BAB IV

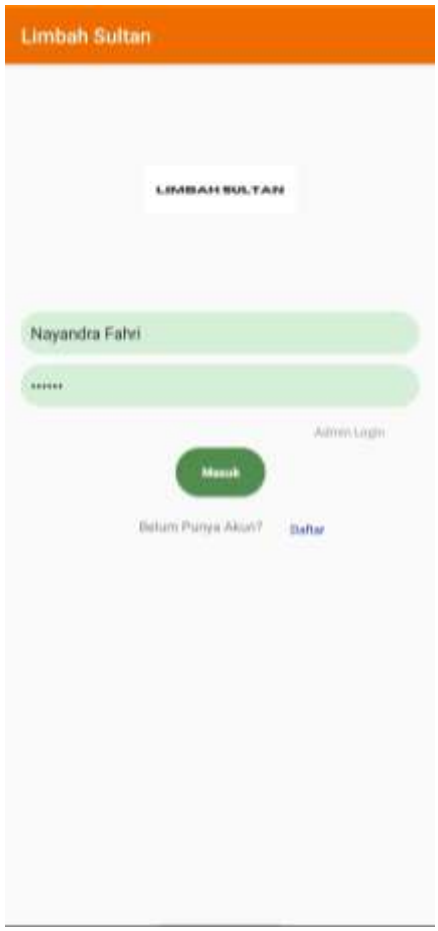
HASIL DAN PEMBAHASAN

4.1 Hasil Penelitian

Setelah melakukan tahapan pengumpulan data, perancangan sistem, dan implementasi ke dalam bentuk program, kemudian dihasilkan sebuah perangkat lunak yang dapat memfasilitasi Toko Limbah Sultan dalam menjual produk-produknya. Hasil perangkat lunak dari aplikasi *e-commerce* dalam bentuk tampilan antarmuka adalah sebagai berikut:

A. Halaman *Login User*

Halaman *Login User* digunakan oleh *user* atau pelanggan yang akan menggunakan aplikasi *e-commerce* untuk belanja di Toko Limbah Sultan. Berikut adalah tampilan Halaman *Login User:*



Gambar 4. 1 Antarmuka *Login User*

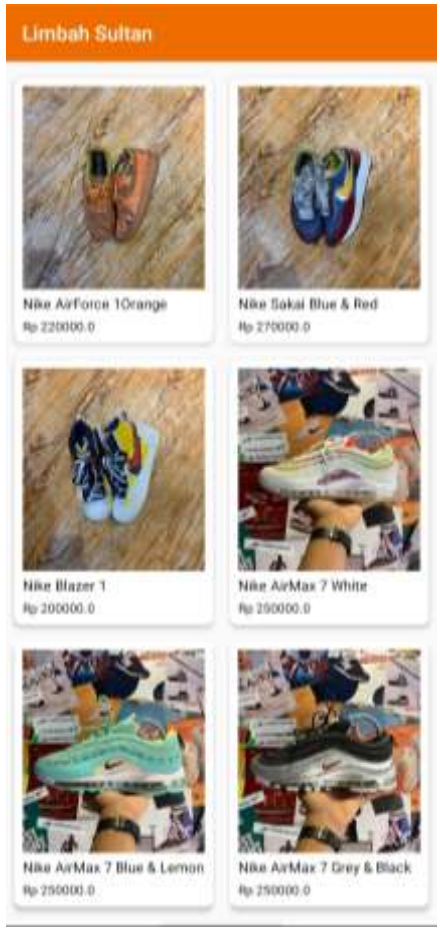
B. Halaman Beranda

Halaman Beranda merupakan halaman yang digunakan oleh *user* atau pelanggan. Halaman ini akan tampil jika *user* atau pelanggan berhasil melakukan *login.* Berikut adalah tampilan Halaman Beranda:

Gambar 4. 2 Antarmuka Beranda

C. Halaman Produk

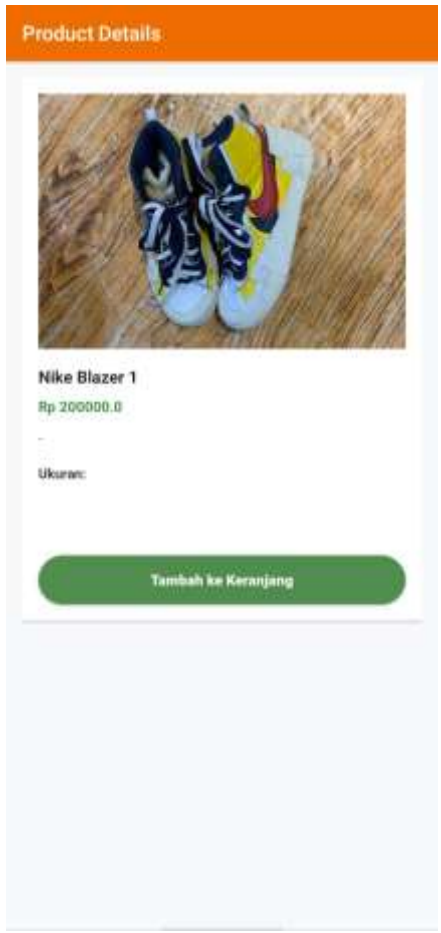
Halaman Produk merupakah halaman yang digunakan oleh *user* atau pelanggan untuk melihat dan memilih produk apa saja yang tersedia. Berikut adalah tampilan Halaman Produk:



Gambar 4. 3 Antarmuka Halaman Produk

D. Halaman Detail Produk

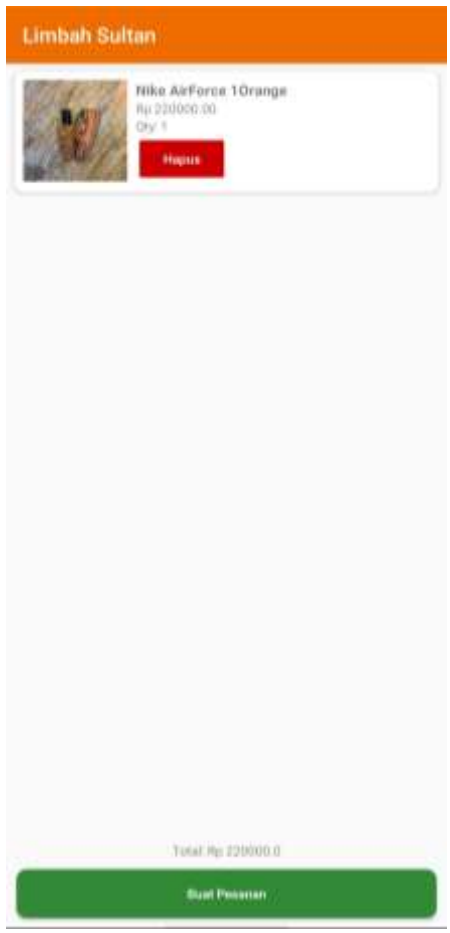
Halaman Detail Produk adalah halaman informasi mengenai suatu produk yang ada di dalam aplikasi. Berikut adalah tampilan Halaman Detail Produk:



Gambar 4. 4 Antarmuka Halaman Detail Produk

E. Halaman Keranjang

Halaman Keranjang merupakan halaman yang digunakan oleh *user* atau pelanggan untuk menyimpan produk sebelum dibeli. Berikut adalah tampilan Halaman Keranjang:



Gambar 4. 5 Antarmuka Halaman Keranjang

F. Halaman Detail Pengiriman

Halaman Detail Pengiriman adalah halaman yang digunakan oleh *user* atau pelanggan untuk melakukan pengisian data pribadi sebelum melakukan pembayaran. Berikut adalah tampilan Halaman Detail Pengiriman:

Gambar 4. 6 Antarmuka Halaman Detail Pengiriman

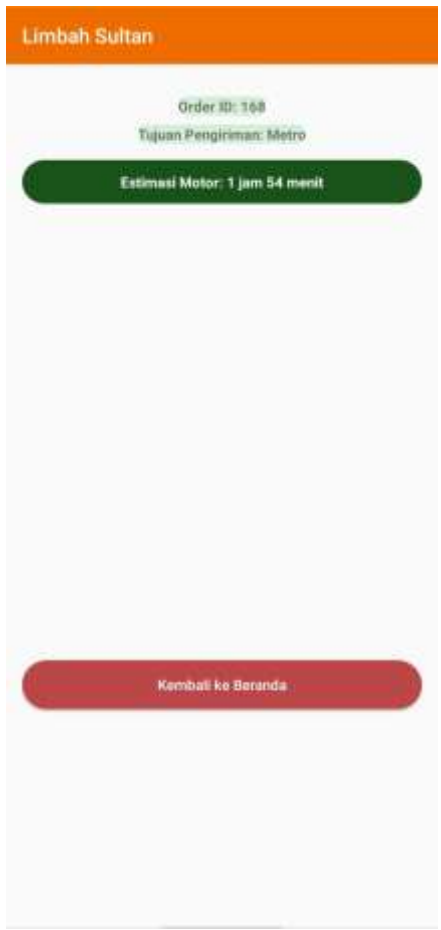
G. Halaman Pembayaran

Halaman Pembayaran digunakan oleh *user* atau pelanggan untuk melakukan transaksi untuk membayaran produk yang akan dibeli. Berikut adalah tampilan Halaman Pembayaran:

Gambar 4. 7 Antarmuka Halaman Pembayaran

H. Halaman Prediksi Kedatangan Paket

Halaman Prediksi Kedatangan Paket adalah halaman yang menampilkan hasil angka prediksi yang terdapat *order id,* tujuan pengiriman, estimasi mobil, estimasi motor. Berikut adalah tampilan Halaman Prediksi Kedatangan Paket:



Gambar 4. 8 Antarmuka Halaman Prediksi Kedatangan Paket

I. Halaman *Login* Admin

Halaman *Login* Admin digunakan oleh admin untuk masuk ke aplikasi. Berikut adalah tampilan Halaman *Login* Admin:

Gambar 4. 9 Antarmuka Halaman *Login* Admin

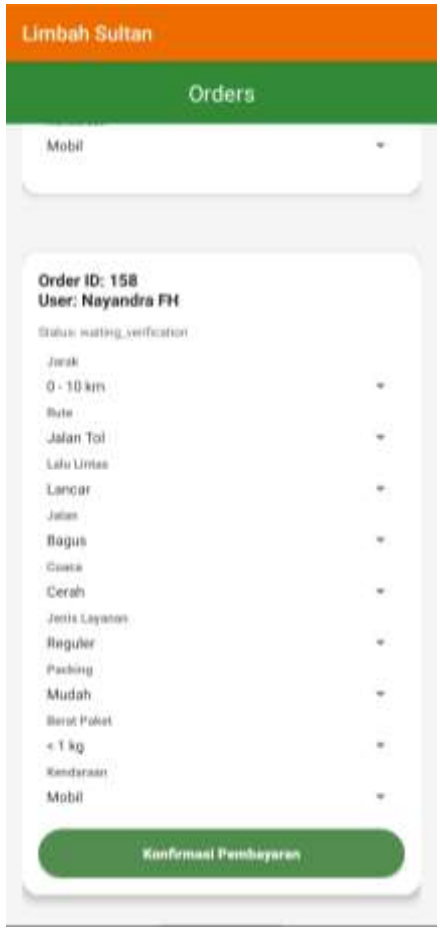
J. Halaman Produk Admin

Halaman Produk Admin digunakan oleh admin untuk mengelola toko seperti menambahkan produk, melakukan validasi pembayaran yang telah dilakukan oleh *user* atau pelanggan. Berikut adalah tampilan Halaman Produk Admin:

Gambar 4. 10 Antarmuka Halaman Produk Admin

K. Halaman Pesanan Admin

Halaman Pesanan Admin digunakan oleh admin untuk melihat dan memvalidasi pembayaran yang telah dilakukan oleh *user* atau pelanggan. Berikut adalah tampilan Halaman Pesanan:



Gambar 4. 11 Antarmuka Halaman Pesanan Admin

L. Halaman Menambah Produk

Halaman Menambahkan Produk digunakan oleh admin untuk menambahkan data produk baru ke dalam sistem. Berikut adalah tampilan Halaman Menambah Produk:

Gambar 4. 12 Antarmuka Halaman Menambah Produk

4.2 Pengujian Sistem

Pengujian sistem adalah tahap terakhir dalam pengembangan sistem. Pada tahap ini, sistem akan diuji untuk mengetahui hasil dari semua proses dan fitur yang telah dibangun. Pengujian *blackbox* adalah metode pengujian perangkat lunak. Sebaliknya, pengujian kotak hitam hanya dapat melihat nilai keluaran berdasarkan nilai masukan itu sendiri[20]. Berikut adalah tabel pengujian antarmuka:

Tabel 4. 1 *Black Box Testing* Antarmuka

Halaman	Aktor	Status
Halaman *Login*	Admin dan *User*	Berhasil
Halaman Produk Admin	Admin	Berhasil
Halaman Lihat Pesanan	Admin	Berhasil

Halaman Tambah Produk Baru	Admin	Berhasil
Halaman Beranda	*User*	Berhasil
Halaman Produk	*User*	Berhasil
Halaman Detail Produk	*User*	Berhasil
Halaman Keranjang	*User*	Berhasil
Halaman Detail Pengiriman	*User*	Berhasil
Halaman Pembayaran	*User*	Berhasil
Halaman Antarmuka Halaman Prediksi Kedatangan Paket	*User*	Berhasil
Halaman Tentang Kami	*User*	Berhasil

Pengujian *blackbox* juga akan menguji fungsionalitas fungsi perangkat lunak saat ini selain menguji tampilan antarmuka. Berikut adalah tabel fungsionalitas:

Tabel 4. 2 *Black Box Testing* Fungsionalitas

Fungsi	Aktor	Status
Fungsi *Login*	Admin dan *User*	Berhasil
Fungsi *Register*	*User*	Berhasil
Fungsi Lihat Pesanan	Admin	Berhasil
Fungsi Tambah Produk Baru	Admin	Berhasil
Fungsi Beranda	*User*	Berhasil
Fungsi Lihat Detail Produk	*User*	Berhasil
Fungsi Keranjang	*User*	Berhasil
Fungsi Detail Pengiriman	*User*	Berhasil
Fungsi Pembayaran	*User*	Berhasil
Fungsi Antarmuka Halaman Prediksi Kedatangan Paket	*User*	Berhasil
Fungsi Tentang Kami	*User*	Berhasil

4.3 Nilai Dasar Jarak

Dalam penelitian ini, nilai dasar jarak digunakan sebagai salah satu variabel penting dalam proses prediksi waktu kedatangan paket. Setiap pengiriman pada Toko Limbah Sultan yang berlokasi di Kota Bandar Lampung memiliki tujuan pengantaran ke berbagai kota dan kabupaten di Provinsi Lampung. Oleh karena itu, jarak antara pusat distribusi di Bandar Lampung dengan daerah tujuan dijadikan sebagai parameter utama dalam perhitungan estimasi waktu tempuh dan waktu kedatangan paket.

Tabel 4. 3 Nilai Dasar Faktor

Kota/Kabupaten	Jarak	Mobil	Motor
Metro	+- 52 km	+- 1 jam 15 menit	+- 1 jam 30 menit
Lampung Selatan	+- 35 km	+- 45 menit	+- 1 jam
Pesawaran	+- 25 km	+- 40 menit	+- 50 menit
Pringsewu	+- 35 km	+- 45 menit	+- 1 jam
Tanggamus	+- 90 km	+- 2 jam	+- 2 jam 30 menit
Lampung Tengah	+- 65 km	+- 1 jam 30 menit	+- 1 jam 45 menit
Lampung Timur	+- 80 km	+- 2 jam	+- 2 jam 30 menit
Lampung Utara	+- 120 km	+- 3 jam	+- 3 jam 30 menit
Way Kanan	+- 220 km	+- 5 jam	+- 6 jam
Lampung Barat	+- 240 km	+- 6 jam	+- 7 jam
Pesisir Barat	+- 270 km	+- 7 jam	+- 8 jam
Tulang Bawang	+- 150 km	+- 3 jam 30 menit	+- 4 jam
Tulang Bawang Barat	+- 140 km	+- 3 jam 15 menit	+- 3 jam 45 menit
Mesuji	+- 220 km	+- 5 jam	+- 6 jam

