BAB III
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem dapat didefinisikan sebagai penguraian dari suatu sistem yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasikan permasalah-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikan. Analisis dapat juga diartikan sebagai penelitian atas sistem yang telah ada dengan tujuan untuk merancang sistem yang baru atau diperbarui. Dalam proses pembuatan suatu sistem mutlak dilakukan penelitian dan penganalisaan tentang sistem yang akan dibangun.

3.1.1 Analisis Masalah

Penggunaan telepon selular untuk membaca pesan SMS dan *Instant Messenger* pada saat mengemudi merupakan hal yang berbahaya. Hal yang sering terjadi yaitu pengemudi mengalihkan pandangannya dari jalan raya untuk membaca pesan yang ada pada telepon selularnya, sementara hal itu memungkinkan pengguna kehilangan konsentrasi yang dapat meningkatkan risiko kecelakaan.

Mengingat pentingnya masalah ini untuk diatasi, sehingga diperlukan aplikasi yang menggunakan metode untuk mempermudah pengendara kendaraan untuk mengetahui isi setiap pesan yang masuk tanpa harus berinteraksi langsung dengan telepon selularnya. Sebuah aplikasi pembaca pesan menerapkan teknologi *text to speech* yang dapat mengkonversi teks pesan menjadi ucapan guna membantu pengemudi kendaraan dalam membacakan setiap pesan yang masuk, sehingga pengguna tidak perlu membacanya secara langsung. Dengan memanfaatkan teknologi *text to speech* yang disediakan oleh Google, aplikasi yang akan dibangun diharapkan dapat menangani kendala tersebut. Kedua, pesan SMS dan *Instant Messenger* yang umumnya berisi kata yang disingkat dapat menyebabkan pengguna tidak menerima informasi secara sempurna jika menggunakan pembacaan pesan melalui *text to speech,* sehingga dibutuhkan fitur

kamus kata yang dapat diisi dengan kata singkatan yang umun digunakan dan padanan kata yang utuhnya.

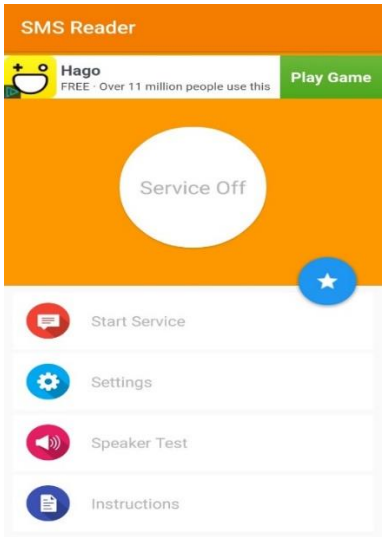
Berdasarkan uraian yang dikemukakan, maka terdapat masalah yaitu bagaimana menerapkan teknologi *text to speech* pada aplikasi pembacaan pesan memanfaatkan Google *text to speech* membantu pengemudi kendaraan bermotor untuk membacakan pesan SMS dan *Instant Messenger* yang masuk, dan bagaimana membuat fitur kamus singkatan yang berisi kata-kata singkatan dan kata-kata yang utuhnya.

3.1.2 Analisis Aplikasi Sejenis

Perbandingan aplikasi sejenis dilakukan dengan tujuan untuk menganalisis berbagai aspek yang dibutuhkan dalam pembangunan aplikasi pembaca pesan. Aplikasi yang dipilih adalah aplikasi yang memiliki fungsionalitas yang sama, yaitu membantu proses *life cycle development*. Adapun perbandingan aplikasi dilakukan kepada SMS *Reader* dan *Message Reader For Whatsapp*. Berikut ini analisis untuk setiap aplikasi :

1. SMS *Reader*

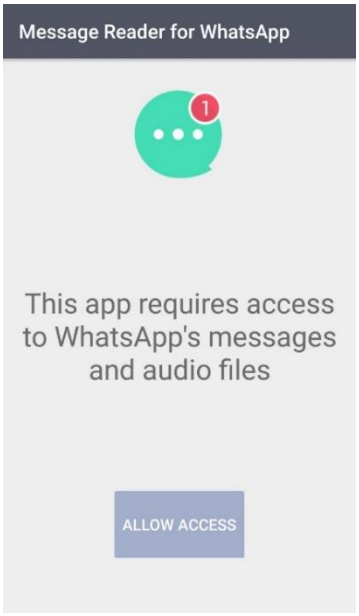
   SMS *Reader* merupakan aplikasi yang memanfaatkan *text to speech* untuk membaca pesan sms yang diterima oleh pengguna. Berikut tampilan antarmuka aplikasi SMS *Reader*.



Gambar III.1 Tampilan Antarmuka Aplikasi SMS *Reader*

2. *Message Reader For Whatsapp*

   Message Reader For Whatsapp merupakan aplikasi yang memanfaatkan *text to speech* untuk membaca pesan Whatsapp yang diterima oleh penggunanya. Berikut tampilan antarmuka aplikasi *Message Reader For Whatsapp* pada Gambar III.2.



Gambar III.2 Tampilan Antarmuka Aplikasi *Message Reader For* Whatsapp

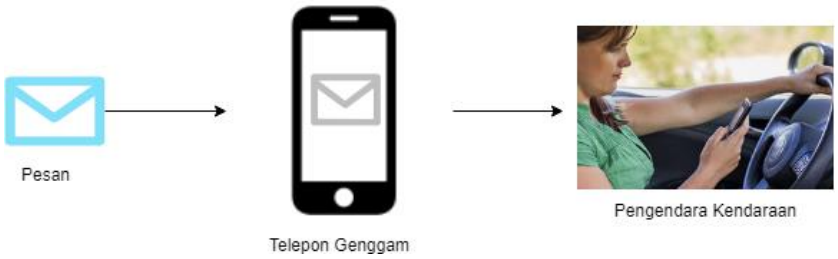
Tabel III.1 Kelebihan dan Kurangan Aplikasi Sejenis

Nama Aplikasi	SMS *Reader*	*Message Reader For* Whatapp
Kelebihan	Dapat membaca pesan SMS yang masuk	Dapat Membaca pesan whatsapp yang masuk
	Tersedih fitur untuk mengatur *volume speaker* dan kecepatan membaca *text to speech*	
Kekurangan	Tidak tersedih kamus singkatan	Tidak tersedia kamus singkatan
		Tidak dapat mengatur *output* suara *text to speech*

3.1.3 Analisis Prosedur yang Sedang Berjalan

Short Message Service (SMS) dan *Instant Messenger* adalah sarana komunikasi berbasis teks yang banyak digunakan karena tidak memakan banyak biaya, tidak menyita banyak waktu dan dapat digunakan untuk berkomunikasi disela-sela kegiatan yang lainnya. Sarana ini banyak digunakan untuk mengirim dan menerima pesan singkat. Pada penelitian ini akan lebih fokus pada penerimaan

pesan singkat saat sedang berkendara. Berikut gambaran umum penerimaan pesan pada saat berkendara pada Gambar 3.1.

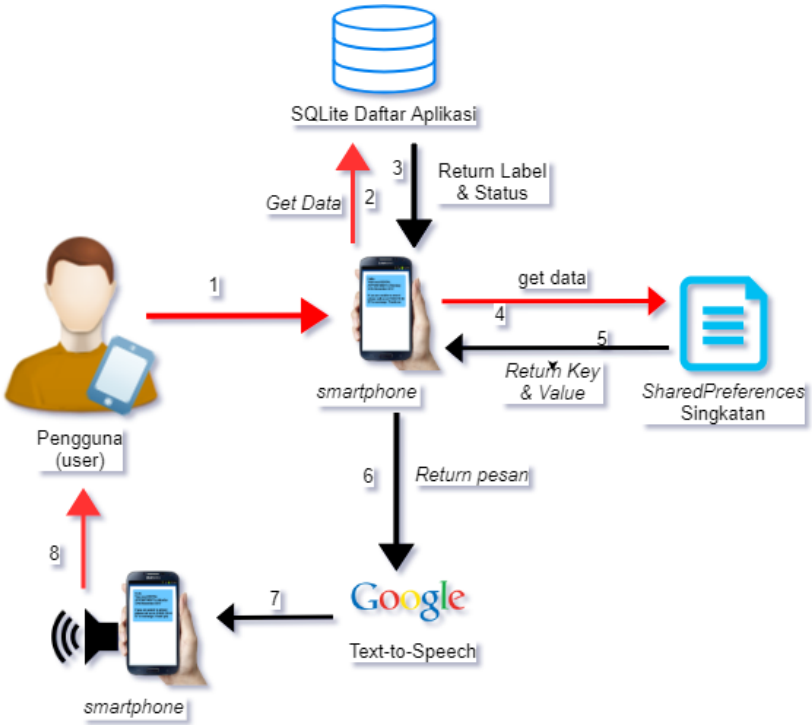


Gambar III.3 Gambaran Umum Penerimaan Pesan Singkat

Gambar III.1 di atas merupakan gambaran umum hubungan antara pesan masuk, perangkat seluler, dan *user* (pengguna). Secara garis besar setiap pesan singkat yang diterima melalui perangkat seluler dan akan dibaca langsung oleh pengguna perangkat seluler tersebut.

3.1.4 Analisis Sistem Yang Dibangun

Sistem yang akan dibangun adalah aplikasi pembaca pesan *Short Message Service* (SMS) dan *instant messenger* berbasis *text to speech* yang dioperasikan pada *platform* Android. Dalam pembangunannya, aplikasi ini mengimplementasi teknologi *text-to speech* yang bertujuan untuk mempermudah pengguna (pengendara kendaraan bermotor) dalam membaca setiap pesan yang masuk. Berikut arsitetur sistem yang akan dibangun pada gambar III.4.

