BAB IV

IMPLEMENTASI DAN EVALUASI

4.1. Implementasi

Untuk mengimplementasikan dan menjalankan Aplikasi Penunjuk Waktu dan Lokasi Untuk Penyandang Tunanetra, dibutuhkan perangkat keras dan perangkat lunak dengan spesifikasi tertentu agar aplikasi yang telah dibuat dapat dijalankan dengan baik. Adapun kebutuhan perangkat keras dan lunak adalah sebagai berikut.

4.1.1. Kebutuhan Perangkat Keras

Aplikasi ini dapat dijalankan pada perangkat bergerak (*smartphone*) Android. Adapun spesifikasi minimal yang harus dimiliki adalah:

1. *Support* HSDPA dan Wifi
2. *Internal Memory* 50MB
3. CPU 1GHz
4. RAM 50MB
5. Layar *touchscreen*
6. GPS *receiver*
7. *Loudspeaker*

4.1.2. Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang dibutuhkan untuk dapat mengembangkan aplikasi ini adalah:

1. Android versi 2.3 (*gingerbread*)
2. Android SDK
3. Eclipse Indigo *Service Release* 2
4. Google *Voice Search*

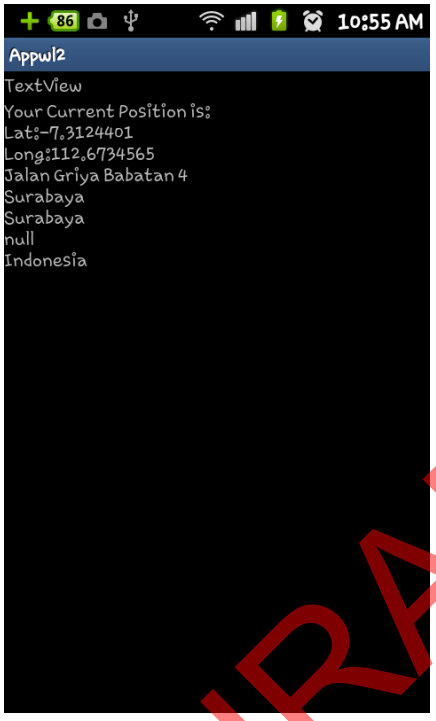
4.2. Pembuatan Program

Aplikasi Penunjuk Waktu dan Lokasi Bagi Penyandang Tunanetra ini dibangun dengan menggunakan Android SDK dan *text editor* Eclipse Indigo *Service Release* 2. Kode program disimpan dalam bentuk file dengan ekstensi *.java untuk *source code* dan *.xml untuk menyimpan *style* tampilan layar. Kemudian kedua file tersebut di-*compile* dengan menggunakan Android SDK. Kemudian kedua file tersebut dibuat menjadi satu *package* file dengan ekstensi *.apk. File *.apk ini yang nantinya dipakai untuk meng-*install* dan menjalankan aplikasi tersebut di dalam *smartphone* Android.

4.3. Implementasi Sistem

Setelah kebutuhan sistem terpenuhi, maka langkah selanjutnya adalah mengimplementasikan rancangan sistem ke dalam aplikasi. Aplikasi ini terbagi dalam beberapa modul, yaitu modul penunjuk waktu yang terbagi menjadi dua bagian ,yaitu penunjuk waktu jam dan penunjuk waktu tanggal, kemudian modul penunjuk lokasi, dan modul untuk menambah alarm.

4.3.1. Implementasi Tampilan Utama



Gambar 4.1 Tampilan Utama

Pada saat aplikasi dijalankan tampilan yang muncul adalah seperti gambar 4.1 diatas. Tampilan ini tidak banyak menampilkan grafis, karena digunakan untuk pengguna tunanetra. Maka interaksi yang digunakan adalah berupa sentuhan dan suara.

Tampilan utama ini menerapkan sistem *gesture detector*. Sistem ini berfungsi untuk mendeteksi gerakan jari dari pengguna. Gerakan jari tersebut kemudian digunakan sebagai dasar perintah kerja bagi aplikasi. Jenis gerakan yang dideteksi adalah keatas, kebawah, kesamping kiri dan kanan, serta *double tap*.

Selain *gesture detector* fitur lain yang diterapkan adalah penggunaan *text to speech*, yang digunakan sebagai media *output* suara.

```
@Override
public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX,
  float velocityY) {

 final float xDistance = Math.abs(e1.getX() - e2.getX());
 final float yDistance = Math.abs(e1.getY() - e2.getY());

 if(xDistance > this.swipe_Max_Distance || yDistance > this.swipe_Max_Distance)
  return false;

 velocityX = Math.abs(velocityX);
 velocityY = Math.abs(velocityY);
       boolean result = false;

 if(velocityX > this.swipe_Min_Velocity && xDistance > this.swipe_Min_Distance){
  if(e1.getX() > e2.getX()) // right to left
   this.listener.onSwipe(SWIPE_LEFT);
  else
   this.listener.onSwipe(SWIPE_RIGHT);

  result = true;
 }
 else if(velocityY > this.swipe_Min_Velocity && yDistance > this.swipe_Min_Distance){
  if(e1.getY() > e2.getY()) // bottom to up
   this.listener.onSwipe(SWIPE_UP);
  else
   this.listener.onSwipe(SWIPE_DOWN);

  result = true;
 }

  return result;
}


 @Override
 public boolean onDoubleTap(MotionEvent arg0) {
  this.listener.onDoubleTap();;
  return true;
 }
```

