BAB IV

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dilakukan implementasi dan pengujian terhadap sistem. Tahapan ini dilakukan setelah perancangan selesai dilakukan. Selanjutnya akan diimplementasikan pada bahasa pemrograman java yang menghasilkan sebuah aplikasi. Setelah diimplementasikan maka dilakukan pengujian terhadap sistem dengan menggunakan pengujian *Black box*. Kemudian dilihat kekurangan pada aplikasi untuk pengembangan sistem selanjutnya.

4.1 Implementasi

Implementasi sistem merupakan proses pembuatan aplikasi dan nantinya saling berintegrasi dengan tahap selanjutnya. Proses implementasi dibuat berdasarkan hasil dari tahap analisis dan desain sistem. Pada tahap ini dilakukan implementasi atau kode pembangunan aplikasi servis sepeda motor online berbasis *android*. Aplikasi yang dibangun memiliki 3 aktor yang memiliki tugas dan fungsi yang berbeda beda. Aktor aktor tersebut yaitu *admin*, *user*/pengguna, dan bengkel/montir.

A. Form *Login*

Form *login* merupakan halaman *login* ke aplikasi servis sepeda motor online. Untuk bisa login harus mengisikan *username* dan *password* terlebih dahulu.

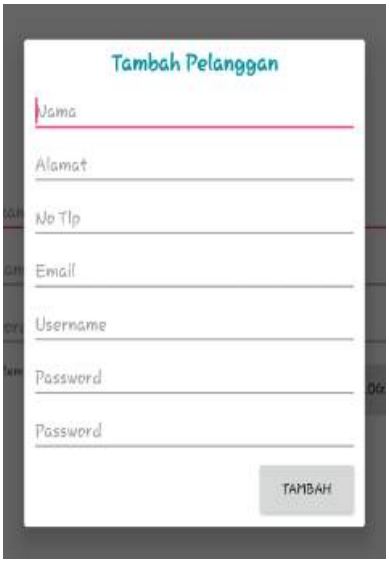
Gambar 4.1 Form *Login*

```
private class CheckLogin extends AsyncTask<Void,Void,Void>{
    String Respons="";
    @Override
    protected void onPreExecute() {
        //add loading
        super.onPreExecute();
    }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        params.put("username",et_username.getText().toString());
        params.put("password",et_password.getText().toString());
        params.put("token_fb",Token_Firebase);
        Log.e( tag: "Firebase",Token_Firebase);
```

Gambar 4.2 *Coding Login*

B. Form *Register*

Form *register/*tambah akun adalah form yang berfungsi membuat akun untuk masuk kedalam aplikasi.



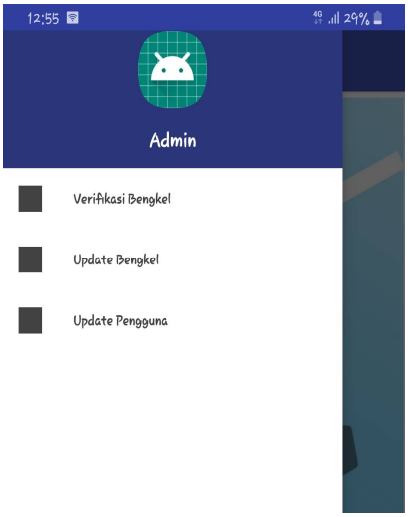
Gambar 4.3 Form *Register*

```
private class AddPengguna extends AsyncTask<Void, Void, Void> {
    String Respons = "";

    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        params.put("PenggunaID", PenggunaID);
        params.put("NamaPengguna", et_nama.getText().toString());
        params.put("Alamat", et_alamat.getText().toString());
        params.put("No_Tlp", et_notlp.getText().toString());
        params.put("email", et_email.getText().toString());
        params.put("Username", et_username.getText().toString());
        params.put("Password", et_password.getText().toString());
```

Gambar 4.4 *Coding Register*

C. Tampilan Menu Admin

Terdapat tiga menu pada admin yaitu verifikasi bengkel, *update* bengkel dan *update* pengguna.

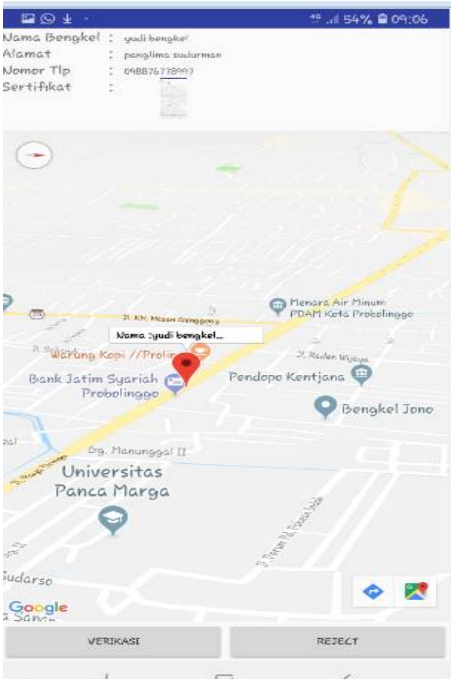


Gambar 4.5 Tampilan Menu Admin

D. Tampilan Verifikasi Bengkel

Pada tampilan verifikasi bengkel akan terdapat list bengkel baru yang akan terdaftar di aplikasi.