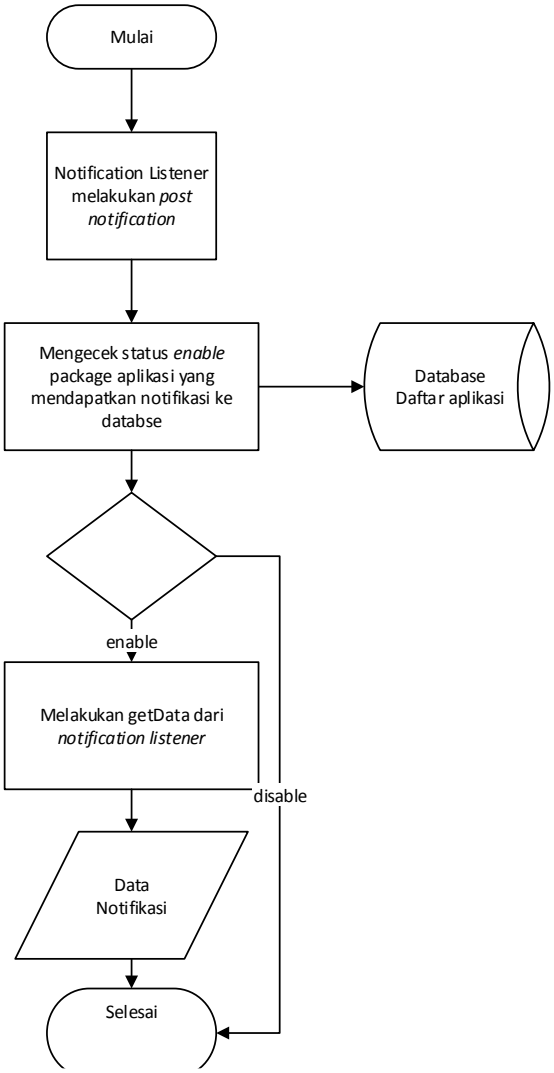


Gambar III.4 Arsitekur Sistem

Berikut skenario sistem yang akan dibangun :

1. Pengguna melakukan interaksi dengan melakukan pengaturan pada aplikasi
    a. Pengaktifan *service* di sini merupakan suatu persetujuan atau *permission* dari aplikasi terhadap sistem android agar aplikasi dapat mengakses dan membaca semua notifikasi, termasuk informasi personal seperti nama kontak, dan pesan yang diterima *smartphone* android.
    b. Pengguna memilih aplikasi mana saja yang notifikasinya akan dibaca oleh *Text-To-Speech*. Aplikasi akan mengakses data dari *database* yang daftar aplikasi yang telah di-*install* pada *device* android. Database akan merespon dan akan menampilkan daftar aplikasi dalam bentuk *ListView* dan *Checkbox*. Dari daftar aplikasi tersebut, pengguna akan memilih dengan mencentang aplikasi mana saja yang nantinya akan dibaca.

2. Setiap pesan yang masuk mulanya akan tampil pada status bar notifikasi. Aplikasi kemudian melakukan *getpackegename* dari *Notification Listener Service*. Aplikasi kemudian memvalidasi ke *database* apakah notifikasi yang masuk berasal dari aplikasi yang dipilih atau tidak dengan cara mencocokkan nama *package* aplikasi yang mendapatkan notifikasi dengan nama *package* yang terdapat pada *database*. Jika sama, maka sistem akan mengecek apakah status aplikasi *enable* (bernilai 1) atau tidak, Jika ya maka sistem akan melakukan *get data* dari *notification listener service*.

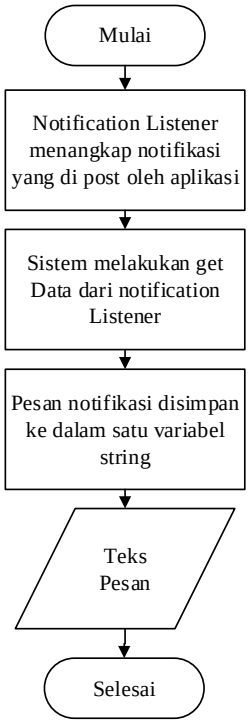


Gambar III.5 *Flowchart* pengecekan aplikasi

3. Setelah divalidasi dan sesuai, notifikasi tersebut diproses dan akan diambil isi dari notifikasi tersebut. Notifikasi suatu aplikasi biasanya berisi *ticker, content title, content text,* dan *content info text* namun untuk pembacaan pesan hanya diperlukan nama aplikasi, *content title* dan *content text*-nya.
4. *Content* notifikasi yang telah diambil, kemudian divalidasi apakah *content* notifikasi mengandung kata singkatan atau tidak dengan membandingan satu per satu kata dari *content* notifikasi dengan *key* pada *sharedpreferences* kamus singkatan. Jika terdapat kata yang sama dengan *key* pada *sharedpreferences* kamus singkatan, *value* dari *key* tersebut akan menggantikan kata dari *content* notifikasi.
5. Setelah proses pengubahan kata selesai dan tidak ada lagi kata yang mengadung kata singkatan, *content* notifikasi diarahkan ke *text-to-speech engine* untuk proses pengubah dari teks ke ucapan/suara dimana teks tersebut berisi nama aplikasi, dan *content* dari notifikasi tersebut.

3.1.4.1 Analisis Get Data Pesan

Pesan merupakan data utama pada dalam pembangunan aplikasi ini, dimana data didapatkan dari notifikasi bar dengan bantuan *Notification Listener Service*. Berikut alur *get* data notifikasi pada gambar III.6.



Gambar III.6 Alur *Get* Data Notifikasi

1. *Notification listener* bertugas untuk menangkap notifikasi yang dikirim oleh pihak ketiga (aplikasi lain) dan menampilkannya ke dalam daftar pesan yang diterima terakhir di perangkat seluler. *Listener service* akan melakukan *return* untuk setiap objek yang terdapat pada notifikasi. Beberapa data yang dapat didapatkan antara lain:
   a. Nama *Package* (untuk mengetahui aplikasi mana yang menerima notifikasi),
   b. Title (judul) dan *text* (berisi isi dari notifikasi),
   c. Nomor telepon dan nama kontak (berasal dari kontak di perangkat seluler).

   Untuk menerapkan dalam projek perlu dilakukan yaitu menambahkan *permission BIND_NOTIFICATION LISTENER_SERVICE* ke dalam *file manifest*.

```
<service android:name=".MyNotificationListener"
          android:label="@string/service_name"

android:permission="android.permission.BIND_NOTIFI
CATION_LISTENER_SERVICE">
     <intent-filter>
         <action
android:name="android.service.notification.Notific
ationListenerService" />
     </intent-filter>
 </service>
```

2. Untuk proses *get data* dari *notification listener* dapat dilakukan dengan, mendeklarasikan *code* berikut:

```
@Override
public void
onNotificationPosted(StatusBarNotification sbn) {
    // We can read notification while posted.
    for (StatusBarNotification sbm :
MyNotification.this.getActiveNotifications()) {
String title =
sbm.getNotification().extras.getString("android.title
");
String text =
sbm.getNotification().extras.getString("android.text"
);
String package_name = sbm.getPackageName();
     Log.v("Notification title is:", title);
     Log.v("Notification text is:", text);
     Log.v("Notification Package Name is:",
package name);}}
```

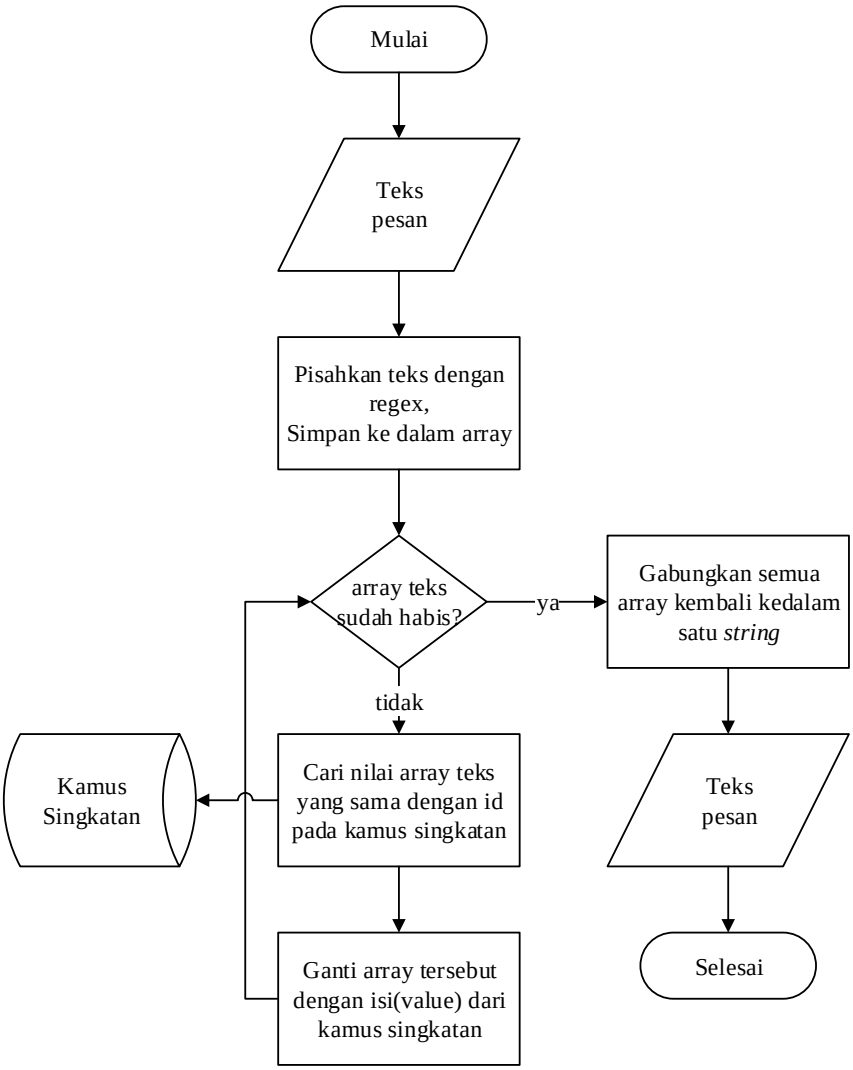
a. *sbm.getNotification().extras.getString("android.title");*

   Merupakan fungsi untuk mengambil atau mendapatkan judul dari notifikasi. Judul tersebut berisi informasi nomor telepon (SMS masuk)atau nama pengirim pesan (berasal dari kontak di perangkat seluler).

b. *sbm.getNotification().extras.getString("android.text");*

   Merupakan fungsi untuk mengambil atau mendapatkan teks atau isi pesan dari notifikasi.

3.1.4.2 Analisis Pengubahan Singkatan



Gambar III.7 *Flow Chart* Pengubahan Singkatan

1. Menginisialisasikan data teks pesan yang telah didapatkan dari notifikasi
2. Setiap kata dari *String* pesan kemudian dicocokkan dengan semua data singkatan yang tersimpan *sharedpreferences* kamus singkatan dengan fungsi *regex replaceall()*. Fungsi *regex replaceall()* akan mengecek satu persatu kata pada teks pesan apakah terdapat kata yang sama dengan dengan *key* pada kata kamus singkatan, jika kata tersebut sama atau

cocok maka kata tersebut akan digantikan oleh *value* dari *key* kata tersebut. Proses ini terus akan dilakukan sampai teks pesan tidak mengandung kata yang sama dengan *key* atau *id* dari data *sharedpreferences* kamus singkatan. Berikut potongan *code* pencocokan dan penggantian kata singkatan.

```
if (!TextUtils.isEmpty(ttsMessage)) {
    String ttsTextReplace =
prefs.getString(context.getString(R.string.key_ttsTextReplace),
null);
    List<Pair<String, String>> textReplaceList =
TextDictionary.convertStringToList(ttsTextReplace);
    for (Pair<String, String> pair : textReplaceList) {
        ttsMessage = ttsMessage.replaceAll("(?i)" +
Pattern.quote(pair.first), pair.second);
    }}
```

3. Setelah proses pencocokan dan pengubahan selesai, akan menghasilkan teks pesan yang tidak mengandung singkatan.

3.1.4.3 Analisis Android Google Text To Speech

Penggunaan Google *Text To Speech* memungkinkan developer untuk mengkonversi teks menjadi suara. Layanan ini dapat diproses secara *offline* dan *online*. Pada pemrosesan secara *offline*, bahasa yang didukung terbatas pada bahasa yang terdapat pada masing-masing *device* atau *smartphone*.