Gambar 4.2 Proses Gesture Detector

Pada gambar 4.2 dapat dilihat rangkaian kode dari proses *gesture detector*, yang berisikan perintah-perintah untuk mendeteksi gerakan jari pengguna. Fungsi onFling() berfungsi untuk mendeteksi ada tidaknya sentuhan pada layar *smartphone*. Dengan menghitung selisih nilai xDistance dan yDistance, maka dapat ditentukan jenis gerakan jari oleh pengguna. Sedangkan fungsi DoubleTap() berfungsi untuk mendeteksi apakah ada sentuhan *double tap* pada layar oleh pengguna.

```
public void onInit(int status) {
 // TODO Auto-generated method stub
  // status can be either TextToSpeech.SUCCESS or TextToSpeech.ERROR.
    if (status == TextToSpeech.SUCCESS) {
        // Set preferred language to US english.
        // Note that a language may not be available, and the result will indicate this.
        int result = tts.setLanguage(Locale.US);
        // Try this someday for some interesting results.
        // int result mTts.setLanguage(Locale.FRANCE);
        if (result == TextToSpeech.LANG_MISSING_DATA ||
                result == TextToSpeech.LANG_NOT_SUPPORTED) {
            // Language data is missing or the language is not supported.
            Log.e(TAG, "Language is not available.");
        } else {
            // Check the documentation for other possible result codes.
            // For example, the language may be available for the locale,
            // but not for the specified country and variant.

            // The TTS engine has been successfully initialized.
            // Allow the user to press the button for the app to speak again.
            //btn.setEnabled(true);
            // Greet the user.
            //sayHello();
        }
    } else {
        // Initialization failed.
        Log.e(TAG, "Could not initialize TextToSpeech.");
    }
}
```

Gambar 4.3 Proses Text To Speech

Gambar 4.3 menjelaskan rangkaian kode yang digunakan untuk memanggil *text to speech*. Kode tersebut akan dijalankan segera setelah aplikasi mulai menyala.

Agar aplikasi dapat lebih mudah digunakan, maka aplikasi harus dapat berjalan sejak *smartphone* nyalakan. Pada gambar 4.4 merupakan kode yang terdapat pada AndroidManifest.xml, yang digunakan sebagai penerima sinyal bahwa *smartphone* selesai *booting* atau telah dinyalakan.

```
<receiver android:enabled="true" android:name=".BootUpReceiver"
android:permission="android.permission.RECEIVE_BOOT_COMPLETED">

<intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED" />
        <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
</receiver>
```

Gambar 4.4 Boot Up Receiver XML

Setelah membuat *receiver* pada AndroidManifest.xml, maka langkah selanjutnya adalah membuat *class* penerima dari *receiver* tersebut.

```
public class BootUpReceiver extends BroadcastReceiver{

    @Override
    public void onReceive(Context context, Intent intent) {
        // TODO Auto-generated method stub
        Intent i = new Intent(context, Appwl2Activity.class);
        i.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        context.startActivity(i);
    }
```

Gambar 4.5 Class BootUpReceiver

Untuk menjalankan fungsi-fungsi tertentu, seperti BootUpReceiver, GPS, dan menambah alarm. Dibutuhkan izin-izin tertentu yang disebut dengan *uses-permission*. Kode tersebut dimasukkan ke dalam AndroidManisfest.xml

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>"
<uses-permission android:name="android.permission.READ_CALENDAR" />
<uses-permission android:name="android.permission.WRITE_CALENDAR" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
```

Gambar 4.6 Uses Permission

Pada gambar 4.6 merupakan macam-macam *uses permission* yang digunakan oleh aplikasi. *Permission internet* berarti memperbolehkan aplikasi untuk dapat menggunakan jaringan *internet, permission access fine location* memperbolehkan aplikasi untuk menggunakan perangkat keras GPS, *permission read calendar* memperbolehkan aplikasi mengambil data kalender, *permission write calendar* memperbolehkan aplikasi untuk memasukkan data ke dalam kalender, dan *receive boot completed* berguna untuk aplikasi agar dapat menyala otomatis setelah *smartphone* selesai *booting up*(dinyalakan)

4.3.2. Implementasi Penunjuk Waktu

Penunjuk waktu pada aplikasi ini dapat diaktifkan dengan cara melakukan gerakan jari ke kiri untuk penunjuk waktu tanggal dan gerakan jari ke atas untuk penunjuk waktu jam.