Gambar 4.6 Tampilan Verifikasi Bengkel 1



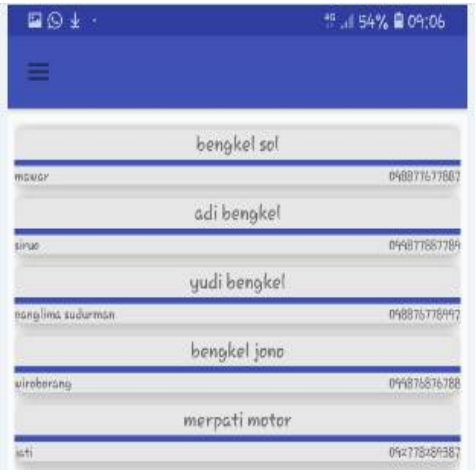
Gambar 4.7 Tampilan Verifikasi Bengkel 2

```
}
private class SendNotification extends AsyncTask<Void, Void, Void> {
    String Respons = "";

    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        params.put("title", "Notification");
        params.put("message", "Bengkel "+(Akses.equals("1") ? "Sudah Di Verifikasi" : "Di Tolak Oleh Admin"));
        params.put("push_type", "individual");
        params.put("include_image", "FALSE");
        params.put("InternalID", iBundle.getString( key: "InternalID"));
        jsonObj = jsonParser.makeHttpRequest( url: "http://"+Link_Url.Url+Link_Url.SendNotification,  method: "POST", params);
        try {
            Respons = jsonObj.getString( name: "Respons");
            jsonArray = jsonObj.getJSONArray( name: "data");
        } catch (JSONException e) {
            Log.e( tag: "Error Login", e.toString());
        }
        return null;
    }
```

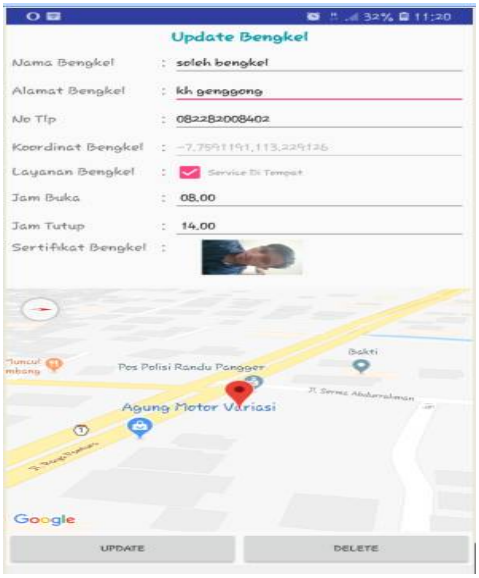
Gambar 4.8 *Coding* verifikasi bengkel

E. Tampilan *Update* Bengkel

Pada tampilan update bengkel akan terdapat list bengkel yang sudah terdaftar di aplikasi. Data bengkel yang sudah dimasukkan bisa diubah ataupun di hapus oleh admin.



Gambar 4.9 Tampilan *Update* Bengkel 1



Gambar 4.10 Tampilan *Update* Bengkel 2

```
public class UpdateBengkel extends AsyncTask<Void,Void,Void> {
    String Respons="";
    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        //7zGetCoordinate();
        params.put("InternalID", ibundle.getString( key: "InternalID"));
        params.put("NamaBengkel", et_nama.getText().toString());
        params.put("AlamatBengkel", et_alamat.getText().toString());
        params.put("noTlp", et_notlp.getText().toString());
        params.put("layanan", ch_layanan.isChecked() ? "1" : "0");
        params.put("Keterangan", "Jam Buka : "+et_buka.getText().toString()+" Jam Tutup : "+et_tutup.getText().toString());
```

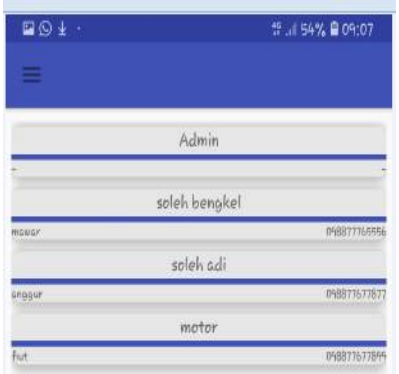
Gambar 4.11 *Coding Update* Bengkel

```
private class DeleteBengkel extends AsyncTask<Void,Void,Void> {
    String Respons="";
    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        //7zGetCoordinate();
        params.put("InternalID", ibundle.getString( key: "InternalID"));
        params.put("Table", "tb_bengkel");
        jsonObj = jsonParser.makeHttpRequest( url: "http://"+Link_Url.Url+ Link_Url.Delete_Data,  method: "POST", params);
        try {
            Respons = jsonObj.getString( name: "Respons");
```

Gambar 4.12 *Coding Delete* Bengkel

F. Tampilan *Update* Pengguna

Pada tampilan *update* pengguna terdapat list *user*, *admin* dan bengkel yang terdaftar di aplikasi. Data user dan bengkel yang terdaftar bisa diubah ataupun dihapus oleh admin. Data tersebut sesuai dengan yang ada di database.



Gambar 4.13 Tampilan *Update* Pengguna 1

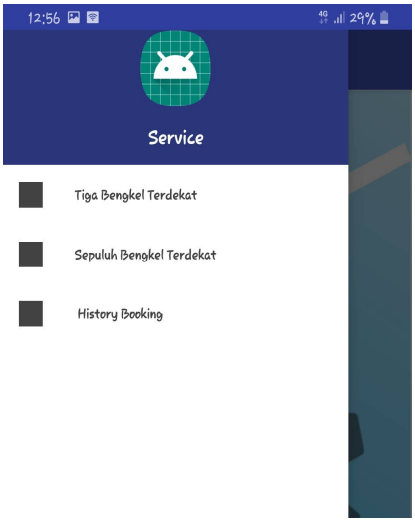
Gambar 4.14 Tampilan *Update* pengguna 2

```
public class UpdatePengguna extends AsyncTask<Void,Void,Void> {
    String Respons="";
    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        //7zGetCoordinate();
        params.put("InternalID", ibundle.getString( key: "InternalID"));
        params.put("NamaPengguna", et_nama.getText().toString());
        params.put("Alamat", et_alamat.getText().toString());
        params.put("No_Tlp", et_notlp.getText().toString());
        params.put("email", et_email.getText().toString());
        jsonObj = jsonParser.makeHttpRequest( url: "http://"+Link_Url.Url+ Link_Url.Update_Pengguna,  method: "POST", params);
        try {
            Respons = jsonObj.getString( name: "Respons");
        } catch (JSONException e) {
            Log.e( tag: "Error Login", e.toString());
        }
        return null;
    }

    @Override
    protected void onPostExecute(Void aVoid) {
        super.onPostExecute(aVoid);
        Log.e( tag: "Respons",Respons);
        if (Respons.equals("succes")) {
            Toast.makeText( context: UpdateDetailPenggunaFragment.this, text: "Update Berhasil",Toast.LENGTH_LONG).show();
            SetControllForm(Respons);
        }else {
```

