

PROPOSAL PROYEK AKHIR DATABASE SYSTEMS

Sistem Basis Data Pengiriman Paket

Expedia



Disusun Oleh :

Irene Maria Joseph 00000053330

Kelly Mae 00000051428

Leony Hana Noah Zebua 00000042544

Reuben Ryan Peter 00000043424

Fareza Ananda Putra 00000051475

Program: Information Systems
Matakuliah: Database Systems

UNIVERSITAS MULTIMEDIA NUSANTARA
TANGERANG
GANJIL 2021/2022

HALAMAN PENGESAHAN PROPOSAL

Topik	:	Pengiriman Paket
Judul Proposal (Indonesia)	:	Sistem Basis Data Pengiriman Paket Expedia
Judul Proposal (Inggris)	:	Database Systems on Package Delivery Expedia
Usulan Pembimbing	:	<<L00079 – Agus Sulaiman>>

HALAMAN PENILAIAN REVIEWER

Topik Final (Diisi oleh Dosen) :

--

Hasil Review Proposal Skripsi (Diisi oleh Dosen)

Tanggal Review	:	
Keputusan	:	Diterima / Revisi
Paraf Reviewer	:	

DAFTAR ISI

DAFTAR ISI	4
BAB I	7
PENDAHULUAN	7
1.1. Latar Belakang	7
1.2. Rumusan Masalah	8
1.3. Tujuan	8
1.4. Profil Perusahaan	9
1.5. Proses Aplikasi	10
BAB II	11
ANALISIS MASALAH	11
2.1. Masalah Bisnis	11
2.2. System Requirements	11
2.2.1. Functional Requirements	11
2.2.2. Non-Functional Requirement	12
2.3.. Use Case Diagram	16
BAB III	18
PENYELESAIAN MASALAH	18
3.1. Proses Normalisasi	18
3.1.1. UNF (Unnormalized Form)	18
3.1.2. 1NF (First Normal Form)	21
3.1.3. 2NF (Second Normal Form)	24
3.1.4. 3NF (Third Normal Form)	26
3.2. Entity Relationship Diagram (ERD)	33
BAB IV	34
HASIL PENYELESAIAN DAN KESIMPULAN	34
4.1. Tabel Relasi	34
4.2. Penjelasan dan Gambar User Interface Aplikasi & Kodingan	34
4.2.1. Menu Loader	34
4.2.2. Menu Sign	36
4.2.2.1. Button Sign In	37
4.2.2.2. Button Sign Up	37
4.2.3. Menu Sign In	37
4.2.3.1. Button Sign In	41
4.2.4. Menu Sign Up	41
4.2.4.1. Button Register	44
4.2.5. Main Menu	45
4.2.6. Menu User	46
4.2.6.1. Button Account Settings	48
4.2.6.2. Button Log Out	49

4.2.6.3. Button About Company	49
4.2.7. Menu User Settings	49
4.2.7.1. Button Change Email	50
4.2.7.2. Button Change Password	50
4.2.8 Menu Change Email	51
4.2.8.1 Button Change Email	51
4.2.9 Menu Change Password	51
4.2.9.1 Button Change Password	53
4.2.10 Menu About Company	54
4.2.10.1 Button Direction Maps	56
4.2.10.2 Button Call Customer Service	56
4.2.11 Menu API Google Maps	56
4.2.11.3 Button Know More	59
4.2.12 Menu Know More	59
4.2.13 Call Customer Service	61
4.2.14 Menu Order Service	62
4.2.14.1 Button Mulai Mengirim Paket	63
4.2.15 Menu Order Process	63
4.2.15.1 Button Pin Pick Up Location	68
4.2.15.2 Button Pin Deliver Location	68
4.2.15.3 Button Package Details	68
4.2.15.4 Button Informasi Pengiriman Lengkap	69
4.2.16 Menu Form Address	69
4.2.16.1 Button Done	74
4.2.17 Menu Package Details	74
4.2.17.1 Button Done	81
4.2.17.2 Button Cancel	81
4.2.18 Menu Order Confirm Order	81
4.2.18.1 Button Payment	84
4.2.18.2 Button Cancel	84
4.2.19 Menu Order Payment	84
4.2.19.1 Button Done	85
4.2.20 Menu History Transaction	86
4.2.21 Menu Transaction Details	89
4.2.21.1 Button Done Tracking	91
4.2.22 Menu Kurir Tracking Update	91
4.2.22.1 Button Notes	94
4.2.22.2 Button Update Done	94
4.2.22.4 Button Log Out	94
4.2.23 Menu Notes Driver	94
4.2.24 Menu Notes Tambah	97
4.2.25 Menu List Tracking Kurir	99
4.3 Query Pembuatan Tabel Basis Data	100
4.3.1. Create dan Use Database	100

4.3.2. Create Tabel users	100
4.3.3. Create Tabel daftar_alamat_pengirim	100
4.3.3. Create Tabel daftar_alamat_penerima	100
4.3.4. Create Tabel Tabel transaksi_paket	101
4.4 Advanced Query Basis Data	101
4.4.1. View	101
4.4.2. Procedure	102
4.4.3. Cursor	103
4.4.4. Trigger	104
4.4.5. Index	104
4.5 Kesimpulan	105
PERANAN ANGGOTA	105
LAMPIRAN	114

BAB I

PENDAHULUAN

1.1. Latar Belakang

Faktor utama pendorong kemajuan suatu negara dan masyarakat adalah perkembangan teknologi pengiriman surat dan paket. Kemudahan dan efisiensi dalam pengiriman surat dan paket antar pulau dan negara membuat masyarakat semakin maju dan berkembang. Hal ini dikarenakan masyarakat tidak harus datang ke tempat yang mereka ingin tuju. Perancangan Website atau pembuatan aplikasi berbasis Android system telah menghasilkan suatu rancangan untuk mendukung proses pelacakan paket, alur berjalannya paket, dan berupaya mempercepat proses update status kiriman paket. Berkembangnya teknologi dan informasi yang cepat saat ini mendukung adanya peluang untuk menggunakan media internet sebagai salah satu alat untuk mendapatkan informasi dan membuat informasi menjadi hal yang sangat penting bagi perusahaan-perusahaan yang ada sekarang ini, terutama perusahaan *e-commerce* (*electronic commerce*). Perusahaan ekspedisi adalah perusahaan yang bergerak dalam bidang jasa pengiriman paket baik keluar negeri maupun domestik. Permasalahan yang umum ditemukan pada perusahaan ekspedisi adalah masalah pengaturan data pengirim, penerima, paket dan lokasi keberadaan paket serta pelacakan lokasi keberadaan paket yang dikirimkan melalui perusahaan ekspedisi. Untuk menyelesaikan masalah tersebut dibuatlah perancangan *software* (perangkat lunak) yang dapat membantu perusahaan ekspedisi dalam mengatur data pengirim, penerima, paket dan keberadaan paket. Dengan menggunakan program aplikasi yang dibuat dapat membantu perusahaan ekspedisi dalam menyelesaikan masalah pengaturan data dan membantu pengirim supaya dapat melacak lokasi keberadaan paket yang dikirimkan melalui web atau aplikasi pelacakan paket.