```
Calendar cal=Calendar.getInstance();
SimpleDateFormat date_date = new SimpleDateFormat("d");
SimpleDateFormat month_date = new SimpleDateFormat("MMMMMMMMM");
SimpleDateFormat year_date = new SimpleDateFormat("yyyy");
String date_name = date_date.format(cal.getTime());
String month_name = month_date.format(cal.getTime());
String year_name = year_date.format(cal.getTime());
//String mydate = java.text.DateFormat.getDateTimeInstance().format(Calendar.getInstance().getTime());
String mydate = date_name + " " + month_name + " " + year_name;
tv.setText(mydate);
tts.speak(mydate, TextToSpeech.QUEUE_ADD, null);
```

Gambar 4.7 Proses Penunjuk Waktu Tanggal

Pada gambar 4.7 merupakan kode-kode yang digunakan untuk menjalankan proses pengambilan nilai tanggal sistem, kemudian dirubah menjadi bentuk text yang kemudian akan dibacakan dalam bentuk suara melalui TTS (*Text To Speech*).

```
String mytime=java.text.DateFormat.getTimeInstance().format(Calendar.getInstance().getTime());
tv.setText(mytime);
tts.speak(mytime, TextToSpeech.QUEUE_ADD, null);
```

Gambar 4.8 Proses Penunjuk Waktu Jam

Agar fungsi pengingat waktu jam dapat berjalan maka diperlukan *thread*, yang digunakan untuk mengecek waktu dalam satuan menit secara terus-menerus. Dan kemudian akan menjalankan fungsi penunjuk waktu jam, jika masuk dalam periode waktu 15 menit. Kode-kode program yang dibutuhkan dalam menjalankan proses *thread* tersebut dapat dilihat pada gambar 4.9.

```
Thread runner;
final Runnable updater = new Runnable(){
   public void run(){
       updateClockValues();
   };
};
final Handler mHandler = new Handler();


public void run() {
 // TODO Auto-generated method stub
    while (runner != null)
      {
          try
          {
           Thread.sleep(1000);
           Log.i("Tick", "Tock");
          } catch (InterruptedException e) { };
          mHandler.post(updater);
      }
}
private void updateClockValues() {
    currentTime = Calendar.getInstance().getTime();
    tvjam.setText(currentTime.getHours()+":"+currentTime.getMinutes()+":"+currentTime.getSeconds());
    if(currentTime.getMinutes()%2==0 && currentTime.getSeconds()==0){
        tts.speak(currentTime.getHours()+":"+currentTime.getMinutes(), TextToSpeech.QUEUE_ADD, null);
    }
}
```

Gambar 4.9 Proses Thread

Pada gambar 4.9 merupakan kode proses *thread*. Pada fungsi *run()* terlihat bahwa proses dijalankan dengan periode 1000 *milisecond*, yang dapat diartikan sebagai 1 detik. Dalam tiap periode 1 detik, *thread* akan menjalankan fungsi *updateClockValues()* yang berfungsi untuk mengecek dan menjalankan fungsi pengingat waktu tiap 15 menit. Variabel *updater* berfungsi sebagai *trigger* untuk menjalankan *updateClockValues()*, ketika perode 1 detik tercapai.

4.3.3. Implementasi Penunjuk Lokasi

Penunjuk Lokasi dapat diaktifkan dengan cara menggerakkan jari ke arah kanan pada layar *smartphone*. Dalam penggunaan fitur ini dibutuhkan perangkat GPS yang sudah terdapat di dalam *smartphone* dan koneksi internet yang stabil. Koneksi internet dibutuhkan untuk menjalankan perangkat A-GPS dan pengambilan alamat dari lokasi.

```
LocationManager locationManager;
String context = Context.LOCATION_SERVICE;
locationManager = (LocationManager)getSystemService(context);

Criteria criteria = new Criteria();
criteria.setAccuracy(Criteria.ACCURACY_FINE);
criteria.setAltitudeRequired(false);
criteria.setBearingRequired(false);
criteria.setCostAllowed(true);
criteria.setPowerRequirement(Criteria.POWER_LOW);
String provider = locationManager.getBestProvider(criteria, true);

Location location = locationManager.getLastKnownLocation(provider);
updateWithNewLocation(location);

locationManager.requestLocationUpdates(provider, 2000, 10, locationListener);
```

Gambar 4.10 Proses Pemanggilan Location Manager

Gambar 4.10 menjelaskan kode yang diperlukan untuk pemanggilan fungsi location manager, yang akan menjalankan fungsi GPS dan sinkronisasi dengan google *map*. Tingkat akurasi dapat diatur pada tingkat akurasi yang terbaik dengan cara menggunakan perintah *setAccuracy(Criteria.ACCURACY_FINE). setCostAllowed(true)* diatur pada *true* agar aplikasi diperbolehkan menggunakan fasilitas A-GPS, yang penggunaannya menggunakan fasilitas *internet* dan dapat dikenakan biaya. Digunakannya A-GPS bertujuan untuk mempercepat proses penguncian lokasi oleh GPS. *setPowerRequirement(Criteria.POWER_LOW)* diatur dengan *power low* agar aplikasi tetap dapat menggunakan GPS walaupun baterai sudah lemah.