4.4 Konversi Nilai Faktor-Faktor Waktu Kedatangan Paket

Berdasarkan proses prediksi waktu kedatangan paket terhadap proses logistik dan pengiriman di wilayah Lampung, sejumlah variabel digunakan untuk memprediksi waktu kedatangan paket. Setiap variabel akan mempengaruhi estimasi waktu kedatangan. Nilai-nilai ini digunakan sebagai input untuk pelatihan model *Random Forest.*

Berikut adalah daftar variabel beserta nilai yang dikonversi:

4.4.1 Jarak

Faktor Jarak juga memengaruhi waktu kedatangan paket. Berikut adalah nilai dari Jarak:

Tabel 4. 4 Jarak

Jarak	**Nilai**
0 - 10 km	+0.00
11 - 50 km	+0.05
51 – 100 km	+0.10
101 – 200 km	+0.15
200+ km	+0.20

4.4.2 Rute

Waktu pengantaran dipengaruhi oleh rute pengiriman paket. Nilai rutenya adalah sebagai berikut:

Tabel 4. 5 Rute

Rute	**Nilai**
Jalan Tol	+0.05
Jalan Kota	+0.05
Via Jalur Lintas Barat	+0.10

Via Jalur Lintas Timur	+0.10

4.4.3 Kondisi Lalu Lintas

Kondisi Lalu Lintas juga dapat memengaruhi waktu kedatangan paket. Berikut adalah nilai Kondisi Lalu Lintas:

Tabel 4. 6 Kondisi Lalu Lintas

Kondisi Lalu Lintas	**Nilai**
Normal	+0.00
Macet	+0.20

4.4.4 Kondisi Jalan

Selama proses pengiriman, kecepatan kendaraan dipengaruhi oleh kondisi jalan. Jalan yang rusak cenderung memperlambat laju pengiriman.. Berikut nilai Kondisi Jalan:

Tabel 4. 7 Kondisi Jalan

Kondisi Jalan	**Nilai**
Bagus	+0.00
Rusak	+0.10

4.4.5 Kondisi Cuaca

Cuaca buruk dapat mengganggu transportasi secara langsung maupun tidak langsung, seperti jalanan licin atau genangan air. Berikut nilai Kondisi Cuaca:

Tabel 4. 8 Kondisi Cuaca

Kondisi Cuaca	**Nilai**
Cerah	+0.00
Hujan Ringan	+0.10
Hujan Deras	+0.20

4.4.6 Jenis Layanan

Estimasi waktu pengantaran juga dipengaruhi oleh jenis layanan yang dipilih pengguna. Penyedia layanan biasanya memprioritaskan layanan ekspres.

Tabel 4. 9 Jenis Layanan

Jenis Layanan	**Nilai**
Reguler	+0.00
Ekspres	-0.10

4.4.7 Packing

Sebelum barang dapat dikirimkan, proses *packing* yang lebih sulit dapat membutuhkan waktu tambahan.

Tabel 4. 10 *Packing*

Tingkat Kesulitan *Packing*	**Nilai**
Mudah	+0.00
Sulit	+0.10

4.4.8 Berat Paket

Kecepatan distribusi dan pengangkutan dipengaruhi oleh berat paket, terutama dalam pengiriman massal.

Tabel 4. 11 Berat Paket

Berat Paket	**Nilai**
< 1 kg	+0.00
1 - 5 kg	+0.02
5 - 10 kg	+0.05
10+ kg	+0.10

Nilai prediksi untuk waktu kedatangan paket ke Metro adalah 1 jam 35 menit menggunakan mobil dan 1 jam 54 menit menggunakan motor. Waktu tempuh dipengaruhi oleh berbagai kondisi yang dapat menambah waktu pengiriman.