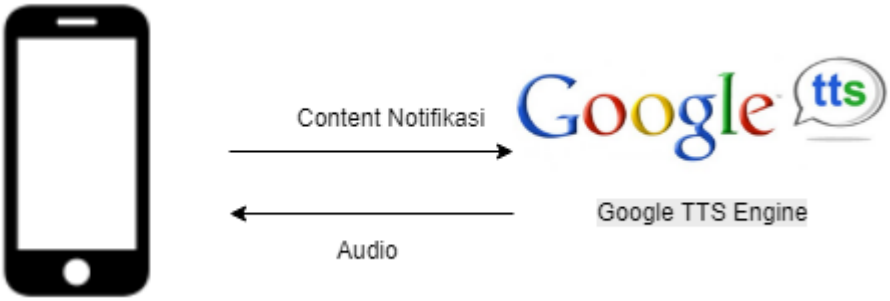
Layanan Google *Text To Speech* digunakan pada aplikasi ini untuk membaca pesan dimana *input* pada proses ini merupakan teks dari bar notifikasi yang telah diproses sebelumnya pada proses pengubahan kata singkatan dan *output* yang dihasilkan pada proses ini berupa suara (ucapan).

Prosedur yang digunakan untuk mengkonversi teks menjadi ucapan pada aplikasi yang akan dibangun yaitu dengan menambahkan metode *speak(),* yang menggunakan *text-to-speech engine* untuk membaca teks apa pun yang diteruskan ke sana. Sebelum memproses, *text-to-speech engine* akan memeriksa apakah pengguna sudah menyetujui pembacaan teks. Berikut *method* pembacaan teks oleh *text to speech engine* pada Gambar III.8.

```
01  public void speak(String text){
02
03      // Speak only if the TTS is ready
04      // and the user has allowed speech
05
06      if(ready && allowed) {
07          HashMap<String, String> hash = new HashMap<String,String>();
08          hash.put(TextToSpeech.Engine.KEY_PARAM_STREAM,
09                  String.valueOf(AudioManager.STREAM_NOTIFICATION));
10          tts.speak(text, TextToSpeech.QUEUE_ADD, hash);
11      }
12  }
```

Gambar III.8 *Method speak()*

Prosedur *speak()* nantinya akan inisialisasikan pada *class service* untuk membaca pesan yang telah melalui proses pengubahan kata singkatan.



Gambar III.9 Alur Penggunaan Google *Text To Speech Engine*

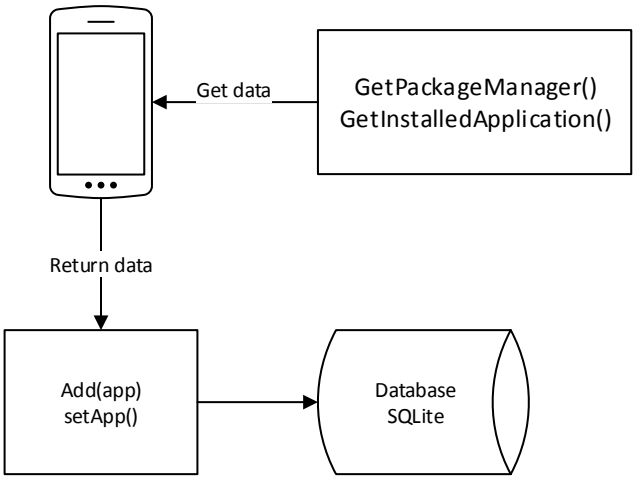
1. *Content* notifikasi hasil proses pengubahan kata singkatan dikirim oleh device ke Google TTS Engine melalui prosedur contoh *speak(ttsmessage).*
2. Google TTS mengkonversi tulisan yang dikirimkan melalui prosedur *speak(ttsmessage)* sebelumnya kedalam bentuk suara (audio)
3. Setelah proses konversi selesai Google TTS akan menjalankan suara ke *device* secara *realtime.*



Gambar III.10 Konversi teks ke ucapan Google *TTS Engine*

3.1.4.4 Analisis *SQlite* Daftar Aplikasi

Pada fitur Daftar Aplikasi memungkinkan pengguna untuk memilih aplikasi mana saja yang notifikasinya akan dibaca oleh aplikasi. Data aplikasi pada daftar aplikasi merupakan aplikasi-aplikasi yang sudah ter-install di *device* android. Data-data berupa kolom nama aplikasi, nama *package* aplikasi seperti com.whatsapp, dan kolom nilai {0,1}, nilai 0 berarti aplikasi tersebut tidak dibaca dan nilai 1 berarti aplikasi akan dibaca notifikasinya.



Gambar III.11 Alur proses pada SQlite untuk Daftar Aplikasi

1. Data pada *database* daftar aplikasi *SQLite* didapatkan dengan menginisialasikan PackcageManager pada *class* AppListActivity. PackageManager digunakan untuk mengambil informasi tentang paket aplikasi yang saat ini diinstal pada perangkat android. Hal tersebut dilakukan dengan membuat *instance* kelas *PackageManager* dengan

memanggil *getPackageManager().* Berikut potongan *code* untuk melakukan *instance* dengan *PackageManager.*

```
PackageManager packageManager = getPackageManager();
List<ApplicationInfo> list =
packageManager.getInstalledApplications(PackageManager.GET_META_DATA)
```

PackageManager.getInstalledApplications() untuk melakukan *return* daftar semua paket aplikasi yang diinstal pada perangkat android.

2. Sebelum dilakukan penyimpanan data ke *database*, perlu dilakukan *filter* terhadap aplikasi agar hanya aplikasi yang dibutuhkan saja yag tersimpan pada *database*. Berikut potongan *code filter* aplikasi:

```
if((appInfo.flags & ApplicationInfo.FLAG_SYSTEM) == 1 )
{

}else{
        apps.add(app);
        setApp();
    }
```

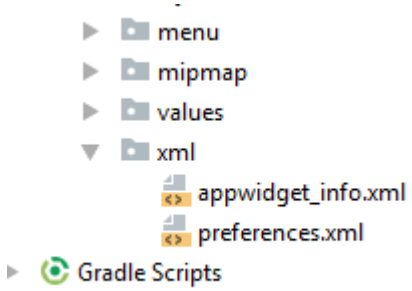
Melalui *code* di atas aplikasi akan difilter apakah aplikasi tersebut merupakan aplikasi sistem android atau aplikasi yang diinstal oleh pengguna. *Code* diatas akan mencocokkan satu per satu alamat *package* aplikasi, jika ya atau sama maka akan disimpan ke dalam *database* daftar aplikasi.

3.1.4.5 Analisis SharedPreferences Kamus Singkatan

Kamus Singkatan merupakan data yang berisi kata singkatan dan kata asalnya, dimana kata singkatan sebagai *key (id)* dan kata asal sebagai *value.* Seperti pada Gambar 3.6 di atas, *key* dari setiap kata singkatan akan dibandingkan dengan kata dari isi pesan, dimana jika terdapat *key* yang sama dengan kata pada pesan, maka *value* dari *key* tersebut akan menggantikan kata dari isi pesan.

Pada fitur kamus singkatan, data kata singkatan belum ditentukan dan di-input-kan sebelumnya, yang artinya pengguna dapat menambahkan atau menghapus data kata singkatan. Karena data kata singkatan yang disimpan hanya berupa string, dan hanya berisi *key* dan *value* maka data kamus singkatan disimpan kedalam

SharedPreferences. *SharedPreferences* memiliki fungsi yang serupa dengan *cookies* pada *web browser*. Penyimpanan data pada SharedPreferences memiliki ukuran yang lebih kecil bila dibandingkan dengan penyimpanan kedalam *database* atau *file*. *SharedPreferences* menyimpan data yang bertipe *key-value* dan disimpan dalam bentuk xml.. Berikut folder peyimpanan *SharedPreferences.*



Gambar III.12 Penyimpanan *Shared Preferences* Kamus Singkatan

3.1.5 Analisis Kebutuhan Non Fungsional

Analisis dan kebutuhan non fungsional meliputi analisis kebutuhan perangkat keras serta analisis kebutuhan perangkat lunak. Adapun kebutuhan non fungsional untuk menjalankan aplikasi ini meliputi kebutuhan perangkat keras, kebutuhan perangkat lunak, kebutuhan data dan pengguna sistem yang akan menggunakan aplikasi. Analisis kebutuhan non fungsional bertujuan agar aplikasi yang dibangun dapat digunakan sesuai dengan kebutuhan pengguna aplikasi dalam mencari informasi yang dibutuhkan.

3.1.5.1 Analisis Kebutuhan Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam pembangunan aplikasi ini adalah:

1. Perangkat Keras Komputer

Tabel III.2 Spesifikasi Minimun Perangkat Keras Komputer

No	Perangkat Keras	Spesifikasi
1	RAM	4 GB
2	Harddisk	500 GB
3	Processor	Intel Core i3 1.33 GHz

2. Perangkat keras *Smartphone*:

Tabel III.3 Spesifikasi Perangkat keras *smartphone*

No	Perangkat Keras	Spesifikasi
1	RAM	2 GB
2	Penyimpanan Internal	16 GB
3	Processor	1.8 GHz

Spesifikasi minimum perangkat keras Smartphone yang dibutuhkan untuk dapat menjalankan aplikasi ini adalah:

Tabel III.4 Spesifikasi minimum Perangkat keras *smartphone*

No	Perangkat Keras	Spesifikasi
1	RAM	1 GB
2	Penyimpanan Internal	8 GB
3	Processor	800 GHz

3.1.5.2 Analisis Kebutuhan Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam pembangunan aplikasi ini adalah:

1. Perangkat lunak komputer:

Tabel III.5 Spesifikasi Perangkat Lunak Komputer

No	Perangkat Lunak	Spesifikasi
1	Sistem Operasi	Windows 8.1 64 bit
2	Android Studio	Versi 3.0
3	ADT (*Android Development Tools*)	Versi v22.6.2-1085508
4	Java Development Kit	Versi 7.0.510.13

2. Perangkat lunak *smartphone* yang digunakan adalah sistem operasi Android versi 5.1.1 *Lollipop*.

   Spesifikasi minimum perangkat lunak *smartphone* yang dibutuhkan untuk dapat menjalankan aplikasi ini adalah sistem operasi Android versi 5.0 *Lollipop*.

3.1.5.3 Analisis Kebutuhan Pengguna

Analisis pengguna sistem dimaksudkan untuk mengetahui siapa saja pengguna yang terlibat dalam menjalankan sistem. Pengguna yaitu publik

khususnya pengendara kendaraan bermotor yang dapat mengoperasikan smartphone khususnya platform android. Aplikasi ini tidak hanya dapat digunakan oleh pengendara melainkan dapat digunakan oleh semua kalangan sebagai media pembaca pesan disaat pengguna sedang tidak dapat membaca pesan secara langsung.