```
private final LocationListener locationListener = new LocationListener() {
    public void onLocationChanged(Location location) {
        updateWithNewLocation(location);
    }

    public void onProviderDisabled(String provider) {
        updateWithNewLocation(null);
    }

    public void onProviderEnabled(String provider) { }

    public void onStatusChanged(String provider, int status, Bundle extras) { }
};
```

Gambar 4.11 Proses Deteksi Perubahan Lokasi

Gambar 4.11 merupakan kode yang berisi perintah untuk mendeteksi perubahan lokasi. Jika ada perubahan lokasi yang terdeteksi maka fungsi *updateWithNewLocation()* akan dipanggil untuk mendapatkan nilai lokasi yang baru. Fungsi *onProviderDisabled()* digunakan untuk mendeteksi apakah *provider* atau internet tersedia, jika tidak tersedia maka fungsi *updateWithNewLocation* tidak dijalankan.

```
private void updateWithNewLocation(Location location) {
    // TODO Auto-generated method stub
    String latLongString;
    String addressString = "No address found";

    if(location != null){
        double lat = location.getLatitude();
        double lng = location.getLongitude();
        latLongString = "Lat:" + lat + "\nLong:" + lng;

        double latitude = location.getLatitude();
        double longitude = location.getLongitude();
        Geocoder gc = new Geocoder(this, Locale.getDefault());
        try{
            List<Address> addresses = gc.getFromLocation(latitude, longitude, 1);
            StringBuilder sb = new StringBuilder();

            if(addresses.size() > 0 ){
                Address address = addresses.get(0);

                for(int i = 0; i < address.getMaxAddressLineIndex(); i++)
                    sb.append(address.getAddressLine(i)).append("\n");
                    sb.append(address.getLocality()).append("\n");
                    sb.append(address.getPostalCode()).append("\n");
                    sb.append(address.getCountryName());
            }
            addressString = sb.toString();
        } catch (IOException e){ }
    } else {
        latLongString = "No Location Found";
    }
    myLocationText.setText("Your Current Position is:\n" + latLongString + "\n" + addressString);
    myLocation="Your Current Position is:\n" + addressString;
}
```

Gambar 4.12 Proses Pengambilan Alamat Lokasi

Gambar 4.12 merupakan serangkaian kode yang berguna untuk menjalankan proses pengambilan nilai *latitude* dan *longitude*, kemudian mensinkronisasikannya dengan Google *map* untuk mendapatkan nama lokasi dari kedua nilai tersebut.

4.3.4. Implementasi Menambah Alarm

Untuk menambah alarm, pengguna dapat melakukan *double tap*. Setelah melakukan double tapi maka aplikasi akan meminta pengguna untuk memasukkan jam alarm. Untuk memasukkan jam alarm dengan cara menggerakkan jari ke atas dan kebawah untuk mengaturnya. Setelah memasukkan jam, maka selanjutnya diminta memasukkan menit alarm, kemudian tahun alarm, bulan alarm, dan tanggal alarm. Yang terakhir harus dimasukkan adalan pesan alarm. Pesan alarm dimasukkan dengan menggunakan fitur *voice recognition*.

Gambar 4.13 Tampilan Voice Recognition

```
private void startVoiceRecognitionActivity()
{
    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
            RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
    intent.putExtra(RecognizerIntent.EXTRA_MAX_RESULTS, 5);
    startActivityForResult(intent, REQUEST_CODE);
    final ToneGenerator tg = new ToneGenerator(AudioManager.STREAM_NOTIFICATION, 100);
    tg.startTone(ToneGenerator.TONE_PROP_BEEP);
}
```

Gambar 4.14 Proses Pemanggilan Voice Recognition

Pada gambar 4.14 merupakan kode program yang digunakan untuk memanggil proses *voice recognition* dengan menggunakan intent. *Voice recognition* diseting untuk menggunakan bahasa sesuai dengan bahasa yang digunakan pada setelan *smartphone*. Sedangkan *tone generator* digunakan untuk menghasilkan bunyi beep ketika *voice recognition intent* dijalankan.

```
public void onDoubleTap() {
   Toast.makeText(this, "Double Tap", Toast.LENGTH_SHORT).show();

   mode +=1;


   if (mode==1) {
       tts.speak("What hours do you want to set the alarm?", TextToSpeech.QUEUE_ADD, null);
       counter=0;
   }
   if (mode==2){
       jam = counter;
       tts.speak(jam.toString(), TextToSpeech.QUEUE_ADD, null);
       tts.speak("What minutes do you want to set the alarm?", TextToSpeech.QUEUE_ADD, null);
       counter = 0;
   }
   if (mode==3){
       menit=counter;
       tts.speak(menit.toString(), TextToSpeech.QUEUE_ADD, null);
       tts.speak("What years do you want to set the alarm?", TextToSpeech.QUEUE_ADD, null);
       counter = Calendar.getInstance().get(Calendar.YEAR);
   }
   if (mode==4) {
       tahun=counter;
       tts.speak(tahun.toString(), TextToSpeech.QUEUE_ADD, null);
       tts.speak("What month do you want to set the alarm?", TextToSpeech.QUEUE_ADD, null);
       counter = Calendar.getInstance().get(Calendar.MONTH);
   }
   if (mode==5) {
       bulan=counter;
       tts.speak(bulan.toString(), TextToSpeech.QUEUE_ADD, null);