Berikut faktor-faktor yang memengaruhi:

Tabel 4. 12 Nilai Faktor Tujuan Metro

Variabel	**Kondisi**	**Nilai**
Jarak	51 – 100 km	+0.10
Rute	Jalan Kota	+0.05
Lalu Lintas	Macet	+0.20
Kondisi Jalan	Bagus	+0.00
Cuaca	Cerah	+0.00
Layanan	Ekspres	-0.10
Packing	Mudah	+0.00
Berat Paket	1-5 kg	+0.02

Dengan total penjumlahan (1.00) + (0.10) + (0.05) + (0.20) + (0.00) + (0.00) - (0.10) + (0.00) + (0.02) = **1.27.** Maka:

$$\text{Estimasi Akhir} = \text{Estimasi Dasar} \times 1{,}27$$

Dengan hitungan estimasi sebagai berikut:

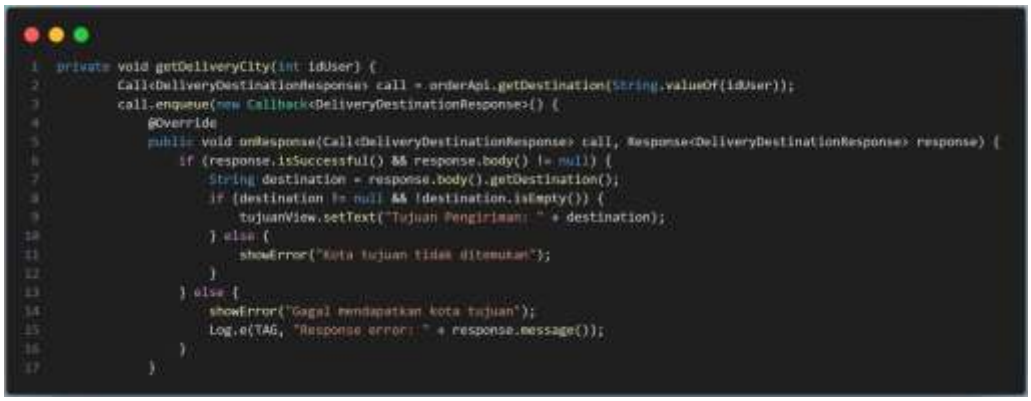
a. Mobil: 75 menit × 1.27 = 1 jam 35 menit
b. Motor: 90 menit × 1.27 = 1 jam 54 menit

4.5 Implementasi Algoritma

Berikut merupakan hasil penerapan algoritma pada sistem dalam bahasa pemrograman python:

4.5.1 GetDeliveryCity

Kode ini menggunakan Retrofit untuk mengambil dan menampilkan kota tujuan pengiriman (destination) berdasarkan ID user (idUser) dari server. Di bawah ini merupakan kode untuk melakukan *getDeliveryCity:*