Gambar 4.15 *Coding Update* pengguna

G. Tampilan menu *Service*/Pengguna

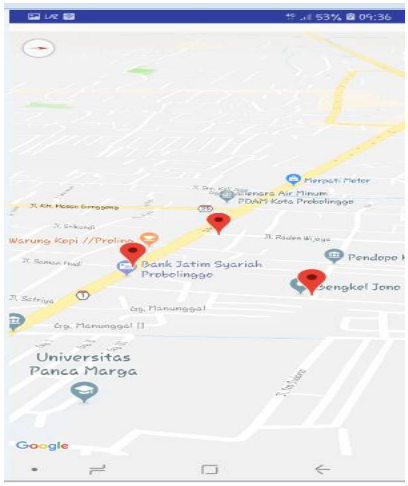
Tampilan menu *Service* terdiri dari tiga komponen yaitu : tiga bengkel terdekat, 10 bengkel terdekat dan *history booking*/pemesanan.



Gambar 4.16 Tampilan Menu *Service*/Pengguna

H. Tampilan menu 3 bengkel terdekat

Tampilan 3 bengkel terdekat merupakan tampilan yang hanya menampilkan 3 bengkel yang ada disekitar pengguna.



Gambar 4.17 Tampilan 3 Bengkel Terdekat

Algoritma menampilkan 3 bengkel terdekat :

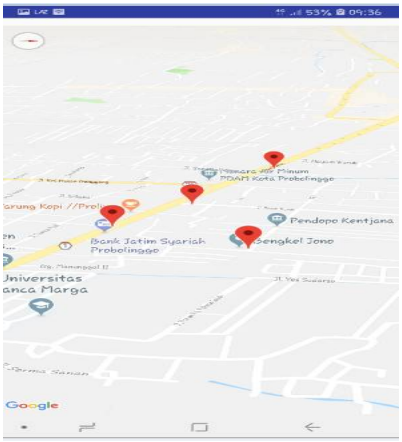
Koordinat dari bengkel sudah ada di database. aplikasi melakukan cek koordinat *smartphone* sekarang. Koordinat yang didapat dikirimkan ke *webservice*. Di *webservice* terdapat *query* untuk mendapatkan koordinat terdekat. Setelah mendapat koordinat, kemudian diteruskan ke aplikasi. Aplikasi akan mengelola data dari *webservice* dan akan menampilkan data tersebut.

```
if(position < 3){
    BengkelModel FC = list_data.get(position);
    final LatLng Lokasi = new LatLng(Double.valueOf(FC.getLatitude()), Double.valueOf(FC.getLongitude()));
    mMap = map;
    mMarker = mMap.addMarker(new MarkerOptions().position(Lokasi).title(position+"Nama :" + FC.getNamaBengkel()
            + "\nAlamat :" + FC.getAlamatBengkel() + "\nTelepon :" + FC.getNoTlp() + "\nKeterangan :" + FC.getKetera
            + "\nLayanan : "+ (FC.getLayanan().toString().equals("1") ? "Layanan Jemput":"Tidak Ada Layanan Jemput")
    array_data_InternalID.add(FC.getInternalID());
    array_data_BengkelID.add(FC.getBengkelID());
    array_data_Nottlp.add(FC.getNoTlp());
      Log.e("Get Layanan",FC.getLayanan().toString());
      Log.e("Marker","Nama :" + FC.getNamaBengkel()
              + "\nAlamat :" + FC.getAlamatBengkel() + "\nTelepon :" + FC.getNoTlp() + "\nKeterangan :" + FC.getKete
              + "\nLayanan : "+ (FC.getLayanan().toString().equals("1") ? "Layanan Jemput":"Tidak Ada Layanan Jemput
    mMarker.setTag(0);
}
position++;
}
```

Gambar 4.18 *Coding* 3 Bengkel Terdekat

I. Tampilan 10 bengkel terdekat

Tampilan 10 bengkel terdekat merupakan tampilan yang hanya menampilkan 10 bengkel yang ada disekitar pengguna.



Gambar 4.19 Tampilan 10 Bengkel Terdekat

Algoritma menampilkan 10 bengkel terdekat :

Koordinat dari bengkel sudah ada di database. Aplikasi Melakukan cek koordinat *smartphone* sekarang. Koordinat yang didapat dikirimkan ke *webservice*. Di *webservice* terdapat *query* untuk mendapatkan koordinat terdekat. Setelah mendapat koordinat, kemudian diteruskan ke aplikasi. Aplikasi akan mengelola data dari *webservice* dan akan menampilkan data tersebut.

```
if(position < 10){
    BengkelModel FC = list_data.get(position);
    final LatLng Lokasi = new LatLng(Double.valueOf(FC.getLatitude()), Double.valueOf(FC.getLongitude()));
    mMap = map;
    mMarker = mMap.addMarker(new MarkerOptions().position(Lokasi).title(position+"Nama :" + FC.getNamaBengkel()
            + "\nAlamat :" + FC.getAlamatBengkel() + "\nTelepon :" + FC.getNoTlp() + "\nKeterangan :" + FC.getKetera
            + "\nLayanan : "+ (FC.getLayanan().toString().equals("1") ? "Layanan Jemput":"Tidak Ada Layanan Jemput")
    array_data_InternalID.add(FC.getInternalID());
    array_data_BengkelID.add(FC.getBengkelID());
    array_data_Nottlp.add(FC.getNoTlp());
    mMarker.setTag(0);
}
position++;
}
```

Gambar 4.20 *Coding* 10 Bengkel Terdekat

J. Tampilan Pemesanan Kerusakan

Tampilan pemesanan kerusakan dilakukan oleh *user* yang ditujukan kepada bengkel. tampilan pemesanan kerusakan berisi form keluhan kerusakan.