       if(bulan==1||bulan==3||bulan==5||bulan==7||bulan==8||bulan==10||bulan==12){
           maxdate=31;
       }
       if(bulan==4||bulan==6||bulan==9||bulan==11){
           maxdate=30;
       }
       if(bulan==2){
           maxdate=28;
       }
       if(bulan==2 && tahun%4==0){
           maxdate=29;
       }
       tts.speak("What date do you want to set the alarm?", TextToSpeech.QUEUE_ADD, null);
       counter=Calendar.getInstance().get(Calendar.DATE);
   }

   if (mode==6) {
       tanggal=counter;
       tts.speak(tanggal.toString(), TextToSpeech.QUEUE_ADD, null);
       tts.speak("What message do you want to set for the alarm?", TextToSpeech.QUEUE_ADD, null);
       mode=0;
       startVoiceRecognitionActivity();
   }
   if (mode==7) {
     mode=0;
   }
   tv.setText("Double Tap");
}
```

Gambar 4.15 Proses Input Alarm

Proses *input* alarm akan dimulai ketika terdeteksi *double tap* pada layar oleh fungsi *onDoubleTap()*. Pada fungsi tersebut terdapat 7 *mode* yang berfungsi agar pengguna dapat memasukkan data alarm secara berurutan dengan menggunakan media suara dengan menggunakan gabungan fitur *text to speech*, *voice recognition* dan *gesture detector*..

Pada gambar selanjutnya, yaitu gambar 4.16 merupakan serangkaian kode yang digunakan untuk memasukkan data alarm pada *database calendar*. *Calendar_id* merupakan data numerik yang dimasukkan sebagai nomor identifikasi data *calendar*. *Title* merupakan data yang berisikan judul dari alarm, *addInfo* merupakan data yang berisikan informasi tambahan atau deskripsi dari data alarm itu sendiri. *Places* merupakan data tempat atau lokasi kegiatan. *Status* merupakan data *Boolean* yang berfungsi sebagai penanda aktif atau tidaknya alarm tersebut, *startDate* dan *endDate* merupakan data tanggal yang berisikan dimulai dan berakhirnya alarm. Sedangkan *needReminder* berfungsi jika bernilai benar maka alarm akan diaktifkan 5 menit sebelum jadwalnya. Kemudian *mailService* jika bernilai benar, maka data alarm akan disinkronisasikan dengan *google calendar*, sehingga data alarm juga bisa diakses melalui *google account.*

```
public static long pushAppointmentsToCalender(Activity curActivity, String title, String addInfo, String place, int status,
        long startDate, boolean needReminder, boolean needMailService) {
    /***************** Event: note(without alert) ******************/

    String eventUriString = "content://com.android.calendar/events";
    ContentValues eventValues = new ContentValues();

    eventValues.put("calendar_id", 1); // id, We need to choose from
                                       // our mobile for primary
                                       // its 1
    eventValues.put("title", title);
    eventValues.put("description", addInfo);
    eventValues.put("eventLocation", place);

    long endDate = startDate + 1000 * 60 * 60; // For next 1hr

    eventValues.put("dtstart", startDate);
    eventValues.put("dtend", endDate);

    // values.put("allDay", 1); //If it is bithday alarm or such
    // kind (which should remind me for whole day) 0 for false, 1
    // for true
    eventValues.put("eventStatus", status); // This information is
    // sufficient for most
    // entries tentative (0),
    // confirmed (1) or canceled
    // (2):
    eventValues.put("visibility", 3); // visibility to default (0),
                                      // confidential (1), private
                                      // (2), or public (3):
    eventValues.put("transparency", 0); // You can control whether
                                        // an event consumes time
                                        // opaque (0) or transparent
                                        // (1).
    eventValues.put("hasAlarm", 1); // 0 for false, 1 for true

    Uri eventUri = curActivity.getApplicationContext().getContentResolver().insert(Uri.parse(eventUriString), eventValues);
    long eventID = Long.parseLong(eventUri.getLastPathSegment());
if (needReminder) {
    /***************** Event: Reminder(with alert) Adding reminder to event ******************/

    String reminderUriString = "content://com.android.calendar/reminders";

    ContentValues reminderValues = new ContentValues();

    reminderValues.put("event_id", eventID);
    reminderValues.put("minutes", 5); // Default value of the
                                      // system. Minutes is a
                                      // integer
    reminderValues.put("method", 1); // Alert Methods: Default(0),
                                     // Alert(1), Email(2),
                                     // SMS(3)