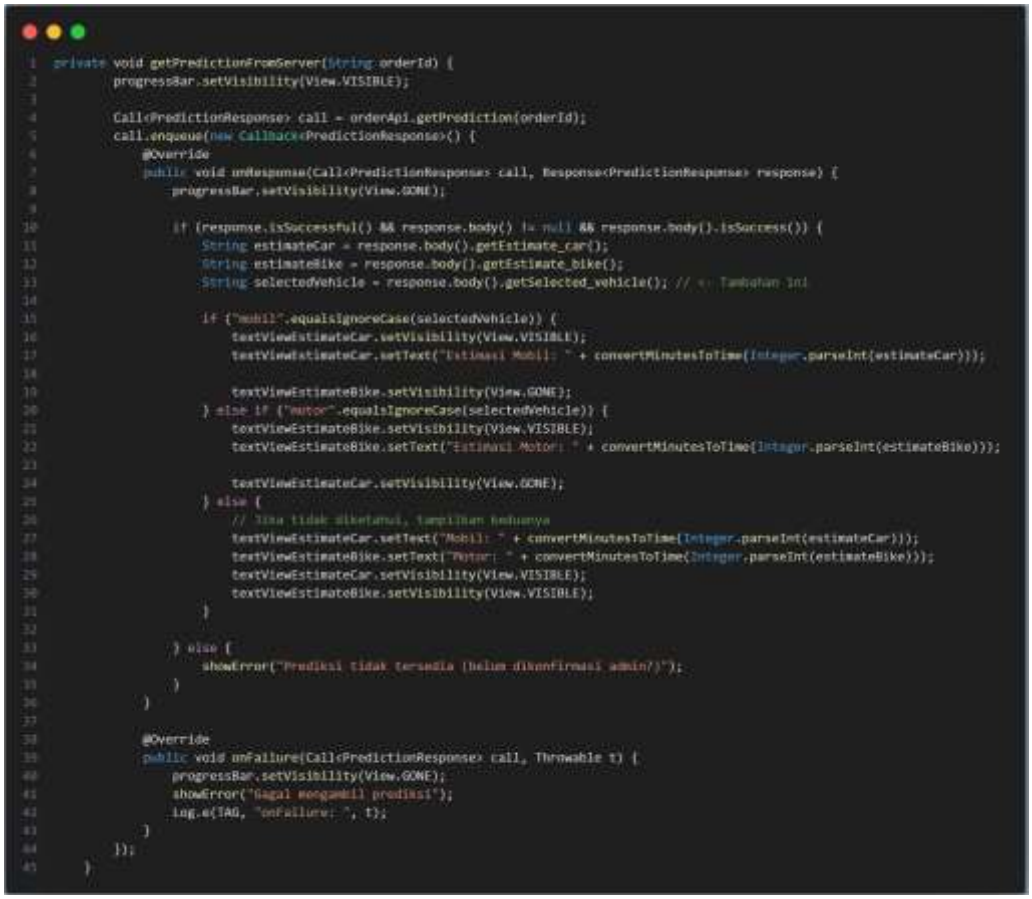
```
private void getDeliveryCity(int idUser) {
        Call<DeliveryDestinationResponse> call = orderApi.getDestination(String.valueOf(idUser));
        call.enqueue(new Callback<DeliveryDestinationResponse>() {
            @Override
            public void onResponse(Call<DeliveryDestinationResponse> call, Response<DeliveryDestinationResponse> response) {
                if (response.isSuccessful() && response.body() != null) {
                    String destination = response.body().getDestination();
                    if (destination != null && !destination.isEmpty()) {
                        tujuanView.setText("Tujuan Pengiriman: " + destination);
                    } else {
                        showError("Kota tujuan tidak ditemukan");
                    }
                } else {
                    showError("Gagal mendapatkan kota tujuan");
                    Log.e(TAG, "Response error: " + response.message());
                }
            }
```

Namun, jika terjadi *error* seperti respons kosong, identifikasi *user* tidak ditemukan, atau koneksi gagal, aplikasi akan menampilkan pesan kesalahan kepada pengguna dan mencatat kesalahan log di *logcat* untuk *debugging.*

```
@Override
            public void onFailure(Call<DeliveryDestinationResponse> call, Throwable t) {
                showError("Gagal koneksi ke server");
                Log.e(TAG, "onFailure: ", t);
            }
        });
    }
```