Gambar 4.21 Tampilan Pemesanan Kerusakan

```
private class BookingBengkeng extends AsyncTask<Void, Void, Void> {
    String Respons = "";

    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        params.put("BookingID", Booking_ID);
        params.put("UserInternalID", Config.UserInternalID);
        params.put("BengkelInternalID", BengkelInternalID);
        params.put("Keterangan", "Kendala "+SetKendala()+""+et_ket.getText().toString());
        params.put("Status", "pending");
        params.put("latitude", String.valueOf(Latitude));
        params.put("longitude", String.valueOf(Longitude));
        jsonObj = jsonParser.makeHttpRequest( url: "http://" + Link_Url.Url + Link_Url.Booking_Url,  method: "POST", params);
        try {
            Respons = jsonObj.getString( name: "Respons");
        } catch (JSONException e) {
            Log.e( tag: "Error Login", e.toString());
        }
        return null;
    }

    @Override
    protected void onPostExecute(Void aVoid) {
        super.onPostExecute(aVoid);
        if (Respons.equals("succes")) {
            Toast.makeText( context: MapFragment.this,  text: "Booking Berhasil !!!",  Toast.LENGTH_LONG).show();
```

Gambar 4.22 *Coding* Pemesanan Kerusakan

K. Tampilan *history booking*/pemesanan

Tampilan *history booking* terdapat daftar riwayat pemesanan yang dilakukan oleh *user*/pengguna

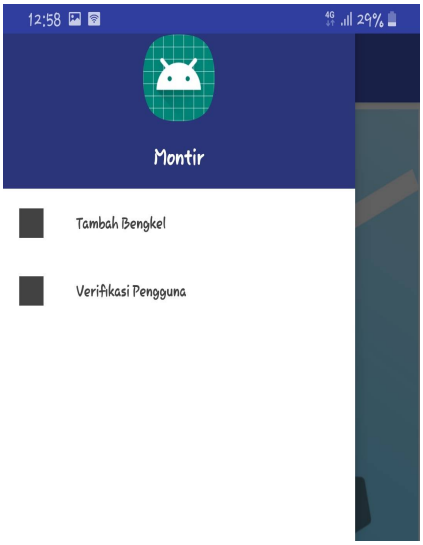
Gambar 4.23 Tampilan *History Booking*/Pemesanan

```
private class GetDataBengkel extends AsyncTask<Void, Void, Void> {
    String Respons = "";

    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        params.put("UserInternalID", Config.UserInternalID);
        jsonObj = jsonParser.makeHttpRequest( url: "http://"+ Link_Url.Url+ Link_Url.GetBookingUser,  method: "POST", params);
        try {
            Respons = jsonObj.getString( name: "Respons");
            jsonArray = jsonObj.getJSONArray( name: "data");
        } catch (JSONException e) {
            Log.e( tag: "Error Login", e.toString());
        }
        return null;
    }

    @Override
    protected void onPostExecute(Void aVoid) {
        super.onPostExecute(aVoid);
        if (Respons.equals("succes")) {
            SetListDataList(jsonArray);
        } else {
```

Gambar 4.24 *Coding History Booking*/Pemesanan

L. Tampilan Menu Montir/Bengkel

Terdapat 2 menu pada tampilan di aplikasi montir yaitu tambah bengkel dan verifikasi pengguna



Gambar 4.25 Tampilan Menu Montir/Bengkel

M. Tampilan Tambah Bengkel

Tampilan tambah bengkel berisi form untuk pendaftaran bengkel. Pihak bengkel yang ingin tempat usahanya terdaftar di aplikasi, maka harus mengisi semua data yang diperlukan.

Gambar 4.26 Tampilan Tambah Bengkel

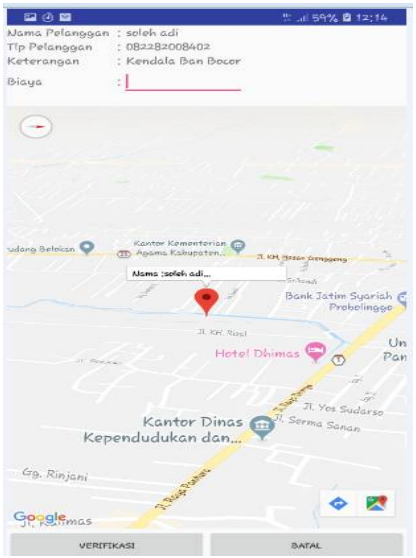
```
private class AddBengkel extends AsyncTask<Void,Void,Void>{
    String Respons="";
    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        //7zGetCoordinate();
        params.put("UserInternalID", Config.UserInternalID );
        params.put("BengkelID", BengkelID);
        params.put("NamaBengkel", et_nama.getText().toString());
        params.put("AlamatBengkel", et_alamat.getText().toString());
        params.put("noTlp", et_notlp.getText().toString());
        params.put("Gambar",BengkelID.replace( target: ".", replacement: "_"));
        params.put("Akses", "-");
        params.put("Latitude", et_lat.getText().toString());
        params.put("Longitude", et_long.getText().toString());
        params.put("Keterangan", "Jam Buka : "+et_buka.getText().toString()+" Jam Tutup : "+et_tutup.getText().toString());
        params.put("Layanan",chlayanan.isChecked() == true ? "1" : "0");
        jsonObj = jsonParser.makeHttpRequest( url: "http://"+Link_Url.Url+Link_Url.AddBengkel_Url,  method: "POST", params);
        try {
            Respons = jsonObj.getString( name: "Respons");
        } catch (JSONException e) {
            Log.e( tag: "Error Login", e.toString());
        }
        return null;
    }

    @Override
    protected void onPostExecute(Void aVoid) {
```

