*

Membuat ringkasan dari artikel, dokumen atau email secara otomatis. Dengan *auto-summarization*, *user* akan mendapatkan hasil ringkasan secara langsung. Ringkasan yang didapatkan telah berisi poin-poin penting dari keseluruhan dokumen.

- *Machine Translation*

Area NLP ini dapat menerima masukan berupa bahasa alami dan menghasilkan keluaran berupa terjemahan bahasa alami dalam bahasa lain (Inggris, Jerman, dll).

- *Speech Recognition*

Area yang merupakan cabang dari NLP yang cukup sulit karena mampu mengenali bahasa lisan. Biasanya merespon pada suara berupa perintah atau pertanyaan.

- *Document classification*

Salah satu cabang dari NLP yang paling banyak dipakai. Dapat menentukan dokumen apa termasuk jenis yang mana. Biasanya dipakai untuk filter *spam*, *movie classification*, dll.

NLP juga memiliki tiga aspek utama yaitu sintaks, semantik, dan pragmatik. Sintaks merupakan aturan tata bahasa, semantik menjelaskan arti dari suatu kalimat, dan pragmatik menjelaskan bagaimana pernyataan yang berhubungan dengan dunia. Selain itu juga yang terkait dengan NLP adalah morfologi, yang merupakan pengetahuan tentang kata dan bentukannya.

## 2.2.2 Information Retrieval[3]

*Information Retrieval* atau IR merupakan proses mencari, menemukan lalu mengambil dokumen yang relevan dengan informasi yang dicari user. *Search engine* yang populer seperti *Google* merupakan contoh dari sistem IR. Kata kunci yang biasa diketikkan pada *search engine* merupakan *query* yang akan digunakan oleh IR untuk mencari dan ditampilkan pada layar. Berikut merupakan karakteristik dari sistem IR:

- *Corpus*. *Document Corpus* merupakan kumpulan teks dalam jumlah besar yang biasanya digunakan untuk analisis statistik atau untuk disimpan dan diproses lagi. *Document Corpus* memiliki bagian-bagian yang berbeda. Dalam sebuah corpus terdapat judul, sub-judul, dan paragraph.
- *Queries*. Merupakan kata kunci atas apa yang ingin dicari. *Query* bisa berupa kata kunci atau kata-kata yang harus berdekatan.
- *A result set*. Hasil keluaran dari proses IR yang dinilai sebagai dokumen yang relevan dengan *query*.
- *Presentation of the result set*. Menampilkan list judul dari dokumen yang telah diberi ranking.

### 2.2.3 Teori Automata[4]

Teori automata merupakan teori mengenai mesin-mesin abstrak dan memiliki hubungan yang dekat dengan teori bahasa formal. Dalam tulisan ini akan dipergunakan istilah *automaton* sebagai bentuk tunggal dan *automata* sebagai bentuk jamak. Teori automata adalah teori tentang mesin abstrak yang:

- bekerja secara sekuensial
- menerima *input*
- menghasilkan *output*

Pengertian mesin di sini bukan hanya mesin elektronis/mekanis, tetapi juga perangkat lunak yang memenuhi ketiga kriteria tersebut. Seperti yang telah disampaikan sebelumnya, teori automata berhubungan erat dengan bahasa formal. Secara umum terdapat dua fungsi automata dalam hubungannya dengan bahasa, yaitu:

- sebagai pengenalan string dari suatu bahasa, dalam kasus ini bahasa sebagai masukan dari automata
- sebagai pengeksktraksi string dari suatu bahasa, dalam kasus ini bahasa sebagai keluaran dari automata

Hal yang akan ditekankan adalah poin pertama. Untuk mengenali string dari suatu bahasa, akan dimodelkan automaton yang memiliki komponen sebagai berikut:

- pita masukan, untuk menyimpan string masukan yang akan dikenali
- kepala pita, untuk membaca/menulis ke pita masukan
- *Finite State Controller* yang berisi status dan aturan yang mengatur langkah apa yang dilakukan oleh automaton berdasarkan status setiap saat dan simbol masukan yang dibaca oleh kepala pita
- pengingat, yang berfungsi sebagai tempat penyimpanan dan pemrosesan sementara

Setelah membaca string masukan dan melakukan langkah pemrosesan yang dibutuhkan, akan dihasilkan keputusan apakah string dapat dikenali. *Konfigurasi* merupakan suatu mekanisme untuk menggambarkan keadaan mesin pengenalan yang terdiri dari:

- status *Finite State Controller*
- isi pita masukan dan posisi kepalanya
- isi pengingat

Mesin pengenalan disebut *deterministik* bila di setiap konfigurasinya hanya ada satu kemungkinan yang dapat dilakukan mesin.

### 2.2.4 Teori Finite State Automata (FSA)[4]

Tiap jenis automata memiliki keunikan yang membedakan fungsinya dengan automata lain. Berikut adalah sifat-sifat FSA:

- pita masukan hanya bisa dibaca, berisi string yang berasal dari suatu abjad
- setelah membaca satu simbol pada pita, kepala pita akan bergeser ke posisi simbol berikutnya
- kepala pita tidak bisa mundur
- memiliki sejumlah status, setiap saat FSA berada dalam status tertentu

Setiap FSA bisa diasosiasikan dengan sebuah diagram transisi berupa graf berarah. Setiap simpulnya mewakili tiap status pada FSA. Jika ada transisi dari status  $a$  ke  $b$  pada input  $i$ , maka ada busur dari  $a$  ke  $b$  dengan label  $i$ . Status awal ditandai dengan kata START dan status akhir ditandai dengan dua lingkaran. Jadi cara kerja suatu FSA dapat digambarkan dengan diagram transisi.

### 2.2.5 Teori Deterministic FSA (DFSA)[4]

Sebelumnya telah disebutkan bahwa automaton dapat bekerja secara deterministik. Setiap bahasa reguler dapat dikenali oleh DFSA. Secara formal DFSA dapat dinyatakan dengan  $(Q, \Sigma, \delta, q_0, F)$  di mana

- $Q$  = himpunan berhingga status
- $\Sigma$  = himpunan berhingga simbol masukan
- $\delta$  = fungsi transisi yang memetakan  $Q \times \Sigma$  ke  $Q$
- $q_0$  = status awal  $q_0 \in Q$
- $F$  = himpunan status akhir,  $F \subseteq Q$

Cara kerja:

- pertama DFSA akan berada pada status  $q_0$ , kepala pita berada pada simbol pertama pita,
- lalu kepala pita akan membaca simbol dari pita dan bergeser maju,
- untuk tiap simbol, DFSA akan berpindah status sesuai fungsi  $\delta$ ,
- proses berakhir jika simbol masukan pita sudah habis,
- jika di akhir proses didapatkan status akhir, maka string masukan diterima dan bila tidak, maka string masukan ditolak.

## 2.3 Steganografi

Kriptografi tradisional berhasil mengamankan pesan secara matematis[7]. Aman secara matematis berarti pesan yang ada bisa saja dibuka, tetapi membutuhkan waktu yang sangat lama. Ketika pesan berhasil dibuka, informasinya sudah tidak lagi bermanfaat. Steganografi menyembunyikan informasi sehingga tidak dapat ditemukan. Kedua hal tersebut memiliki tujuan yang sama, namun melewati proses yang berbeda. Pada steganografi proses penyembunyian dan ekstraksi kembali informasi yang disembunyikan dikenal dengan istilah *embedding* dan *extracting information*. Sementara untuk informasi rahasia yang akan disembunyikan dikenal dengan istilah *secret*. Proses *embedding* memerlukan *secret* dan *stego-cover* sebagai media penyembunyiannya. Hasil dari proses *embedding* disebut dengan *stego-object*.

Untuk menyembunyikan informasi tentunya dibutuhkan suatu algoritma. Sebagian algoritma menyembunyikan informasi dengan menggunakan kunci untuk mengatur bagaimana penyisipan informasi. Sebagian lagi menyembunyikan informasi tidak menggunakan kunci, sehingga proses ekstraksinya pun tidak memerlukan kunci. Hal ini mirip dengan kriptografi, walaupun dilakukan dengan cara menyembunyikan informasi.

Steganografi teks menggunakan teks sebagai media untuk penyisipan (*embed*) informasi. Steganografi teks merupakan jenis yang paling jarang digunakan. Hal ini disebabkan sulitnya menyembunyikan teks dalam jumlah besar pada teks lain sebagai medianya. Walaupun demikian, teks sebagai media memiliki kapasitas lebih besar daripada suara, gambar, atau video sebagai medianya karena memiliki ukuran yang lebih kecil dan dapat menampung lebih banyak informasi. Steganografi teks dibedakan menjadi dua kategori, yaitu steganografi dengan memanfaatkan ilmu bahasa (*linguistic steganography*) dan steganografi yang memanfaatkan bentuk visual dari huruf (*format based steganography*).

### 2.3.1 Format Based Steganography[5]

Metode ini menggunakan format fisik dari teks seperti spasi, tinggi huruf, dan format fisik lainnya yang dapat dimanfaatkan untuk menyisipkan informasi. Metode ini secara umum mengubah teks yang ada untuk menyembunyikan pesan rahasia. Spasi atau karakter yang tidak ditampilkan, kesalahan penulisan yang disengaja disembunyikan pada teks, mengubah ukuran tulisan merupakan contoh dari *format-based steganography*. Beberapa cara di atas seperti kesalahan penulisan dapat mengecoh pembaca yang menganggapnya wajar, namun jika memiliki dokumen asli, dapat dilakukan perbandingan antara dokumen yang asli dan dokumen hasil steganografi (*stego-object*), sehingga bagian-bagian yang diganti dapat terlihat.

#### The Word-Shift Coding

Mirip dengan *line-shift coding*, teknik ini menggeser kata secara horizontal untuk memberikan sebuah tanda. Pada laporan teknisnya, Neil F. Johnson memberikan contoh bahwa suatu teks dapat mengandung teks lain di dalamnya dengan menggunakan *line-shift coding*[8]. Diberikan suatu kalimat S0:

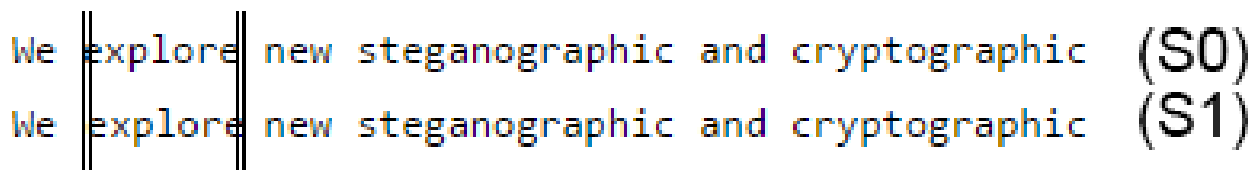


```
We explore new steganographic and cryptographic
algorithms and techniques throughout the world to
produce wide variety and security in the electronic web
called the Internet.
```

Gambar 2.1: Kalimat awal (S0)

lalu setelah disembunyikan suatu pesan dengan menggunakan *line-shift coding*, muncul kalimat S1.

```
We explore new steganographic and cryptographic
algorithms and techniques throughout the world to
produce wide variety and security in the electronic web
called the Internet.
```

Gambar 2.2: Kalimat setelah melalui proses *line-shift coding* (S1)


Gambar 2.3: Perbedaan S0 dan S1

Dengan menumpukkan kedua kalimat S0 dan S1, didapatkan hasil sebagai berikut:

```
We explore new steganographic and cryptographic
algorithms and techniques throughout the world to
produce wide variety and security in the electronic web
called the Internet.
```

Gambar 2.4: Hasil penumpukkan kalimat S0 dan S1

Hasil ini dapat dicapai dengan melebarkan spasi sebelum kata *explore*, *the*, *wide*, dan *web*, juga menyempitkan spasi setelah kata *explore*, *world*, *wide*, dan *web* pada S1. Dengan demikian, kalimat yang mengandung *shifted-word* akan tetap memiliki arti yang sama.

### Feature Coding

*Feature coding* memanfaatkan ciri yang ada dalam teks yang dapat diubah. Seperti garis lurus yang ada pada huruf b, d, h, k, dan lain-lain. Panjang dari garis tersebut dapat diubah tanpa membuat curiga pembaca pada umumnya. Dapat juga dilakukan substitusi pada beberapa kata dengan kata sinonimnya. Dengan itu kita dapat menyisipkan informasi 0 pada sinonim pertama dan 1 pada sinonim kedua. Cara ini harus melalui kesepakatan antara dua belah pihak tentang pasangan sinonimnya.

### Inter-sentence Spacing

Cara ini akan menyandikan teks yang sudah dalam bentuk biner ke dalam teks dengan menyisipkan satu atau dua spasi setelah tiap kalimat. Misalkan satu spasi menyandikan 0 dan dua spasi menyandikan 1. Sayangnya cara ini memiliki beberapa masalah, yang pertama jika ingin menyisipkan

teks yang memiliki ukuran yang besar, akan membutuhkan teks cover yang banyak pula karena 1 kalimat hanya dapat menyandikan 1 bit. Kedua, beberapa perangkat pengolah kata secara otomatis mengubah banyaknya spasi menjadi satu.

### **End of Line Spacing**

Cara ini menyisipkan spasi pada akhir baris. Cara ini meningkatkan banyaknya informasi yang dapat disandikan dibandingkan cara sebelumnya. Sama seperti sebelumnya, cara ini dapat terganggu oleh beberapa program yang dapat menghapus spasi yang berlebihan dari teks dan cara ini tidak dapat menampilkan pesan rahasia dari *hard copy*.

### **Inter-word Spacing**

Cara ketiga menggunakan spasi untuk menyandikan data yang menggunakan *right-justification*. Data disandikan dengan mengatur di mana spasi ekstra akan diletakkan. Satu spasi di antara kata menyandikan 0 dan dua spasi menyandikan 1. Cara ini memungkinkan beberapa bit dapat disembunyikan dalam satu baris.