3.1.5.4 Analisis Kebutuhan Data

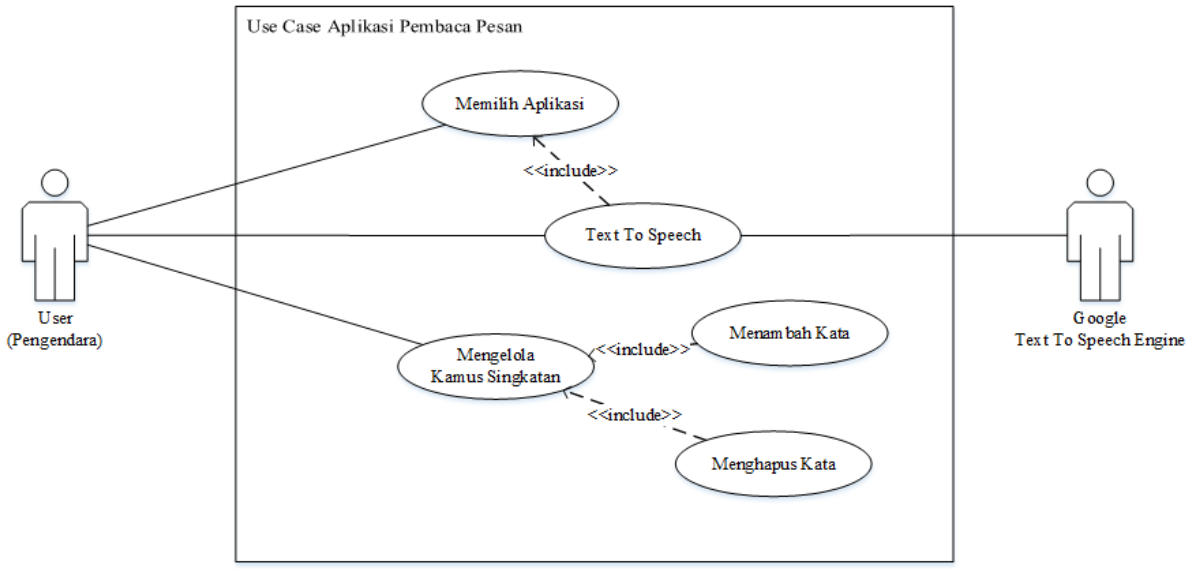
Data yang di butuhkan dalam pembangunan aplikasi ini yaitu data kata singkatan (**Lampiran A-1**). Data ini nantikan akan berguna dalam proses pengubahan singkatan ke kata yang lebih natural.

3.1.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik. Analisis yang dilakukan dimodelkan dengan menggunakan UML (*Unified Modeling Language*). Tahaptahap pemodelan dalam analisis tersebut antara lain *Use Case Diagram, Activity Diagram, Sequence Diagram,* dan *Class Diagram.*

3.1.6.1 Use Case Diagram

Use Case Diagram mendeskripsikan sebuah interaksi antar satu atau lebih aktor dengan sistem yang akan dibuat. Secara kasar *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. *Use Case* terdiri dari tiga bagian yaitu identifikasi aktor, identifikasi *use case* dan skenario *use case*. Analisis *Use Case Diagram* pada sistem yang akan bibangun dapat dilihat pada Gambar III.15.



Gambar III.13 *Use Case Diagram* Aplikasi Pembaca Pesan

1. Identifikasi Aktor

   Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibangun. Berikut adalah aktor-aktor yang terdapat pada sistem yang akan dibangun.

Tabel III.6 Identifikasi Aktor

No	Aktor	Deskripsi
A-01	**User (Pengendara)**	Merupakan aktor dari aplikasi yang dibangun
A-02	**Google Text To Speech**	Merupakan aktor yang berperan mengkonversi teks menjadi suara (ucapan)

2. Identifikasi *Use Case*

   Use case merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. Berikut adalah *use case* yang terdapat pada sistem yang akan dibangun.

Tabel III.7 Identifikasi *Use Case*

No	Use Case	Deskripsi
UC-01	Memilih Aplikasi	Fungsionalitas untuk memilih aplikasi yang pesannya akan dibaca oleh *text to speech*
UC-02	Text To Speech	Fungsionalitas untuk mengkonversi pesan teks ke dalam bentuk ucapan (suara)
UC-03	Mengelolah Kamus Singkatan	Fungsionalitas untuk menampilkan data kamus singkatan yang tersimpan di *SharedPrefences*
UC-04	Menambah kata	Fungsionalitas untuk menambah data pada kamus singkatan
UC-05	Menghapus Kata	Fungsionalitas untuk menghapus pada kamus singkatan

3. Skenario *Use Case*

   Skenario *Use Case* mendeskripsikan urutan langkah-langkah dalam proses bisnis, baik yang dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap aktor.

   a. Skenario *Use Case* Memilih Aplikasi

Tabel III.8 *Use Case* Mengelola Daftar Aplikasi

Identifikasi	
Nama *Use Case*	Memilih Aplikasi
Aktor	User
Tujuan	Menampilkan dan mengubah Daftar Aplikasi
Kondisi Awal	Menampilkan menu pengaturan aplikas
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan *button* Daftar Aplikasi	
	2. Menampilkan Daftar Aplikasi yang terdapat di *database* dan disertai *checkbox* untuk mengatur apakah pesan aplikasi dapat dibacakan atau tidak.
3. Menekan *checkbox* pada aplikasi yang akan dibaca pesannya	
	4. Menyimpan perubah status aplikasi pada *database*
	5. Menampilkan *toast* "Aplikasi ….. akan dibaca"
Kondisi akhir	Menampilkan daftar aplikasi telah mengalami pengubahan

b. Skenario *Use Case Text-To-Speech*

Tabel III.9 Tabel Skenario *Text-To-Speech*

Identifikasi	
Nama *Use Case*	Text-To-Speech
Aktor	Google Text-to-speech Engine
Tujuan	Text-To-Speech membacakan pesan yang masuk
Kondisi Awal	Terdapat pesan SMS atau instant messenger yang masuk ke telepon genggam pengguna
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Sistem akan mengambil *content* dari notifikasi bar
	2. Sistem akan mengecek apakah ke *database* apakah aplikasi statusnya *enable* atau *disable* . Jika enable maka akan diteruskan untuk diproses, dan jika tidak maka proses tidak dilanjutkan
	3. *Content* dari notifikasi kemudian diubah formatnya menjadi *ticker, content title, content text*, dsb.
	4. Sistem akan mengecek format pesan yang akan dibaca disesuai dengan setelan pengguna atau default
	5. Kemudian sistem akan menggabungkan hasil pengecekan menjadi "ttsmessage".
	6. *Ttsmessage* kemudian dicek dan dibandingkan apakah mengadung kata yang sama dengan *key* pada *sharedprefences* kamus singkatan. Jika terdapat kata yang sama dengan *key* tersebut, maka *value* dari *key* akan menggantikan kata yang berada di dalam ttsmessage
	7. Kemudian ttsmessage dicek apakah kosong atau tidak sebelum dibacakan memakai *method* speak(). Jika kosong maka akan dibaca nama aplikasinya saja dan jika tidak maka tts akan membaca nama aplikasi dan isi dari *ttsmessage*
8. Mengkonversi teks menjadi ucapan dengan *method speak().*	
Kondisi Akhir	Pengguna mendengarkan nama aplikasi dan isi pesan melalui telepon genggamnya

c. Skenario Mengelola Kamus Singkatan

Tabel III.10 Skenario Mengelola Kamus Singkatan

Identifikasi	
Nama *Use Case*	Mengelolah Kamus Singkatan
Aktor	User
Tujuan	Menampilkan Kamus Singkatan
Kondisi Awal	Menampilkan menu pengaturan
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan *button* kamus singkatan	
	2. Menampilkan *dialog preference* kamus singkatan
Kondisi akhir	Menampilkan dialog kamus singkatan

d. Skenario Menambha Kata Singkatan

Tabel III.11 Skenario Menambah Kata Singkatan

Identifikasi	
Nama *Use Case*	Menambah Kata Singkatan
Aktor	User
Tujuan	Menambah data pada kamu s singkatan
Kondisi Awal	Menampilkan Dialog Kamus Singkatan
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Meng-*input*-kan kata singkatan dan kata pengganti	
2. Menekan button OK pada dialog kamus singkatan	
	3. Data yang di-*input*-kan disimpan ke dalam *sharedpreferences*.
Kondisi akhir	Data kat singkatan berhasil ditambahkan

e. Skenario Mengahapus Kata singkatan

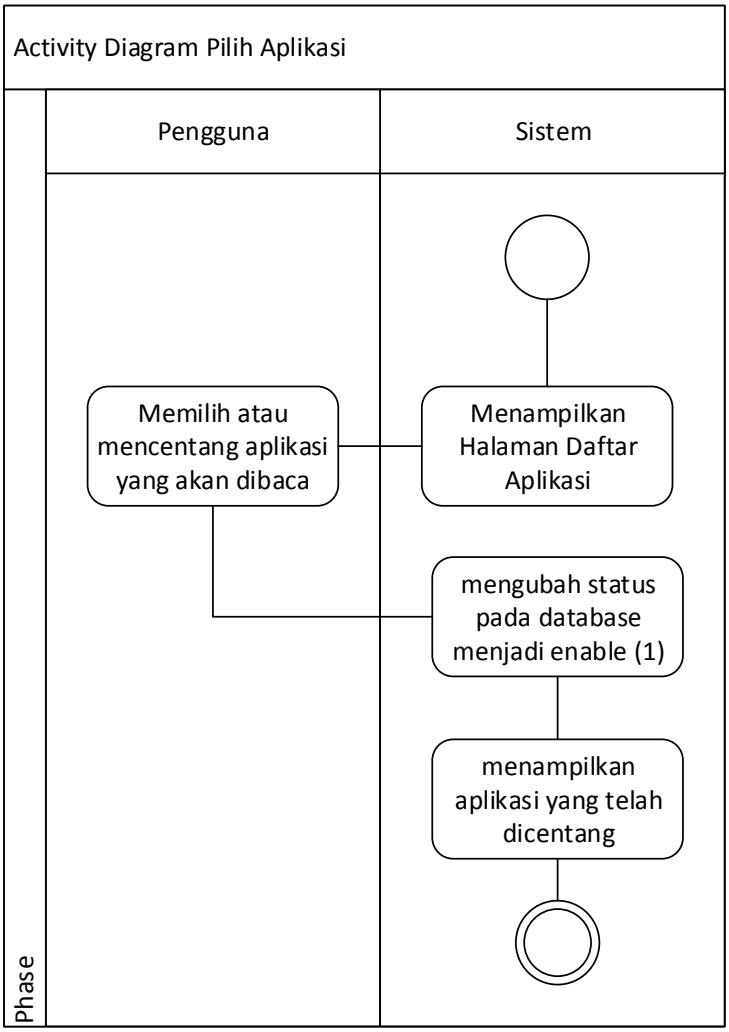
Tabel III.12 Skenario Menghapus Kata Singkatan