Gambar 4.27 *Coding* Tambah Bengkel

N. Tampilan Verifikasi Pengguna

Tampilan verifikasi pengguna berisi daftar pengguna yang sudah melakukan pemesanan kerusakan. Pemesanan kerusakan dilakukan pada aplikasi *service*.

Gambar 4.28 Tampilan Verifikasi Pengguna 1



Gambar 4.29 Tampilan Verifikasi Pengguna 2

```
private class GetDataBengkel extends AsyncTask<Void, Void, Void> {
    String Respons = "";

    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... voids) {
        HashMap<String, String> params = new HashMap<~>();
        params.put("UserInternalID", Config.UserInternalID);
        jsonObj = jsonParser.makeHttpRequest( url: "http://"+ Link_Url.Url+Link_Url.Get_Data_Booking,  method: "POST", params);
        try {
            Respons = jsonObj.getString( name: "Respons");
            jsonArray = jsonObj.getJSONArray( name: "data");
        } catch (JSONException e) {
            Log.e( tag: "Error Login", e.toString());
        }
        return null;
    }

    @Override
    protected void onPostExecute(Void aVoid) {
        super.onPostExecute(aVoid);
        if (Respons.equals("succes")) {
            SetRecycleViewDataList(jsonArray);
        } else {

        }
    }
```

Gambar 4.30 *Coding* Verifikasi Pengguna

O. Tampilan *Database*

Tampilan database berisi 3 tabel yang digunakan untuk menyimpan ataupun mengambil data. Terdiri dari 3 tabel yaitu : tabel bengkel, tabel booking dan tabel user.

Tabel	Tindakan	Baris	Jenis	Penyortiran	Ukuran	Beban
tb_bengkel	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	6	InnoDB	latin1_swedish_ci	16 KB	-
tb_booking	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	6	InnoDB	latin1_swedish_ci	16 KB	-
tb_users	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	8	InnoDB	latin1_swedish_ci	16 KB	-
3 tabel	Jumlah	20	InnoDB	latin1_swedish_ci	48 KB	0 B

Gambar 4.31 Tampilan *database*

P. Tampilan Tabel Bengkel

Tampilan tabel bengkel merupakan tampilan yang didalamnya terdapat daftar bengkel yang sudah terdaftar di aplikasi

	InternalID	UserInternalID	BengkelID	NamaBengkel	AlamatBengkel	noTlp	Gambar	Akses	layanan	Keterangan	Latitude	Longitude
Ubah Salin Hapus	30	14	BN.1905.000001	yudi bengkel	pangsud	087867865667	BN_1905_000001	1	1	Jam Buka : 09.00 Jam Tutup : 15.00	-7.760773	113.232331
Ubah Salin Hapus	31	16	BN.1905.000002	rajawali bengkel	raden wijaya	0987667877667	BN_1905_000002	1	1	Jam Buka : 08.00 Jam Tutup : 16.00	-7.759694	113.230597
Ubah Salin Hapus	33	17	BN.1905.000003	bengkel jono	priksan	0822789776677	BN_1905_000003	1	0	Jam Buka : 09.00 Jam Tutup : 16.00	-7.758999	113.233784
Ubah Salin Hapus	34	20	BN.1905.000004	halil bengkel	pangsud	082234544377	BN_1905_000004	1	1	Jam Buka : 08.00 Jam Tutup : 17.00	-7.761118	113.232605
Ubah Salin Hapus	35	19	BN.1905.000005	merpati bengkel	pangsud	0811929937272	BN_1905_000005	1	1	Jam Buka : 09.00 Jam Tutup : 16.00	-7.758246	113.227246
Ubah Salin Hapus	36	21	BN.1905.000006	budi bengkel	yos sudarso	098887776667	BN_1905_000006	1	1	Jam Buka : 09.00 Jam Tutup : 16.00	-7.762453	113.238051

Gambar 4.32 Tampilan tabel bengkel

Q. Tampilan Tabel *Booking*

Tampilan tabel *Booking* merupakan tampilan yang didalamnya terdapat daftar pesanan kerusakan yang dikirimkan oleh user kepada bengkel

	InternalID	BookingID	UserInternalID	BengkelInternalID	Keterangan	Biaya	Status	latitude	longitude	dtRecord	dtModified
Ubah Salin Hapus	1	BK.1905.000001	13	33	Kendala Mogok	bayar nanti	verified	-7.7595117	113.2287829	2019-05-24 16:08:40	
Ubah Salin Hapus	2	BK.1905.000002	13	31	Kendala Ganti Oli		pending	-7.7595117	113.2287829	2019-05-24 16:09:16	
Ubah Salin Hapus	3	BK.1905.000003	13	36	Kendala rantai putus	6000	verified	-7.7615189	113.236923	2019-05-25 04:11:03	
Ubah Salin Hapus	4	BK.1905.000004	13	34	Kendala Mogok		pending	-7.7614055	113.2371414	2019-05-25 04:14:21	
Ubah Salin Hapus	5	BK.1905.000005	13	31	Kendala Tali Kopling Putus putus		pending	-7.7594522	113.2287436	2019-05-28 08:51:48	
Ubah Salin Hapus	6	BK.1905.000006	13	31	Kendala Lampu Depan Mati		canceled	-7.7594522	113.2287436	2019-05-28 08:52:34	

Gambar 4.33 Tampilan tabel *booking*

R. Tampilan Tabel *User*

Tampilan tabel user merupakan tampilan yang didalamnya terdapat daftar user yang terdaftar di aplikasi.