### **2.3.2 Linguistic Steganography**

Metode ini memanfaatkan *natural language processing* atau pemrosesan bahasa alami yang menyebabkan pesan dapat disembunyikan tanpa mengubah bahasa asli. Contohnya bisa dengan melakukan substitusi terhadap sinonim yang termasuk steganografi leksikal. Perubahan dari kalimat aktif ke pasif atau sebaliknya untuk contoh dari steganografi sintaksis. Pemanfaatan suku kata untuk menyisipkan pesan yang juga termasuk ke dalam steganografi linguistik yang akan menjadi bahasan utama dalam skripsi ini.

## BAB 3

### ANALISIS

Pada bab ini akan dibahas mengenai algoritma yang dipilih untuk menyembunyikan *secret message*. Untuk itu penjelasannya akan dibagi menjadi beberapa bagian. Bagian pertama akan menjelaskan tentang gambaran umum algoritma yang dipakai. Selanjutnya akan dijelaskan bagaimana pemenggalan kata akan dilakukan. Ada pula bagian kamus sinonim, penjelasan lebih lengkapnya terdapat pada Bab 3.1 dan Bab 3.1.3. Setelah penjelasan tentang kamus sinonim, akan dijelaskan juga mengenai *stego-cover* seperti apa yang akan dipakai, serta bagaimana proses *embedding* dan *extracting information*.

#### 3.1 Gambaran Algoritma Umum

*Stego-cover* yang akan digunakan berupa *text file* yang tentu terdiri dari banyak kata. Algoritma yang dipilih akan memanfaatkan banyaknya suku kata dalam suatu kata. Notasi  $c(w)$  akan digunakan untuk merepresentasikan banyaknya suku kata dalam suatu kata, dimana suatu katanya adalah  $w$ . Diketahui bahwa  $c(w)$  dapat bernilai ganjil atau genap. Dua kemungkinan tersebut identik dengan bilangan biner. Dengan demikian diputuskan bahwa  $c(w)$  yang bernilai genap akan merepresentasikan 0 dan  $c(w)$  bernilai ganjil merepresentasikan 1. Dapat ditarik kesimpulan berarti 1 kata dapat diubah menjadi 1 bit bilangan biner. Jika sebuah *stego-cover* memiliki 700 kata, berarti *stego-cover* tersebut memiliki kapasitas sebesar 700 bit.

Setelah menemukan cara mengubah *stego-cover* menjadi bilangan biner, *secret message* juga harus diubah menjadi bilangan biner. Diketahui bahwa setiap karakter yang ditampilkan pada layar komputer, memiliki kode ASCII yang unik. Kode ASCII merupakan representasi numerik dari karakter seperti 'a', '1', atau '@'. Kode ASCII merupakan representasi numerik, berarti kode ASCII dapat diubah lagi menjadi bilangan biner. Sebagai contoh, jika sebuah *secret message* terdiri dari  $n$  karakter dan tiap karakter diubah menjadi 7 bit bilangan biner dengan memanfaatkan kode ASCII, maka dibutuhkan minimal  $7*n$  bit *stego-cover* untuk dapat menyembunyikan *secret message* tersebut.

*Stego-cover* dan *secret message* kini dapat diubah menjadi deret bilangan biner, *stego-cover* harus dapat diubah lagi sedemikian rupa agar  $7*n$  deret biner pertamanya identik dengan deret biner *secret message*. Sehingga saat proses ekstraksi, deret biner dari *stego-cover* dapat diubah lagi menjadi karakter yang sesuai dengan *secret message*. Karena  $c(w)$  dalam *stego-cover* dapat bernilai 0 atau 1, ada kemungkinan  $c(w)$  tidak sesuai dengan angka biner dari *secret message* yang akan disembunyikan. Untuk mengatasi hal ini, kata dengan nilai  $c(w)$  yang tidak sesuai, akan diganti

dengan sinonim yang memiliki  $c(w)$  yang sesuai.

Tentunya semua kata yang ada pada *stego-cover* harus memiliki sinonim agar setiap kata dapat memiliki  $c(w)$  yang bernilai 0 atau 1. Tesaurus merupakan buku referensi berupa daftar kata dengan sinonimnya<sup>1</sup>. Namun karena format tesaurus yang memiliki banyak pengelompokkan kata, seperti kata "bobol" bisa berarti "ambrol"/"hancur", bisa juga berarti "buyar"/"tembus". Lalu dari kata "bobol" bisa jadi "membobol" yang juga bisa berarti "membobok", bisa juga berarti "mencuri". Karena format yang terlalu rumit untuk dapat dibaca secara otomatis oleh program, maka diputuskan untuk membuat kamus sinonim yang lebih ringkas. Untuk kamus pada skripsi ini, telah disiapkan kamus sinonim yang daftar sinonimnya sebagian besar diambil dari tesaurus. Kamus sinonim ini berisi pasangan kata dengan sinonimnya pada tiap barisnya. Karena kamus sinonim dibuat sendiri, maka daftar kata yang ada di dalamnya, masih terbatas, artinya jika pengguna menambahkan *stego-cover* baru, pengguna juga harus melengkapi sinonim yang belum terdaftar pada kamus sinonim.

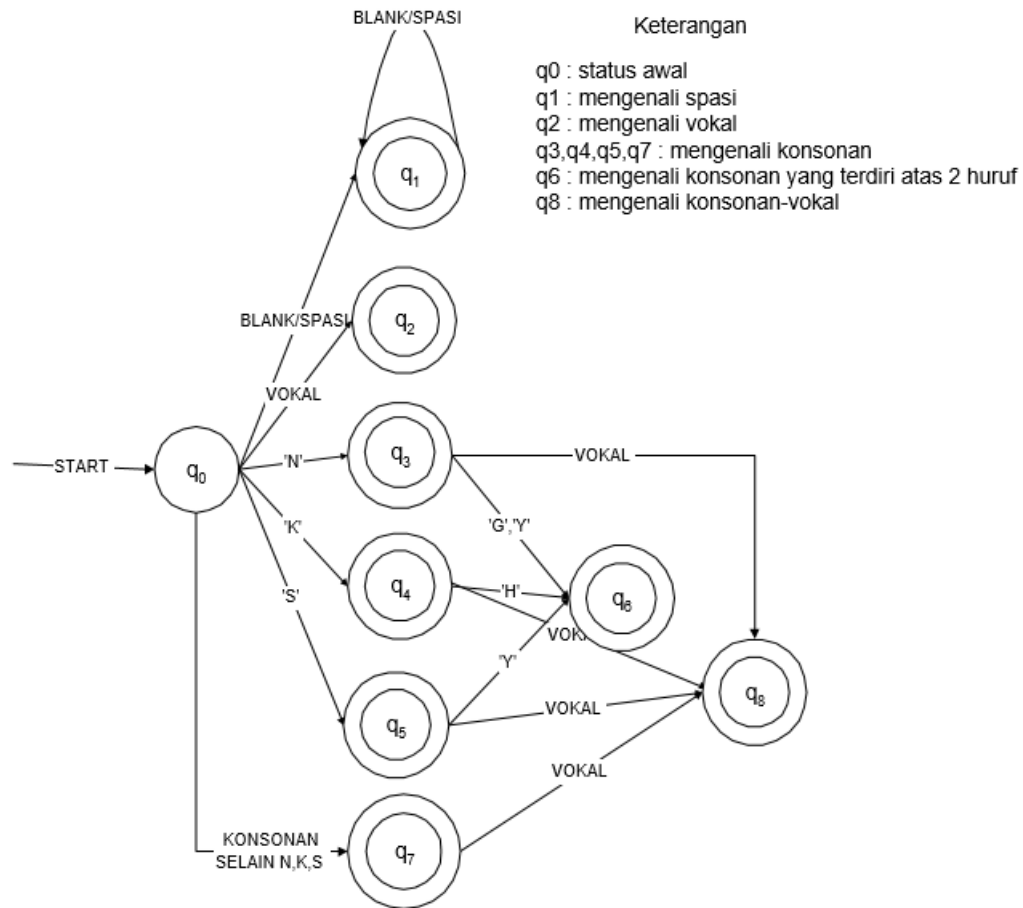
### 3.1.1 Pemenggalan Kata

Untuk proses pemenggalan kata, akan menggunakan skripsi Frisca Sumarlin 2010 yang berjudul Aplikasi Pendongeng. Untuk mengenali suku kata dalam bahasa Indonesia, akan digunakan tiga tahap DFSA. Setiap masukan dari DFSA merupakan hasil DFSA tahap sebelumnya. Seperti yang telah dijelaskan, FSA memiliki prinsip kerja membaca hanya satu arah dan tidak bisa membaca mundur masukannya. Sedangkan untuk mengenali suku kata, untuk beberapa kasus kata dibutuhkan kemampuan untuk membaca mundur masukannya. Sebagai contoh, pada kata "anda", huruf ketiga adalah huruf konsonan sehingga pemenggalan suku kata dapat dilakukan pada saat membaca huruf ketiga.

Pada tahap 1 ini memang beberapa kata belum sesuai dengan aturan penyukuan kata yang benar, karena itu akan dilakukan tiga tahapan. Untuk diagramnya dapat dilihat pada Gambar 3.1.

---

<sup>1</sup><http://kbbi.web.id/tesaurus>



Gambar 3.1: Diagram Transisi DFSA tahap 1[6]

Penjelasan tentang Diagram Transisi DFSA tahap 1 untuk mengenali spasi, vokal (V), konsonan yang terdiri dari 1 huruf (K), konsonan yang terdiri dari 2 huruf (KK), dan konsonan-vokal (KV) adalah sebagai berikut.

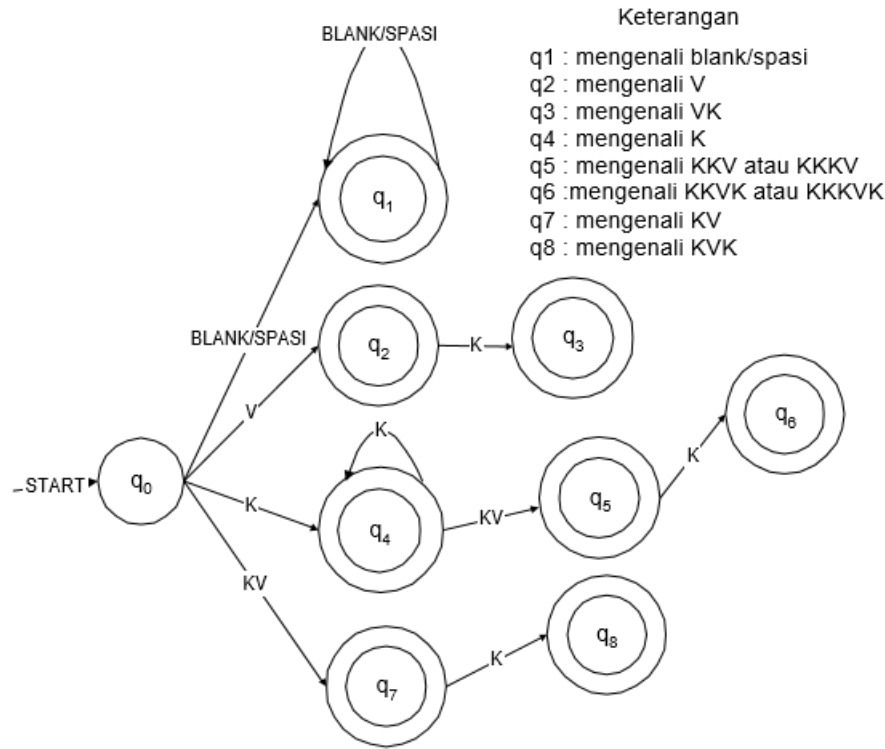
- Perpindahan dari status awal ( $q_0$ ) ke  $q_1$  adalah untuk mengenali spasi atau *string* kosong.
- Perpindahan dari  $q_0$  ke  $q_2$  adalah untuk mengenali huruf vokal.
- Perpindahan dari  $q_0$  ke  $q_3$  adalah untuk mengenali huruf 'N'.
- Perpindahan dari  $q_0$  ke  $q_4$  adalah untuk mengenali huruf 'K'.
- Perpindahan dari  $q_0$  ke  $q_5$  adalah untuk mengenali huruf 'S'.
- Perpindahan dari  $q_0$  ke  $q_7$  adalah untuk mengenali huruf konsonan selain 'N', 'K', dan 'S'.
- Perpindahan dari  $q_3$  ke  $q_8$  adalah untuk mengenali pola suku kata VK.
- Perpindahan dari  $q_3$  ke  $q_6$  adalah untuk mengenali pola suku kata KK (ng dan ny).
- Perpindahan dari  $q_3$  ke  $q_8$  adalah untuk mengenali pola suku kata KV.
- Perpindahan dari  $q_4$  ke  $q_6$  adalah untuk mengenali pola suku kata KK (kh).

- Perpindahan dari  $q_4$  ke  $q_8$  adalah untuk mengenali pola suku kata KV.
- Perpindahan dari  $q_5$  ke  $q_6$  adalah untuk mengenali pola suku kata KK (sy).
- Perpindahan dari  $q_5$  ke  $q_8$  adalah untuk mengenali pola suku kata KV.
- Perpindahan dari  $q_7$  ke  $q_8$  adalah untuk mengenali pola suku kata KV.

Sebagai contoh, jika melakukan pemenggalan kata hanya dengan menggunakan DFSA tahap 1 saja, kata "gondok" tidak dapat dipotong secara sempurna. Berikut langkah-langkahnya.