Identifikasi	
Nama *Use Case*	Menghapus Kata Singkatan
Aktor	User
Tujuan	Menghapus data pada kamu s singkatan
Kondisi Awal	Menampilkan Dialog Kamus Singkatan
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan button "X" pada kamus singkatan	
	2. Data yang telah dihapus telah hilang dari Kamus Singkatan
3. Menekan *button* OK pada dialog kamus singkatan	
	4. Data yang telah diubah kemudian disimpan kembali ke *SharedPreferences*
Kondisi akhir	Data kata singkatan berhasil dihapus

3.1.6.2 Activity Diagram

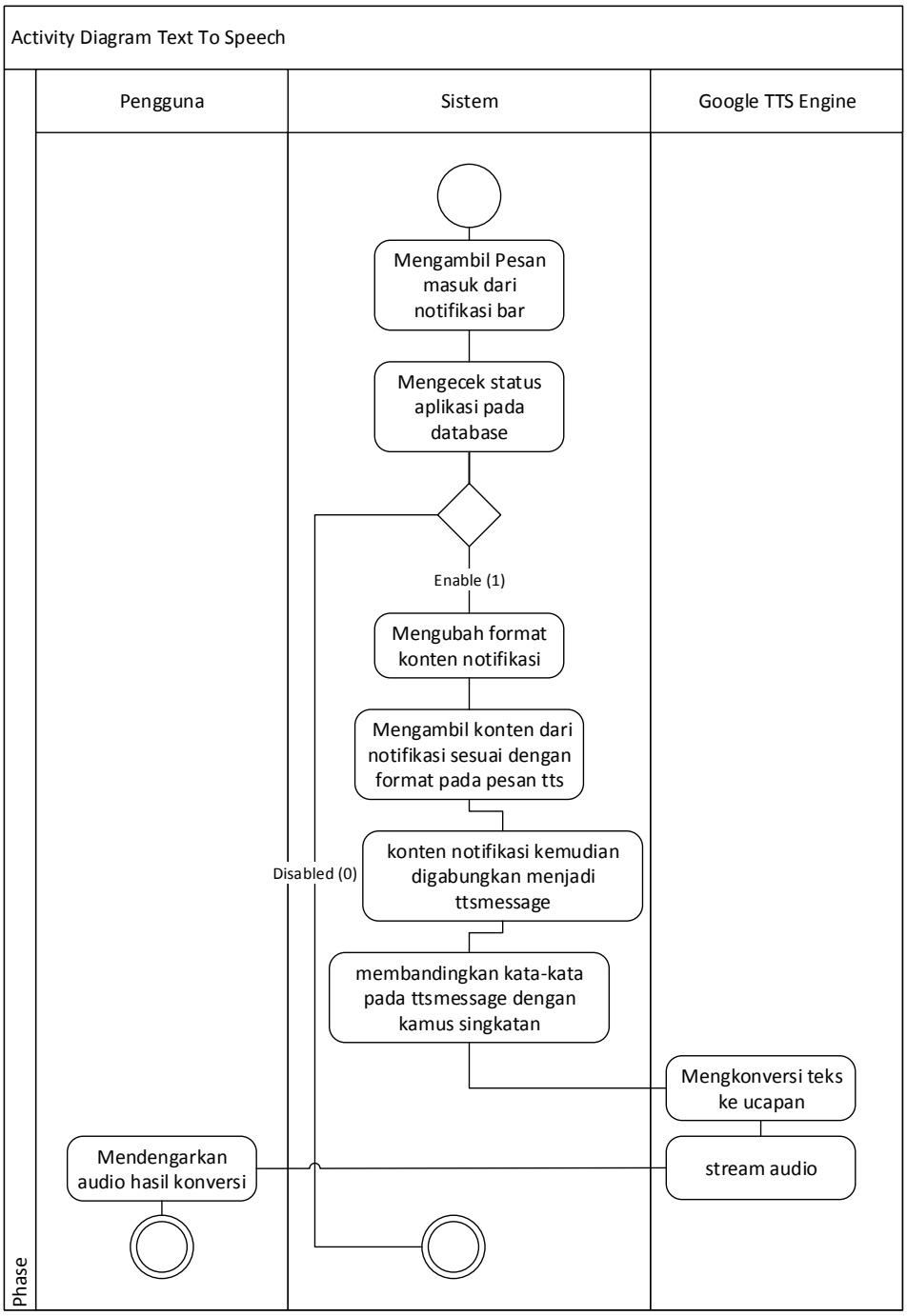
Activity Diagram menggambarkan aliran kerja (*workflow*) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Penggambaran *activity diagram* memiliki kemiripan dengan *flowchart diagram*.

1. *Activity Diagram* Memilih Aplikasi



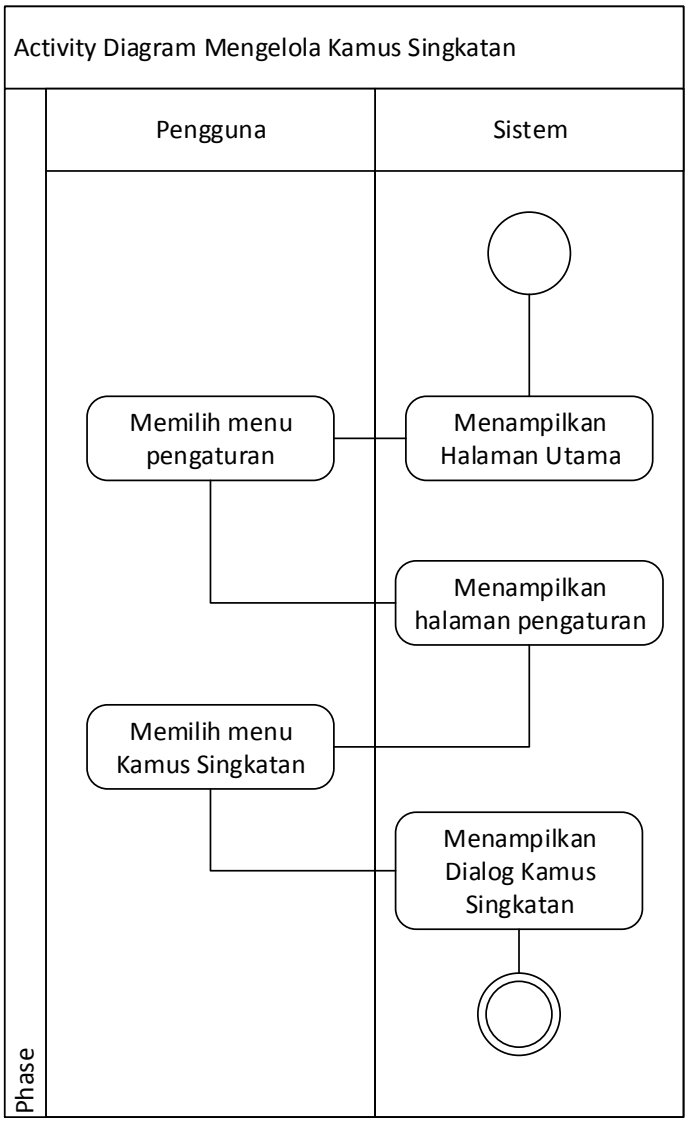
Gambar III.14 *Activity Diagram* Memilih Daftar Aplikasi

2. *Activity Diagram Text To Speech*



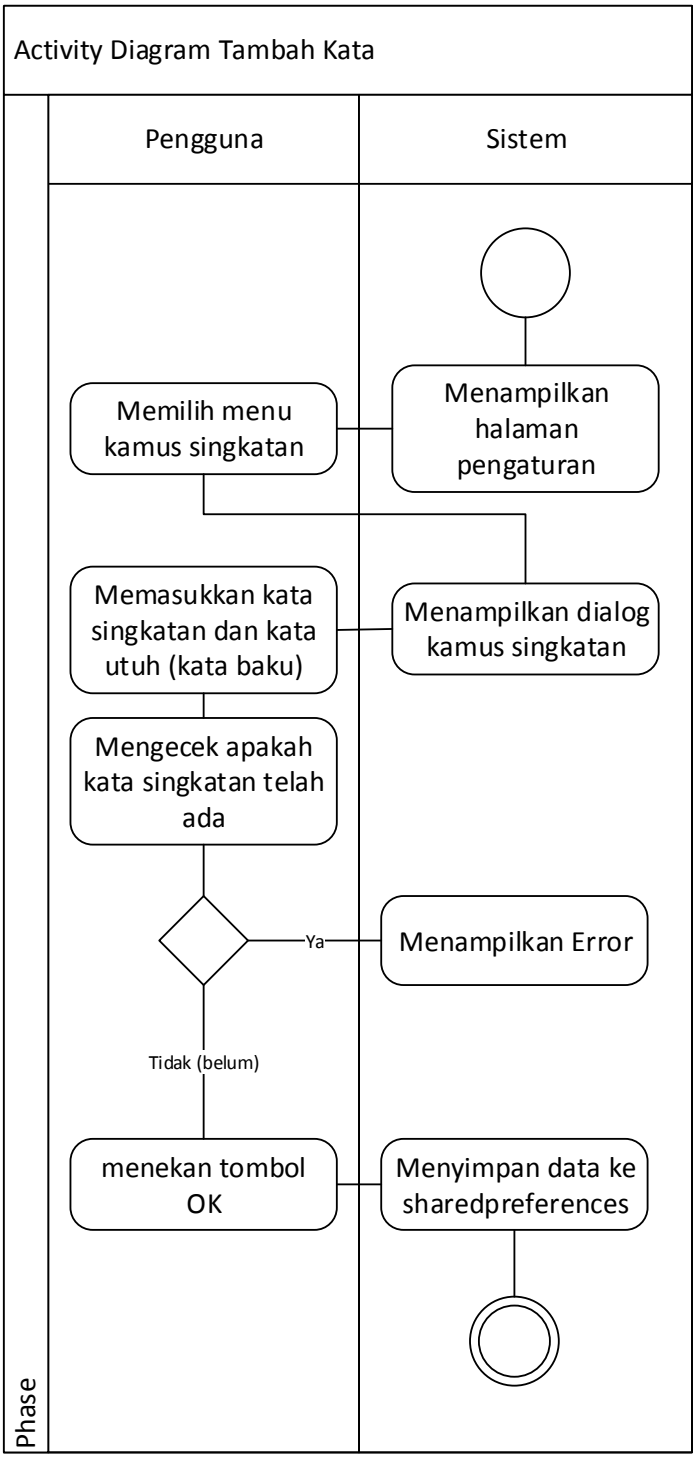
Gambar III.15 ***Activity Diagram Text To Speech***

