

BAB III

PERANCANGAN DAN IMPLEMENTASI

3.1. Gambaran Umum

Aplikasi Hallo Depok merupakan aplikasi Location Based Service (LBS). Location Based Service adalah layanan informasi berbasis geografis yang dapat diakses oleh mobile device dengan memberikan informasi suatu lokasi yang dibutuhkan. Aplikasi ini dibuat untuk memudahkan masyarakat pengguna Android dalam mencari suatu tempat penting di kota Depok. Sehingga dengan aplikasi ini kita dapat mengetahui lokasi suatu tempat penting dan informasi yang terkait seperti alamat, nomor telepon, jarak lokasi dengan pengguna, jam buka dan online map. Tahap-tahap pembuatan aplikasi Hallo Depok berbasis Android sebagai berikut :

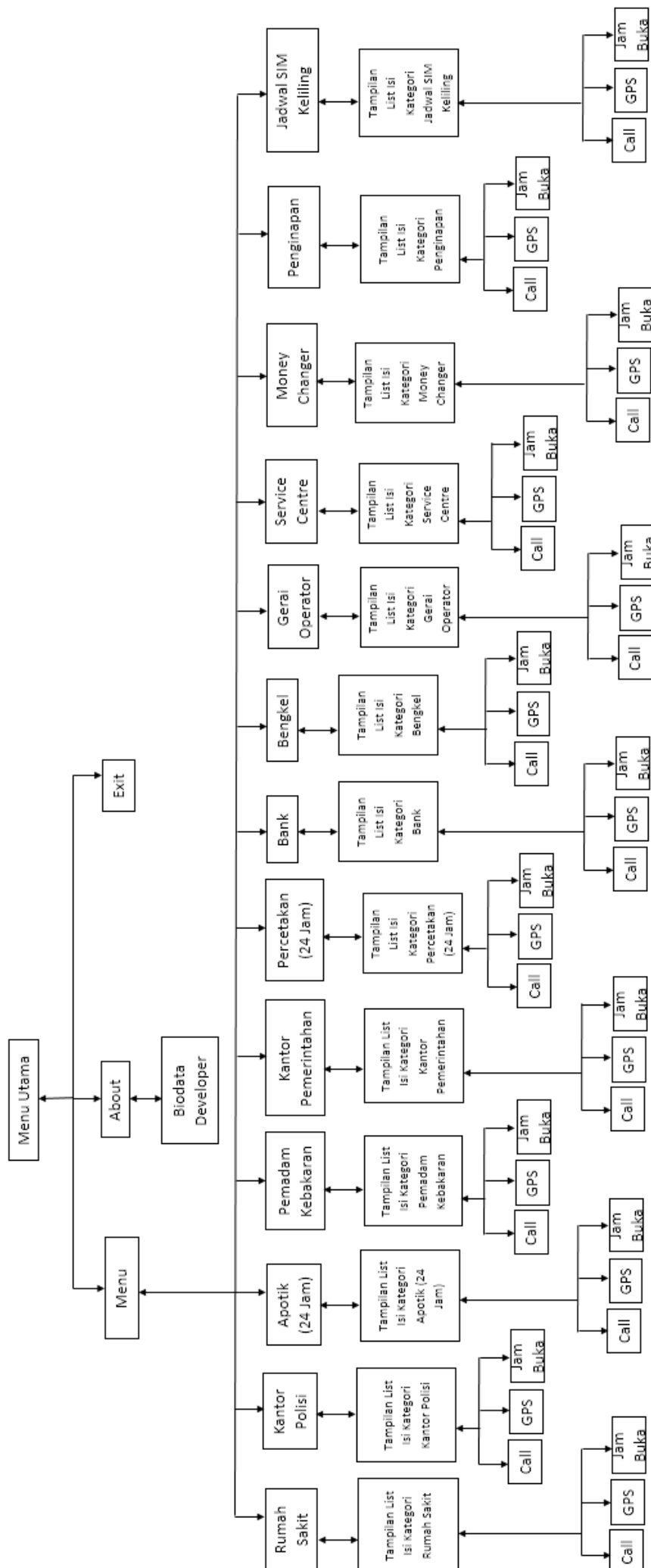
1. Merancang konsep desain interface
2. Merancang struktur navigasi aplikasi.
3. Merancang desain interface aplikasi.
4. Melakukan pengcodingan aplikasi.
5. Melakukan uji coba aplikasi.

Penggunaan aplikasi ini sangat mudah dan cukup sederhana. Pengguna masuk ke dalam aplikasi Hallo Depok maka akan muncul tampilan menu utama yang terdiri dari “Menu”, “About”, “Exit”. Menu yang pertama adalah “Menu”, yang berisi kategori tempat-tempat yang berada di kota Depok seperti gerai operator, rumah sakit, kantor polisi, apotik (24 jam), service centre, percetakan (24 jam), money changer, pemadam kebakaran, kantor pemerintahan, bank, bengkel, penginapan, dan jadwal SIM keliling. Jika kita memilih kategori gerai operator maka akan muncul daftar gerai operator yang berada di kota Depok, lengkap dengan alamat, no telepon, jam buka, jarak lokasi dengan pengguna, dan online map. Menu kedua adalah “About”, pengguna dapat melihat biodata developer dari aplikasi ini. Menu berikutnya adalah menu “Exit”, yang digunakan untuk keluar dari aplikasi Hallo Depok.

3.2. Struktur Navigasi

Struktur navigasi merupakan susunan menu yang terdapat pada suatu aplikasi. Struktur navigasi yang digunakan pada aplikasi Hallo Depok adalah struktur navigasi hirarki karena menggunakan konsep treemap. Tidak hanya itu, struktur ini mempresentasikan suatu pokok informasi secara teratur dengan membagi suatu pokok informasi menjadi beberapa sub-pokok informasi yang dapat mengurangi kompleksitas suatu informasi sehingga pengguna dapat mudah dalam mencari suatu tempat yang dibutuhkan.