Oleh karena e-commerce terus mendorong lanskap belanja ritel, pelacakan pesanan daring (seperti pada *online shopping*) menjadi semakin penting untuk keberhasilannya. Pelacakan pesanan daring (*online*) adalah salah satu faktor terpenting dalam *e-commerce* karena memungkinkan klien untuk melayani pelanggan online mereka dengan lebih baik dengan detail pengiriman yang akurat. Dengan memberikan informasi pengiriman yang akurat dan pembaruan pelacakan kepada pelanggan memungkinkan mereka mengetahui bahwa bisnis mereka dapat diandalkan dan dapat dipercaya, keduanya sangat penting untuk retensi pelanggan. Pelacakan pesanan secara daring membantu meminimalkan kecemasan pelanggan dan penyesalan pembeli.

Melalui penerapan sistem pelacakan pesanan *online*, perusahaan dapat menghilangkan banyak tekanan dari tim layanan pelanggan (*customer*) yang seharusnya dihabiskan untuk menangani pertanyaan, pertanyaan, dan keluhan

melalui panggilan telepon. Pelacakan pesanan memungkinkan tim layanan customer menghabiskan lebih banyak waktu menangani masalah penting.

Selain pelacakan pesanan, pelaporan berkelanjutan berperan penting bagi keberhasilan *shipping management* (manajemen pengiriman). Semua bisnis yang menangani semua jenis pengiriman harus memiliki serangkaian indikator kinerja utama yang mereka lacak secara teratur untuk mengukur hasil. Waktu penyelesaian, waktu kedatangan, dan akurasi inventaris adalah semua tolok ukur yang harus ditinjau secara berkala untuk menentukan apakah ada area untuk perbaikan atau tidak. Pelaporan manajemen kargo dapat membantu bisnis mengidentifikasi area untuk perbaikan dan membuat penyesuaian kecil untuk meningkatkan efisiensi.

Dari pemaparan di atas, maka kelompok kami berinisiatif untuk membuat perancangan aplikasi berbasis sistem Android untuk pelacakan pesanan atau paket atau biasa disebut *expedition tracker*. Kami harap melalui aplikasi ini dapat membantu banyak pelanggan yang melakukan pembayaran secara daring dan memudahkan perusahaan untuk menggunakan waktu dan tenaga secara efisien dalam menangani bagian *shipping* (pengiriman).

1.2. Rumusan Masalah

Dari pemaparan latar belakang di atas, dapat diketahui terdapat beberapa masalah, di antaranya sebagai berikut:

1. Customer kesulitan untuk mengetahui informasi mengenai proses pengiriman dan posisi barang yang mereka pesan.
2. Pelaporan lanjutan yang dilakukan kurang efektif karena membutuhkan banyak sumber daya manusia untuk menjadi *Customer Service* yang siap menerima telepon dari pihak Customer.
3. Kurang akuratnya pelacakan posisi barang yang diinformasikan ke pihak Customer.
4. Trend belanja daring yang semakin tinggi menuntut sistem ekspedisi yang semakin baik.

1.3. Tujuan

Keberadaan Expedia ditujukan untuk membantu melacak keberadaan paket secara online. Pengguna dapat mengetahui segala kondisi serta lokasi saat proses pengiriman paket berlangsung secara real-time atau dalam waktu yang nyata.

Proses pelacakan paket secara online pada aplikasi Expedia disertai dengan dukungan *User Interface* (UI) yang bersifat *user-friendly* sehingga hal ini menjadi mudah untuk digunakan bagi seluruh pengguna. Expedia juga bertujuan untuk memberikan informasi mengenai lokasi dan status paket sesuai dengan jenis layanan pengiriman paket kepada pengguna. Dengan demikian, pengguna dapat mengetahui secara jelas tentang estimasi waktu penyampaian paket menuju lokasi tujuannya.

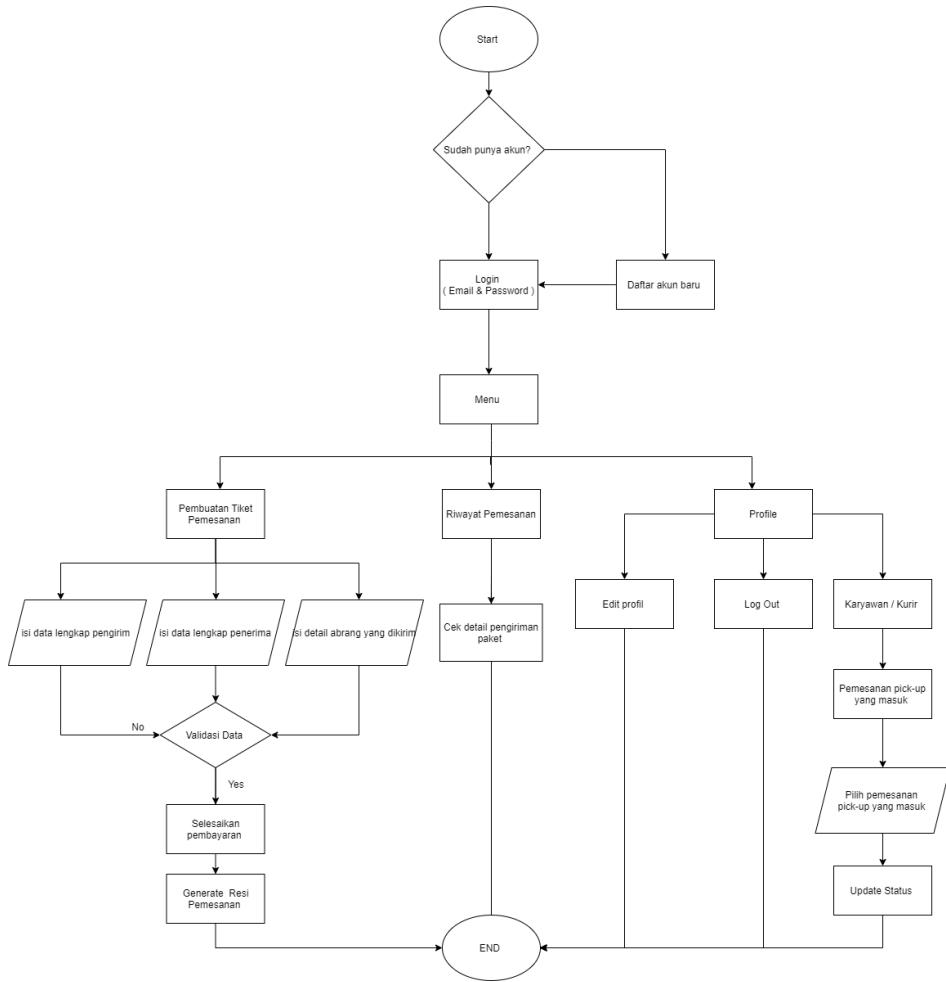
Semua pengecekan status paket dalam aplikasi Expedia akan dijalankan secara otomatis dari sistem. Aplikasi ini diharapkan mampu membantu pengguna memeriksa pelacakan paket sehingga dapat sampai di tempat tujuan sesuai dengan rencana.