3. *Activity Diagram* Mengelola Kamus Singkatan



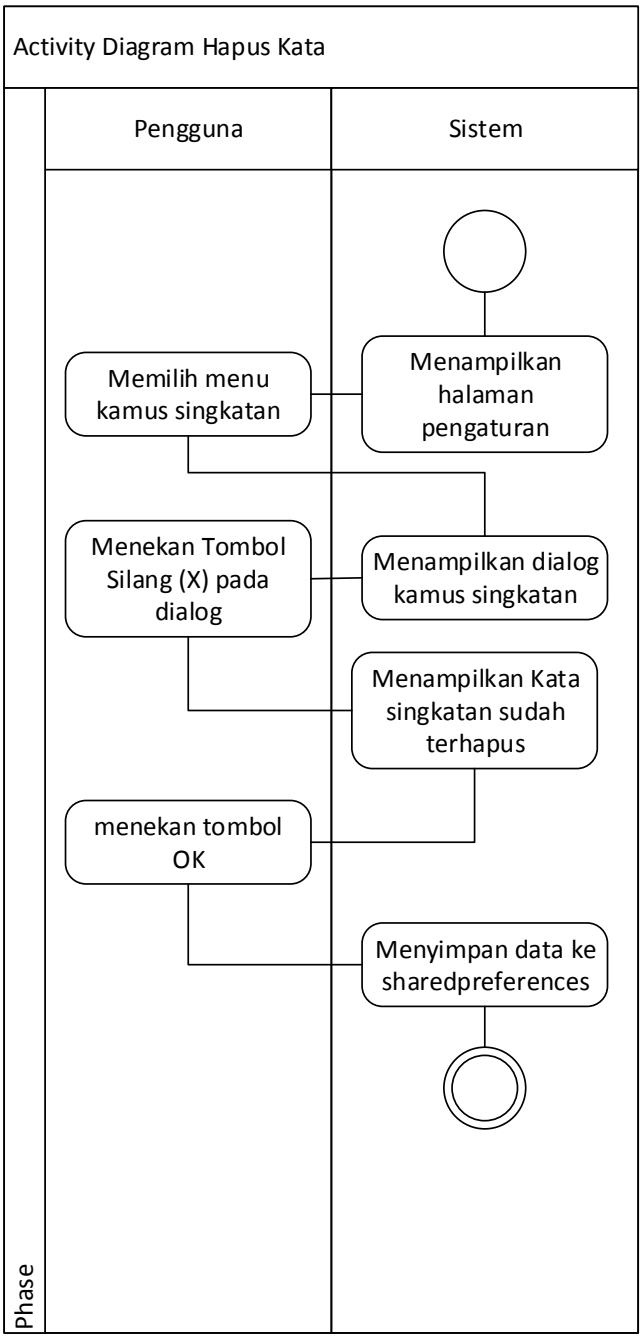
Gambar III.16 *Activity Diagram* Mengelola Kamus Data

4. *Activity Diagram* Menambah Data Kamus Singkatan



Gambar III.17 *Activity Diagram* Meambah Data Kamus Singkatan

5. *Activity Diagram* Menghapus Data Kamus Singkatan

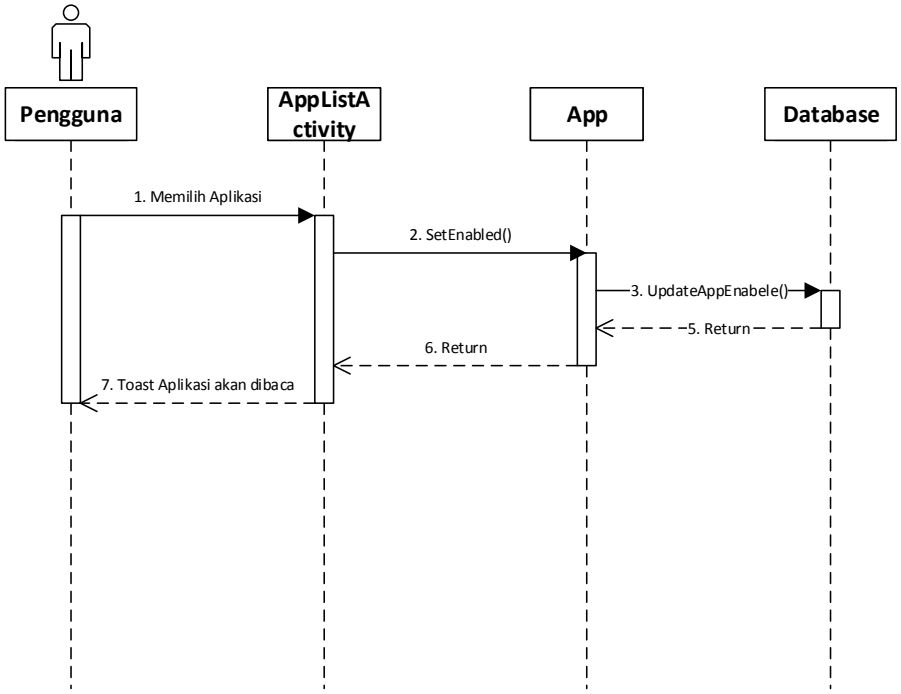


Gambar III.18 *Activity Diagram* Hapus Data Kamus Singkatan

3.1.6.3 Sequence Diagram

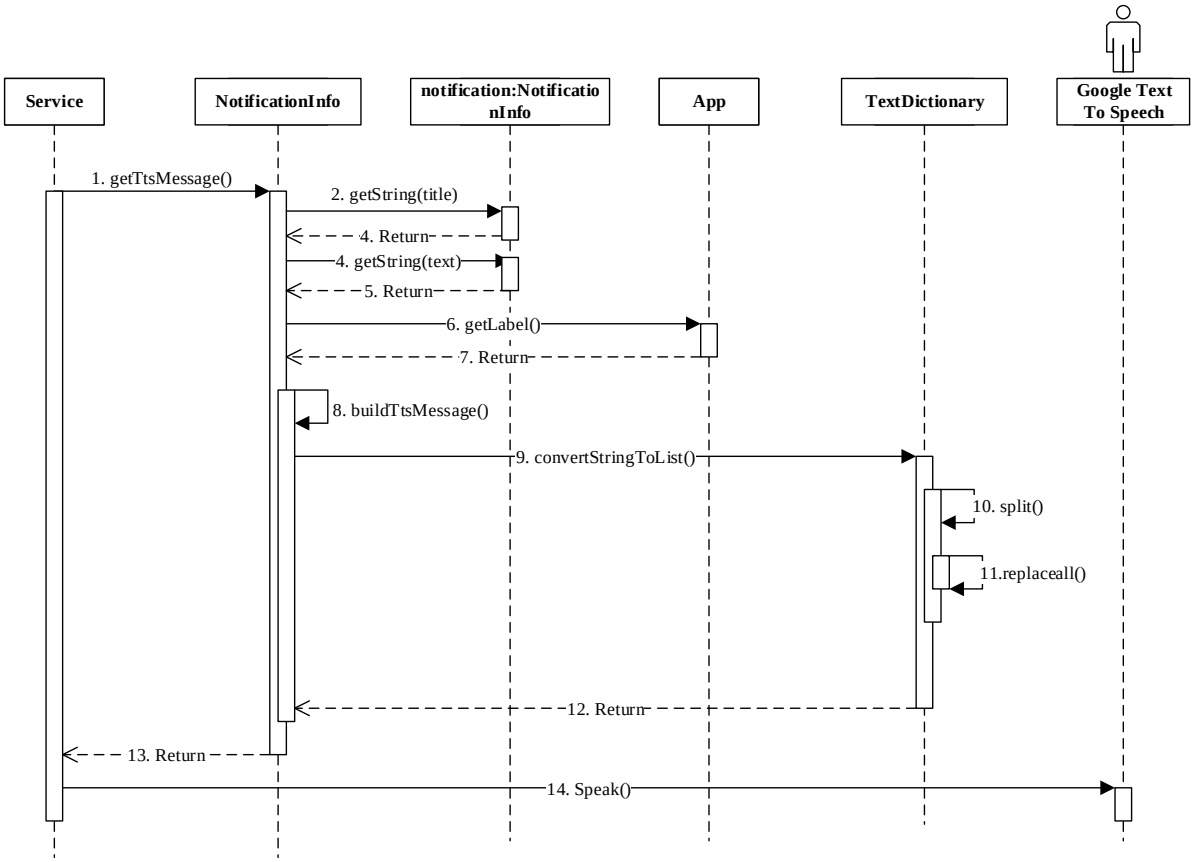
Sequence Diagram, menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

1. *Sequence Diagam* Memilih Aplikasi



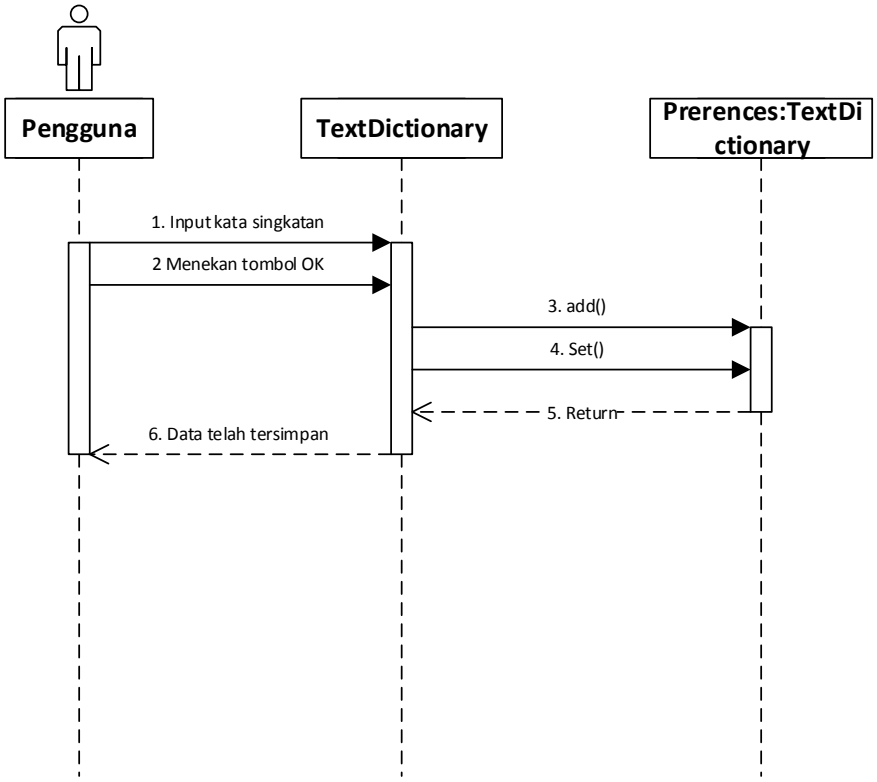
Gambar III.19 *Sequence Diagram* Memilih Aplikasi