- Pada status awal ( $q_0$ ) huruf ke-1 'g' akan diperiksa.
- Pindah ke  $q_7$  karena huruf 'g' merupakan huruf konsonan selain huruf 'N', 'K', dan 'S'.
- Huruf ke-2 'o' diperiksa dan pindah ke  $q_8$  karena huruf 'o' merupakan huruf vokal. Suku kata ke-1 "go" disimpan, lanjut ke huruf selanjutnya dan mulai lagi dari  $q_0$ .
- Huruf ke-3 'n' diperiksa dan pindah ke  $q_3$  karena huruf ke-3 adalah huruf 'n'.
- Huruf ke-4 'd' diperiksa dan tidak memenuhi syarat untuk ke  $q_8$  ataupun  $q_6$ . Suku kata ke-2 "n" disimpan. Huruf ke-4 'd' kembali ke  $q_0$ .
- Pindah ke  $q_7$  karena huruf 'd' merupakan huruf konsonan selain huruf 'N', 'K', dan 'S'.
- Huruf ke-5 'o' diperiksa dan pindah ke  $q_8$  karena 'o' merupakan huruf vokal. Suku kata ke-3 "do" disimpan, lanjut ke huruf selanjutnya dan mulai lagi dari  $q_0$ .
- Huruf ke-6 'k' diperiksa dan pindah ke  $q_4$  karena huruf ke-6 adalah huruf 'K'. Suku kata ke-4 "k" disimpan. Tahap 1 selesai karena semua huruf telah melewati diagram transisi DFSA tahap 1.
- Didapatkan hasil pemenggalan suku kata "gondok" menjadi "go-n-do-k".

Dapat dilihat bahwa jika hanya menggunakan DFSA tahap 1, hasilnya masih belum sempurna. Pada tahap 1 hanya dikenali pola suku kata V, K, KV, dan KK. Selanjutnya, setelah melewati tahapan 1 DFSA, *output*nya akan menjadi masukan untuk DFSA tahap 2. Pada tahap 2 ini harus dikenali V, KV, dan pengembangan dari ketiga pola yang dikenali pada tahap pertama. Pada tahap ini, pengembangan yang bisa dilakukan adalah V+K, K+KV, K+KV+K, K+K+KV, K+K+KV+K, K+KV+K, dan KV+K. Untuk digram transisi tahap kedua dapat dilihat pada Gambar 3.2.

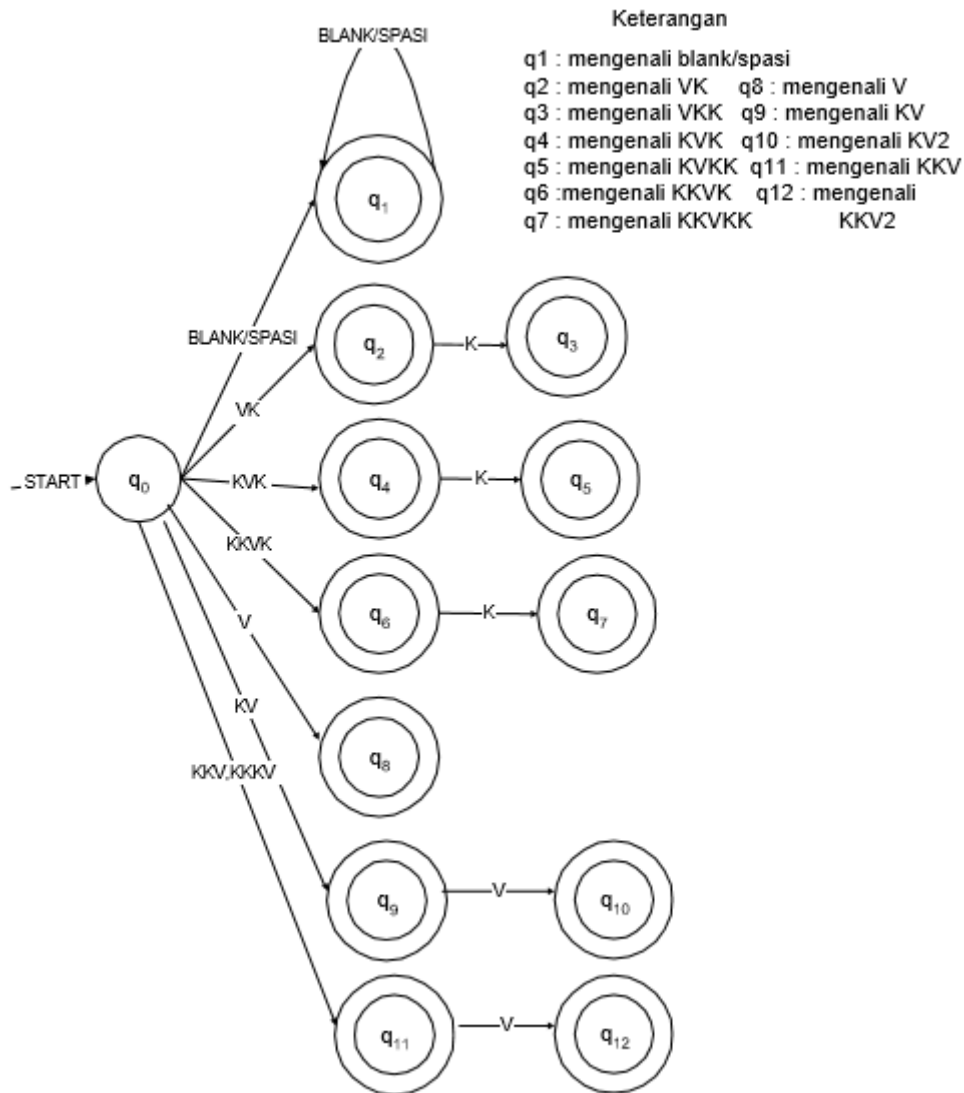


Gambar 3.2: Diagram Transisi DFSA tahap 2[6]

Penjelasan tentang Diagram Transisi DFSA tahap 2 untuk mengenali pola suku kata V, VK, K, KKV, KKKV, KKVK, KKKVK, KV, dan KVK adalah sebagai berikut.

- Perpindahan dari status awal ( $q_0$ ) ke  $q_1$  adalah untuk mengenali spasi atau *string* kosong.
- Perpindahan dari  $q_0$  ke  $q_2$  adalah untuk mengenali pola suku kata V.
- Perpindahan dari  $q_0$  ke  $q_4$  adalah untuk mengenali pola suku kata K.
- Perpindahan dari  $q_0$  ke  $q_7$  adalah untuk mengenali pola suku kata KV.
- Perpindahan dari  $q_2$  ke  $q_3$  adalah untuk mengenali pola suku kata VK.
- Perpindahan dari  $q_4$  ke  $q_4$  adalah untuk mengenali pola suku kata KK.
- Perpindahan dari  $q_4$  ke  $q_5$  adalah untuk mengenali pola suku kata KKV atau KKKV.
- Perpindahan dari  $q_5$  ke  $q_6$  adalah untuk mengenali pola suku kata KKVK atau KKKVK.
- Perpindahan dari  $q_7$  ke  $q_8$  adalah untuk mengenali pola suku kata KVK.

Pada tahap 2, masih ada tiga pola suku kata yang belum dapat dikenali, yaitu VKK, KVKK, dan KKVKK. Untuk itu masih dibutuhkan satu tahapan lagi untuk dapat mengenali pola-pola tersebut dan mengenali huruf diftong. Pada tahap ini hasil dari DFSA tahap kedua akan menjadi masukan bagi DFSA tahap terakhir yaitu tahap 3.



Gambar 3.3: Diagram Transisi DFSA tahap 3[6]

Penjelasan tentang Diagram Transisi DFSA tahap 3 untuk mengenali pola suku kata VK, V, VKK, KV, KVK, KVV, KVKK, KKV, KKKV, KKKVK, dan KKKV adalah sebagai berikut.

- Perpindahan dari status awal ( $q_0$ ) ke  $q_1$  adalah untuk mengenali spasi atau *string* kosong.
- Perpindahan dari  $q_0$  ke  $q_2$  adalah untuk mengenali pola suku kata VK.
- Perpindahan dari  $q_0$  ke  $q_4$  adalah untuk mengenali pola suku kata KVK.
- Perpindahan dari  $q_0$  ke  $q_6$  adalah untuk mengenali pola suku kata KKKV.
- Perpindahan dari  $q_0$  ke  $q_8$  adalah untuk mengenali pola suku kata V.
- Perpindahan dari  $q_0$  ke  $q_9$  adalah untuk mengenali pola suku kata KV.
- Perpindahan dari  $q_0$  ke  $q_{11}$  adalah untuk mengenali pola suku kata KKV atau KKKV.
- Perpindahan dari  $q_2$  ke  $q_3$  adalah untuk mengenali pola suku kata VKK.



- Perpindahan dari  $q_4$  ke  $q_5$  adalah untuk mengenali pola suku kata KVKK.
- Perpindahan dari  $q_6$  ke  $q_7$  adalah untuk mengenali pola suku kata KKVKK.
- Perpindahan dari  $q_9$  ke  $q_{10}$  adalah untuk mengenali pola suku kata KVV.
- Perpindahan dari  $q_{11}$  ke  $q_{12}$  adalah untuk mengenali pola suku kata KKVV atau KKKVV.

Untuk dapat membagi suku kata, diperlukan automata yang dapat menerima masukan berupa kata dan keluaran berupa suku kata. *Finite State Transducer* merupakan *finite state machine* yang memiliki dua pita, yaitu pita masukan dan pita keluaran. Automata yang telah dirancang adalah DFSA, di mana keluaran yang dihasilkan adalah dikenali atau tidak dikenali. Dengan merubah keluaran dari DFSA tersebut menjadi suku kata untuk setiap sekuens karakter yang telah dikenali, maka DFSA yang telah dirancang dapat dimanfaatkan untuk membagi kata menjadi suku kata. Atas dasar itu, DFSA dipilih untuk melakukan penyukuan kata.

Pemenggalan suku kata akan dipakai untuk mendapatkan banyak suku kata kata-kata yang terdapat pada *stego-cover*, sehingga dapat dihitung banyak suku kata tiap katanya. Namun karena kumpulan *stego-cover* yang akan dipakai dapat berupa cerita pendek(cerpen) atau puisi, maka isi dari *stego-cover* tidak hanya berupa kata-kata alfabet saja, tetapi juga ada angka yang bisa berupa tanggal, kata serapan, dan lain-lain. Hal ini kemudian dapat menjadi hambatan, karena seperti yang kita ketahui bahwa angka tidak memiliki suku kata. Hambatan lainnya juga datang dari kata-kata dalam bahasa asing dan singkatan.

Setelah menemui hambatan-hambatan di atas, akhirnya diputuskan beberapa solusi. Pertama-tama, dari keseluruhan isi *stego-cover*, tanda baca akan diabaikan (kecuali tanda penghubung '-'), sehingga yang akan dipenggal hanyalah kata-katanya saja. Satu digit angka akan dihitung sebagai satu suku kata, namun pada *stego-cover* diputuskan untuk tidak boleh mengandung angka sama sekali, karena tidak memiliki sinonim. Semua singkatan atau kata dalam bahasa asing, akan dilakukan pemenggalan kata apa adanya.

### 3.1.2 *Stego-cover*

*Stego-cover* yang telah dikumpulkan berupa puisi dan cerpen. Alasan dipilihnya cerpen dan puisi sebagai *stego-cover*, karena puisi umumnya menggunakan satu kata berulang-ulang, sehingga dapat mengurangi banyaknya kata yang disimpan pada kamus sinonim. Sedangkan cerpen yang dipilih memiliki jumlah kata yang cukup banyak, ini menandakan kapasitas penyimpanan yang cukup besar.

Dengan dipilihnya cerpen dan puisi sebagai *stego-cover*, maka dibutuhkan skema komunikasi yang dapat menyamarkan artikel tersebut agar tidak menimbulkan kecurigaan oleh pihak lain. Skema komunikasi yang dipilih berupa komunikasi antara dua mahasiswa yang gemar mengirim puisi atau cerpen melalui *email*. *Email* yang dikirim mengandung *stego-object* yang merupakan hasil dari *embedding secret message* pada *stego-cover*. Dengan skema komunikasi seperti ini, jika ada pihak lain yang membaca *email* kedua mahasiswa tersebut, tidak akan mencurigai adanya kejangalan karena kedua mahasiswa gemar berkirim puisi dan cerpen.

### 3.1.3 Kamus sinonim

Proses *embedding* akan memanfaatkan daftar sinonim yang ada pada kamus sinonim. Kamus sinonim akan disimpan dalam sebuah *file* dengan ekstensi *.txt*. Kamus sinonim dibuat dengan format seperti pada Gambar 3.4. Dimana kata pertama merupakan kata yang ada pada *stego cover* dan kata di sebelahnya adalah sinonimnya. Format ini dipilih agar pencarian sinonim lebih mudah karena sudah berpasangan antara kata dan sinonimnya. Untuk skripsi ini telah disiapkan terlebih dahulu kamus sinonim yang dibuat secara manual. Pada kamus sinonim setiap kata yang ada pada *stego-cover* memiliki sinonimnya sendiri. Referensi utama pembuatan kamus sinonim ini adalah tesaurus. Tesaurus merupakan buku referensi berupa daftar kata dengan sinonimnya<sup>2</sup>. Kata-kata yang terdaftar di tesaurus adalah kata baku, sehingga jika pada *stego-cover* terdapat kata yang tidak dapat ditemukan pada tesaurus, diperlukan penanganan tersendiri.

```
jelek-jelekkan ejek-ejek
terang-terangan langsung
kagumi idolakan
peri batari
wanita perempuan
cinta asmara
di pada
```

Gambar 3.4: Format kamus sinonim

Solusinya adalah dengan mengubah-ubah imbuhan pada awal atau akhir kalimat, namun tetap memperhatikan konteks kalimat tersebut. Sebagai contoh, pada *stego-cover* terdapat kata 'menggendongnya', namun kata tersebut tidak dapat ditemukan pada tesaurus, karena terdapat imbuhan. Sehingga untuk sinonimnya dapat digunakan kata 'menggendong'. Kedua kata tersebut memiliki jumlah suku kata yang berbeda, namun memiliki arti yang sama.