1.4. Profil Perusahaan

Aplikasi Expedia adalah aplikasi yang digunakan untuk memantau melacak paket yang pelanggan pesan agar kita bisa mengetahui tempat dan status dari paket yang telah dipesan. Selain itu informasi yang pelanggan dapatkan tersiar secara langsung sehingga terdapat sedikit *delay*.

Dari aplikasi Expedia pengguna dapat melihat dan memantau paket yang pelanggan telah pesan sehingga pelanggan dapat mengetahui estimasi waktu dan kondisi paket. Hal ini juga dibantu dengan kooperasi para pengantar paket agar kita mendapatkan informasi secepat mungkin. Tujuan dari aplikasi Expedia adalah memberikan pelanggan informasi yang real-time agar mereka mendapatkan informasi yang mereka perlu untuk mengetahui keadaan paket mereka.

1.5. Proses Aplikasi



BAB II

ANALISIS MASALAH

2.1. Masalah Bisnis

Dalam menjalankan proses bisnis mengirim paket perusahaan masih menggunakan metode kertas dalam pembuatan bukti pembayaran yang diperlukan. Pencatatan menggunakan pulpen dan kertas ini tentu sangat rawan untuk terjadinya kecurangan hingga kehilangan data. Tidak adanya sistem yang mengawasi membuat risiko kesalahan dan kecurangan semakin tinggi. Kesalahpahaman yang terjadi antara satu divisi dengan divisi yang lainnya sangat berpotensi terjadi akibat tulisan tangan yang bisa saja tidak semua orang dalam divisi tersebut memahaminya.

2.2. System Requirements

2.2.1. Functional Requirements

- Melakukan *log in* dan menerima inputan berupa informasi *user* yang telah dimasukkan ketika pertama kali digunakan.
- Memperoleh informasi berupa *user_id*.
- Menyimpan data dari pengisian informasi pengirim, penerima, dan pengiriman secara terperinci.
- Melakukan perhitungan total ongkos pengiriman paket sesuai dengan lokasi tujuan pengiriman dan jasa pengiriman paket.
- Menampilkan konfirmasi pemesanan jasa pengiriman paket berupa *invoice* pengiriman dari data pengiriman yang telah didapatkan.
- Menampilkan pemesanan jasa pengiriman paket yang telah terkonfirmasi.
- Menerima pemesanan pengiriman secara berulang.
- Menampilkan pemesanan jasa pengiriman paket yang telah dilakukan oleh pengirim.
- Menampilkan status pengiriman bagi penerima dalam waktu nyata (*real-time*).
- Memperbarui status pengiriman secara berkala.
- Mengubah status pengiriman agar pengiriman dapat sesuai dengan estimasi waktu yang telah diberikan.
- Terdapat berbagai informasi yang harus dihasilkan oleh sistem, antara lain:
 - Data *User*
 - Data Pengirim
 - Data Penerima
 - Data Pengiriman Paket
 - Data Alamat Pengiriman
 - *Invoice* Pengiriman Paket

2.2.2. Non-Functional Requirement

1. Manageability

Title	Requirements
Availability	Database Expedia akan aktif selama 24 jam dan tidak memiliki <i>downtime</i> , kecuali pada saat melakukan pemeliharaan (<i>maintenance</i>).
Recoverability	Database Expedia akan dipulihkan dalam waktu 30 menit apabila terjadi <i>downtime</i> yang tidak terencana.
Maintainability	Setiap tampilan pada aplikasi Expedia akan diberikan tombol kembali (<i>back</i>) untuk memudahkan <i>user</i> berpindah dari menu satu menuju menu lainnya.
Supportability	Aplikasi Expedia harus memberikan informasi secara terperinci tentang kondisi kesalahan beserta sumbernya sehingga perusahaan dapat mengidentifikasi dan mengatasi permasalahan secara efisien.
Data Integrity	Perusahaan harus memastikan bahwa setiap informasi yang telah ditulis atau diberikan oleh <i>user</i> telah tersimpan dalam database.
Process Notification	<i>Process notification</i> mampu memperkirakan sisa estimasi waktu yang dibutuhkan dalam suatu proses. Apabila sisa estimasi waktu tidak memungkinkan, maka aplikasi tidak akan menampilkan <i>process notification</i> .
Confirmation Box	Setiap kali <i>user</i> melakukan aksi yang dapat membawa pengaruh pada keputusan terhadap data. Maka, aksi tersebut memicu munculnya <i>confirmation box</i> pada layar untuk memastikan bahwa aksi <i>user</i> telah dilakukan dengan benar.

Monitoring Report	Perusahaan memastikan bahwa tidak ada kesalahan yang muncul dalam mengawasi atau memonitor setiap data yang tersimpan dalam database.
-------------------	---

2. Security

Title	Requirements
Authentication	Sistem hanya dapat digunakan oleh <i>customer</i> (pelanggan) dan kurir.
Authorization	<i>Customer</i> dapat mengakses data paket yang dipesan dan pembayaran (<i>payment</i>). Kurir dapat mengakses data pemesanan paket.
Compliance	Aplikasi akan dijalankan dengan sistem <i>firewall</i> .