	InternalID	PenggunaID	NamaPengguna	Alamat	No_Tlp	email	Username	Password	akses	token_fb
Ubah Salin Hapus	1	AD.0119.000001	Admin	-	-		admin	admin	admin	ffJW9Ep7IzA:APA91bGf8O1UtDufCsmxpmbxOcBqxO8kl-rVQQ...
Ubah Salin Hapus	13	AD.1905.000001	soleh	saman hudi	082282008402		soleh	1234	user	dRLe5JpeZeU:APA91bEvTogPSDxK63A8IR4wUZZkS0ucTNHc0S
Ubah Salin Hapus	14	AD.1905.000001	yudi	pangsud	086788997766		yudi	1234	bengkel	c3d55ssKy2M:APA91bEybTo-qqdzPQj3zSElbZm7VJsWrbxrh2...
Ubah Salin Hapus	16	AD.1905.000001	rajawali	raden wijaya	098666788877766		rajawali	1234	bengkel	c3d55ssKy2M:APA91bEybTo-qqdzPQj3zSElbZm7VJsWrbxrh2...
Ubah Salin Hapus	17	AD.1905.000001	jono	priksan	082267554567		jono	1234	bengkel	exRtyyKhY0E:APA91bFcxO4ghI3HJte9hB4V-rP1UveaPbC5rJ...
Ubah Salin Hapus	19	AD.1905.000001	merpati	jati	0865457766		merpati	1234	bengkel	eBc5MFka3BU:APA91bHhUwoXvgt0a14GsbMXp7k0UgI6nrl3ie...
Ubah Salin Hapus	20	AD.1905.000001	halil	pangsud	082234667866		halil	1234	bengkel	eBc5MFka3BU:APA91bHhUwoXvgt0a14GsbMXp7k0UgI6nrl3ie...
Ubah Salin Hapus	21	AD.1905.000001	budi	yos sudarso	098887776667		budi	1234	bengkel	eBc5MFka3BU:APA91bHhUwoXvgt0a14GsbMXp7k0UgI6nrl3ie...

Gambar 4.34 Tampilan tabel *user*

4.2 Pengujian Sistem

Pengujian sistem merupakan tahap setelah implementasi dilakukan, dimana tahap ini merupakan tahap terpenting karena berkaitan dengan kelayakan program yang dibangun. Pengujian sistem dilakukan dengan dua metode yaitu *black box testing* dan pengujian sistem diberbagai *platform android*.`

4.2.1 *Black Box Testing*

Pengujian *black box* merupakan tahap yang berfokus pada pernyataan fungsional perangkat lunak. Pengujian black box memungkinkan pembuat perangkat lunak mendapat serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program.

1. Tabel Pengujian Aplikasi Android

   a. Pengujian *Register*

Tabel 4.1 Hasil Pengujian *Register Android*

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**

Form *login*	Klik tombol buat akun	-	Muncul form register	Benar
Form *Register*	Klik tombol tambah	Data terisi dengan benar	Tambah pelanggan berhasil	Benar
Form *Register*	Klik tombol tambah	Data kosong	Muncul pesan "Harap mengisi form yang kosong"	Benar

b. Pengujian *Login*

Tabel 4.2 Hasil Pengujian *Login Android*

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Form *login*	Klik tombol *login*	Username : Benar Password : Benar	Muncul Tampilan Utama	Benar
Form *login*	Klik tombol *login*	Username : Salah Password : Salah	Muncul pesan : "Login gagal !!!"	Benar
Form *login*	Klik	Username :	Muncul	Benar

	tombol *login*	Kosong Password : Kosong	pesan : “Login gagal !!!”	

c. Pengujian verifikasi bengkel

Tabel 4.3 Hasil Pengujian verifikasi bengkel

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Bengkel	Klik tombol verifikasi	Data Benar	Verifikasi	Benar
Bengkel	Klik tombol *reject*	Data Salah	*Reject* berhasil	Benar

d. Pengujian *Update* bengkel

Tabel 4.4 Hasil Pengujian *Update* bengkel

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Update Bengkel	Klik tombol *Update*	Data : diganti	Muncul pesan “*Update* berhasil”	Benar

Update Bengkel	Klik tombol *Delete*	Data : dihapus	Muncul pesan "*Delete* berhasil"	Benar

e. Pengujian *Update* pengguna

Tabel 4.5 Hasil Pengujian *Update* pengguna

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Update Pengguna	Klik tombol *update*	Data : diganti	Muncul pesan "*Update* berhasil"	Benar
Update Pengguna	Klik tombol *delete*	Data : dihapus	Muncul pesan "*Delete* berhasil"	Benar

f. Pengujian Tiga Bengkel Terdekat

Tabel 4.6 Hasil Pengujian Tiga Bengkel Terdekat

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Tombol tiga bengkel terdekat	Klik tombol tiga bengkel terdekat	-	Muncul tiga bengkel terdekat di map	Benar

g. Pengujian Sepuluh Bengkel Terdekat

Tabel 4.7 Pengujian Sepuluh Bengkel Terdekat

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Tombol sepuluh bengkel terdekat	Klik tombol sepuluh bengkel terdekat	-	Muncul lebih dari tiga bengkel terdekat atau maksimal sepuluh bengkel terdekat	Benar