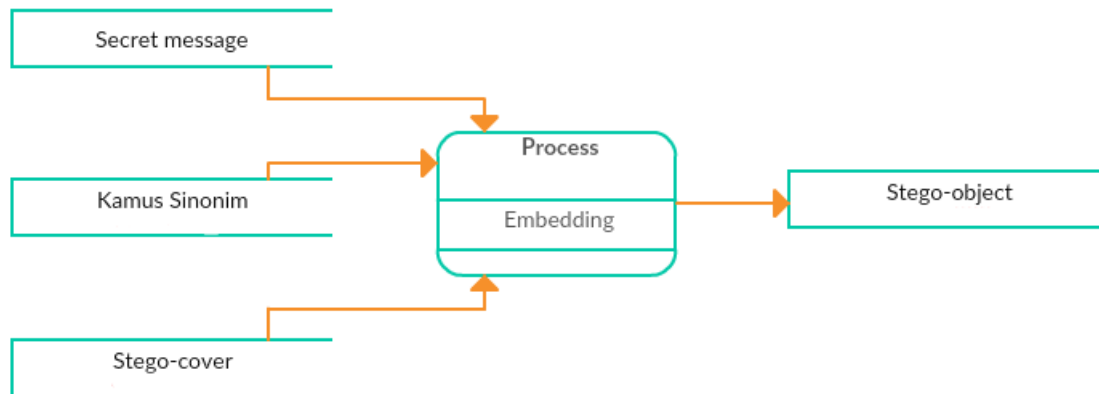
Terdapat beberapa kata yang memang tidak memiliki sinonim dengan jumlah suku kata yang berbeda. Kata 'lama' memiliki 2 suku kata, sedangkan sinonim yang ada pada Tesaurus adalah 'lamban' dan 'lelet'. Kedua sinonimnya memiliki  $c(w)$  genap, tetapi yang dibutuhkan adalah  $c(w)$  ganjil. Hal ini dapat diatasi dengan menyamakannya sebagai *typo* atau kesalahan pengetikkan. Contohnya dengan memasukkan kata 'lambaan' atau 'lamaa' sebagai sinonim dari kata 'lama'. Kata 'lambaan' dan 'lamaa', jika dipotong suku katanya dengan menggunakan pemotong suku kata yang telah dibuat, kedua kata tersebut memiliki  $c(w)$  ganjil.

### 3.1.4 Embedding

Proses *embedding* akan memanfaatkan suku kata dan sinonim kata itu sendiri. Ide utamanya adalah dengan mengganti kata-kata tertentu yang ada dalam *stego-cover* dengan kata-kata yang telah disediakan pada kamus sinonim, agar  $7*n$  deret bilangan biner pertama pada *stego-cover* identik dengan deret bilangan biner *secret message*. Kamus sinonim berisi semua kata yang ada pada *stego cover* berpasangan dengan sinonimnya. Pasangan kata yang ada pada *database* merupakan sinonim dengan  $c(w)$  yang berbeda (ganjil dan genap) dengan kata aslinya. *Data Flow Diagram Embedding* dapat dilihat pada Gambar 3.5.

<sup>2</sup><http://kbbi.web.id/tesaurus>

Penggantian kata pada *stego-cover* ditentukan dari ASCII *secret message* yang akan disembunyikan. ASCII (*American Standard Code for Information Interchange*) merupakan format yang paling umum untuk *file* teks yang ada di komputer dan internet<sup>3</sup>. Kode ASCII yang akan dipakai direpresentasikan dengan 7-bit angka biner, yang berarti ada 7 digit angka 0 atau 1. Pemilihan 7-bit ini dikarenakan kode ASCII pada bilangan desimal hanya ada dari 0 sampai 127, sehingga jika menggunakan 8-bit, angka pertama(paling kiri) akan selalu bernilai 0. Dapat disimpulkan bahwa untuk menyisipkan 7-bit dibutuhkan 7 kata, 8-bit dibutuhkan 8 kata. Untuk alasan memaksimalkan kapasitas penyisipan, maka diputuskan untuk merepresentasikan 1 karakter menjadi 7-bit.



Gambar 3.5: *Data Flow Diagram Embedding*

Setelah *secret message* diubah menjadi deret bilangan biner dengan memanfaatkan kode ASCII, satu per satu digit kodenya akan dicocokkan dengan  $c(w)$  pada *stego cover*. Kode digit pertama akan disisipkan pada kata pertama *stego-cover*, digit kedua disisipkan pada kata kedua, dst. Jika  $c(w)$  pada *stego-cover* tidak sesuai dengan digit kodenya, maka kata tersebut akan diganti dengan sinonimnya yang ada pada kamus sinonim. Pemberitahuan akan muncul jika ada kata yang sinonimnya tidak ditemukan pada kamus sinonim.

### Algoritma

Seperti yang telah dijelaskan sebelumnya, algoritma ini akan memanfaatkan kata yang ada pada *stego-cover* untuk menyisipkan kode ASCII dari pesan rahasia ditambah kode ASCII karakter '#' (untuk menandakan bahwa pesan rahasia telah berakhir). Proses penyisipan ini juga memanfaatkan kamus sinonim untuk mengubah kata asli yang ada pada *stego-cover* dengan sinonim yang ada pada kamus sinonim jika  $c(w)$  tidak sesuai dengan kode ASCII pesan rahasia.

Algoritma *embedding* pesan rahasia pada *stego-cover* dengan memanfaatkan sinonim

1. Meminta input berupa pesan rahasia(*secret message*).
2. Tiap karakter pada *secret message* diubah menjadi kode ASCII 7-bit dan disimpan menjadi *secret code*.
3. Tambahkan deret biner 0100011, yang merupakan kode ASCII untuk karakter '#' untuk menandakan bahwa *secret message* telah berakhir.

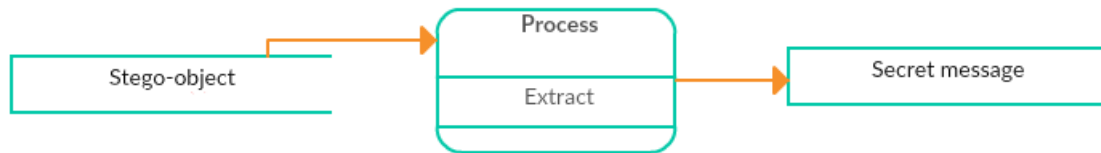
<sup>3</sup><http://whatis.techtarget.com/definition/ASCII-American-Standard-Code-for-Information-Interchange>

4. Jika  $n$  adalah banyak karakter yang ada pada *secret message*, maka panjang *secret code* =  $7*n+1$  (ditambah karakter '#').
5. Cari *stego cover* yang dapat menampung *secret code*. (Banyaknya kata yang ada dalam *stego cover*  $\geq$  panjang *secret code*).
6. Bangkitkan angka acak ( $r$ ) dari 0 sampai banyaknya *stego cover* yang dapat menampung *secret code* - 1.
7. Buka *file* ke- $(r)$  dan baca per kata dengan mengabaikan tanda baca dan spasi.
8. Untuk  $7*n+1$  kata pertama, lakukan pemenggalan kata dan dapatkan nilai  $c(w)$ .
9. Cocokkan nilai  $c(w)$  dengan angka ke- $n$  yang ada pada *secret code* secara berurutan. ( $c(w)$  kata ke- $n$  dicocokkan dengan angka ke- $n$  *secret code*).
10. Jika pada tahap 9 didapatkan  $c(w)$  yang tidak sesuai dengan angka ke- $n$  *secret code*, lanjut ke tahap 11. Jika sesuai, ulangi tahap 9.
11. Cari sinonim kata tersebut pada kamus sinonim. Kata yang diambil hanya kata dengan nilai  $c(w)$  yang berbeda.
12. Dari kata-kata yang didapatkan dari tahap 11, akan dilakukan pengambilan satu kata secara acak.
13. Kata yang telah diambil dari tahap 12, akan menggantikan kata yang tidak sesuai sebelumnya.
14. Kembali ke tahap 9.

### 3.1.5 *Extracting Information*

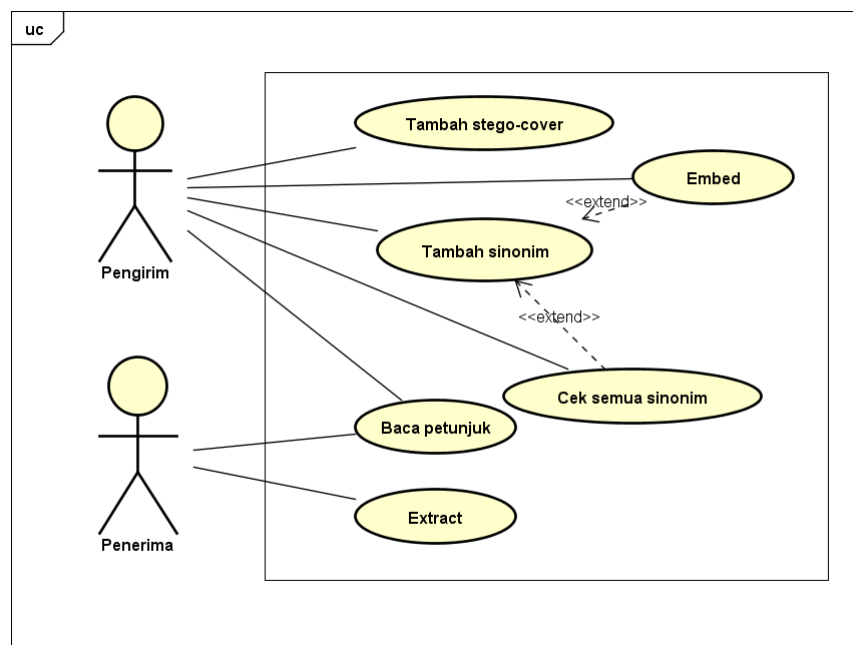
Pada proses pengestraksian kembali pesan rahasia akan dibutuhkan input berupa *stego-object* yang merupakan hasil dari *embedding*. Perangkat lunak pertama-tama akan membaca secara keseluruhan inputnya, lalu melakukan pemenggalan kata sama seperti yang dilakukan pada saat proses *embedding*. Selesai melakukan pemenggalan kata, dari tiap kata akan didapatkan *sum*nya. Saat ini telah didapatkan deretan angka 0 dan 1 yang dinamakan *secret code*. Saat proses *embedding*, memang 1 karakter diubah menjadi kode ASCII 7-bit, namun *java* menyediakan fungsi untuk mengubah deret biner 8-bit menjadi karakter. Sehingga untuk setiap 7-bit pada *secret code* dapat ditambahkan angka 0 di paling kiri sebelum dijadikan *input* pada fungsi *java* tersebut.

Setelah selesai mengubah bit-bit yang ada menjadi karakter, hasilnya akan ditampilkan pada layar. Proses ekstraksi ini melibatkan semua kata yang ada pada *stego-object*, sehingga kata yang tidak disisipkan pesan rahasia pun ikut terekstraksi. Namun karena saat proses *embedding* telah ditambahkan kode ASCII dari karakter '#', maka *secret message* berakhir saat ditemukan karakter '#'. *Data Flow Diagram Extract* dapat dilihat pada Gambar 3.6.

Gambar 3.6: *Data Flow Diagram Extract*

## 3.2 Analisis Use Case

Diagram *use case* perangkat lunak steganografi ini memiliki dua aktor, yaitu pengirim dan penerima. Pengirim berarti aktor yang akan melakukan *embedding* dan mengirimkan *stego-object*, sedangkan penerima berarti aktor yang menerima *stego-object* dan melakukan *extract information* untuk mendapatkan *secret message*. Diagram *use case* dari perangkat lunak yang akan dibangun dapat dilihat pada Gambar 3.7.



Gambar 3.7: Use Case Diagram Perangkat Lunak Steganografi

Dari diagram dapat dilihat dua *use case*, yakni:

1. **Embed**, pengirim mengetikkan pesan rahasia yang akan disembunyikan.
2. **Tambah stego-cover**, pengirim menyalin teks sebagai *stego-cover*. Jika pengirim menambahkan *stego-cover*, maka pengirim wajib melengkapi sinonim yang belum terdaftar agar *stego-cover* dapat digunakan.
3. **Tambah sinonim**, pengirim mengetikkan kata beserta dengan sinonimnya pada masing-masing kotak yang telah disediakan.
4. **Cek semua sinonim**, pengirim dapat memeriksa kata apa saja yang belum memiliki sinonim dengan menekan tombol yang telah disediakan.

5. **Extract**, penerima akan memasukkan *stego-object* yang diterima.
6. **Baca petunjuk**, pengirim dan penerima dapat membaca petunjuk terkait cara menggunakan perangkat lunak ini.

### 3.2.1 Skenario Use Case

#### 1. *Embed*

- Nama: *Embed*
  - Aktor: Pengirim
  - Prakondisi: Pengirim mengetahui pesan rahasia yang akan disembunyikan
  - Tujuan: Menghasilkan *stego-object*.
  - Skenario:
    1. Pengirim mengetikkan pesan rahasia yang akan disembunyikan.
    2. Pengirim menekan tombol *Embed*.
    3. Sistem lalu akan mengubah pesan yang diketik menjadi kode ASCII.
    4. Sistem mencari *stego-cover* yang dapat menampung pesan yang sudah dalam bentuk ASCII.
    5. Sistem melakukan proses *embedding* dengan bantuan kamus sinonim.
    6. *Stego-object* sebagai hasil dari proses *embedding* akan muncul pada kotak hasil.
- Alternatif:
- 4.1 Tidak ada *stego-cover* yang dapat menampung pesan.
  - 4.2 Sistem mengeluarkan peringatan bahwa pesan rahasia melebihi kapasitas.
  - 5.1 Proses *embedding* gagal karena ada sinonim yang tidak ditemukan.
  - 5.2 Sistem mengembalikan daftar kata yang tidak terdaftar sinonimnya.
  - 5.3 Pengguna harus menambahkan sinonim kata yang belum terdaftar.