3. Performance

Title	Requirements
Responsiveness	Berikut terdapat <i>performance load time</i> yang diharapkan dalam aplikasi Expedia: <ul style="list-style-type: none"> • Maksimal 3 detik untuk memasukkan data pemesanan. • Maksimal 20 detik untuk melakukan <i>view</i> pada data pemesanan. • Maksimal 20 detik untuk melakukan <i>view</i> pada data pembayaran. • Maksimal 20 detik untuk melakukan <i>view</i> pada data <i>tracking</i> pemesanan.
Concurrency	Jumlah maksimum <i>concurrent user</i> adalah 10 <i>user</i> .
Resource Usage	Aplikasi Expedia harus dirancang untuk menggunakan sumber daya berupa perangkat keras (<i>hardware</i>), baik di sisi klien maupun sisi server.

Agility	Aplikasi Expedia harus bisa menangani dan memastikan akan integritas data apabila terjadi kondisi <i>overload</i> pada data yang ada.
---------	---

4. User Interface Requirement

Title	Requirements
Ease of Use	Sistem harus memiliki tampilan yang bersifat <i>user-friendly</i> . Tampilan harus mewakili fungsi tersendiri secara efisien yang dapat dimengerti dan tidak menimbulkan kebingungan bagi pengguna.
Attractiveness	Sistem harus memiliki tampilan semenarik mungkin dengan tujuan agar pengguna dapat mengetahui umpan balik (<i>feedback</i>) dari tindakan yang telah dilakukan terhadap sistem melalui respon yang diberikan oleh sistem.
Learn Ability	<i>User</i> seperti kurir harus memiliki pengetahuan dasar dalam menggunakan aplikasi dengan benar melalui pelatihan selama satu hari.

5. Interoperability

Title	Requirements
Interoperability	Aplikasi sebaiknya dijalankan pada <i>hardware</i> berupa <i>smartphone</i> dengan spesifikasi ruang memori 64GB, RAM 2GB, serta koneksi internet melalui data seluler maupun wifi untuk menjamin kelancaran transmisi data.

6. Backups

Title	Requirements
Backups	Cadangan data (<i>backup data</i>) akan tersimpan dalam memori yang berfungsi untuk melakukan <i>backup</i> data yang bersifat penting. Setiap data yang dimasukkan akan secara otomatis tersimpan ke dalam memori cadangan.

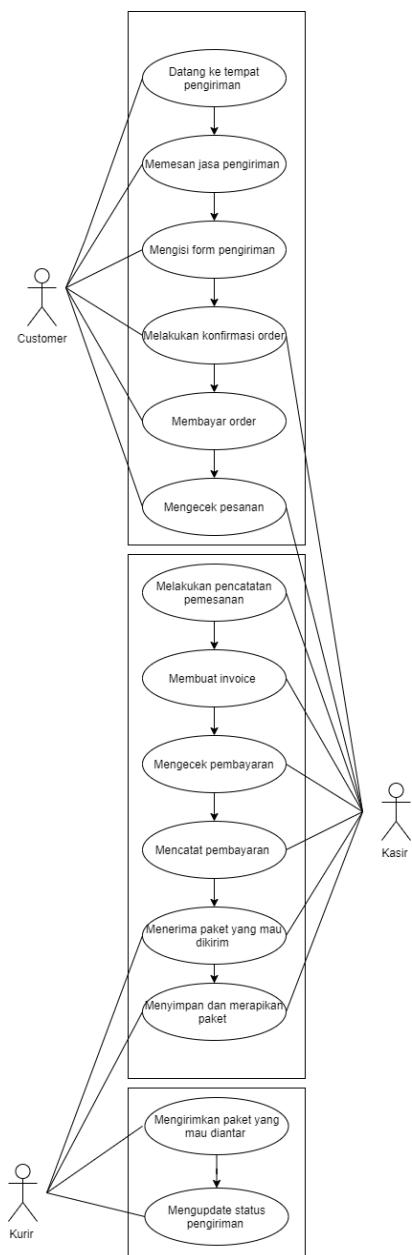
Data Protection	Data <i>protection</i> dilakukan dengan sistem <i>firewall</i> yang aktif setiap saat dan dibutuhkan proses verifikasi oleh pihak yang berwenang setiap kali ada <i>user</i> yang mencoba untuk melakukan perubahan pada data yang ada.
Availability	Data <i>backup</i> yang berada dalam memori cadangan akan selalu diekspor ke server utama (<i>main server</i>).
Disaster Recovery	Data <i>backup</i> yang berada dalam memori cadangan dapat diekspor ke <i>main server</i> apabila terjadi peristiwa kerusakan data, serangan <i>cyber</i> , bencana alam, <i>human error</i> (kesalahan disebabkan oleh manusia), kerusakan <i>hardware</i> yang digunakan pada <i>main server</i> , dan peristiwa lainnya yang tidak terduga.

7. Infrastructure Requirements

Title	Requirements
Clients	Perangkat yang digunakan untuk servers harus memiliki spesifikasi ruang memori 64GB, RAM 2GB. Sistem operasi yang digunakan adalah Android dengan minimum prosesor Android 5.1, yaitu Lollipop.
Servers	Perangkat yang digunakan untuk servers harus memiliki spesifikasi ruang memori 64GB, RAM 2GB. Sistem operasi yang digunakan adalah Android dengan minimum prosesor Android 5.1, yaitu Lollipop.
Network	Koneksi internet disertai dengan data seluler maupun wifi yang dihubungkan pada setiap perangkat. Bandwidth yang akan digunakan sebesar 10 Mbit/s dengan kecepatan sebesar 1 Gbps.
Peripheral	Hardware yang diperlukan untuk mengoperasikan aplikasi ini adalah perangkat Android dan data seluler atau wifi untuk koneksi internet.