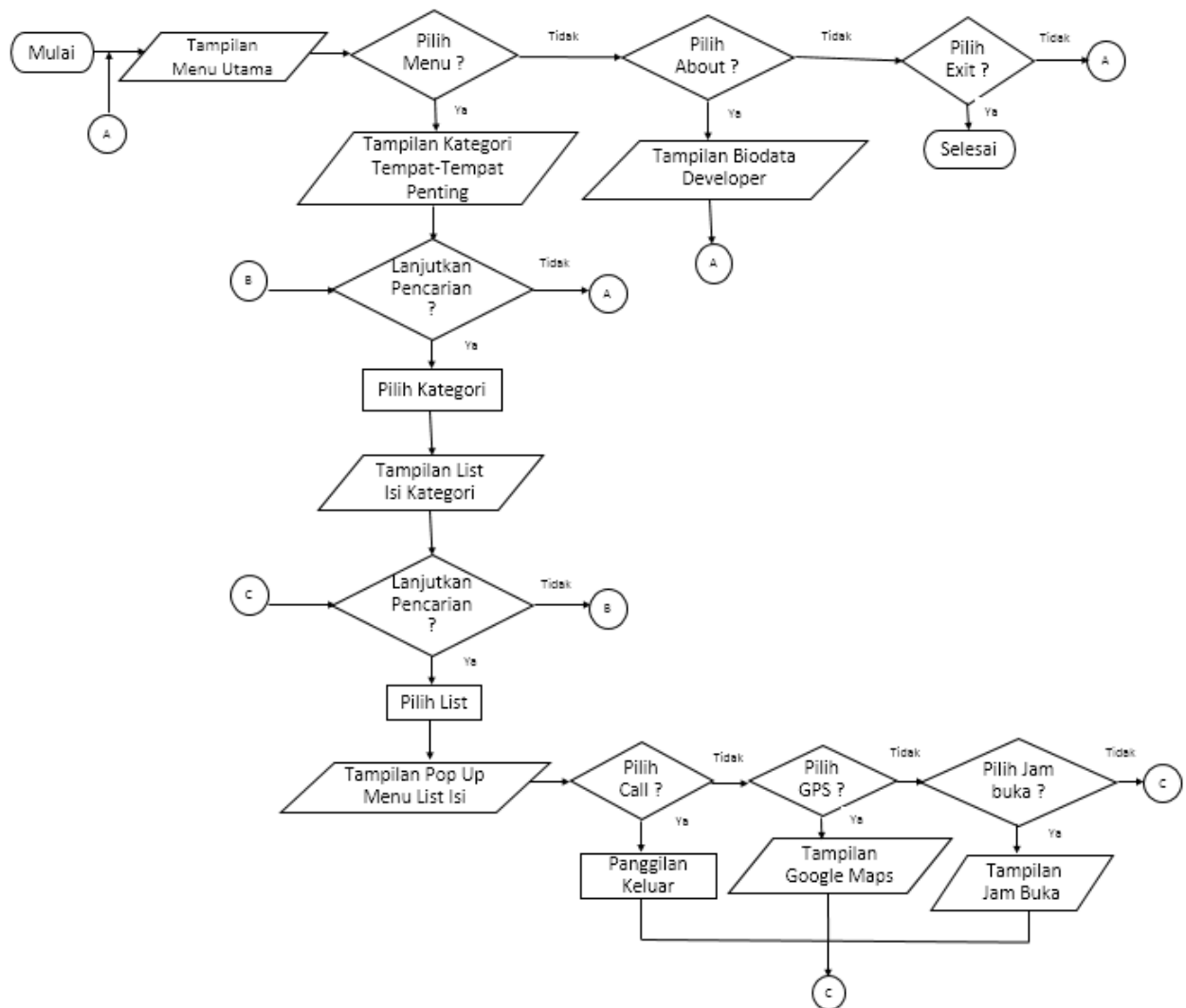
Dari struktur navigasi di bawah dapat terlihat susunan menu dari aplikasi tersebut yaitu pengguna masuk ke dalam aplikasi Hallo Depok maka akan muncul tampilan menu utama yang terdiri dari “Menu”, “About”, dan “Exit”. Jika pengguna memilih menu maka akan keluar tampilan menu kategori tempat-tempat penting di kota Depok. Kemudian pada saat pengguna memilih salah satu dari kategori tersebut maka akan keluar tampilan isi dari kategori yang kita pilih berbentuk list. Jika kita memilih salah satu dari list tersebut maka akan keluar tiga opsi berbentuk pop up menu yang terdiri dari “Call”, “GPS”, dan “Jam Buka”. Apabila kita memilih opsi “Call” maka pengguna akan langsung menelepon tempat tersebut. Lalu apabila pengguna memilih opsi “GPS” maka pengguna akan langsung masuk ke dalam aplikasi google maps yang akan menunjukkan tempat yang dipilih oleh pengguna. Kemudian apabila pengguna memilih “Jam Buka” maka pengguna akan mendapatkan informasi mengenai waktu buka dan tutup dari tempat yang dipilih. Pada saat di menu utama pengguna memilih menu “About” maka pengguna akan mendapatkan informasi mengenai biodata developer aplikasi Hallo Depok. Kemudian jika pengguna memilih menu “Exit” maka pengguna akan keluar dari aplikasi Hallo Depok.



Gambar 3.1. Struktur Navigasi

3.2.1. Flowchart

Flowchart di bawah ini menjelaskan bagaimana proses atau alur yang berjalan pada aplikasi Hallo Depok. Sehingga diharapkan dengan flowchart ini konsep yang dirancang dapat diimplementasikan dengan baik pada aplikasi ini. Berikut adalah diagram flowchart dari aplikasi Hallo Depok :



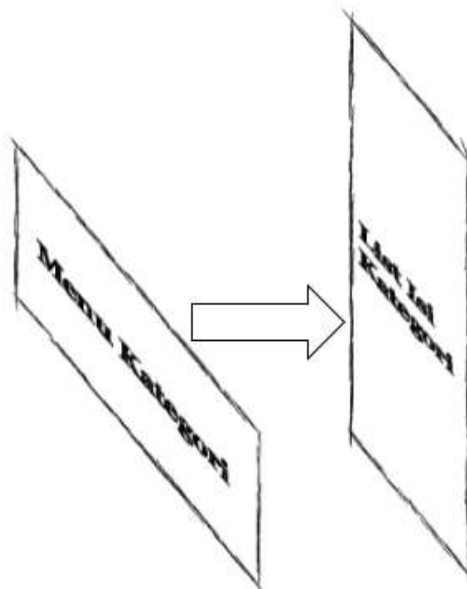
Gambar 3.2. Flowchart Hallo Depok

3.3. Perancangan Aplikasi

Pada tahapan ini akan menjelaskan rancangan tampilan pada aplikasi Hallo Depok. Di mana pada user interface ini diharapkan akan memudahkan pengguna dalam menjalankan aplikasi ini. Berikut adalah gambar beserta penjelasannya dari rancangan tersebut.