#### 2. Tambah *stego-cover*

- Nama: Tambah *stego-cover*
- Aktor: Pengirim
- Prakondisi: Telah memiliki teks yang memenuhi syarat untuk dijadikan *stego-cover*. Pengirim telah menuliskan judul untuk file tersebut.
- Tujuan: Menambahkan variasi *stego-cover*.
- Skenario:
  1. Pengirim mengetikkan judul untuk *stego-cover* yang baru.
  2. Pengirim menyalin teks yang akan dijadikan *stego-cover*.
  3. Pengirim menekan tombol Tambahkan Cover.
  4. Sistem akan membuat file dengan nama *file* yang dimasukkan pengirim, dan menuliskan *stego-cover* dalam *file* tersebut.

5. Sistem juga menuliskan nama *file* beserta kapasitas *stego-cover* pada file lain yang merupakan daftar *stego-cover* yang ada.
6. Akan tampil peringatan jika *stego-cover* berhasil / gagal ditambahkan.

### 3. Tambah sinonim

- Nama: Tambah sinonim
- Aktor: Pengirim
- Prakondisi: -
- Tujuan: Menambahkan daftar kata dan sinonimnya.
- Skenario:
  1. Pengirim mengetikkan kata yang ada pada *stego-cover*, yang akan ditambahkan sinonimnya.
  2. Pengirim mengetikkan sinonim dari kata pada tahap 1.
  3. Sistem akan menghitung nilai  $c(w)$  kata pada tahap 1 dan  $c(w)$  sinonim pada tahap 2.
  4. Jika nilai  $c(w)$  sama, maka akan ditampilkan hasil pemenggalan kata dari sinonim yang diketikkan.
  5. Jika nilai  $c(w)$  berbeda, maka sistem akan menambahkan kata beserta sinonimnya pada kamus sinonim.
  6. Akan tampil peringatan jika sinonim berhasil atau gagal ditambahkan.

### 4. Cek semua sinonim

- Nama: Cek semua sinonim
- Aktor: Pengirim
- Prakondisi: -
- Tujuan: Menginformasikan kata mana saja yang belum memiliki sinonim.
- Skenario:
  1. Pengirim menekan tombol Periksa semua sinonim.
  2. Sistem akan memeriksa semua kata yang ada pada semua *stego-cover* untuk dicari sinonimnya pada kamus sinonim.
  3. Kata yang belum memiliki sinonim akan ditampilkan.  
Alternatif:
    - 3.1 Pengirim harus menambahkan sinonim untuk kata yang ditampilkan.

### 5. *Extract*

- Nama: *Extract*
- Aktor: Penerima
- Prakondisi: Telah memiliki *stego-object* yang dihasilkan dari perangkat lunak yang sama.
- Tujuan: Mengkestraksi pesan rahasia dari *stego-object*.

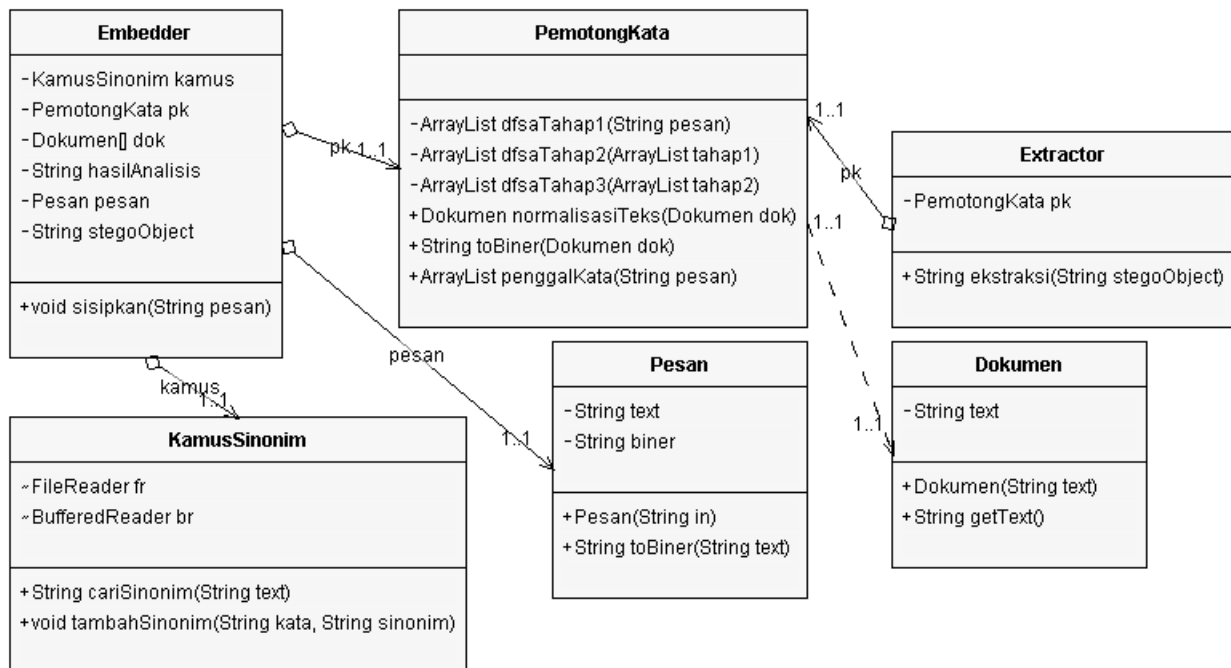
- Skenario:
  1. Penerima menyalin *stego-object* yang didapat dari perangkat yang sama.
  2. Penerima menekan tombol *Extract*.
  3. Sistem akan mengeksekusi fungsi ekstraksi.
  4. *Secret message* yang merupakan hasil ekstraksi akan ditampilkan.
  5. Karakter '#' menandakan akhir dari *secret message*.

## 6. Baca petunjuk

- Nama: Baca petunjuk
- Aktor: Pengirim atau penerima
- Prakondisi: -
- Tujuan: Memberi pemahaman kepada pengirim dan penerima mengenai cara menggunakan perangkat lunak ini.
- Skenario:
  1. Pengirim atau penerima membaca petunjuk tentang *embedding*.
  2. Pengirim atau penerima membaca petunjuk tentang *extract*.
  3. Pengirim atau penerima membaca petunjuk tentang *cover (stego-cover)*.
  4. Pengirim atau penerima membaca petunjuk tentang sinonim.

### 3.2.2 Analisis Diagram Kelas

Diagram kelas perangkat lunak steganografi dapat dilihat pada Gambar 3.8



Gambar 3.8: Class Diagram perangkat lunak steganografi

Keterangan atas diagram kelas akan dijelaskan sebagai berikut:



### 1. Kelas *Embedder*

- Atribut kamus, untuk mencari sinonim.
- Atribut pk, untuk melakukan pemenggalan kata.
- Atribut dok, untuk menyimpan kumpulan *stego-cover*.
- Atribut hasilAnalisis, untuk menyimpan hasil dari penyukuan kata.
- Atribut pesan, untuk menyimpan pesan rahasia yang akan disembunyikan.
- Atribut stegoObject, untuk menyimpan *stego-object* yang akan diektrak.
- Fungsi sisipkan, memiliki parameter pesan, untuk melakukan proses *embedding*.

### 2. Kelas *KamusSinonim*

- Atribut fr, untuk membuka *file*.
- Atribut br, untuk membaca file yang dibuka fr.
- Fungsi cariSinonim, memiliki parameter text, untuk mencari sinonim dari parameter.
- Fungsi tambahSinonim, memiliki parameter kata dan sinonim, untuk menambah sinonim baru.

### 3. Kelas *PemotongKata*

- Fungsi dfsaTahap1, memiliki parameter pesan, mengembalikan keluaran berupa hasil proses dfsa tahap pertama.
- Fungsi dfsaTahap2, memiliki parameter tahap1 yang merupakan keluaran dfsaTahap1 dan mengembalikan hasil proses dfsa tahap kedua.
- Fungsi dfsaTahap3, memiliki parameter tahap2 yang merupakan keluaran dfsaTahap2 dan mengembalikan hasil penyukuan kata akhir.
- Fungsi normalisasiText, memiliki parameter dokumen dan menghasilkan keluaran berupa dokumen tanpa tanda baca dan simbol, sehingga siap untuk dipotong berdasarkan suku katanya
- Fungsi toBiner, memiliki parameter dokumen. Parameter dokumen merupakan dokumen yang telah dilakukan penyukuan, sehingga fungsi ini hanya menghitung jumlah suku kata untuk setiap katanya. Jumlah kata akan disandikan dengan 0 untuk jumlah genap dan 1 untuk jumlah ganjil.
- Fungsi penggalKata, memiliki parameter pesan, melakukan pemenggalan kata dengan menggunakan fungsi dfsaTahap1, dfsaTahap2, dan dfsaTahap3.

### 4. Kelas *Pesan*

- Atribut *text*, untuk menampung teks dari pesan berupa *string*.
- *Constructor* Pesan, memiliki parameter bertipe *string*, parameter akan mengisi atribut teks.
- Fungsi toBiner, memiliki parameter *text*. Fungsi ini akan mengembalikan bentuk biner dari isi atribut *text*.

### 5. Kelas *Extractor*

- Atribut *pk*, untuk melakukan pemenggalan kata.
- Fungsi ekstraksi, memiliki parameter *stegoObject*, untuk melakukan proses *extract*.

### 6. Kelas *Dokumen*

- Atribut *text*, untuk menampung teks dari dokumen berupa *string*.
- *Constructor* *Dokumen*, memiliki parameter bertipe *string*, parameter akan mengisi atribut teks.
- Fungsi *getText*, merupakan *getter* untuk atribut *text*.
- Fungsi *setText*, memiliki parameter *text*, merupakan *setter* untuk atribut *text*.

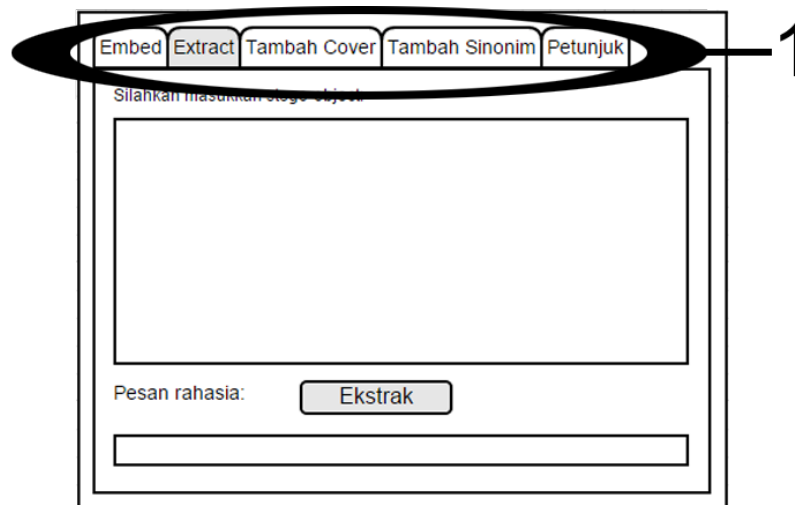
## BAB 4

### PERANCANGAN

Bab ini akan membahas mengenai perancangan aplikasi. Perancangan aplikasi akan meliputi tampilan antarmuka, diagram kelas lengkap beserta dengan deskripsi dan fungsinya.

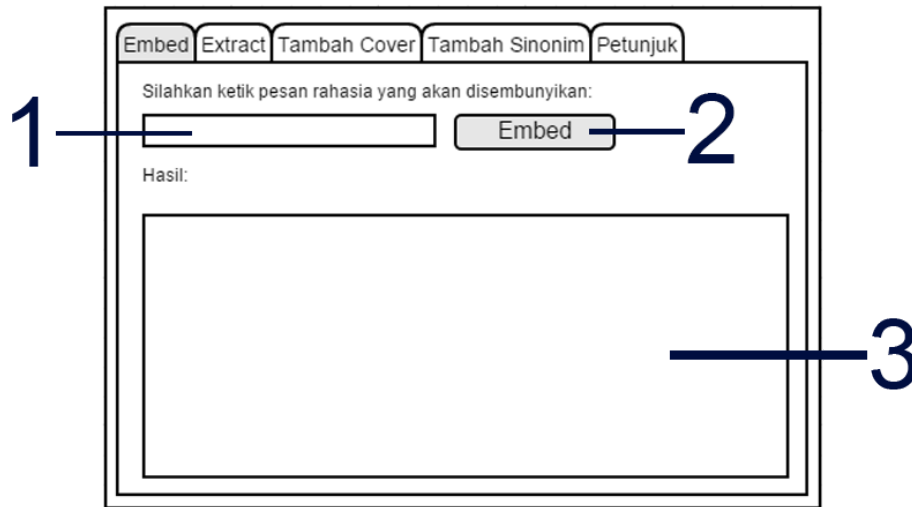
#### 4.1 Perancangan Antarmuka

Agar pengguna dapat menggunakan perangkat lunak ini dengan nyaman, maka dibutuhkan antarmuka. Rancangan antarmuka yang dibuat terdiri dari satu *frame* dan di dalamnya terdapat beberapa *tab*, yaitu *tab embed*, *tab extract*, *tab* tambah *cover*, *tab* tambah sinonim, dan *tab* petunjuk seperti yang dapat dilihat pada Gambar 4.1(1).