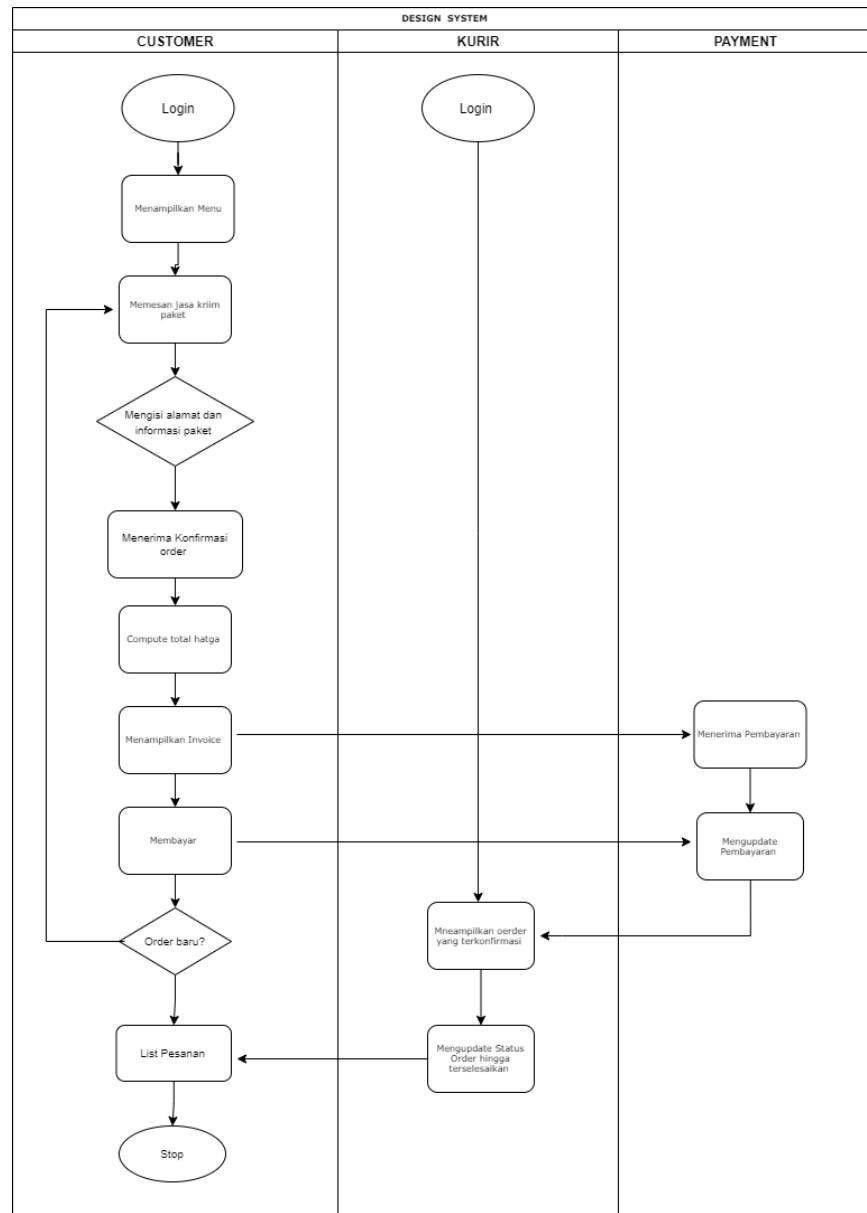
2.3.. Use Case Diagram

Pada masalah bisnis yang ada, berikut gambaran *use case* sebelum didigitalisasi:



2.4. Design System

Berdasarkan Requirement dan masalah yang dihadapi oleh maka disusunlah *Design System* Expedia sebagai berikut:



BAB III

PENYELESAIAN MASALAH

3.1. Proses Normalisasi

3.1.1. UNF (*Unnormalized Form*)

transaksi										
id	name	password	email	join_date	role	id_alamat	nama_pengirim	telpon	alamat_lengkap	provinsi
US001	Abbey Heracles	abbeyh2804	abbeyhera@gmail.com	2021-10-03	user	AD001	Abbey Heracles	081287390467	Jalan Taman Patra Kuningan No. 4, Kuningan Timur, Kuningan, Jakarta Selatan, DKI Jakarta - Indonesia 12114	DKI Jakarta
						AD002	Klemens Rosalina	08119733086	Jalan H. M. Kasim	Sulawesi Selatan

							No. 9, Baji Pamai, Maros Baru, Maros, Sulawesi - Indonesia 90516	
--	--	--	--	--	--	--	---	--

transaksi							
kota	kodepos	latitude	longitude	notes_tambah	id_transaksi	nama_barang	kuantitas
Jakarta Selatan	12114	-6.261493	106.834570	Lokasi tepi jalan	TR001	Baju Y2K	2
Maros	90516	-5.005840	119.574310	Seberang Museum Maros			

transaksi									
unit_paket	berat(kg)	fragile	asuransib_arang	tipe_peng_ambilan	jarak(km)	harga	status_paket	tanggal_pickup	tanggal_deliveredto_post
pcs	1.783	0	1	Delivered to post	2.689	349.000	Arrived	-	2021-08-13

transaksi				
tanggal_warehousetransit	tanggal_acceptedbykurir	tanggal_ongoing	tanggal_arrived	tanggal_failed
2021-08-13	2021-08-13	2021-08-14	2021-08-19	-

3.1.2. 1NF (First Normal Form)

transaksi										
id	name	password	email	join_date	role	id_alamat	nama	telpon	alamat_lengkap	provinsi
US001	Abbey Heracles	abbeyh2804	abbeyhera@gmail.com	2021-10-03	user	AD001	Abbey Heracles	081287390467	Jalan Taman Patra Kuningan No. 4, Kuningan Timur, Kuningan, Jakarta Selatan, DKI Jakarta - Indonesia 12114	DKI Jakarta
US001	Abbey Heracles	abbeyh2804	abbeyhera@gmail.com	2021-10-03	user	AD002	Klemens Rosalina	08119733086	Jalan H. M. Kasim No. 9, Baji Pamai, Maros Baru, Maros, Sulawesi -	Sulawesi Selatan

									Indonesia 90516	
--	--	--	--	--	--	--	--	--	--------------------	--

transaksi									
kota	kodepos	latitude	longitude	notes_tam bahana	id_transa ksi	id_pengiri m	id_peneri ma	nama_bar ang	kuantitas
Jakarta Selatan	12114	-6.261493	106.83457 0	Lokasi tepi jalan	TR001	SE001	RC001	Baju Y2K	2
Jakarta Selatan	12114	-6.261493	106.83457 0	Lokasi tepi jalan	TR001	SE001	RC001	Baju Y2K	2

transaksi									
unit_pake t	berat(kg)	fragile	asuransib arang	tipe_peng ambilan	jarak(km)	harga	status_pa ket	tanggal_pi ckup	tanggal_d eliveredto post
pcs	1.783	0	1	Delivered to post	2.689	349.000	Arrived	-	2021-08-1 3
pcs	1.783	0	1	Delivered to post	2.689	349.000	Arrived	-	2021-08-1 3

transaksi				
tanggal_warehousetransit	tanggal_acceptedbykurir	tanggal_ongoing	tanggal_arrived	tanggal_failed
2021-08-13	2021-08-13	2021-08-14	2021-08-19	-
2021-08-13	2021-08-13	2021-08-14	2021-08-19	-

3.1.3. 2NF (Second Normal Form)

a. Tabel *users*

users					
id	name	password	email	role	join_date
US001	Abbey Heracles	abbeyh2804	abbeyhera@gmail.com	user	2021-10-03

users									
id_alamat	nama	telpon	alamat_le ngkap	provinsi	kota	kodepos	latitude	longitude	notes_tam bahan
AD001	Abbey Heracles	081287390467	Jalan Taman Patra Kuningan No. 4, Kuningan Timur, Kuningan, Jakarta Selatan, DKI Jakarta - Indonesia 12114	DKI Jakarta	Jakarta Selatan	12114	-6.261493	106.834570	Lokasi tepi jalan