h. Pengujian *Hystory Booking*/Pemesanan

Tabel 4.8 Pengujian *Hystory Booking/*Pemesanan

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Tombol *history booking*	Klik tpmbol *history booking*	-	Muncul tampilan daftar riwayat pemesanan kerusakan	Benar

i. Pengujian Pemesanan Kerusakan

Tabel 4.9 Pengujian Pemesanan Kerusakan

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Form bengkel	Klik tombol pesan	Data benar	Muncul pesan“*booking* berhasil”	Benar
Form bengkel	Klik tombol pesan	Data kosong	Muncul pesan ”keterangan/kategori masalah harus diisi”	Benar
Form bengkel	Klik tombol telfon	-	Muncul nomor telfon	Benar

j. Pengujian Tambah Bengkel

Tabel 4.10 Pengujian Tambah Bengkel

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Form Tambah Bengkel	Klik tombol tambah	Data benar	Muncul pesan “tambah bengkel berhasil”	Benar

Form Tambah Bengkel	Klik tambah bengkel	Data Kosong	Muncul pesan “harap mengisi semua data”	Benar

k. Pengujian Verifikasi Pengguna

Tabel 4.11 Pengujian Verifikasi Pengguna

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Tampilan pengguna	Klik tombol verifikasi	Biaya diisi	Muncul pesan “notifikasi terkirim”	Benar
Tampilan pengguna	Klik tombol batal	-	Muncul pesan “notifikasi terkirim”	

l. Pengujian Keluar

Tabel 4.12 Pengujian Keluar

Interface	**Yang diuji**	**Kondisi**	***Output***	**Status**
Menu keluar	Klik tombol Keluar	-	Muncul tampilan form *login*	Benar

4.2.3 Pengujian Sistem di Berbagai *Platform Android*

Pengujian di berbagai platform di android sangat diperlukan karena perangkat android sangat beragam. Berikut beberapa perangkat dan platform android yang diujikan.

Tabel 4.13 Pengujian *Platform Android*

Perangkat	**Operating System**	**Resolusi Layar(pixels)**	**Hasil**
Samsung j4+	Android 9.0 (Pie)	(1280x720) 6 inci	Berjalan dengan baik
Samsung A8	Android 8.0 (Oreo)	(1080x2220) 5.6 inci	Berjalan dengan baik
Oppo A11w	Android 4.0 (Kitkat)	(480x854) 4.5 inci	Terjadi kesalahan saat menguraikan Paket
Xperia A4	Android 6.0 (Marsmellow)	(720x1280) 4.6 inci	Berjalan dengan baik
Oppo A37	Android 5.1 (Lolipop)	(720x1280) 5 inci	Berjalan dengan baik

4.2.3 Kuisioner Penggunaan Aplikasi

Pengujian dalam bentuk Kuisioner ini terdiri dari 10 pernyataan yang disebarkan pada 10 responden. Kuisioner dibuat menggunakan skala likert dari skala 1 sampai 4. Berdasarkan data yang dihasilkan dari Kuisioner, dilakukan perhitungan menggunakan skala likert. Skala likert adalah metode perhitungan yang digunakan untuk keperluan riset atas jawaban setuju atau tidaknya seorang responden terhadap suatu pernyataan. Untuk menghitung skor maksimum tiap

jawaban, dengan mengalikan skor dengan jumlah keseluruhan responden, yaitu skor dikali 10 responden.

Tabel 4.14 Skor Maksimum

Jawaban	**Skor**	**Skor Maksimum (Skor * Jumlah Responden)**
Sangat sering	4	40
Sering	3	30
Kadang-kadang	2	20
Tidak pernah	1	10

Setelah itu, dapat dicari presentase masing-masing jawaban dengan menggunakan rumus :

$$Y = \frac{TS}{SKOR\ IDEAL}\ x\ 100\%$$

Dimana:

Y = Nilai persentase

TS= Total skor responden = ∑ skor x responden

Skor ideal= skor x jumlah responden = 4 x 10 = 40

Tabel 4.15 Kriteria Skor

Kategori	**Keterangan**
0%-25%	Tidak pernah
26%-50%	Kadang-kadang
51%-75%	Sering
76%-100%	Sangat sering

Berikut ini adalah hasil persentase masing–masing jawaban yang sudah dihitung nilainya. Kuisioner ini telah diujikan kepada 10 orang responden (hasil Kuisioner dilampirkan).

1. Pernyataan pertama

Hasil Kuisioner pernyataan pertama dapat dilihat pada tabel 4.16.

Tabel 4.16 Hasil Kuisioner Pernyataan Pertama

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
1	Sangat sering	4	0	0	(23:40)x100 = 57.5%
	Sering	3	3	9	
	Kadang-kadang	2	7	14	
	Tidak pernah	1	0	0	
Jumlah			10	23	

Berdasarkan nilai persentase dari pernyataan pertama, dapat disimpulkan sebanyak 57.5% responden menyatakan sering mengalami ban bocor

2. Pernyataan pertama

Hasil Kuisioner pernyataan kedua dapat dilihat pada tabel 4.17.

Tabel 4.17 Hasil Kuisioner Pernyataan kedua

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
2	Sangat sering	4	0	0	(20:40)x100 =50%
	Sering	3	0	0	
	Kadang-kadang	2	10	20	
	Tidak pernah	1	0	0	
Jumlah			10	20	

Berdasarkan nilai persentase dari pernyataan kedua, dapat disimpulkan sebanyak 50% responden menyatakan kadang - kadang mengalami oli habis/ganti oli

3. Pernyataan ketiga

Hasil Kuisioner pernyataan ketiga dapat dilihat pada tabel 4.18.