2. *Sequence Diagam* Text To Speech



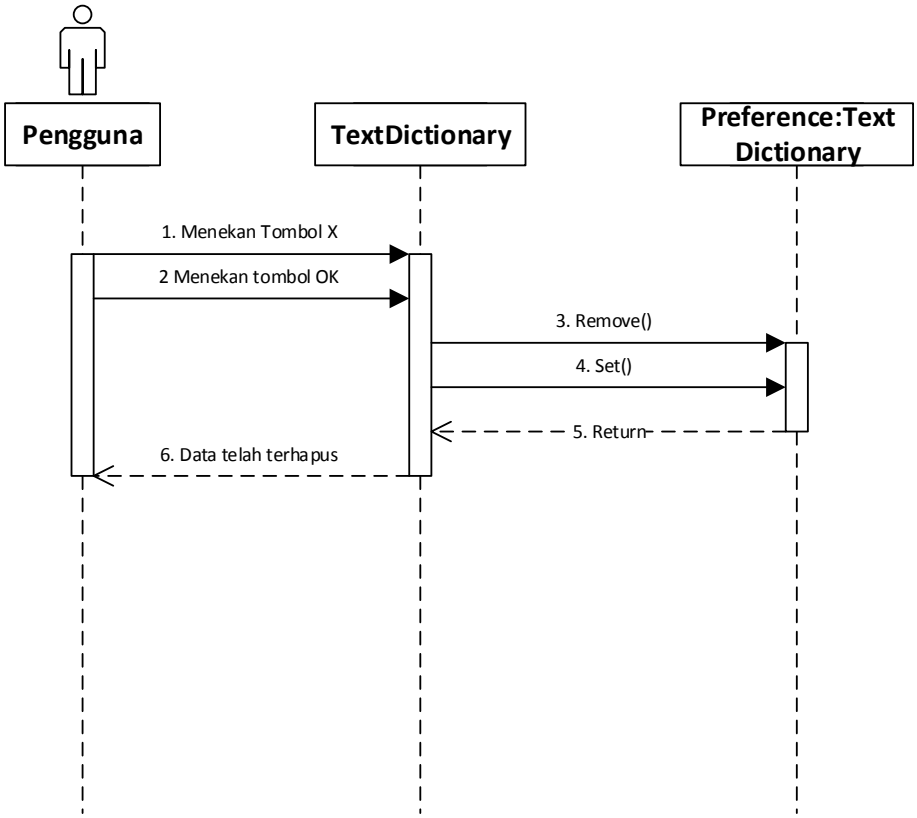
Gambar III.20 *Sequence Diagram Text To Speech*

3. *Sequence Diagam* Menambah Kata singkatan



Gambar III.21 *Sequence Diagram* Menambah Kata Singkatan

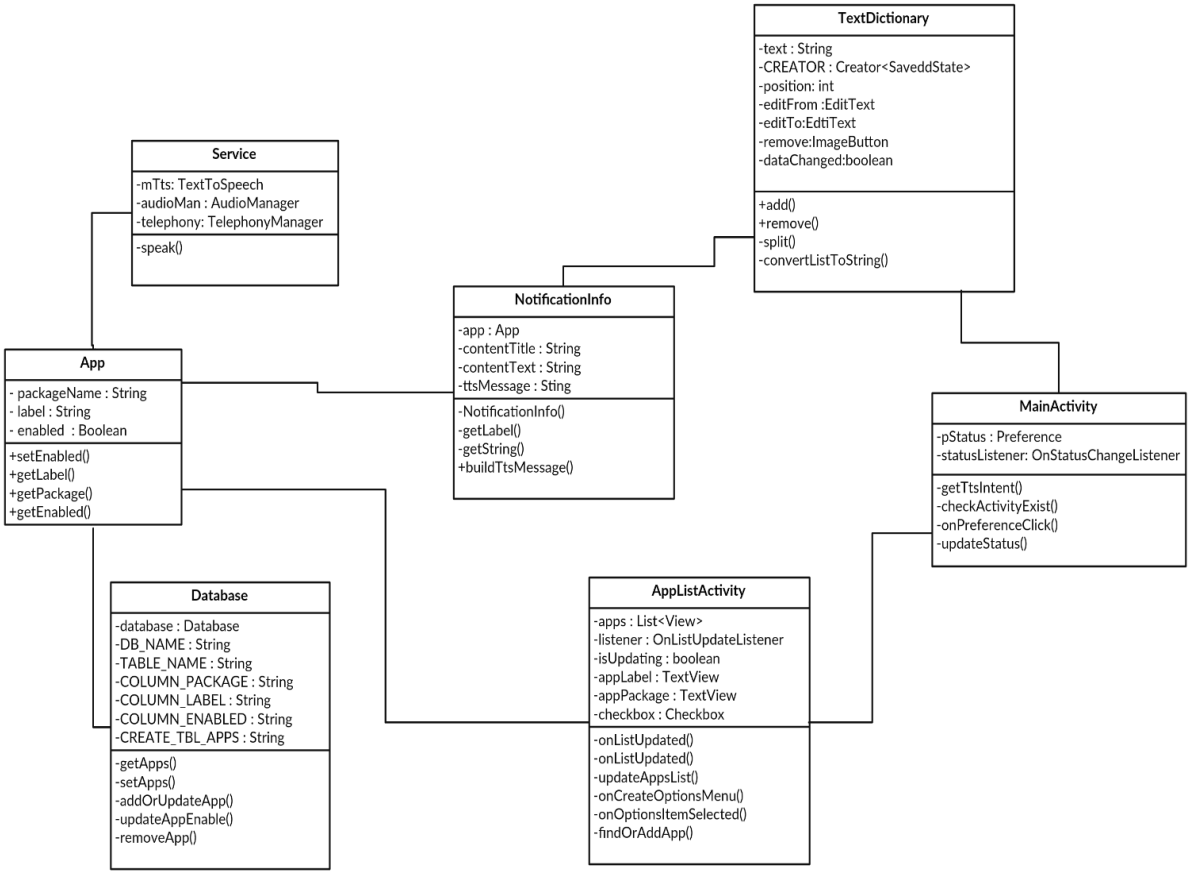
4. *Sequence Diagam* Menghapus kata Singkatan



Gambar III.22 *Sequence Diagram* Menghapus Kata Singkatan

3.1.6.4 *Class Diagram*

Class diagram menggambarkan struktur dan hubungan antar objek-objek yang ada pada sistem. Struktur ini meliputi atribut-atribut dan metode-metode yang ada pada masing-masing class. Hubungannya dari masing-masing class yang ada dapat dilihat pada Gambar III.23.



Gambar III.23 *Class Diagram*

Tabel III.13 Keterangan *Class Diagram*

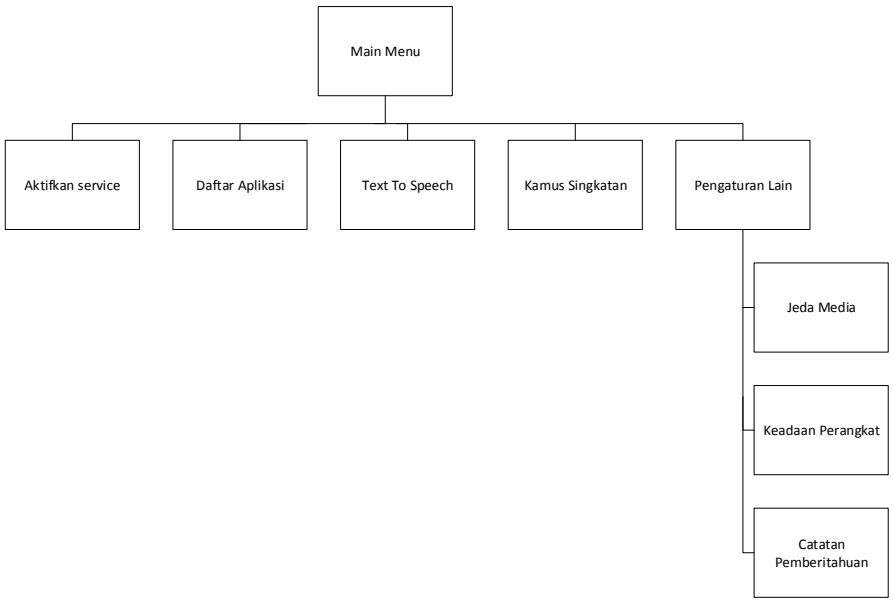
No	Nama Class	Jenis Class	Keterangan
1	MainActivity	Boundary	Class untuk menampilkan halaman utama
2	App	Control	Merupakan class yang digunakan untuk pengaksesan data ke *database*
3	AppListActivity	Control	Merupakan class untuk menampilkan daftar aplikasi
5	Database	Entity	Merupakan class yang digunakan untuk pengaksesan data ke *database*
6	Service	Control	Merupakan class yang menangani pengecekan kata singkatan dan konversi text ke suara
7	NotificationInfo	Control	Merupakan class yang menangani pengambilan content notifikasi bar dan pengubah format *content* notifikasi
9	TextDictionary	Entity	Merupakan class yang digunakan untu pengaksesan data ke sharedpreference

3.2 Perancangan Sistem

Perancangan sistem bertujuan untuk menspesifikasikan aspek-aspek teknik yang menjadi solusi dalam perencanaan. Pada tahap ini perancangan akan didefinisikan secara detail untuk mengatasi masalah-masalah yang lebih teknis, berkaitan dengan kegiatan implementasi seperti perancangan menu, perancangan antarmuka, dan jaringan semantik.

3.2.1 Perancangan Struktur Menu

Perancangan struktur menu merupakan gambaran alur pemakaian aplikasi. Perancangan struktur menu dari aplikasi ini dapat dilihat pada Gambar III.24.