4.5.2 GetPredictionFromServer

Bagian ini adalah kode yang dapat mengambil dan menampilkan perkiraan waktu pengiriman dari server berdasarkan kendaraan yang dipilih oleh admin, seperti mobil atau motor. Di bawah ini merupakan kode untuk melakukan *GetPredictionFromServer:*



```
private void getPredictionFromServer(String orderId) {
        progressBar.setVisibility(View.VISIBLE);

        Call<PredictionResponse> call = orderApi.getPrediction(orderId);
        call.enqueue(new Callback<PredictionResponse>() {
            @Override
            public void onResponse(Call<PredictionResponse> call, Response<PredictionResponse> response) {
                progressBar.setVisibility(View.GONE);

                if (response.isSuccessful() && response.body() != null && response.body().isSuccess()) {
                    String estimateCar = response.body().getEstimate_car();
                    String estimateBike = response.body().getEstimate_bike();
                    String selectedVehicle = response.body().getSelected_vehicle(); // <- Tambahan ini

                    if ("mobil".equalsIgnoreCase(selectedVehicle)) {
                        textViewEstimateCar.setVisibility(View.VISIBLE);
                        textViewEstimateCar.setText("Estimasi Mobil: " + convertMinutesToTime(Integer.parseInt(estimateCar)));

                        textViewEstimateBike.setVisibility(View.GONE);
                    } else if ("motor".equalsIgnoreCase(selectedVehicle)) {
                        textViewEstimateBike.setVisibility(View.VISIBLE);
                        textViewEstimateBike.setText("Estimasi Motor: " + convertMinutesToTime(Integer.parseInt(estimateBike)));

                        textViewEstimateCar.setVisibility(View.GONE);
                    } else {
                        // Jika tidak diketahui, tampilkan keduanya
                        textViewEstimateCar.setText("Mobil: " + convertMinutesToTime(Integer.parseInt(estimateCar)));
                        textViewEstimateBike.setText("Motor: " + convertMinutesToTime(Integer.parseInt(estimateBike)));
                        textViewEstimateCar.setVisibility(View.VISIBLE);
                        textViewEstimateBike.setVisibility(View.VISIBLE);
                    }

                } else {
                    showError("Prediksi tidak tersedia (belum dikonfirmasi admin?)");
                }
            }

            @Override
            public void onFailure(Call<PredictionResponse> call, Throwable t) {
                progressBar.setVisibility(View.GONE);
                showError("Gagal mengambil prediksi");
                Log.e(TAG, "onFailure: ", t);
            }
        });
    }
```

Berdasarkan kode di atas, diketahui bahwa:

a. Baris *Call<PredictionResponse> call = orderApi.getPrediction(orderId);* ini digunakan untuk memanggil API *backend* yang menerima *orderId* sebagai parameter, dan mengembalikan *response* prediksi estimasi waktu kedatangan paket (baik untuk mobil atau motor) berdasarkan data yang telah dikonfirmasi admin sebelumnya.

b. Baris *call.enqueue(new Callback<PredictionRepsonse>(){…})* mengeksekusi permintaan API secara asinkron. Ini berarti bahwa UI tidak akan diblokir saat menunggu tanggapan *server.*

c. Dalam blok *onResponse*(…), sistem mengevaluasi tanggapan server sebagai berhasil atau kosong (*response.isSuccessful*() && *response.body*()!= *null* && *response.body*().*isSuccess*()). Seandainya berhasil:.

   - Dengan menggunakan *getEstimate_car*() dan *getEstimate_bike*(), data estimasi waktu untuk mobil dan motor dikumpulkan.

- Pengecekan dilakukan setelah mendapatkan kendaraan pilihan admin dari *getSelected_vehicle*():

a. Jika kendaraan yang dipilih adalah mobil, maka hanya estimasi untuk mobil yang ditampilkan (*textViewEstimateCar.setVisibility*(*View.VISIBLE*)), sedangkan estimasi motor disembunyikan.

b. Jika kendaraan yang dipilih adalah motor, maka sebaliknya terjadi: estimasi motor ditampilkan dan estimasi mobil disembunyikan. Jika kendaraan tidak diketahui, maka kedua estimasi ditampilkan kepada pengguna.

c. Jika kendaraan tidak diketahui, maka kedua estimasi ditampilkan kepada pengguna.

- Metode *convertMinutesToTime* telah digunakan untuk mengubah estimasi waktu yang ditampilkan dari menit menjadi format waktu, yaitu jam dan menit.

d. Jika tanggapan gagal atau tidak berhasil, pesan kesalahan "Prediksi tidak tersedia (belum dikonfirmasi admin?)" akan ditampilkan dengan menggunakan *showError*(...).

e. Dalam blok *onFailure*(...), jika ada kegagalan koneksi (misalnya, internet tidak berfungsi atau *server down*), pesan kesalahan "Gagal mengambil prediksi" akan ditampilkan dan kesalahan tersebut dicatat ke dalam *log* menggunakan *Log.e*(*TAG*, "*onFailure*:", t); ini memudahkan proses *debugging* selama pengembangan.