Tabel 4.18 Hasil Kuisioner Pernyataan ketiga

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
3	Sangat sering	4	0	0	(11:40)x100 =27.5%
	Sering	3	0	0	
	Kadang-kadang	2	1	2	
	Tidak pernah	1	9	9	
Jumlah			10	11	

Berdasarkan nilai persentase dari pernyataan ketiga, dapat disimpulkan sebanyak 27.5% responden menyatakan kadang - kadang mengalami rantai putus pada sepeda motor

4. Pernyataan keempat

Hasil Kuisioner pernyataan keempat dapat dilihat pada tabel 4.19.

Tabel 4.19 Hasil Kuisioner Pernyataan keempat

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
4	Sangat sering	4	0	0	(20:40)x100 =50%
	Sering	3	1	3	
	Kadang-kadang	2	8	16	
	Tidak pernah	1	1	1	
Jumlah			10	20	

Berdasarkan nilai persentase dari pernyataan keempat, dapat disimpulkan sebanyak 50% responden menyatakan kadang - kadang mengalami lampu depan mati pada sepeda motornya

5. Pernyataan kelima

Hasil Kuisioner pernyataan kelima dapat dilihat pada tabel 4.20.

Tabel 4.20 Hasil Kuisioner Pernyataan kelima

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
5	Sangat sering	4	0	0	(16:40)x100 =40%
	Sering	3	0	0	
	Kadang-kadang	2	6	12	
	Tidak pernah	1	4	4	
Jumlah			10	16	

Berdasarkan nilai persentase dari pernyataan kelima, dapat disimpulkan sebanyak 40% responden menyatakan kadang - kadang mengalami lampu belakang mati pada sepeda motornya

6. Pernyataan keenam

Hasil Kuisioner pernyataan keenam dapat dilihat pada tabel 4.21.

Tabel 4.21 Hasil Kuisioner Pernyataan Keenam

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
6	Sangat sering	4	0	0	(10:40)x100 =25%
	Sering	3	0	0	
	Kadang-kadang	2	0	0	
	Tidak pernah	1	10	10	
Jumlah			10	10	

Berdasarkan nilai persentase dari pernyataan keenam, dapat disimpulkan sebanyak 25% responden menyatakan tidak pernah mengalami tali rem putus pada sepeda motornya

7. Pernyataan ketujuh

Hasil Kuisioner pernyataan ketujuh dapat dilihat pada tabel 4.22.

Tabel 4.22 Hasil Kuisioner Pernyataan Ketujuh

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
7	Sangat sering	4	0	0	(12:40)x100 =30%
	Sering	3	0	0	
	Kadang-kadang	2	2	4	
	Tidak pernah	1	8	8	
Jumlah			10	12	

Berdasarkan nilai persentase dari pernyataan ketujuh, dapat disimpulkan sebanyak 30% responden menyatakan kadang - kadang mengalami tali kopling putus pada sepeda motornya

8. Pernyataan kedelapan

Hasil Kuisioner pernyataan kedelapan dapat dilihat pada tabel 4.23.

Tabel 4.23 Hasil Kuisioner Pernyataan Kedelapan

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
8	Sangat sering	4	0	0	(10:40)x100 =25%
	Sering	3	0	0	
	Kadang-kadang	2	0	0	
	Tidak pernah	1	10	10	
Jumlah			10	10	

Berdasarkan nilai persentase dari pernyataan kedelapan, dapat disimpulkan sebanyak 25% responden menyatakan tidak pernah mengalami tali gas putus pada sepeda motornya

9. Pernyataan kesembilan

Hasil Kuisioner pernyataan kesembilan dapat dilihat pada tabel 4.24.

Tabel 4.24 Hasil Kuisioner Pernyataan Kesembilan

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
9	Sangat sering	4	0	0	(22:40)x100 =55%
	Sering	3	2	6	
	Kadang-kadang	2	8	16	
	Tidak pernah	1	0	0	
Jumlah			10	22	

Berdasarkan nilai persentase dari pernyataan kesembilan, dapat disimpulkan sebanyak 55% responden menyatakan sering mengalami starter mati pada sepeda motornya

10. Pernyataan kesepuluh

Hasil Kuisioner pernyataan kesepuluh dapat dilihat pada tabel 4.25.

Tabel 4.25 Hasil Kuisioner Pernyataan Kesepuluh

Pernyataan	**Jawaban**	**Skor**	**Responden**	**Jumlah Skor**	**Nilai Presentase (%)**
10	Sangat sering	4	0	0	(18:40)x100 =45%
	Sering	3	0	0	
	Kadang-kadang	2	8	16	
	Tidak pernah	1	2	2	
Jumlah			10	18	

Berdasarkan nilai persentase dari pernyataan kesepuluh, dapat disimpulkan sebanyak 45% responden menyatakan kadang - kadang mengalami busi mati pada sepeda motornya.

Hasil dari setiap pertanyaan dilakukan perhitungan rata-rata secara keseluruhan. Kemudian akan dibandingkan dengan Tabel 4.15 untuk diambil kesimpulan.Perhitungan secara keseluruhan pengolahan Kuisioner dapat dilihat pada Tabel 4.26.

Tabel 4.26 Pengolahan Skala

Pertanyaan	**Nilai Presentase**	**Keterangan**
1	57.5%	Sering
2	50%	Kadang – kadang
3	27.5%	Kadang – kadang
4	50%	Kadang – kadang
5	40%	Kadang – kadang
6	25%	Tidak pernah
7	30%	Kadang – kadang
8	25%	Tidak pernah
9	55%	Sering
10	45%	Kadang - kadang
Total Presentase	57.5%+50%+27.5%+50%+40%+25%+30%+25%+55%+45%=405%	Kadang - kadang
Rata-rata	405%:10=40.5%	

Hasil dari pengujian Kuisioner diperoleh dari 10 pernyataan yang diberikan kepada 10 responden menyatakan 40.5%, yang berarti kadang – kadang sepeda motor mengalami kerusakan.