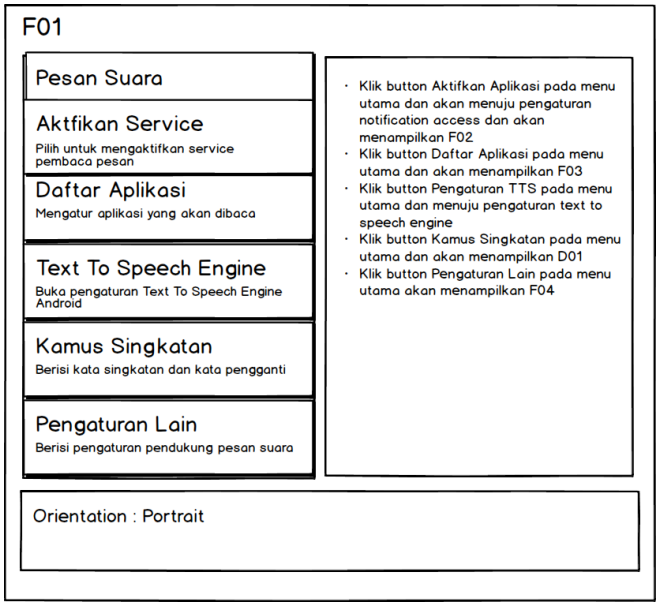


Gambar III.24 Struktur Menu

3.2.2 Perancangan Struktur Menu

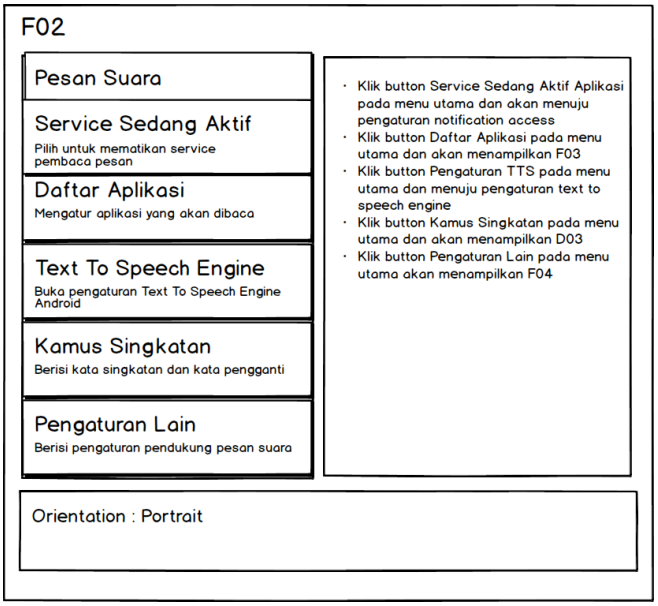
Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga mempermudah dalam mengimplementasikan aplikasi serta memudahkan dalam pembuatan aplikasi.

1. Perancangan Menu Utama



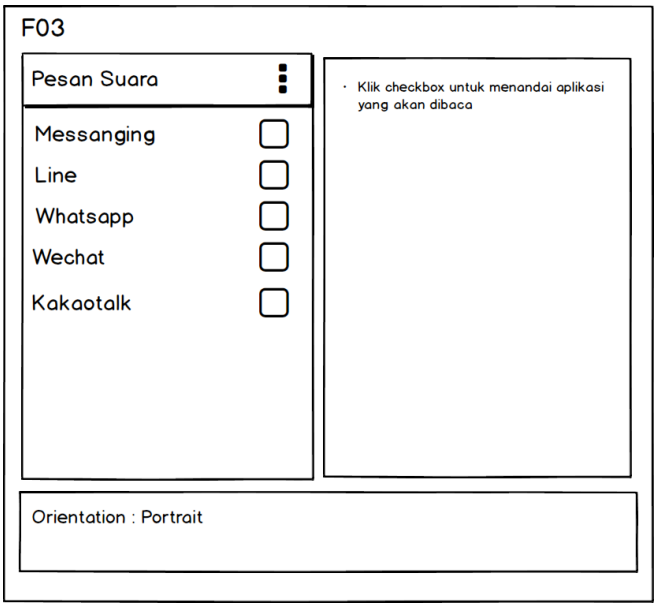
Gambar III.25 Perancangan Menu Utama

2. Perancangan Menu Utama



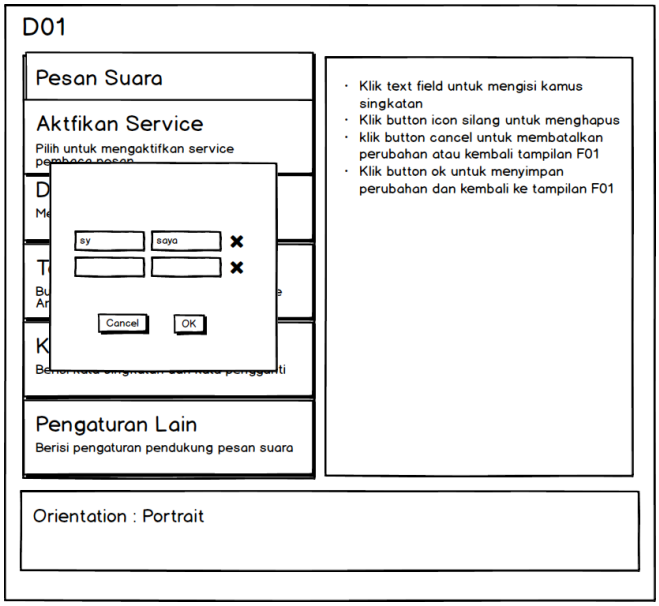
Gambar III.26 Perancangan Menu Utama

3. Perancangan Daftar Aplikasi



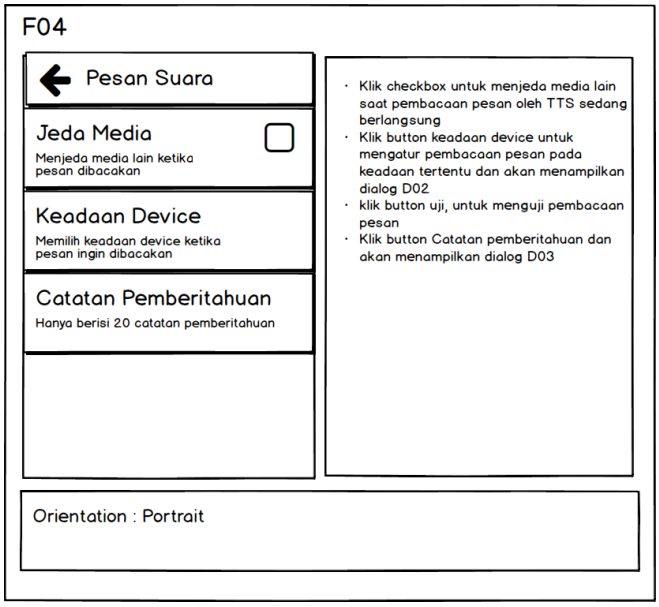
Gambar III.27 Perancangan Daftar Aplikasi

4. Perancangan *Dialog* Kamus Singkatan



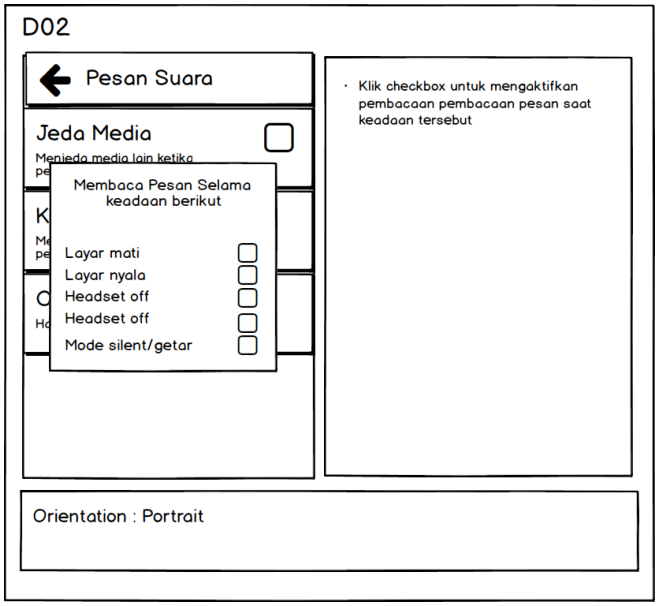
Gambar III.28 Perancangan *Dialog* Kamus Singkatan

5. Perancangan Halaman Pengaturan Lain Aplikasi



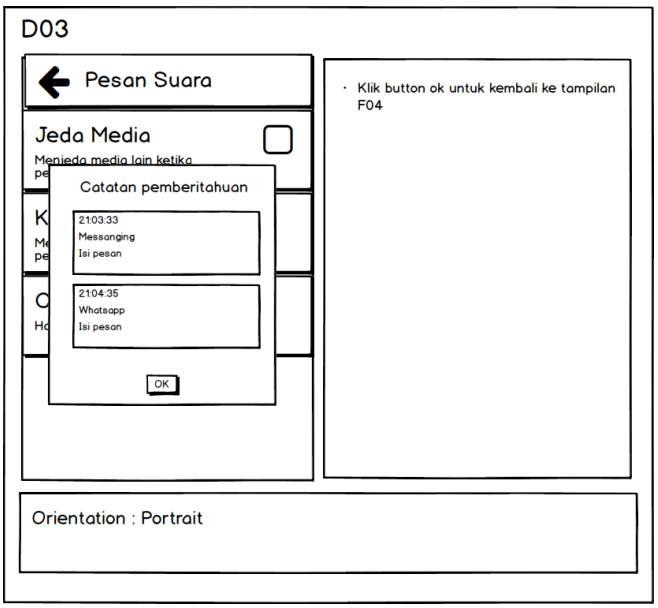
Gambar III.29 Perancangan Halaman Pengaturan Lain Aplikasi

6. Perancangan Dialog Pengaturan Keadaan Saat Membaca Pesan



Gambar III.30 Perancangan Pengaturan Keadaan Saat Membaca Pesan

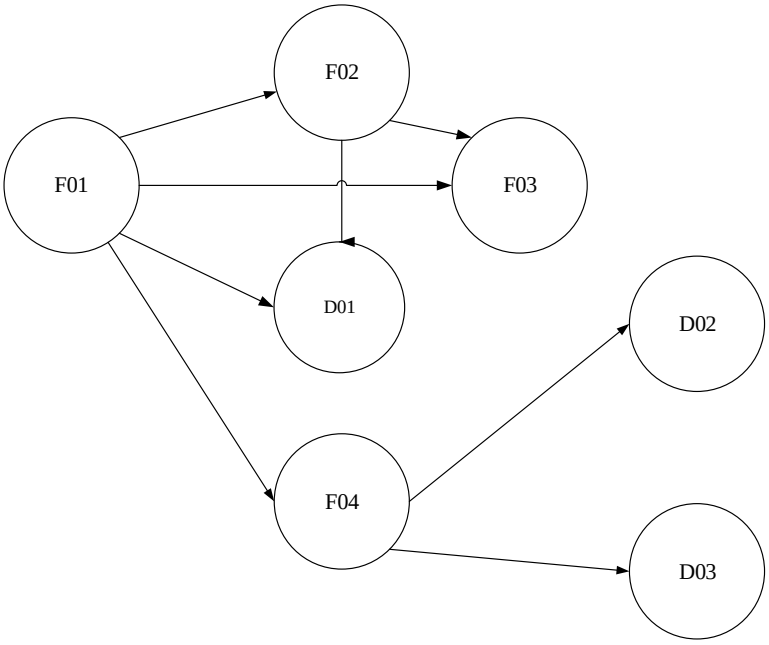
7. Perancangan Catatan Pemberitahuan



Gambar III.31 Perancangan Catatan Pemberitahuan

3.2.3 Jaringan Semantik

Tahap selanjutnya yaitu membuat jaringan semantik, yaitu penggambaran grafis yang menunjukan hubungan antar berbagai objek. Adapan jaringan semantik yang menggambarkan kontrol sistem yang akan dibangun adalah sebagai berikut:



Gambar III.32 Jaringan Semantik