AD002	Klemens Rosalina	08119733086	Jalan H. M. Kasim No. 9, Baji Pawai, Maros Baru, Maros, Sulawesi - Indonesia 90516	Sulawesi Selatan	Maros	90516	-5.005840	119.574310	Seberang Museum Maros
-------	------------------	-------------	--	------------------	-------	-------	-----------	------------	-----------------------

b. Tabel *transaksi_paket*

transaksi_paket										
id_transaksi	id_user	id_pengirim	id_penerima	nama_barang	kuantitas	unit_paket	berat	fragile	asuransi_barang	tipe_pengambilan
TR001	US001	AD001	AP001	Baju Y2K	2	pcs	1.783	0	1	Delivered to Post

transaksi_paket									
jarak	harga	status_paket	tanggal_pickedup	tanggal_deliveredto_post	tanggal_warehousestransit	tanggal_acceptedbykurir	tanggal_ongoing	tanggal_arived	tanggal_failed
2.689	349.000	Arrived	-	2021-08-13	2021-08-13	2021-08-13	2021-08-14	2021-08-19	-

3.1.4. 3NF (*Third Normal Form*)

a. Tabel *users*

users					
id	name	password	email	role	join_date
US001	Abbey Heracles	abbeyh2804	abbeyhera@gmail.com	user	2021-10-03

b. Tabel *daftar_alamat_pengirim*

daftar_alamat_pengirim (tabel_customer)									
id_alamat_pengirim	nama	telpon	alamat_le ngkap	provinsi	kota	kodepos	latitude	longitude	notes_tam bahan
AD001	Abbey Heracles	081287390467	Jalan Taman Patra Kuningan No. 4, Kuningan Timur, Kuningan, Jakarta Selatan, DKI Jakarta - Indonesia 12114	DKI Jakarta	Jakarta Selatan	12114	-6.261493	106.834570	Lokasi tepi jalan

AP001	Klemens Rosalina	08119733086	Jalan H. M. Kasim No. 9, Baji Pamai, Maros Baru, Maros, Sulawesi - Indonesia 90516	Sulawesi Selatan	Maros	90516	-5.005840	119.574310	Seberang Museum Maros
-------	------------------	-------------	--	------------------	-------	-------	-----------	------------	-----------------------

c. Tabel *daftar_alamat_penerima*

daftar_alamat_penerima (tabel_customer)									
id_alamat_penerima	nama	telpon	alamat_le ngkap	provinsi	kota	kodepos	latitude	longitude	notes_tam bahan
AP001	Klemens Rosalina	08119733086	Jalan H. M. Kasim No. 9, Baji Pamai, Maros Baru, Maros, Sulawesi - Indonesia 90516	Sulawesi Selatan	Maros	90516	-5.005840	119.574310	Seberang Museum Maros

			90516						
--	--	--	-------	--	--	--	--	--	--

d. Tabel *transaksi_paket*

transaksi_paket										
id_transaksi	id_user	id_pengirim	id_penerima	nama_barang	kuantitas	unit_paket	berat	fragile	asuransi_barang	tipe_pengambilan
TR001	US001	AD001	AP001	Baju Y2K	2	pcs	1.783	0	1	Delivered to Post

transaksi_paket										
jarak	harga	status_paket	tanggal_pickedup	tanggal_deliveredto_post	tanggal_warehousestransit	tanggal_acceptedbycourir	tanggal_ongoing	tanggal_arived	tanggal_failed	
2.689	349.000	Arrived	-	2021-08-13	2021-08-13	2021-08-13	2021-08-14	2021-08-19	-	

Normalisasi tanpa Tabel

1. Tabel UNF (Unnormalized Form)

Tabel transaksi: id, name, password, email, join_date, role, {id_alamatpengirim, nama_pengirim, telp_pengirim, alamat_pengirim, provinsi_pengirim, kota_pengirim, latitude_pengirim, longitude_pengirim, notes_tambahanpengirim}, {id_alamatpenerima, nama_penerima, telp_ppenerima, alamat_penerima, provinsi_penerima, kota_penerima, latitude_penerima, longitude_penerima, notes_tambahanpenerima}, id_transaksi, nama_barang, kuantitas, unit_paket, berat, fragile,

asuransibarang, tipe_pengambilan, jarak, harga, status_paket, tanggal_pickup, tanggal_deliveredtopost, tanggal_warehousetransit, tanggal_acceptedbykurir, tanggal_ongoing, tanggal_arrived, tanggal_failed

2. Tabel 1NF (First Normalization Form)

Tabel transaksi: @id, name, password, email, join_date, role, id_alamatpengirim, nama_pengirim, telp_pengirim, alamat_pengirim, privinsi_pengirim, kota_pengirim, latitude_pengirim, longitude_pengirim, notes_tambahanpengirim , id_alamatpenerima, nama_penerima, telp_ppenerima, alamat_penerima, privinsi_penerima, kota_penerima, latitude_penerima, longitude_penerima, notes_tambahanpenerima , @id_transaksi, nama_barang, kuantitas, unit_paket, berat, fragile, asuransibarang, tipe_pengambilan, jarak, harga, status_paket, tanggal_pickup, tanggal_deliveredtopost, tanggal_warehousetransit, tanggal_acceptedbykurir, tanggal_ongoing, tanggal_arrived, tanggal_failed

3. Tabel 2NF (Unnormalized Form)

Tabel users: @id, name, password, email, join_date, role, id_alamatpengirim, nama_pengirim, telp_pengirim, alamat_pengirim, privinsi_pengirim, kota_pengirim, latitude_pengirim, longitude_pengirim, notes_tambahanpengirim , id_alamatpenerima, nama_penerima, telp_penerima, alamat_penerima, privinsi_penerima, kota_penerima, latitude_penerima, longitude_penerima, notes_tambahanpenerima

Tabel transaksi: @id_transaksi, nama_barang, kuantitas, unit_paket, berat, fragile, asuransibarang, tipe_pengambilan, jarak, harga, status_paket, tanggal_pickup, tanggal_deliveredtopost, tanggal_warehousetransit, tanggal_acceptedbykurir, tanggal_ongoing, tanggal_arrived, tanggal_failed, #id