3.3.1. Konsep Desain Interface

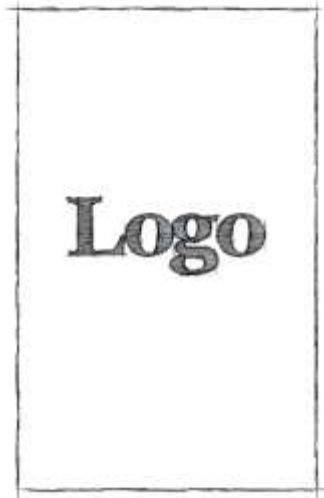
Dalam pembuatan aplikasi, desain interface merupakan salah satu yang terpenting karena desain melakukan interaksi langsung dengan pengguna. Desain menyangkut soal nilai estetika, di mana pengguna akan menilai sangat objektif. Aplikasi Hallo Depok ini dirancang dengan desain yang baik tanpa mengurangi fungsi dari aplikasi itu sendiri yang tangguh (powerful). Penulis mengambil sebagian konsep desain dari metode visualisasi informasi yaitu representasi isi ruang atau treemaps. Treemaps adalah metode untuk menampilkan informasi secara hirarki dengan menampilkan kategori dalam persegi panjang yang berkelompok di mana isi dari kategori tersebut berada di dalamnya.



Gambar 3.3. Konsep Desain Interface

3.3.2. Tampilan Splashscreen

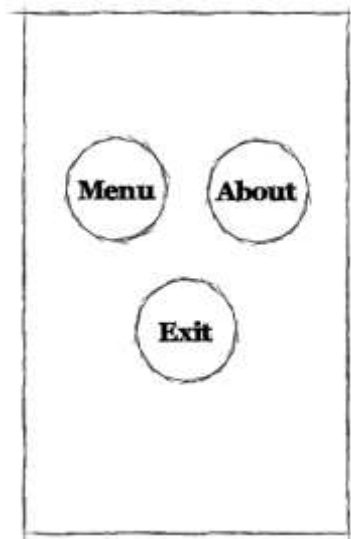
Tampilan ini merupakan tampilan pembuka pada saat pengguna menjalankan aplikasi Hallo Depok. Penambahan splashscreen pada aplikasi ini untuk menambah nilai artistik pada aplikasi Hallo Depok.



Gambar 3.4. Rancangan Tampilan Splashscreen

3.3.3. Tampilan Menu Utama

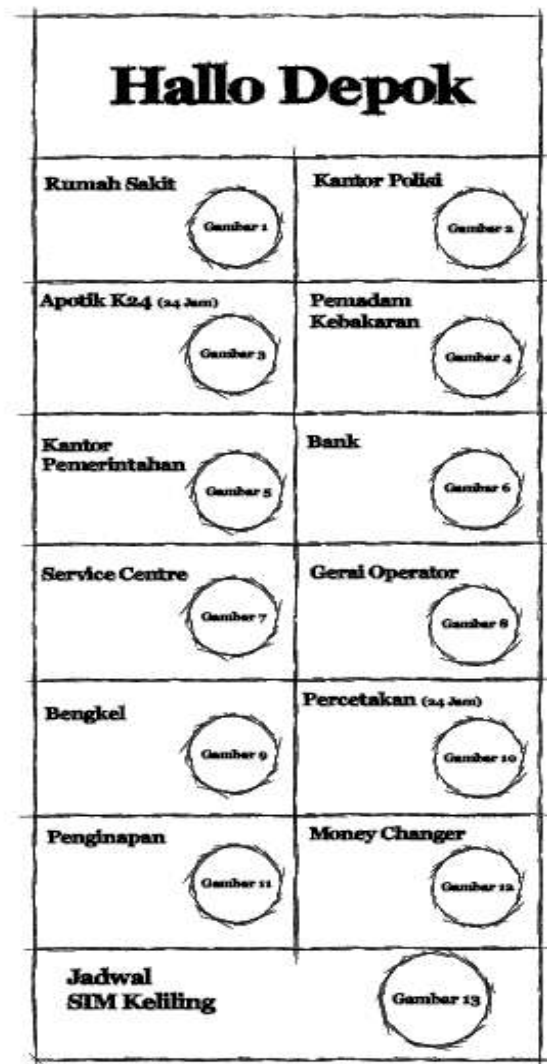
Gambar di bawah ini merupakan tampilan yang akan muncul setelah tampilan splashscreen. Pada tampilan ini terdapat tiga menu yang berupa button yang terdiri dari "Menu", "About", dan "Exit". Di sini ditaruh hanya tiga buah button saja yang bertujuan agar pengguna tidak bingung dalam menggunakan aplikasi ini.



Gambar 3.5. Rancangan Tampilan Menu Utama

3.3.4. Tampilan Menu Kategori

Pada rancangan tampilan menu akan muncul tampilan kategori tempat-tempat penting di kota Depok. Di sini konsep treemap diterapkan di mana setiap kategori diwakili oleh persegi panjang.



Gambar 3.6. Rancangan Tampilan Menu Kategori

3.3.5. Tampilan List Isi Kategori

Pada bagian ini akan ditampilkan isi dari kategori-kategori yang ada dalam berbentuk list. Di mana terdapat nama tempat, alamat, nomor telepon dan jarak.

Nama Tempat 1	Jarak 1
Alamat 1	
No Telepon 1	
Nama Tempat 2	Jarak 2
Alamat 2	
No Telepon 2	
Nama Tempat 3	Jarak 3
Alamat 3	
No Telepon 3	
Nama Tempat 4	Jarak 4
Alamat 4	
No Telepon 4	
Nama Tempat 5	Jarak 5
Alamat 5	
No Telepon 5	

Gambar 3.7. Rancangan Tampilan List Isi Kategori

3.3.6. Tampilan Pop Up Menu List Isi

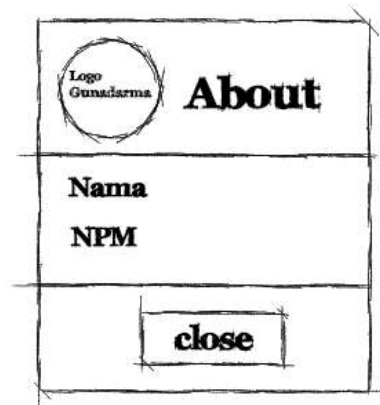
Pada saat di list isi kategori kita menekan salah satu list tersebut maka akan muncul opsi yang berbentuk pop up menu yang terdiri dai “Call”, “GPS”, dan “Jam Buka”.

Call
GPS
Jam Buka

Gambar 3.8. Rancangan Tampilan Pop Up Menu

3.3.7. Tampilan About

Pada tampilan ini akan menampilkan biodata developer yang membuat aplikasi Hallo Depok.



Gambar 3.9. Rancangan Tampilan About

3.4. Pembuatan Aplikasi Hallo Depok

Dalam pembuatan aplikasi Hallo Depok digunakan perangkat lunak Eclipse Juno sebagai editor, ADT (Android Development Tools) 22.0.1 untuk pluginnya. Untuk library dan emulator menggunakan Android SDK_r22 (Standart Development Kit).