    Uri reminderUri = curActivity.getApplicationContext().getContentResolver().insert(Uri.parse(reminderUriString),
            reminderValues);
}

/***************** Event: Meeting(without alert) Adding Attendies to the meeting ******************/

if (needMailService) {
    String attendeuesesUriString = "content://com.android.calendar/attendees";

    /********
     * To add multiple attendees need to insert ContentValues multiple
     * times
     ***********/
    ContentValues attendeesValues = new ContentValues();

    attendeesValues.put("event_id", eventID);
    attendeesValues.put("attendeeName", "xxxxx"); // Attendees name
    attendeesValues.put("attendeeEmail", "yyyy@gmail.com");// Attendee
                                                                                    // E
                                                                                    // mail
                                                                                    // id
        attendeesValues.put("attendeeRelationship", 0); // Relationship_Attendee(1),
                                                        // Relationship_None(0),
                                                        // Organizer(2),
                                                        // Performer(3),
                                                        // Speaker(4)
        attendeesValues.put("attendeeType", 0); // None(0), Optional(1),
                                                // Required(2), Resource(3)
        attendeesValues.put("attendeeStatus", 0); // NOne(0), Accepted(1),
                                                  // Decline(2),
                                                  // Invited(3),
                                                  // Tentative(4)

        Uri attendeuesesUri = curActivity.getApplicationContext().getContentResolver().insert(Uri.parse(attendeuesesUriString),
                attendeesValues);
    }

    return eventID;