4. Tabel 3NF (Third Normal Form)

Tabel users: @id, name, password, email, join_date, role

Tabel daftar_alamat_pengirim: @id_alamatpengirim, nama_pengirim, telp_pengirim, alamat_pengirim, privinsi_pengirim, kota_pengirim, latitude_pengirim, longitude_pengirim, notes_tambahanpengirim

Tabel daftar_alamat_penerima: @id_alamatpenerima, nama_penerima, telp_ppenerima, alamat_penerima, privinsi_penerima, kota_penerima, latitude_penerima, longitude_penerima, notes_tambahanpenerima

Tabel transaksi: @id_transaksi, nama_barang, kuantitas, unit_paket, berat, fragile, asuransibarang, tipe_pengambilan, jarak, harga, status_paket, tanggal_pickup, tanggal_deliveredtopost, tanggal_warehousetransit, tanggal_acceptedbykurir, tanggal_ongoing, tanggal_arrived, tanggal_failed, #id, #id_alamatpengirim, #id_alamatpenerima

PENJELASAN:

1. UNF (*Unnormalized Form*)

Pada *unnormalized form*, terdapat data yang akan menjadi *header* pada setiap tabel transaksi yang pada umumnya muncul berulang kali untuk sebuah transaksi yang sama akan digabungkan menjadi satu *cell*, seperti id, user, password, email, role, dan join_date.

2. 1NF (*First Normal Form*)

Dalam *first normal form*, terdapat penggabungan data yang terdapat pada *unnormalized form* menjadi satu *cell* yang akan dipecah setiap baris sehingga aturan *first normal form* dapat terpenuhi. Aturan *first normal form* tersebut berupa sebuah relasi dimana setiap irisan antara kolom dan baris telah berisikan hanya satu nilai. Dalam hal ini, terdapat *candidate key* berupa *primary key* pada tabel transaksi sebagai tabel *first normal form*, yaitu atribut id dan id_transaksi

3. 2NF (*Second Normal Form*)

Pada *second normal form*, proses normalisasi yang dilakukan telah menghasilkan dua buah tabel sebagai hasil pemecahan dari tabel *first normal form*, yaitu tabel users dan tabel transaksi_paket. Proses ini melibatkan pemisahan antara atribut yang bersifat *partially dependent* ataupun *fully dependent* terhadap suatu *candidate key* yang akan dilakukan sehingga menghasilkan tabel baru sesuai dengan tingkat ketergantungan pada setiap atribut, yaitu sebagai berikut:

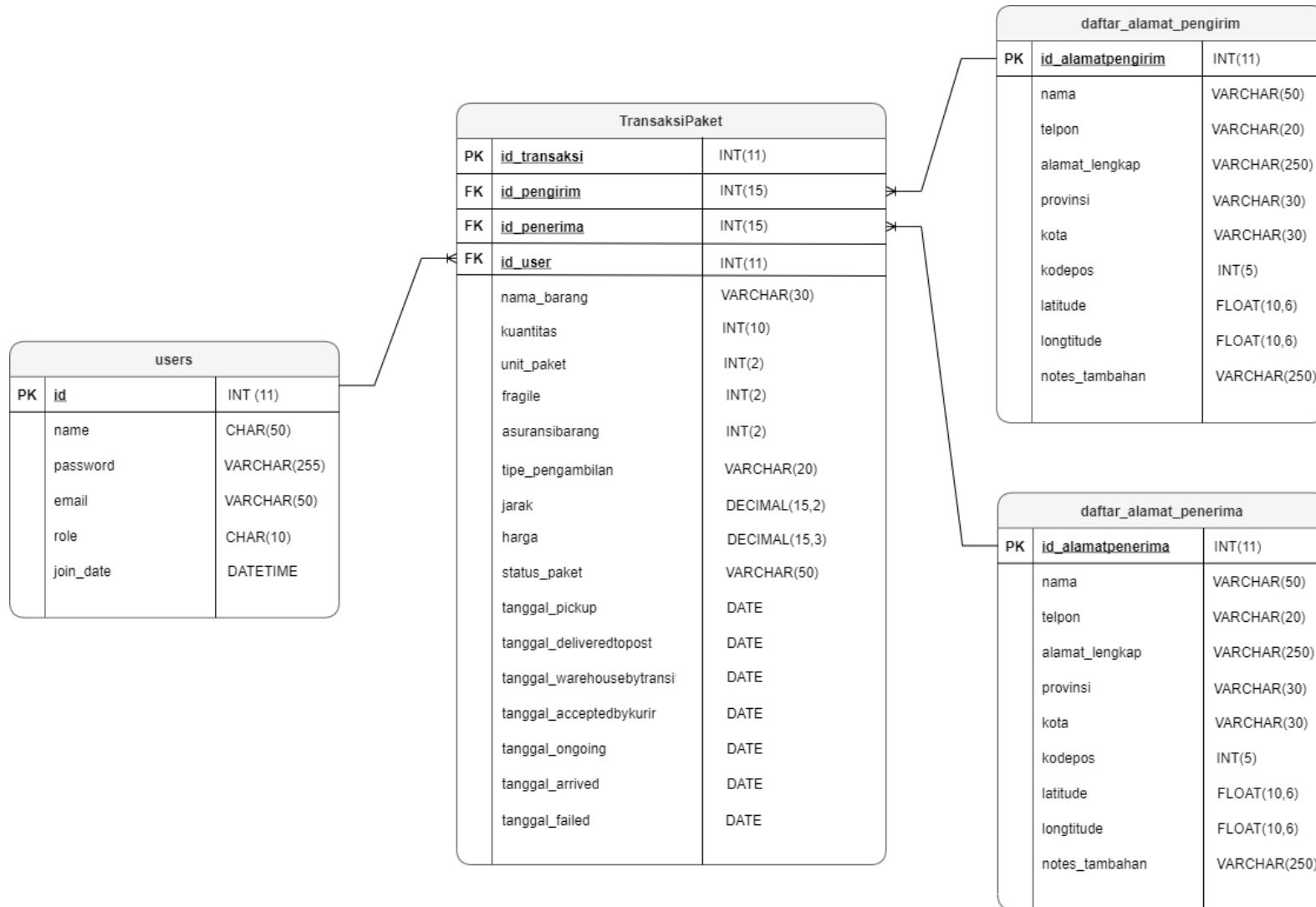
- Terdapat tabel *users* yang tercipta oleh karena adanya atribut name, password, email, role, join_date, id_alamat, nama, telpon, alamat_lengkap, provinsi, kota, kodepos, latitude, longitude, dan notes_tambahan yang bersifat *partially dependent* terhadap atribut id.
- Terdapat tabel *transaksi_paket* yang tercipta oleh karena adanya atribut id_pengirim, id_penerima, nama_barang, kuantitas, unit_paket, berat, fragile, asuransibarang, tipe_pengambilan, jarak, harga, status_paket, tanggal_pickup, tanggal_deliveredtopost, tanggal_warehousetransit, tanggal_acceptedbykurir, tanggal_ongoing, tanggal_arrived, dan tanggal_failed yang bersifat *partially dependent* terhadap atribut id_transaksi.