3.4.1. Pembuatan Splashscreen

Dalam membangun suatu aplikasi android dibutuhkan dua file. Pertama file java untuk pembuatan program dan yang kedua file XML untuk tampilan layout aplikasi android tersebut. Pada bagian rancangan layout splashscreen dipergunakan splashscreen.xml. Berikut potongan script dari splashscreen.xml :

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@drawable/logo">
</LinearLayout>
```

Layout yang digunakan adalah linear layout dengan menggunakan layout ini maka komponen-komponen yang berada di dalamnya akan tersusun secara perbaris atau perkolom. Di sini untuk background yang akan dijadikan tampilan pembuka diambil dari folder drawable dengan nama file logo. Gambar yang dijadikan background lebar dan tingginya bernilai “fill_parent” yang akan

membuat lebar dan tinggi gambar akan memenuhi mobile device. Kemudian dilanjutkan pembuatan script untuk membuat gambar ini muncul pada saat aplikasi dijalankan. Berikut potongan script dari `Splashscreen.java` :

```
public class Splashscreen extends Activity {
    Context context;
    Activity parent;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        context=this;
        parent = this;
        setContentView(R.layout.splashscreen);
        Timer timer=new Timer();
        Advancedtomenu task= new Advancedtomenu();
        timer.schedule(task, 2000);
    }
}
```

Script di atas menggunakan layout file XML `splashscreen` yang berisi gambar untuk tampilan pembuka. File `splashscreen.xml` ini berada di folder `res/layout`. Waktu `splashscreen` ini diatur menggunakan perintah `timer.schedule(task, 2000)` yang durasi `splashscreen` 2000 milisecond atau 2 detik. Setelah durasi selesai maka akan memanggil class `Advancedtomenu`.

```
class Advancedtomenu extends TimerTask {
    public void run() {
        Intent intent=new Intent(context,MenuUtama.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK|Intent.FLAG_ACTIVITY_CLEAR_T
        OP);
        startActivity(intent);
        parent.finish();
    }
}
```

Pada class `Advancedtomenu` ini Activity yang dilakukan adalah melakukan intent (pindah halaman). Pada saat durasi `splashscreen` habis maka akan terbentuk intent baru dengan memanggil file `MenuUtama.java` yang merupakan script pemograman dari tampilan menu utama.



Gambar 3.10. Tampilan Splashscreen

3.4.2. Pembuatan Menu Utama

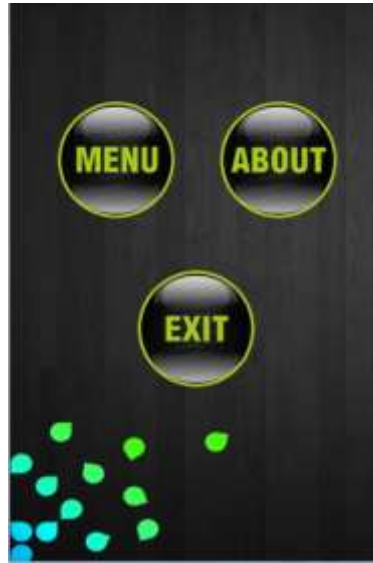
Setelah tampilan splashscreen maka aplikasi akan masuk ke dalam menu utama. Pada pembuatan menu utama di dalamnya menggunakan tiga buah tombol yaitu Menu, About, dan Exit. Dalam file XML tombol ini merupakan widget button maka di sini menggunakan tiga buah widget button untuk masing-masing fungsi. Berikut potongan script XML menu utama :

```
<RelativeLayout
    android:layout_width="fill_parent" android:layout_height="fill_parent"
    android:background="@drawable/menu_utama" >
    <Button
        android:id="@+id/menu" android:layout_width="100dp"
        android:layout_height="100dp" android:layout_alignBaseline="@+id/about"
        android:layout_above="@+id/exit" android:layout_marginBottom="45dp"
        android:layout_marginRight="25dp" android:layout_toLeftOf="@+id/about" android:bac
        kground="@drawable/menu" />
    <Button
        android:id="@+id/about" android:layout_width="100dp" android:layout_height="100dp"
        android:layout_above="@+id/exit" android:layout_alignParentRight="true"
        android:layout_marginBottom="25dp" android:layout_marginRight="39dp"
        android:background="@drawable/about" />
    <Button
        android:id="@+id/exit" android:layout_width="100dp" android:layout_height="100dp"
        android:layout_alignParentBottom="true" android:layout_centerHorizontal="true"
        android:layout_marginBottom="125dp" android:background="@drawable/exit" />
</RelativeLayout>
```

Dalam `menu_utama.xml` ini menggunakan `relative layout` di mana komponen-komponen yang berada di dalamnya dapat diatur bebas tidak harus secara vertikal ataupun horizontal. Untuk tampilan background dari menu utama ini diambil dari folder `drawable` dengan nama filenya `menu_utama`. Untuk button masing-masing memiliki identifiers. Button “Menu” untuk masuk ke dalam menu kategori dengan identifiers “`@+id/menu`”, button “About” untuk menampilkan biodata developer dalam bentuk `pop up` dengan “`@+id/about`”, sedangkan button “Exit” untuk keluar dari program menggunakan identifiers “`@+id/exit`”. Setelah itu pembuatan script `MenuUtama.java`. Berikut ini adalah potongan scriptnya :

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.menu_utama);
    Button next =(Button) findViewById(R.id.menu);
    next.setOnClickListener(new View.OnClickListener() {
        public void onClick(View view) {
            Intent myIntent = new Intent(view.getContext(),MenuKategori.class);
            startActivityForResult(myIntent, 0);});
    Button next2 =(Button) findViewById(R.id.exit);next2.setOnClickListener(new
    View.OnClickListener() { public void onClick(View view) {
        finish();}});}
```

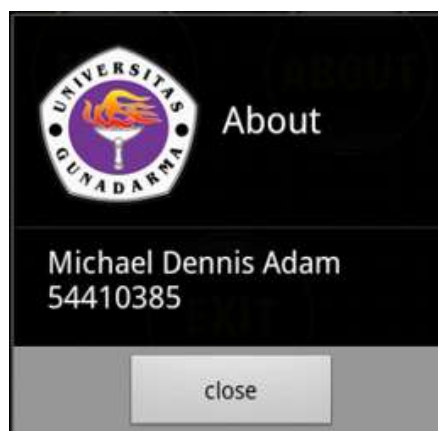
Pada potongan script `MenuUtama.java` ini menggunakan layout `menu_utama.xml`. Fungsi `findViewById` digunakan untuk menghubungkan antar variabel yang dibuat di `xml` dengan deklarasi dalam `android:id="@+id/`". Setiap button diberikan listener agar dapat menangkap event saat button itu dipilih dengan perintah ‘`setOnClickListener`’. Kemudian untuk membaca button yang dipilih maka digunakan `View.OnClickListener` kemudian dimasukkan fungsi yang akan dijalankan di dalam fungsi `onClick`. Di dalam `onClick`, fungsi yang diterapkan adalah `intent` (pindah halaman). Saat memilih button menu maka program akan memanggil `MenuKategori.class` untuk masuk ke menu kategori. Kemudian button `exit` memiliki fungsi `finish` yang berfungsi untuk keluar dari aplikasi.