 }
```

Gambar 4.16 Proses Simpan Alarm

4.4. Evaluasi Sistem

Setelah melakukan implementasi sistem, maka langkah selanjutnya adalah uji coba dan evaluasi dengan tujuan untuk mengetahui apakah aplikasi yang dibuat dapat menghasilkan *output* yang diharapkan. Pengujian dilakukan dengan menggunakan *smartphone* Android dengan sistem operasi Android versi 2.3.6 (Gingerbread.DXML3).

4.4.1. Uji Coba Fungsi Aplikasi

Pengujian ini dilakukan untuk mengetahui apakah fungsi-fungsi yang ada pada aplikasi telah berjalan dengan baik atau tidak. Fungsi-fungsi yang diujikan adalah:

A. Fungsi Menjalankan Aplikasi Setelah Booting Up

Uji Coba ini bertujuan untuk mengetahui apakah fungsi menjalankan aplikasi setelah *booting up* telah berjalan dengan baik. Pada uji ini *smartphone* berada pada keadaan mati untuk pertama kali, kemudian dinyalakan. Saat proses *booting up* selesai, maka aplikasi akan dijalankan.

Tabel 4.1 Hasil Uji Coba Fungsi Menjalankan Aplikasi Setelah Booting Up

Test Case ID	Tujuan	Input	Output yang diharapkan	Status
1	Menjalankan aplikasi setelah *booting up*	Menyalakan *smartphone*	Aplikasi berjalan setelah selesai *booting up*	Sesuai

B. Fungsi Penunjuk Waktu

Uji Coba ini bertujuan untuk mengetahui apakah fungsi penunjuk waktu dapat dilakukan oleh aplikasi. Pada uji coba pertama dilakukan uji coba terhadap

fungsi penunjuk waktu jam, kemudian dilanjutkan dengan uji coba pengingat waktu jam dan pengingat waktu tanggal. Ketika jari digerakkan kearah kiri maka aplikasi akan mengambil tanggal sistem dan melalui fitur *text to speech*, tanggal tersebut akan dibacakan dengan suara. Ketika jari digerakkan ke atas maka aplikasi akan mengambil jam sistem dan melalui fitur *text to speech* akan dibacakan. Sedangkan pada pengingat waktu tidak ada *input* dari pengguna, fungsi tersebut akan berjalan otomatis tiap periode 15 menit.

Tabel 4.2 Hasil Uji Coba Fungsi Penunjuk Waktu

Test Case ID	Tujuan	Input	Output yang diharapkan	Status
2	Meminta informasi waktu “jam”	Menggerakkan jari ke kiri pada layar	Muncul suara berisikan informasi waktu jam	Sesuai
3	Meminta informasi waktu “jam”	-	Muncul suara berisikan informasi waktu jam dengan periode 15 menit	Sesuai
4	Meminta informasi waktu “tanggal”	Menggerakkan jari ke kanan pada layar	Muncul suara berisikan informasi waktu tanggal	Sesuai

C. Fungsi Penunjuk Lokasi

Uji coba fungsi penunjuk lokasi ini bertujuan untuk mengetahui apakah aplikasi dapat menjalankan fungsi ini dengan baik. Proses pengambilan *altitude* dan *longitude* telah dimulai sejak aplikasi pertama kali dinyalakan. Setelah nilai *altitude* dan *longitude* didapatkan, aplikasi akan mensinkronkan data tersebut dengan google *map*. Ketika aplikasi mendeteksi ada gerakan jari kearah kanan,

maka aplikasi akan mengambil nilai alamat yang didapatkan dari google map dan mengubahnya menjadi bentuk suara dengan bantuan *text to speech.*

Tabel 4.3 Hasil Uji Coba Fungsi Penunjuk Lokasi

Test Case ID	Tujuan	Input	Output yang diharapkan	Status
5	Meminta informasi lokasi	Menggerakkan jari ke kanan pada layar	Muncul suara berisikan informasi lokasi	Sesuai

D. Fungsi Menambah Alarm

Uji coba fungsi ini untuk mengetahui apakah aplikasi dapat menjalankan fungsi menambah alarm. Proses ini dimulai ketika pengguna melakukan *double tap* pada layar. Kemudian pengguna diarahkan untuk memasukkan jam, menit, tahun, bulan, tanggal, dan pesan alarm. Pesan alarm akan dimasukkan dengan menggunakan fitur *voice recognition*.

Tabel 4.4 Hasil Uji Coba Fungsi Menambah Alarm

Test Case ID	Tujuan	Input	Output yang diharapkan	Status
6	Menambahkan alarm	*Double tap* pada layar dan pengguna dapat memasukkan pesan suara	*Voice Recognizer* tampil. Alarm pada Android *Calendar* bertambah	Sesuai

E. Fungsi Bantuan

Uji coba ini betujuan untuk mengetahui apakah aplikasi dapat menjalankan fungsi bantuan dengan baik. Dimulai ketika pengguna menggerakkan jari kearah bawah pada layar. Aplikasi akan memperdengarkan cara penggunaan aplikasi.

Tabel 4.5 Hasil Uji Coba Fungsi Bantuan

Test Case ID	Tujuan	Input	Output yang diharapkan	Status
7	Meminta Bantuan	Menggerakkan jari ke bawah pada layar	Suara informasi bantuan berisikan informasi fungsi aplikasi	Sesuai

F. Uji Coba GPS Pada Aplikasi

Uji coba ini bertujuan untuk mengetahui tingkat akurasi dari GPS. Uji coba yang dilakukan meliputi uji coba tingkat cepat tangkap GPS, uji coba tingkat akurasi alamat GPS *Map*, dan uji coba tingkat akurasi GPS.

Tabel 4.6 Hasil Uji Coba Tingkat Cepat Tangkap GPS

Test Case ID	Tempat Uji Coba	Waktu Tangkap
8	Lapangan tanpa halangan	5 detik
9	Lapangan dengan halangan gedung di sekitar	10detik
10	Di dalam gedung 1 tingkat	30 detik
11	Di dalam gedung 2 tingkat	1 menit
12	Di dalam gedung 3 tingkat	-

Tabel 4.6 bertujuan untuk mengetahui cepat tangkap posisi oleh GPS *receiver* dengan berbagai kondisi lokasi uji coba. Dari tabel di atas dapat diketahui bahwa GPS tidak dapat mengunci posisi jika berada di dalam gedung lebih dari 2 tingkat.

Tabel 4.7 Hasil Uji Coba Tingkat Akurasi Alamat Google Map

Test Case ID	Tempat Uji Coba	Terdeteksi