Gambar 4.1: Tampilan *tab*

#### 4.1.1 Tab Embed

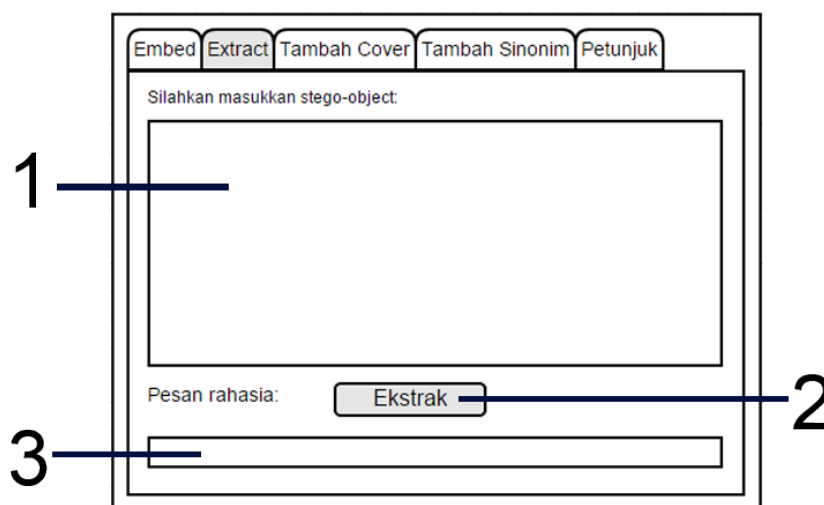


Gambar 4.2: Tampilan *tab Embed*

Pada *tab Embed* (dapat dilihat pada Gambar 4.2) terdapat beberapa elemen sebagai berikut.

1. **Text box secret message**, pengguna dapat mengetikkan pesan rahasia yang akan disembunyikan pada bagian ini.
2. **Tombol Embed**, pengguna dapat menekan tombol ini untuk melakukan proses *embedding*.
3. **Text area stego-object**, hasil proses *embedding* yang merupakan *stego-object* akan muncul pada bagian ini.

#### 4.1.2 Tab Extract

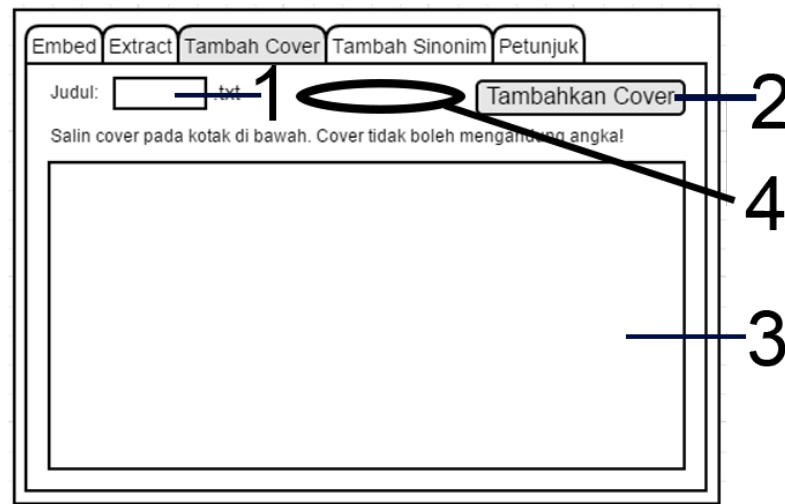


Gambar 4.3: Tampilan *tab Extract*

Pada *tab Extract* (dapat dilihat pada Gambar 4.3) terdapat beberapa elemen sebagai berikut.

1. *Text area stego-object*, pengguna dapat memasukkan *stego-object* yang diterima pada bagian ini.
2. **Tombol Ekstrak**, pengguna dapat menekan tombol ini untuk melakukan proses *extracting*.
3. *Text box secret message*, hasil *extracting* akan ditampilkan pada bagian ini.

#### 4.1.3 Tab Tambah Cover

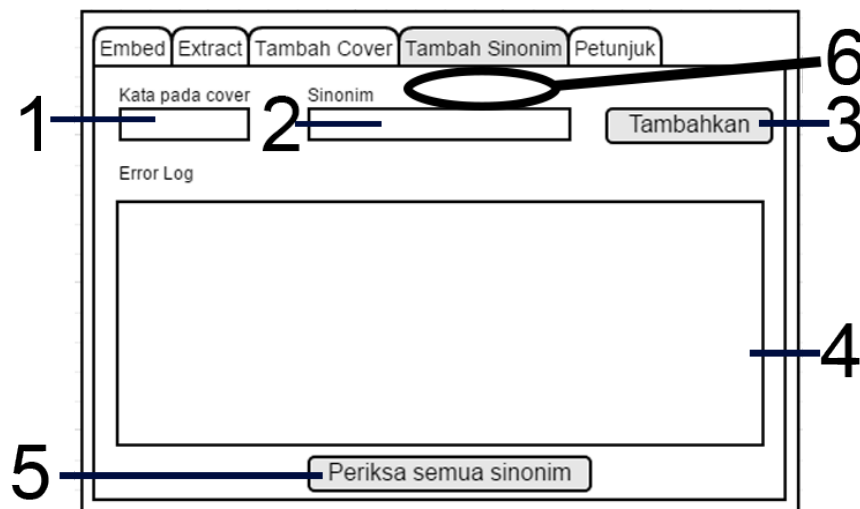


Gambar 4.4: Tampilan *tab* Tambah Cover

Pada *tab* Tambah Cover (dapat dilihat pada Gambar 4.4) terdapat beberapa elemen sebagai berikut.

1. *Text box judul*, pengguna dapat mengetikkan judul dari *stego-cover* yang akan ditambahkan pada bagian ini.
2. **Tombol Tambah Cover**, pengguna dapat menekan tombol ini untuk menambahkan *stego-cover* yang telah disalin pada no 3.
3. *Text area stego-cover*, pengguna dapat menyalin *stego-cover* yang memenuhi persyaratan di bagian ini.
4. **Label peringatan**, peringatan akan muncul jika pengguna belum memasukkan judul *file*, judul yang diketik telah terdaftar, atau saat *stego-cover* berhasil ditambahkan.

#### 4.1.4 Tab Tambah Sinonim

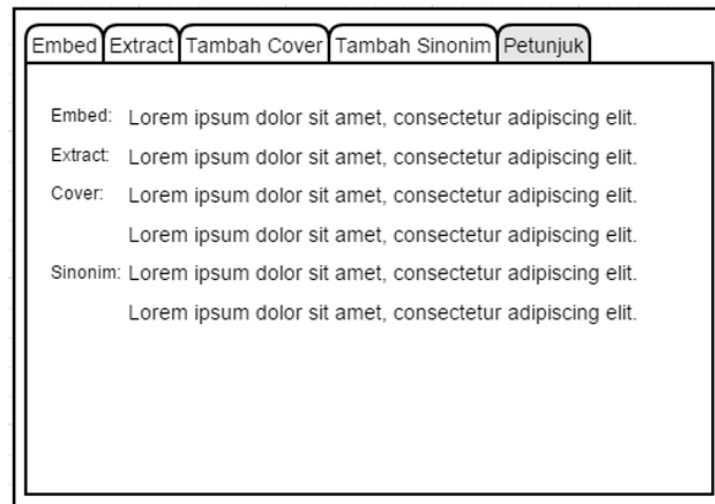


Gambar 4.5: Tampilan *tab* Tambah Sinonim

Pada *tab* Tambah Sinonim (dapat dilihat pada Gambar 4.5) terdapat beberapa elemen sebagai berikut.

1. **Text box kata pada cover**, kata pada *cover* berarti kata yang ada pada *stego-cover*, pengguna dapat mengetikkannya pada bagian ini.
2. **Text box sinonim**, pengguna dapat mengetik sinonimnya pada bagian ini.
3. **Tombol Tambahkan**, pengguna dapat menekan tombol ini untuk memasukkan pasangan kata dan sinonim yang baru ke *file* kamus.
4. **Text Area Error Log**, akan menampilkan daftar kata yang belum memiliki sinonim jika pengguna menekan tombol Periksa semua sinonim.
5. **Tombol Periksa semua sinonim**, akan memeriksa semua kata dari tiap *stego-cover* yang terdaftar dan menampilkan daftar kata yang belum memiliki sinonim.
6. **Label peringatan**, akan memberikan keterangan jika sinonim gagal atau berhasil ditambahkan.

### 4.1.5 *Tab* Petunjuk



Gambar 4.6: Tampilan *tab* Petunjuk

*Tab* Petunjuk (dapat dilihat pada Gambar 4.6) sebenarnya hanya dibuat untuk membantu pengguna yang baru pertama kali memakai perangkat lunak ini. Pada *tab* ini berisi petunjuk-petunjuk yang dapat membantu pengguna untuk mengoperasikan perangkat lunak ini.

## 4.2 Diagram Kelas Lengkap

Diagram kelas sebelumnya yang dapat dilihat pada Gambar 3.8 mengalami beberapa perubahan dan penambahan kelas. Diagram kelas lengkap dapat dilihat pada Gambar 4.7.

Deskripsi setiap kelas beserta dengan atribut dan fungsinya akan dijelaskan sebagai berikut.

### 1. Kelas PemotongKata

Kelas ini merupakan kelas yang merepresentasikan pemotong kata. Fungsi-fungsi yang ada pada kelas ini adalah:

- **public String getPattern(String input)**  
 Berfungsi untuk mendapatkan pola suku kata.  
**Parameter:**
  - **input** Teks yang akan dicari pola suku katanya.
- **public ArrayList generateLevel1(String text)**  
 Berfungsi untuk melakukan DFSA tahap 1.  
**Parameter:**
  - **text** Teks yang akan didapatkan suku katanya.
- **public ArrayList generateLevel2(ArrayList lv1)**  
 Berfungsi untuk melakukan DFSA tahap 2.  
**Parameter:**
  - **lv1** Merupakan kembalian dari fungsi generateLevel1().

- **public ArrayList generateLevel3(ArrayList lv2)**

Berfungsi untuk melakukan DFSA tahap 3.

**Parameter:**

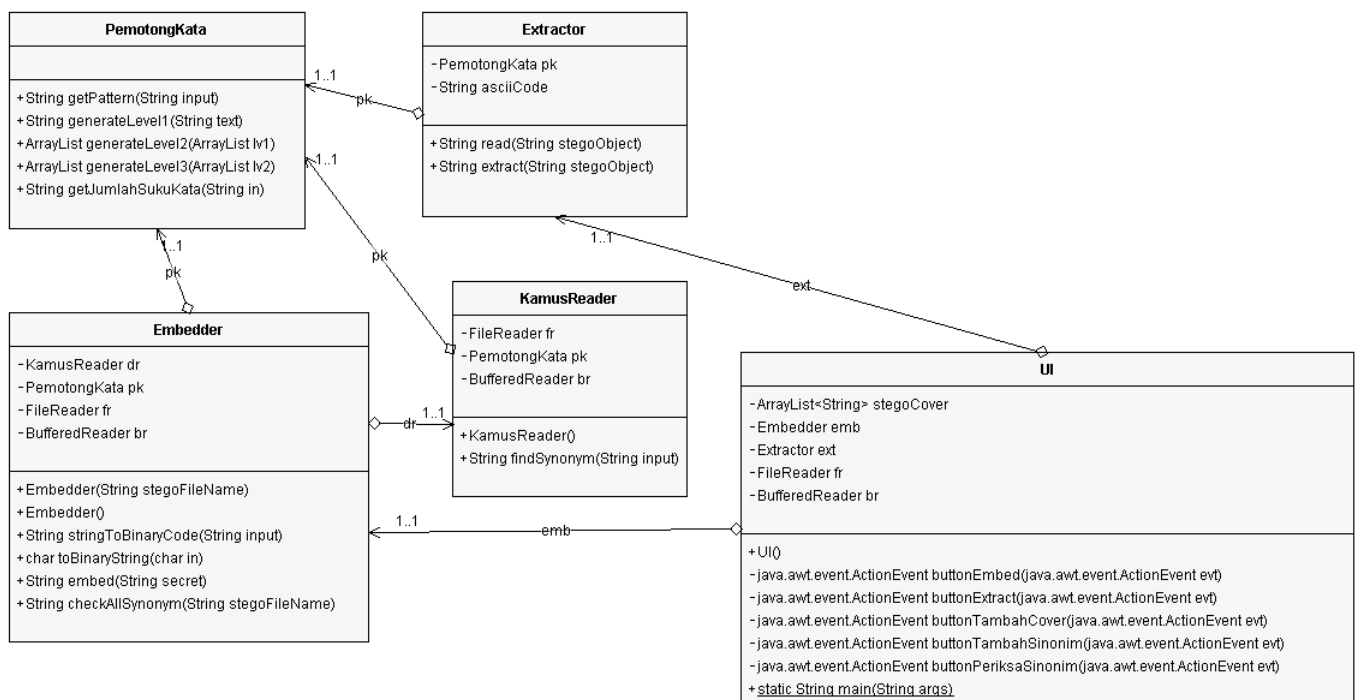
- **lv2** Merupakan kembalian dari fungsi generateLevel2().

- **public int getJumlahSukuKata(String in)**

Berfungsi untuk menghitung banyaknya suku kata.

**Parameter:**

- **in** Merupakan teks yang akan dicari banyak suku katanya.



Gambar 4.7: Diagram kelas lengkap

## 2. Kelas KamusReader

Kelas ini berfungsi untuk membaca *file* kamus dengan tujuan mencari sinonim. Atribut yang ada pada kelas KamusReader adalah:

- **FileReader fr:** Objek FileReader yang berfungsi untuk membuka suatu *file*.
- **BufferedReader br:** Objek BufferedReader yang berfungsi untuk membaca isi dari *file* yang dibuka FileReader.
- **PemotongKata pk:** Objek PemotongKata yang akan berfungsi untuk memotong kata.

Fungsi-fungsi yang dimiliki kelas ini adalah:

- **KamusReader()**

Merupakan *constructor* dari kelas KamusReader yang akan menginisialisasi atribut-atributnya.



- **String findSynonym(String input)**

Berfungsi untuk mencari sinonim dari parameter.