Gambar 3.11. Tampilan Menu Utama

```
Button alert=(Button)findViewById(R.id.about);
alert.setOnClickListener(this);
public void onClick(View view) {
    new AlertDialog.Builder(this)
        .setTitle ("About")
        .setMessage ("Michael Dennis Adam \n54410385 \nUniversitas Gunadarma")
        .setIcon (R.drawable.gunadarma)
        .setNeutralButton ("close", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dlg, int sumthin) { }}).show(); }
```

Script di atas merupakan fungsi `onClick` pada menu `about` yang fungsinya adalah `alertdialog` maka saat button `about` dipilih akan keluar message box yang berisi biodata developer.



Gambar 3.12. Tampilan About

3.4.3. Pembuatan Menu Kategori

Seperti halnya pada menu utama, pada tampilan kategori memiliki logika program menggunakan intent sebagai awal untuk membuka list dari kategori. Pada tampilan ini banyak menggunakan widget button karena pada menu ini ada 13 kategori yang disediakan. Berikut merupakan script dari menu.xml :

<ScrollView

```
android:id="@+id/scrollView1"
android:layout_width="match_parent"
android:layout_height="wrap_content" >
<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical" >
```

Pada menu ini terdapat penggunaan scroll view agar pada saat aplikasi ini dijalankan pada mobile device yang berukuran layar kecil, menu kategori dapat terlihat semua dengan menggeser scroll yang terdapat pada mobile device ke atas atau ke bawah karena di sini orientasinya secara vertikal.

```
<ImageView
android:id="@+id/imageView1"
android:layout_width="fill_parent"
android:layout_height="150dp"
android:clickable="true"
android:contentDescription="@string/todo"
android:scaleType="fitXY"
android:src="@drawable/hallodepok" />
<LinearLayout
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:contentDescription="@string/todo"
android:gravity="center_vertical/center_horizontal"
android:orientation="horizontal" >
<Button
android:id="@+id/rs"
android:layout_width="wrap_content"
android:layout_height="100dp"
android:layout_weight="48.81"
android:scaleType="fitXY"
android:background="@drawable/rs" />
<Button
android:id="@+id/polisi"
android:layout_width="wrap_content"
android:layout_height="100dp"
android:layout_weight="48.81"
```

```

        android:scaleType="fitXY"
        android:background="@drawable/polisi" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:contentDescription="@string/todo"
        android:gravity="center_vertical/center_horizontal"
        android:orientation="horizontal" >
        <Button
            android:id="@+id/apotik"
            android:layout_width="wrap_content"
            android:layout_height="100dp"
            android:layout_weight="48.81"
            android:background="@drawable/apotik" />
        <Button
            android:id="@+id/pemadam"
            android:layout_width="wrap_content"
            android:layout_height="100dp"
            android:layout_weight="48.81"
            android:scaleType="fitXY"
            android:background="@drawable/pemadam" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:contentDescription="@string/todo"
        android:gravity="center_vertical/center_horizontal"
        android:orientation="horizontal" >
        <Button
            android:id="@+id/pemerintahan"
            android:layout_width="wrap_content"
            android:layout_height="100dp"
            android:layout_weight="48.81"
            android:scaleType="fitXY"
            android:background="@drawable/pemerintah" />
        <Button
            android:id="@+id/bank"
            android:layout_width="wrap_content"
            android:layout_height="100dp"
            android:layout_weight="48.81"
            android:scaleType="fitXY"
            android:background="@drawable/bank" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:contentDescription="@string/todo"
        android:gravity="center_vertical/center_horizontal"
        android:orientation="horizontal" >

```

```

<Button
    android:id="@+id/bengkel"
    android:layout_width="wrap_content"
    android:layout_height="100dp"
    android:layout_weight="48.81"
    android:scaleType="fitXY"
    android:background="@drawable/bengkel" />
<Button
    android:id="@+id/galeri"
    android:layout_width="wrap_content"
    android:layout_height="100dp"
    android:layout_weight="48.81"
    android:scaleType="fitXY"
    android:background="@drawable/gerai" />
</LinearLayout>
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:contentDescription="@string/todo"
    android:gravity="center_vertical|center_horizontal"
    android:orientation="horizontal" >
    <Button
        android:id="@+id/service"
        android:layout_width="wrap_content"
        android:layout_height="100dp"
        android:layout_weight="48.81"
        android:scaleType="fitXY"
        android:background="@drawable/service" />
    <Button
        android:id="@+id/percetakan"
        android:layout_width="wrap_content"
        android:layout_height="100dp"
        android:layout_weight="48.81"
        android:scaleType="fitXY"
        android:background="@drawable/percetakan" />
</LinearLayout>
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:contentDescription="@string/todo"
    android:gravity="center_vertical|center_horizontal"
    android:orientation="horizontal" >
    <Button
        android:id="@+id/hotel"
        android:layout_width="wrap_content"
        android:layout_height="100dp"
        android:layout_weight="48.81"
        android:scaleType="fitXY"
        android:background="@drawable/hotel" />
    <Button

```



```

android:id="@+id/changer"
android:layout_width="wrap_content"
android:layout_height="100dp"
android:layout_weight="48.81"
android:scaleType="fitXY"
android:background="@drawable/changer" />
</LinearLayout>
<LinearLayout
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:contentDescription="@string/todo"
android:gravity="center_vertical/center_horizontal"
android:orientation="horizontal" >
<Button
android:id="@+id/sim"
android:layout_width="fill_parent"
android:layout_height="100dp"
android:layout_weight="48.81"
android:scaleType="fitXY"
android:background="@drawable/sim" />
</LinearLayout>
</LinearLayout>
</ScrollView>