13	Jalan Kedung Baruk Raya	Jalan Kedung Baruk Raya
14	Jalan Raya Darmo	Jalan Raya Darmo
15	Jalan Griya Babatan 4	Jalan Griya Babatan 4
16	Jalan Jagir Wonokromo	Jalan Jagir Wonokromo
17	Jalan Citra Raya	Jalan Citra Raya

Pada tabel 4.7 berisikan data hasil uji coba akurasi pengambilan alamat oleh Google *map* dengan alamat yang sebenarnya. Dari seluruh hasil uji coba yang dilakukan, didapatkan bahwa semua alamat terdeteksi sesuai dengan yang sebenarnya.

Tabel 4.8 Hasil Uji Coba Tingkat Akurasi GPS

Test Case ID	Tempat Uji Coba		Terdeteksi		Selisih
	Latitude	Longitude	Latitude	Longitude	
18	-7.311002	112.781232	-7.311016	112.78122	5 m
19	-7.283504	112.740062	-7.283492	112.740065	3 m
20	-7.312395	112.673371	-7.312338	112.67334	6 m
21	-7.304417	112.753922	-7.304434	112.753976	7 m
22	-7.298583	112.668358	-7.298569	112.668423	12 m

Berdasarkan tabel 4.8 yang berisikan data hasil uji coba akurasi GPS dapat diketahui bahwa ada selisih jarak lokasi yang dideteksi oleh GPS *receiver* dengan lokasi yang sebenarnya. Selisih paling besar dari uji coba sebanyak 12 meter dan selisih paling sedikit sebanyak 3 meter.

G. Uji Coba Tingkat Akurasi Voice Recognition

Pada uji coba ini dilakukan pengujian terhadap tingkat akurasi dari fitur *voice recognition* dalam berbagai kondisi tingkat kebisingan. Pengujian ini dilakukan dengan memberikan suara gangguan dalam tingkat kebisingan tertentu, pada saat proses *voice recognition* dijalankan.

Tabel 4.9 Hasil Uji Coba Tingkat Akurasi Voice Recognition

Test Case ID	Tingkat Kebisingan	Terdeteksi	Keterangan Suara
18	20 db	Sesuai	Daun bergesekan
23	30 db	Sesuai	Bisikan pada jarak 3 kaki
24	40 db	Sesuai	Pemukiman yang sepi
25	50 db	Sesuai	Hujan

26	60 db	Tidak Sesuai	Pembicaraan pada jarak 3 kaki
27	70 db	Tidak Sesuai	*Ringtone handphone*
28	80 db	Tidak Sesuai	Alarm mobil
29	90 db	Tidak Sesuai	Mesin pabrik
30	100 db	Tidak Sesuai	Tabrakan kendaraan

Pada tabel 4.9 yang berisikan hasil uji coba tingkat akurasi *voice recognition*, diketahui bahwa *voice recognition* akan mengalami kegagalan jika tingkat kebisingan di sekitar lokasi mencapai 60 db keatas, atau setara dengan ada orang yang berbicara pada jarak 3 kaki.

4.4.2 Uji Coba Aplikasi Pada Pengguna

Untuk mengetahui apakah aplikasi telah memenuhi atau telah sesuai dengan keinginan para pengguna tunanetra, maka dilakukan pengujian aplikasi terhadap pengguna tunanetra. Data pengguna terdapat pada tabel 4.10

Tabel 4.10 Data Pengguna

No	Nama Pengguna	Umur
1	Pengguna 1	22
2	Pengguna 2	17
3	Pengguna 3	30
4	Pengguna 4	22
5	Pengguna 5	20

Pengujian ini dilakukan secara wawancara langsung kepada 5 orang penyandang tunanetra. Identitas asli pengguna tidak dituliskan untuk menjaga hak privasi mereka. Wawancara dilakukan dengan menanyakan 6 poin pertanyaan yang berkaitan dengan fungsi-fungsi yang ada pada aplikasi. Hasil wawancara kemudian diubah menjadi berupa poin atau angka dengan ketentuan penilaian menggunakan angka 1 sampai 5. Angka 5 untuk sangat bagus, angka 4 untuk bagus, angka 3 untuk biasa saja, angka 2 untuk jelek, dan angka 1 untuk sangat

jelek. Tabel 4.11 dibawah nerupakan hasil rekapitulasi dari wawancara yang dilakukan..

Tabel 4.11 Hasil Rekapitulasi Wawancara

No	Pertanyaan	Penilaian					Rata-rata
		Pengguna 1	Pengguna 2	Pengguna 3	Pengguna 4	Pengguna 5	
1	Kemudahan sensor gerakan jari	5	4	4	5	4	4.4
2	Kejelasan suara penunjuk waktu	5	3	4	3	5	4
3	Kejelasan suara penunjuk lokasi	4	3	3	3	4	3.4
4	Kejelasan suara menambah alarm	4	4	3	3	4	3.6
5	Kemudahan menambah alarm	4	3	3	4	3	3.4
6	Kejelasan suara bantuan	5	4	5	4	4	4.4
Rata-rata keseluruhan							3.87

Pada tabel 4.11 berisikan data hasil kuesioner pada pengguna tunanetra, diketahui bahwa aplikasi secara keseluruhan mendapatkan nilai di atas normal dan mendekati baik, yaitu 3,87. Nilai tersebut didapatkan dengan merata-rata semua nilai yang diberikan oleh pengguna pada kuesioner.

4.4.3 Analisis Hasil Uji Coba Aplikasi

Berdasarkan pada hasil dari pengujian yang telah dilakukan, maka dapat disimpulkan dari hasil uji coba keseluruhan bahwa aplikasi layak untuk digunakan oleh para penyandang tunanetra. Semua fungsi aplikasi telah diuji coba dan *output* yang dihasilkan telah sesuai dengan yang diharapkan. Dari uji coba fungsi aplikasi

didapatkan bahwa GPS tidak dapat menangkap sinyal, jika berada di dalam gedung lebih dari 2 tingkat. Selain itu, diketahui juga bahwa GPS mempunyai tingkat akurasi terbaik 3 meter. Fungsi *voice recognition* hanya dapat berjalan dengan baik jika tingkat kebisingan kurang dari 60 db. Pada uji coba pada pengguna didapatkan nilai rata-rata keseluruhan dari aplikasi sebanyak 3.87 yang termasuk di atas normal, dan mendekati kriteria baik. Pada nilai rata-rata tiap fungsi aplikasi, kemudahan menambah alarm dan kejelasan suara penunjuk lokasi mendapatkan nilai yang terendah, yaitu sebanyak 3.4. Sedangkan nilai tertinggi, yaitu sebanyak 4.4 diperoleh oleh kemudahan sensor gerakan jari dan kejelasan suara bantuan. Kejelasan suara penunjuk waktu mendapatkan nilai sebanyak 4, sedangkan kejelasan suara menambah alarm hanya mendapatkan nilai sebanyak 3.6.