**Parameter:**

- **input** Kata yang akan dicari sinonimnya.

### 3. Kelas Embedder

Kelas ini berfungsi untuk menangani seluruh kegiatan *embedding* pesan ke dalam *stego-cover*.

Atribut yang ada pada kelas Embedder adalah:

- **KamusReader dr:** Objek KamusReader berfungsi untuk membaca *file* kamus.
- **PemotongKata pk:** Objek PemotongKata berfungsi untuk memotong kata.
- **FileReader fr:** Objek FileReader yang berfungsi untuk membuka suatu *file*.
- **BufferedReader br:** Objek BufferedReader yang berfungsi untuk membaca isi dari *file* yang dibuka FileReader.

Fungsi-fungsi yang dimiliki kelas ini adalah:

- **Embedder(String stegoFileName)**

Merupakan *constructor* dari kelas Embedder dengan parameter nama *stego-cover* yang akan langsung dibuka oleh FileReader.

**Parameter:**

- **stegoFileName** Nama *file stego-cover* yang akan dibuka oleh FileReader.

- **Embedder()**

Merupakan *constructor* tanpa parameter dari kelas Embedder yang akan menginisialisasi FileReader dan BufferedReader.

- **String stringToBinaryCode(String input)**

Berfungsi untuk mengubah setiap karakter dalam string menjadi biner.

**Parameter:**

- **input** String yang akan diubah menjadi biner.

- **String toBinaryString(char in)**

Berfungsi untuk mendapatkan kode ASCII dari karakter.

**Parameter:**

- **in** Karakter yang akan didapatkan kode ASCII-nya.

- **String embed(String secret)**

Berfungsi untuk melakukan proses *embedding*.

**Parameter:**

- **secret** Pesan rahasia yang akan disembunyikan.

- **String checkAllSynonym(String stegoFileName)**

Berfungsi untuk cek ketersediaan sinonim *stego-cover* pada *file* kamus.

**Parameter:**

- **stegoFileName** Nama *file stego-cover* yang akan dicek.

#### 4. Kelas Extractor

Kelas ini berfungsi untuk menangani seluruh kegiatan *extracting* pesan dari *stego-object*. Atribut yang ada pada kelas Extractor adalah:

- **PemotongKata pk:** Objek PemotongKata berfungsi untuk memotong kata.
- **String asciiCode:** Untuk menyimpan kode yang siap diubah menjadi karakter ASCII.

Fungsi-fungsi yang dimiliki kelas ini adalah:

- **void read(String stegoObject)**

Berfungsi untuk mengubah *stego-object* menjadi kode yang siap diubah menjadi karakter ASCII.

**Paramter:**

- **stegoObject** *Stego-object* yang akan didapatkan kodenya.

- **String extract(String stegoObject)**

Berfungsi untuk melakukan proses *extracting*

**Parameter:**

- **stegoObject** *Stego-object* yang akan diekstraksi.

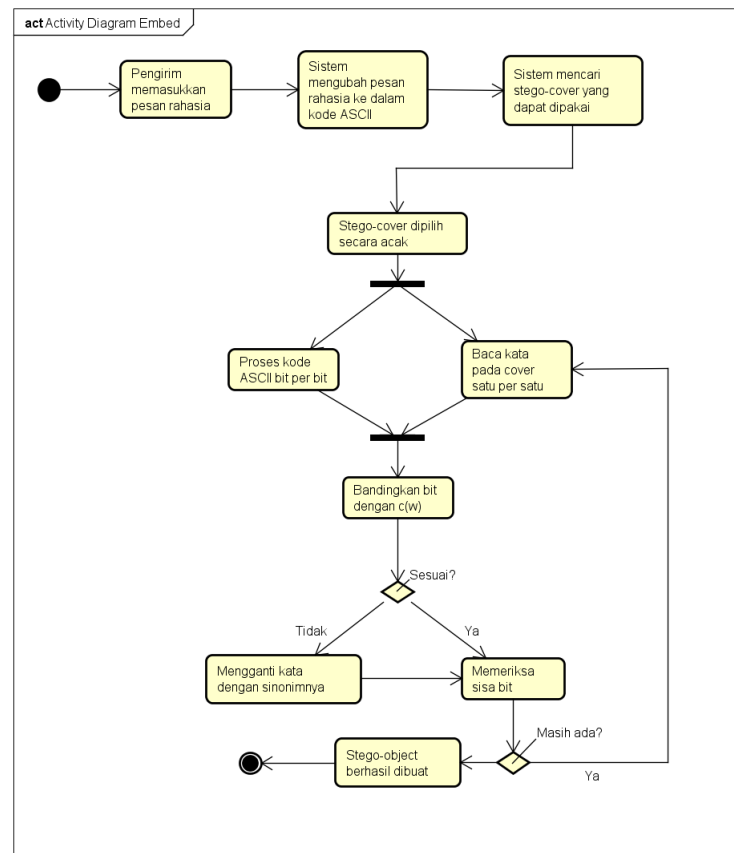
### 4.3 Analisis Diagram Aktivitas

Diagram aktivitas dari perangkat lunak dibagi menjadi dua. Diagram aktivitas saat menyisipkan pesan dapat dilihat pada [4.8](#)

Dari diagram aktivitas dapat dijelaskan sebagai berikut:

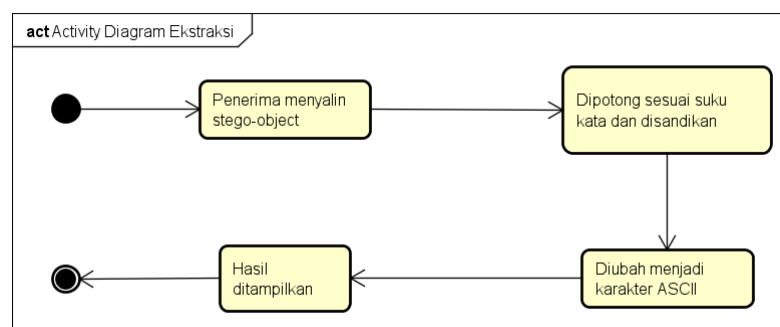
1. Pengguna memasukkan input pesan rahasia berupa string
2. Pesan akan diubah ke bentuk ASCII
3. Sistem akan mencari *stego-cover* yang dapat dipakai (kapasitas mencukupi)
4. Dari beberapa *stego-cover* yang dapat dipakai, akan dipilih satu secara acak
5. Sistem akan membaca satu per satu kata yang ada pada *stego-cover*
6. Kode ASCII dari pesan rahasia akan dibaca bit per bit
7. Bandingkan apakah  $c(w)$  dan bit ASCII yang sedang dibaca sama-sama bernilai genap atau ganjil. Dari sini akan dihasilkan dua kemungkinan:
  - Jika keduanya sesuai, ada dua kemungkinan lagi:
    - Jika masih ada bit yang belum diperiksa, maka akan kembali ke tahap 5 dan 6.
    - Jika semua kode telah diperiksa, maka *stego-cover* berhasil dibuat dan proses *embedding* selesai.
  - Jika tidak sesuai, maka sistem akan mengganti kata tersebut dengan sinonimnya. Setelah itu ada dua kemungkinan lagi:

- Jika masih ada bit yang belum diperiksa, maka akan kembali ke tahap 5 dan 6.
- Jika semua kode telah diperiksa, maka *stego-cover* berhasil dibuat dan proses *embedding* selesai.



Gambar 4.8: Activity Diagram perangkat lunak steganografi saat menyisipkan

Diagram aktivitas ekstraksi dapat dilihat pada Gambar 4.9



Gambar 4.9: Activity Diagram perangkat lunak steganografi saat proses ekstraksi

Dari diagram aktivitas di atas dapat dijelaskan sebagai berikut:

1. Pengguna akan memberikan input berupa *stego-object* dari program yang sama
2. Sistem lalu akan menyukukan kata-kata yang ada. Setelah itu dilakukan penyandian, angka 0 untuk kata dengan jumlah suku kata genap dan angka 1 untuk jumlah ganjil

3. Selesai melakukan penyandian, hasilnya akan diubah menjadi karakter ASCII yang sesuai
4. Kini sudah didapatkan pesan dengan karakter ASCII, hasil akan ditampilkan pada layar

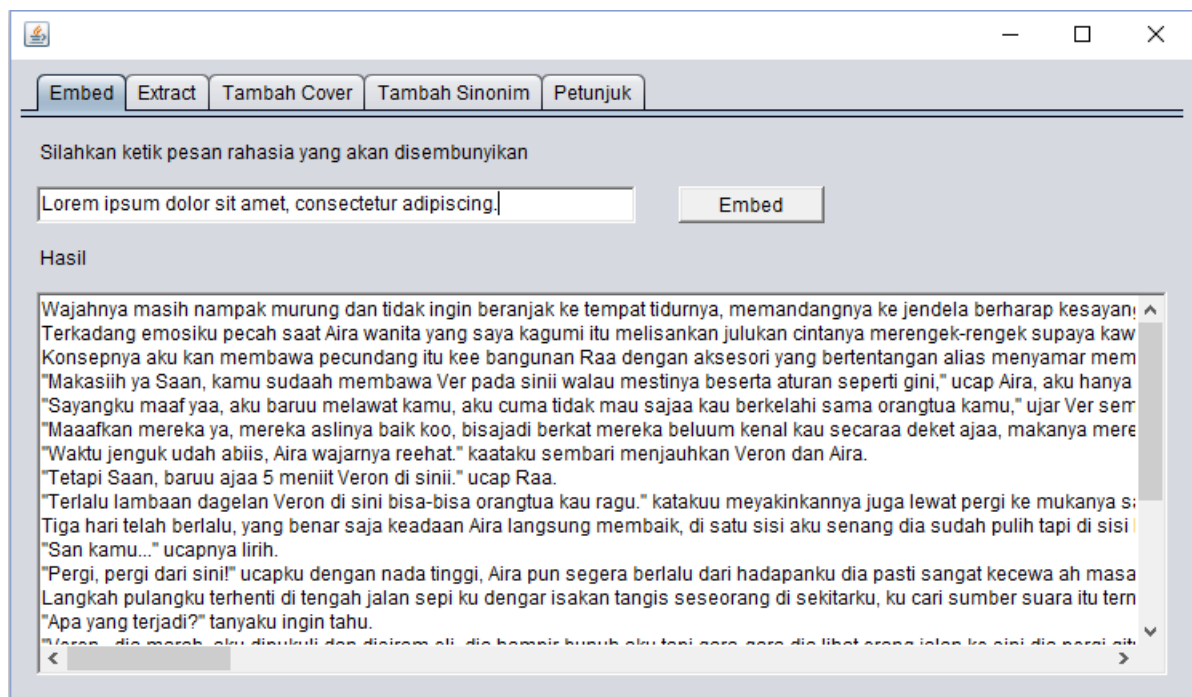
## BAB 5

### IMPLEMENTASI DAN PENGUJIAN

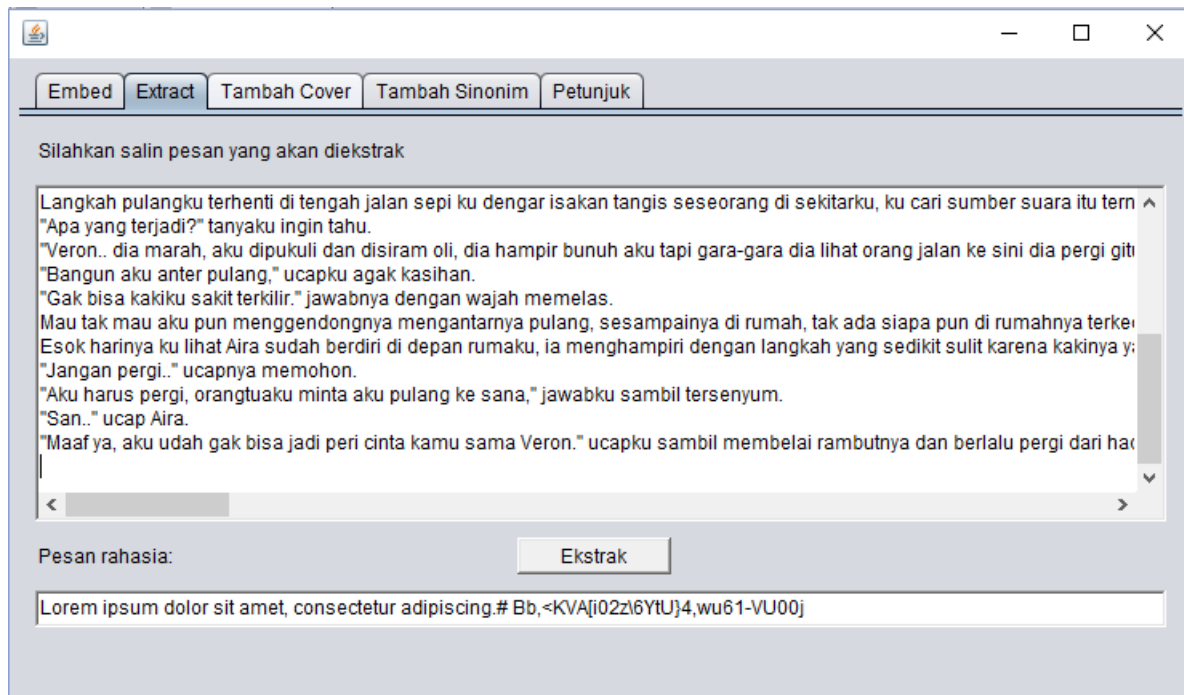
Bab ini akan berisi tentang implementasi perangkat lunak serta pengujian pada perangkat lunak tersebut. Hasil dari pengujian akan digunakan untuk menarik kesimpulan.