```

Script di atas terdapat image view yang digunakan untuk menampilkan gambar yang dijadikan sebagai banner pada aplikasi ini. Kemudian Penggunaan button untuk membuat tombol dengan identifiernya masing-masing. Sebagai contoh pada button gerai operator memiliki identifiernya “@+id/galeri”. Button ini memiliki background yang gambarnya diambil dari folder drawable dengan nama file gerai. Pada button ini lebarnya wrap_content yang artinya lebar dari button ini akan menyesuaikan content yang ada di dalamnya sedangkan untuk tingginya diset 100dp kemudian agar dapat di dalam layout dapat dimasukkan banyak button maka layout weightnya diset menjadi 48.1. Untuk button yang lainnya pengaturannya sama tetapi untuk identifier dan backgroundnya disesuaikan dengan kategori masing-masing. Selanjutnya pembuatan script MenuKategori.java.

```

protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.menu);
Button next =(Button) findViewById(R.id.rs);
next.setOnClickListener(new View.OnClickListener() {
public void onClick(View view) {
Intent myIntent = new

```



```

    });
    Button next9 =(Button) findViewById(R.id.service);
    next9.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
    Intent myIntent = new
    Intent(view.getContext(),Service.class);
    startActivityForResult(myIntent, 0);
    });
    Button next10 =(Button) findViewById(R.id.percetakan);
    next10.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
    Intent myIntent = new
    Intent(view.getContext(),Percetakan.class);
    startActivityForResult(myIntent, 0);
    });
    Button next11 =(Button) findViewById(R.id.hotel);
    next11.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
    Intent myIntent = new
    Intent(view.getContext(),Penginapan.class);
    startActivityForResult(myIntent, 0);
    });
    Button next12 =(Button) findViewById(R.id.changer);
    next12.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
    Intent myIntent = new
    Intent(view.getContext(),Moneychanger.class);
    startActivityForResult(myIntent, 0);
    });
    Button next13 =(Button) findViewById(R.id.sim);
    next13.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
    Intent myIntent = new
    Intent(view.getContext(),Sim.class);
    startActivityForResult(myIntent, 0);
    });}

```

Pada potongan script sama halnya dengan fungsi lainnya yaitu untuk intent (pindah halaman) dengan mengaktifkan button-button yang telah dibuat pada menu.xml dengan menggunakan id button masing-masing. Intent yang dipanggil sesuai dengan class java kategori. Sebagai contoh saat memilih menu kategori gerai operator maka fungsi Onclick akan memanggil Galeri.class yang merupakan list isi kategori dari gerai operator. Begitu juga dengan kategori-kategori yang lain akan memanggil class masing-masing kategori.



Gambar 3.13. Tampilan Menu Kategori

3.4.4. Pembuatan Menu List Isi Kategori

Dalam pembuatan menu list isi kategori ini dibutuhkan dua file XML dan dua file java. File XML tersebut adalah listmenu.xml dan list.xml sedangkan file javanya AdapterItem.java dan Item.Java. File listmenu.xml ini sebagai kerangka dari list kategori sedangkan file list.xml untuk isi dari list kategori tersebut seperti nama tempatnya, alamat, nomor telepon dan jarak. Pada aplikasi Hallo Depok jumlah kategori yang disediakan cukup banyak dengan kata lain list isi kategori yang dibuat disesuaikan dengan jumlah kategorinya. Agar program menjadi lebih efektif maka dibuatlah AdapterItem.java agar listmenu.xml dan list.xml dapat digunakan sebagai kerangka untuk membuat list kategori lainnya. Jadi, tidak perlu membuat banyak kerangka meskipun list untuk kategorinya banyak cukup satu kerangka yang dapat digunakan bersama. Sedangkan file Item.java digunakan sebagai konstruktor atribut objek list kategori yang ada. Berikut potongan script dari Item.java :

```

public class Item {
    private String nama;
    private String alamat;
    private String telp;
    private String buka;
    private Location location;
    public Item (String nama, String alamat, String telp, String buka, double lat,double lon ){
        this.nama= nama;
        this.alamat= alamat;
        this.telp= telp;
        this.buka=buka;
        this.location= new Location ("HalloDepok") ;
        this.location.setLatitude(lat);
        this.location.setLongitude(lon);
    }

```

Perintah di atas menggunakan parameter di mana parameternya adalah `public Item (String nama, String alamat, String telp, String buka, double lat,double lon)` kemudian parameter yang ada dideklarasikan sama dengan atribut pada class Item.

```

public void setNama(String nama) {
    this.nama = nama;
}
public void setAlamat(String alamat) {
    this.alamat = alamat;
}
public void setTelp(String telp) {
    this.telp = telp;
}
public void setBuka(String buka){
    this.buka= buka;
}
public void setLocation(Location location) {
    this.location = location;
}

```

Penggunaan `public void setNama(String nama) { this.nama = nama; }` merupakan method yang digunakan untuk memberi nilai baru atribut nama melalui parameter yang nantinya diinput pada class java list kategori. Sama halnya dengan atribut alamat, telp, buka dan location yang nantinya akan diberi nilai baru atributnya sesuai dengan parameternya masing-masing.

```

public String getNama() {
return nama;
}
public String getAlamat() {
return alamat;
}
public String getTelp() {
return telp;
}
public String getBuka(){
return buka;
}
public double getLon() {
return location.getLongitude() ;
}
public double getLat() {
return location.getLatitude();
}
public Location getLocation() {
return location;
}

```

Script diatas digunakan untuk mengambil nilai dari atribut di dalam class Item.java yang nilai atributnya sesuai dengan parameter masing-masing atribut. Setelah Item.java selesai maka dibuatlah AdapterItem.java, berikut potongan scriptnya :

```

public class AdapterItem extends ArrayAdapter<Item>{
Item[] list;
Location currentlocation;
public Item[] getList() {
return list; }
Context parent;
public AdapterItem(Context context,
int textViewResourceId, Item[] list, Location currentlocation) {
super(context, textViewResourceId, list);
this.list=list;
this.parent=context;
this.currentlocation=currentlocation;
}
public View getView(int position, View convertView, ViewGroup parent) {
LayoutInflater inflater=(LayoutInflater)
this.parent.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
View view=inflater.inflate(R.layout.list,parent,false);
TextView v1=(TextView) view.findViewById(R.id.nama);
TextView v2=(TextView) view.findViewById(R.id.alamat);
TextView v3=(TextView) view.findViewById(R.id.telp);
TextView v4=(TextView) view.findViewById(R.id.jarak);
Item depok= list[position];

```

```

v1.setText(depok.getNama());
v2.setText(depok.getAlamat());
v3.setText(depok.getTelp());
if (currentlocation != null) { double
jarak=currentlocation.distanceTo(depok.getLocation());
v4.setText(String.format("%.3f Km", jarak/1000));
} else { v4.setText("None");}
return view;}}

```

Perintah-perintah di atas digunakan untuk menghubungkan list.xml dengan data-data dari list kategori yang ada. Dengan fungsi inflater ini maka kedua file XML tadi menjadi view bagi data-data pada list kategori. Kemudian fungsi findViewById untuk mengaktifkan texview pada list.xml menjadi tempat output data seperti nama, alamat, telepon dan jarak. Pada jarak formatnya tiga angka di belakang koma dengan satuan kilometer tetapi jika jaraknya bernilai kosong maka hanya ada tulisan “None”. Data-data tersebut dibaca dengan menggunakan method get pada class Item.java. Selanjutnya pembuatan file java list kategori, di sini akan diambil salah satu contoh dari kategori gerai operator. Berikut potongan script Gerai.java :

```

protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
Item[] list=new Item[7];
list[0]= new Item ("Telkomsel","Jl. Margonda Raya No.130","155","Senin-Jum'at
\n08.00-17.00", -6.366007,106.83411);list[1]= new Item ("Indosat","Ruko Pesona
Khayangan Blok.5 Jl. Margonda Raya No. 45","0217756397","Senin-Jum'at \n08.00-
17.00 \n\nSabtu \n09.00-17.00", -6.382769,106.830207); list[2]= new Item
("IM2","Depok Town Square Lantai Dasar GS 7","02178870346","Senin-Jum'at \n08.00-
17.00 \nSabtu \n09.00-17.00", -6.372485,106.832753); list[3]= new Item ("XL","Ruko
Margonda Residence Jl Margonda Raya No 2-3","817","Senin-Sabtu \n08.00-19.00
\n\nLibur Nasional \n10.00 - 16.00", -6.363942,106.833903); list[4]= new Item
("Axis","ITC Depok Lt.2 Blok A-38","838","Senin-Minggu \n10.00-19.00", -
6.392711,106.823429); list[5]= new Item ("Smartfren","Ruko ITC Depok No.12 RT
04/12","888","Senin-Minggu \n09.00-17.30", -6.393764,106.823215); list[6]= new Item
("Esia","Jl.Margonda Raya no 27","02191009999","Senin-Jum'at \n09.00-18.00
\n\nSabtu - Minggu \n09.00-14.00", -6.365954,106.83412);
setContentView(R.layout.listmenu);
AdapterItem adapter=new AdapterItem (this,R.layout.list,list,mylocation);
setListAdapter(adapter);
registerForContextMenu(getListView()); }

```

Perintah di atas digunakan untuk memasukkan data-data yang terkait sesuai dengan kategori masing-masing. Di sini digunakan array satu dimensi yang ketentuan parameternya telah ditentukan pada Item.java. Sebagai contoh `list[0]= new Item ("Telkomsel","Jl. Margonda Raya No.130","155","Senin-Jum'at \n08.00-17.00", -6.366007,106.83411)` Telkomsel masuk ke dalam atribut nama, Jl. Margonda Raya No.130 masuk ke dalam atribut alamat, 155 masuk ke dalam atribut telp, Senin-Jum'at \n08.00-17.00 masuk ke dalam atribut buka, -6.366007 masuk atribut lat serta 106.83411 masuk ke dalam atribut lon. Output data-datanya menggunakan list.xml yang menggunakan listmenu.xml sebagai kerangka layoutnya.

Telkomsel Jl. Margonda Raya No.130 155	None
Indosat Ruko Pesona Khayangan Blok.5 Jl. Margonda Raya No. 45 0217756397	None
IM2 Depok Town Square Lantai Dasar GS 7 02178870346	None
XL Ruko Margonda Residence Jl Margonda Raya No 2-3 817	None
Axis ITC Depok LT.2 Blok A-38 838	None
Smartfren Ruko ITC Depok No.12 RT 04/12 888	None
Esia Jl.Margonda Raya no 27 02191009999	None

Gambar 3.14. Tampilan List Isi Kategori

3.4.5. Pembuatan Pop Up Menu List Isi

Untuk pembuatan pop up menu, scriptnya digabung dengan masing-masing class list isi kategori agar informasi yang didapat sesuai dengan informasi yang ada pada masing-masing list isi kategori. Berikut potongan scriptnya :

```
public boolean onContextItemSelected(MenuItem menu) {
    AdapterContextMenuInfo info=(AdapterContextMenuInfo) menu.getMenuInfo();
```



```

int pos=info.position;
Item item=((AdapterItem)getListView().getAdapter()).getList()[pos];
int menuid=menu.getItemId();
if (menuid==MENU_CALL){
Intent intent=new Intent (Intent.ACTION_CALL,Uri.parse("tel:"+item.getTelp()));
startActivity(intent);
}
else if (menuid==MENU_GPS){
Intent intent=new Intent
(Intent.ACTION_VIEW,Uri.parse("http://maps.google.com/?f=d&saddr=&daddr="+item
.getLat()+","+item.getLon()+" (" +item.getNama()+")"));
startActivity(intent);
}
else if (menuid==MENU_BUKA){
AlertDialog.Builder alertbox = new AlertDialog.Builder(this);
alertbox.setMessage(item.getBuka());
alertbox.setNeutralButton("Ok", new DialogInterface.OnClickListener() {
public void onClick(DialogInterface arg0, int arg1) {
}});
alertbox.show();
}
return super.onContextItemSelected(menu);
}
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo
menuInfo) {
super.onCreateContextMenu(menu, v, menuInfo);
menu.add(Menu.NONE,MENU_CALL,Menu.NONE,"Call");
menu.add(Menu.NONE,MENU_GPS,Menu.NONE,"GPS");
menu.add(Menu.NONE,MENU_BUKA,Menu.NONE,"Jam Buka");}

```

Perintah context menu digunakan untuk menampilkan menu saat list kategori ditekan. Aksi-aksi yang terdapat pada menu id akan disimpan pada context menu dengan fungsi menu.add (). Dengan adanya context item selected maka program akan melakukan aksi saat menu dipilih. Pada saat pengguna pilih menu “Call” maka yang dipilih menu id MENU_CALL dengan melakukan aksi ACTION_CALL yaitu melakukan panggilan keluar dengan method getTelp untuk mendapat nomor teleponnya. Sedangkan jika memilih menu “GPS” maka yang dipilih menu id MENU_GPS dengan melakukan aksi ACTION_VIEW yaitu membuka aplikasi google maps. Disini d&saddr kosong maka aplikasi akan mencari lokasi keberadaan mobile devicenya kemudian &daddr="+item.getLat()+","+item.getLon()+" (" +item.getNama()+") aplikasi akan mencari tujuan kita dengan mendapatkan latitude, longitude dan nama lokasinya dengan metode getLat, getLong, dan getNama. Lalu apabila pengguna memilih

menu “Jam Buka” maka yang dipilih menu id MENU_BUKA dengan melakukan aksi menampilkan message dialog yang berisi informasi jam buka dari kategori tersebut.