#### 5.1 Implementasi

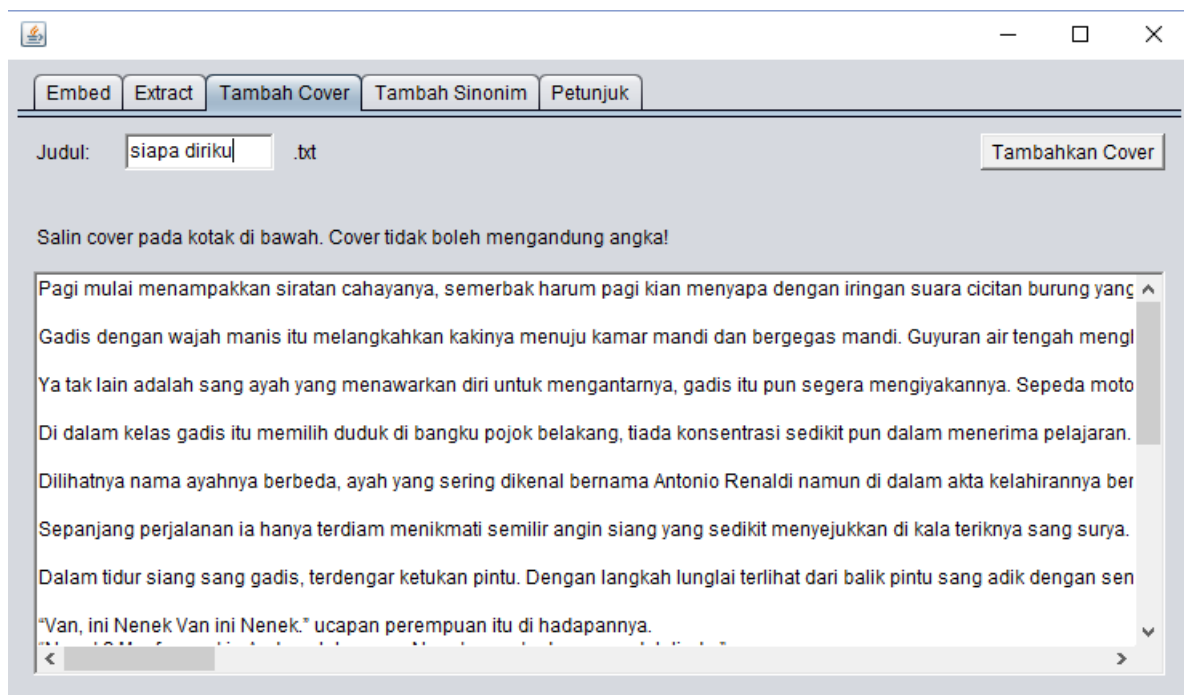
Perangkat lunak akan diimplementasikan menjadi sebuah program dengan menggunakan bahasa Java. Rancangan antarmuka yang telah dijelaskan pada Bab 4.1, telah berhasil diimplementasikan dan dapat dilihat pada gambar-gambar di bawah ini.

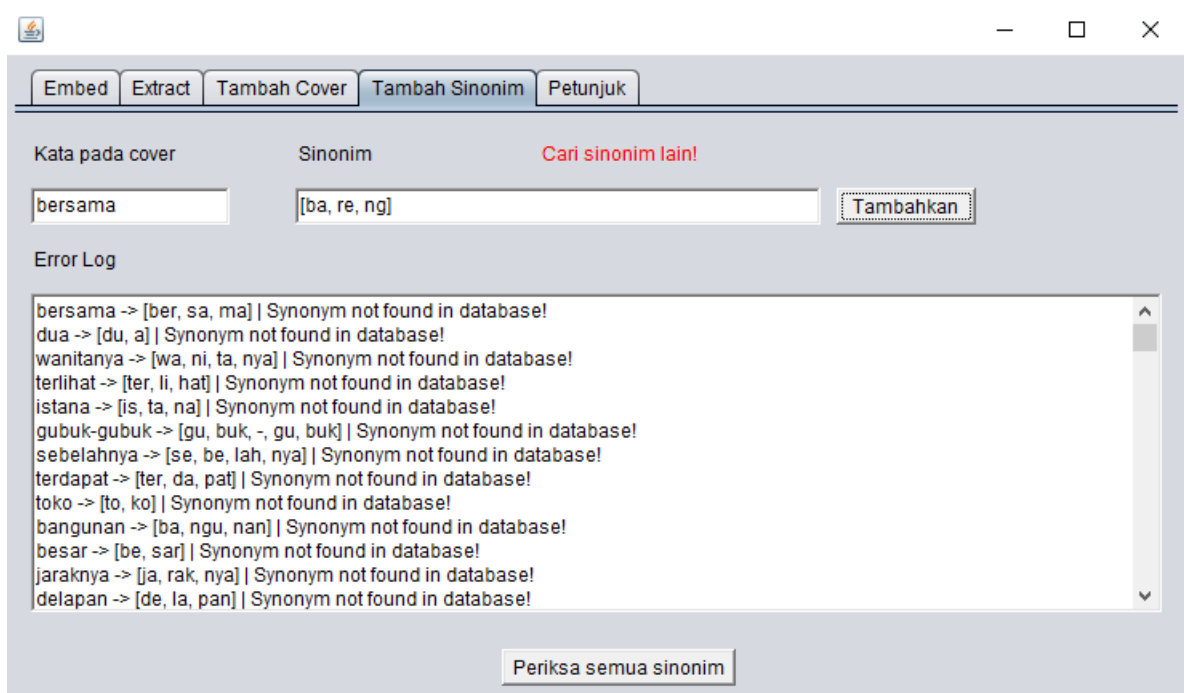


Gambar 5.1: Tampilan *tab Embed*

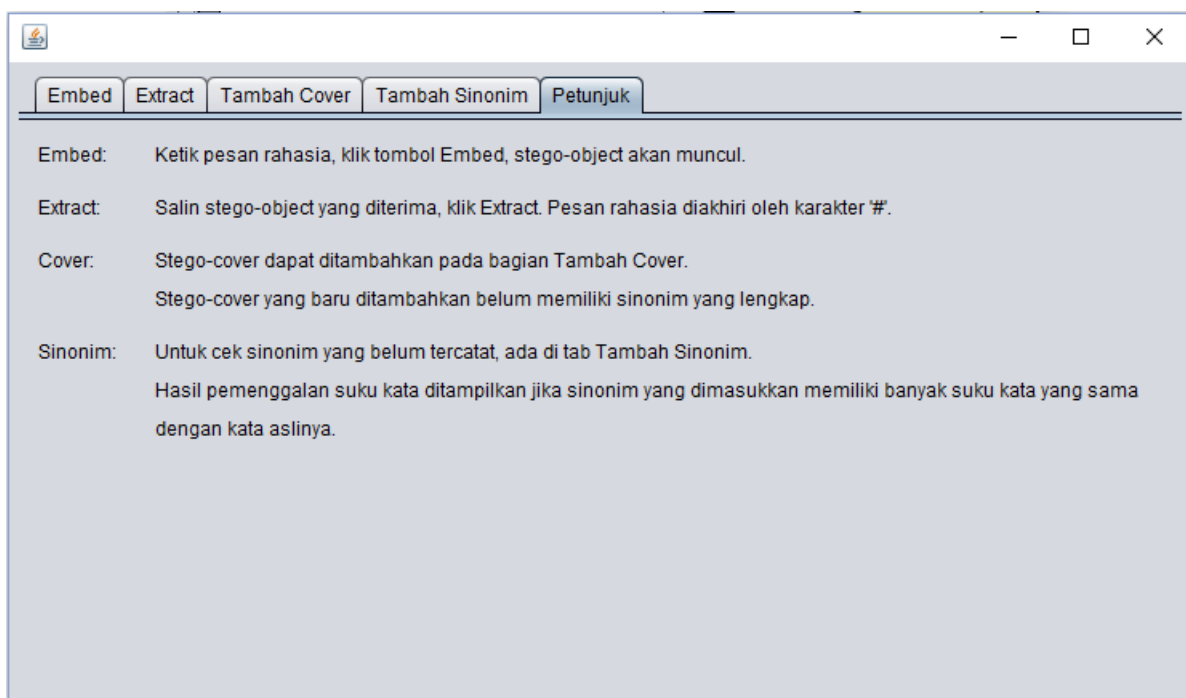
Gambar 5.2: Tampilan *tab Extract*

Gambar 5.2 menunjukkan bahwa ada karakter '#' yang memberitahu penerima bahwa pesan rahasia telah berakhir, seperti yang telah dijelaskan pada Bab 3.1.4.

Gambar 5.3: Tampilan *tab Tambah Cover*

Gambar 5.4: Tampilan *tab Tambah Sinonim*

Gambar 5.4 merupakan contoh saat pengirim memasukkan pasangan kata dan sinonim yang memiliki  $c(w)$  yang memiliki nilai yang sama. Seperti yang telah dijelaskan pada Bab 4.1, jika penambahan sinonim gagal, *textbox* sinonim akan diisi dengan hasil pemenggalan kata oleh perangkat lunak. Hal ini dilakukan untuk memberitahu pengirim bagaimana perangkat lunak memenggal kata tersebut. Pemenggalan kata yang dilakukan oleh perangkat lunak tidak sempurna, namun hal ini tidak menjadi masalah karena yang terpenting pemenggalan katanya konsisten.

Gambar 5.5: Tampilan *tab Petunjuk*

## 5.2 Pengujian

Pengujian yang dilakukan adalah pengujian fungsional dan pengujian eksperimental. Pengujian ini dilakukan untuk memeriksa apakah perangkat lunak yang dibuat telah berfungsi dengan baik. Hasil pengujian ini juga akan dipakai untuk menarik kesimpulan pada bab terakhir.

### 5.2.1 Pengujian Fungsional

Pengujian fungsional dilakukan untuk memeriksa apakah perangkat lunak memberikan reaksi yang seharusnya terhadap aksi dari pengguna perangkat lunak. Pengujian ini dilakukan pada sistem operasi Windows. Terdapat beberapa tes kasus yang diujikan, untuk lengkapnya dapat dilihat pada Tabel 5.2.1.

Tabel 5.1: Tabel Pengujian Fungsional

| No. | Aksi  | Reaksi seharusnya  | Hasil  |
|-----|---|--|--------|
| 1   | Pengguna mengetikkan pesan rahasia lalu menekan tombol Embed        | Perangkat lunak menampilkan <i>stego-cover</i>   | Sesuai |
| 2   | Pengguna memasukkan <i>stego-object</i> lalu menekan tombol Extract | Perangkat lunak menampilkan pesan rahasia, karakter '#' menandakan pesan rahasia telah berakhir            | Sesuai |
| 3   | Pengguna menambahkan <i>stego-cover</i>                             | Perangkat lunak membuat <i>file stego-cover</i> yang baru dan mendaftarkannya pada <i>stego-cover list</i> | Sesuai |
| 4   | Pengguna menambahkan sinonim  | Perangkat lunak menambahkan sinonim ke kamus sinonim   | Sesuai |
| 5   | Pengguna memeriksa semua sinonim                                    | Perangkat lunak menampilkan kata-kata yang sinonimnya tidak terdaftar pada kamus sinonim                   | Sesuai |

### 5.2.2 Pengujian Eksperimental

Pengujian eksperimental ini dilakukan untuk menguji respon perangkat lunak terhadap beberapa kemungkinan yang dilakukan pengguna pada perangkat lunak.



Tabel 5.2: Tabel Pengujian Eksperimental

| No. | Aksi  | Reaksi   |
|-----|---|--|
| 1   | Pengguna memasukkan pesan rahasia dengan ukuran yang tidak bisa ditampung <i>stego-cover</i>        | Perangkat lunak tidak mengeksekusi proses <i>embed</i> . Muncul peringatan bahwa pesan rahasia melebihi kapasitas yang ada |
| 2   | Pengguna menekan tombol <i>embed</i> , namun <i>stego-cover</i> belum memiliki sinonim yang lengkap | Perangkat lunak menampilkan kata apa saja yang belum terdaftar sinonimnya  |
| 3   | Pengguna memasukkan teks sembarang (bukan <i>stego-object</i> )                                     | Perangkat lunak mengeksekusi proses <i>extract</i> , namun pesan yang diekstraksi tidak memiliki arti                      |
| 4   | Pengguna menambahkan <i>stego-cover</i> tanpa memasukkan nama <i>file</i>                           | Muncul peringatan bahwa nama <i>file</i> belum diisi   |
| 5   | Pengguna menambahkan <i>stego-cover</i> dengan nama yang telah ada sebelumnya                       | Muncul peringatan bahwa nama <i>file</i> telah dipakai   |
| 6   | Pengguna menambahkan sinonim dengan nilai $c(w)$ yang sama dengan katanya                           | Muncul peringatan untuk mencari sinonim lain dan ditampilkan hasil pemenggalan sinonim yang dimasukkan pengguna            |



## DAFTAR REFERENSI

- [1] Permendiknas in *EYD Terbaru (Sesuai Permendikbud No. 46 Tahun 2009)*, JENDELA PUBLISHING, 2009.
- [2] J. Pustejovsky and A. Stubbs, *Natural Language Annotation for Machine Learning*. O'Reilly Media, 2012.
- [3] G. Kowalski, *Information Retrieval Architecture and Algorithms*. Computer science, Springer US, 2010.
- [4] F. Sumarlin in *Sistem Text-To-Speech Untuk Membacakan Dongeng Berbahasa Indonesia*, 2014.
- [5] H. Singh, P. K. Singh, and K. Saroha, "A survey on text based steganography," 2009.
- [6] T. A. Basuki, "Pengenal suku kata bahasa indonesia menggunakan finite-state automata," *Integral*, vol. 5, pp. 67–74, 2000.
- [7] P. Wayner, *Disappearing Cryptography: Information Hiding : Steganography & Watermarking*. The Morgan Kaufmann Series in Software Engineering and Programming Series, Morgan Kaufmann Publishers, 2009.
- [8] N. F. Johnson, "Information system and software engineering," tech. rep., George Mason University, 1995.