Gambar 3.15. Tampilan Menu List

3.5. Uji Coba Aplikasi

Setelah pembuatan program aplikasi maka dilakukan uji coba dengan menggunakan ponsel berbasis Android dengan sistem operasi berbasis android 2.3 (GingerBread). Spesifikasi ponsel yang digunakan sebagai berikut :


- **Layar** : Tipe TFT capacitive touchscreen, 256K colors
Ukuran 240 x 320 pixels, 3.2 inches (~125 ppi pixel density)
- **Dimensi** : Ukuran/Berat 102.6 x 61.6 x 11.7 mm
- **Memori** : Internal 1 GB storage, 384MB RAM
Eksternal microSD, up to 32GB
- **Data** : 3G HSDPA, 3.6 Mbps; HSUPA
EDGE : Ya
GPRS : Ya
WLAN : Wi-Fi 802.11 b/g/n

Bluetooth : Ya, v2.1 with A2DP

USB/Port v2.0 microUSB

- Fitur : OS Android OS, v2.3 (Gingerbread) CPU 800 MHz
Browser HTML
GPS : Ya, A-GPS

Untuk melaksanakan uji coba ada beberapa hal yang harus dilakukan seperti berikut ini :

1. Copy file Hallo Depok.apk berada di folder Hallo Depok/bin ke dalam ponsel menggunakan kabel data.
2. Setelah file Hallo Depok berada di ponsel, maka install aplikasi Hallo Depok
3. Setelah terinstal, pilih icon  untuk masuk ke dalam aplikasi Hallo Depok.

3.5.1. System Internal Testing

System Internal Testing bertujuan untuk memastikan bahwa aplikasi Hallo Depok berjalan dengan baik atau tidak dan sesuai rancangan yang ada. Selain itu hasil yang didapat menjadi bahan evaluasi sebelum pemakai akhir melakukan test terhadap aplikasi Hallo Depok. Ada beberapa hal yang diamati dalam proses uji coba ini :

Tabel 3.1. Hasil Uji Coba Aplikasi

No	Uji Coba	Keterangan	Hasil
1.	Struktur navigasi	Struktur navigasi sesuai dengan struktur yang dirancang.	Ok
2.	Ukuran gambar splashscreen pada ponsel	Gambar splashscreen memenuhi layar ponsel	Ok
3.	Tampilan desain menu utama	Desain sesuai dengan rancangan menu utama yang ada	Ok

4.	Tampilan About	Menampilkan biodata developer	Ok
5.	Tampilan desain menu kategori	Desain sesuai dengan rancangan menu kategori yang ada	Ok
6.	List isi kategori	List isi kategori sesuai dengan kategorinya masing-masing	Ok
7.	Vertikal scrollview pada aplikasi	Vertikal scrollview berjalan dengan baik	Ok
8.	Output jarak pada list isi kategori	Output jarak pada list isi kategori keluar pada saat GPS atau paket data aktif	Ok
9.	Fungsi Call	Dapat melakukan panggilan keluar sesuai nomor yang ada pada list isi kategori	Ok
10.	Fungsi GPS	Dapat menampilkan google maps dan arah ke lokasi yang dituju	Ok
11.	Fungsi Jam buka	Dapat menampilkan output keterangan waktu buka dan tutup	Ok
12.	Fungsi Exit	Dapat keluar dari aplikasi Hallo Depok	Ok

3.5.2. User Acceptance Test

Penulis melakukan pengujian kepada pengguna akhir (end user) untuk mendapatkan informasi mengenai aplikasi Hallo Depok. Jenis pengujian ini memberikan pengguna akhir keyakinan bahwa aplikasi yang disampaikan kepada mereka memenuhi persyaratan mereka. Untuk itu penulis memberikan tiga

pertanyaan yang berkaitan dengan desain, kemudahan dan manfaat berbentuk kuesioner kepada 20 responden yang diambil secara acak serta meminta penilaiannya. Penilaian kuesioner dengan menggunakan Skala Likert yang mempunyai bobot nilai pada masing-masing jawaban pertanyaan. Bobot nilai pada masing-masing pertanyaan sebagai berikut :

Tabel 3.2. Bobot Jawaban Skala Likert

Skala Likert	Bobot Nilai
Sangat Setuju	4
Setuju	3
Kurang Setuju	2
Tidak Setuju	1

Cara menghitung skor dan presentase penggolongan skor penilaian adalah sebagai berikut :

a. Cara Menghitung Skor

Skor = frekwensi x bobot nilai

Jumlah skor = jumlah skor skala penilaian 1 sampai dengan 4

b. Cara penghitungan presentase penggolongan skor penilaian

Penggolongan skor penilaian dilakukan berdasarkan skor ideal, dimana repondennya berjumlah 20 orang, maka :

Skor Ideal (skor tertinggi) = 20 x bobot nilai tertinggi

Skor terendah = 20 x bobot nilai terendah

sehingga persentase penggolongan skor penilaian adalah :

$$\frac{\text{Jumlah skor}}{\text{Skor ideal}} \times 100 \%$$

sedangkan kriteria interpretasi skor berdasarkan persentase kelompok responden :

76% – 100% = sangat baik/sangat menarik/sangat setuju/ sangat efektif

51% – 75% = baik/menarik/setuju/efektif

26% – 50% = kurang baik/kurang menarik/kurang setuju/kurang efektif

0% – 25% = tidak baik/tidak menarik/tidak setuju/efektif

Berikut ini merupakan tabel hasil kuesioner yang telah diisi oleh 20 responden :

Tabel 3.3. Hasil Kuesioner

No.	Pertanyaan	4	3	2	1	Persentase (%)
1.	Bagaimana desain dari aplikasi Hallo Depok ?	13	7	0	0	91.25
2.	Apakah aplikasi Hallo Depok mudah digunakan ?	16	3	1	0	93.75
3.	Apakah aplikasi Hallo Depok dapat membantu anda untuk mencari tempat penting di kota Depok ?	11	5	4	0	83.75

3.6. Spesifikasi Hardware dan Software Pendukung

Peralatan pendukung yang digunakan oleh penulis untuk membuat aplikasi Hallo Depok ini terdiri dari hardware dan software dengan spesifikasi sebagai berikut :

Hardware

- Komputer Intel Core 2 Duo CPU E7500 2.9 GHz
- RAM 4 GB
- VGA Nvidia GeForce GTS 450
- Hardisk 250 GB

Software

- Microsoft Windows 7
- JDK 7 for Windows
- Eclipse Juno
- ADT (Android Development Tools) 22.0.1
- Android SDK_r22 (Standart Development Kit).