4. 3NF (*Third Normal Form*)

Pada *third normal form*, terdapat *non-primary key* yang memiliki *transitive dependencies* terhadap *primary key* dalam suatu tabel dalam proses normalisasi pada *second normal form*. Oleh karena itu, dibutuhkan pemecahan secara terperinci menjadi sebuah tabel baru.

- Terdapat atribut *id_alamatpengirim*, *nama*, *telpon*, *alamat_lengkap*, *provinsi*, *kota*, *kodepos*, *latitude*, *longitude*, dan *notes_tambahan* yang bersifat *transitive dependent* terhadap primary key *id* dalam tabel *users* sehingga hal ini menimbulkan terciptanya tabel baru, yaitu tabel *daftar_alamat_pengirim*.
- Terdapat atribut *id_alamatpenerima*, *nama*, *telpon*, *alamat_lengkap*, *provinsi*, *kota*, *kodepos*, *latitude*, *longitude*, dan *notes_tambahan* yang bersifat *transitive dependent* terhadap primary key *id* dalam tabel *users* sehingga hal ini menimbulkan terciptanya tabel baru, yaitu tabel *daftar_alamat_penerima*.

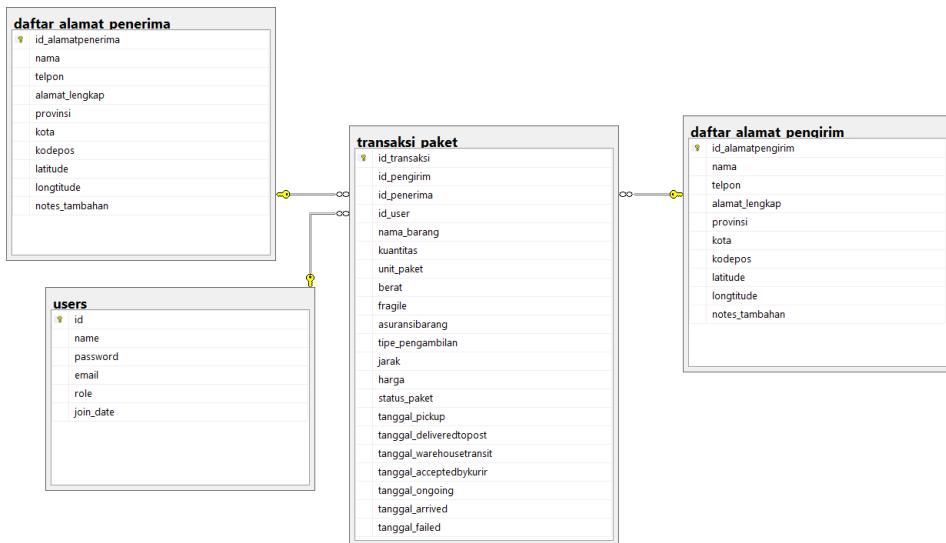
3.2. Entity Relationship Diagram (ERD)



BAB IV

HASIL PENYELESAIAN DAN KESIMPULAN

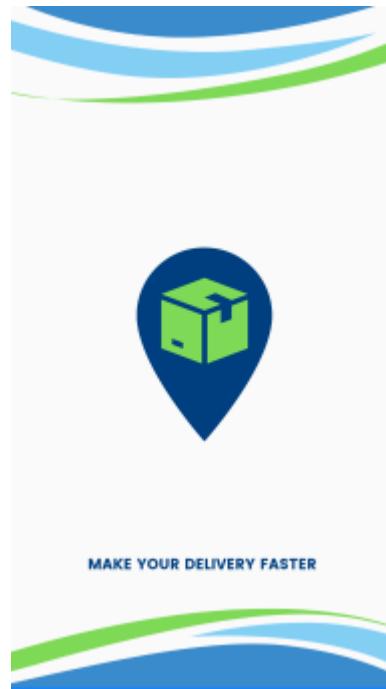
4.1. Tabel Relasi



4.2. Penjelasan dan Gambar *User Interface* Aplikasi & Kodingan

4.2.1. Menu Loader

Menu loader merupakan menu yang merupakan tampilan pertama saat user, baik customer maupun admin memproses aplikasi Expedia. Pada menu ini tertampil logo Expedia dengan jargonnya “Make Your Delivery Faster” yang dalam bahasa Indonesiaanya berarti “Membuat Pesanan Anda Semakin Cepat”.



Berikut kodingan yang digunakan untuk memberikan tampilan *loader* yang dapat menghentikan tampilan ketika loading timenya telah selesai, lalu masuk ke menu *Sign*:

```
import androidx.appcompat.app.AppCompatActivity;

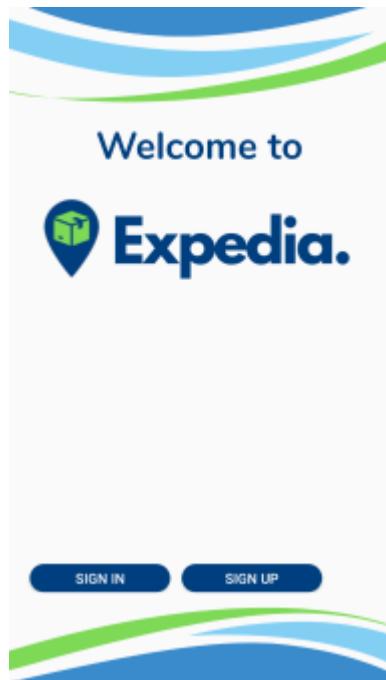
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;

public class Loader extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_loader);

        final int loading_time = 3000;
        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                Intent i =new Intent(getApplicationContext(), Sign.class);
                startActivity(i);
                finish();
            }
        },loading_time);
    }
}
```

4.2.2. Menu Sign

Pada menu Sign, customer dan/atau admin akan diperhadapkan pada dua pilihan untuk dapat mengakses aplikasi Expedia dan masuk ke menu utama, yaitu pilihan Sign In dan Sign Up. Sign In dan Sign Up dapat dipilih melalui button yang sudah tersedia pada tampilan menu ini. Button Sign In berada di sebelah kiri, sedangkan button Sign Up berada di sebelah kanan.



Berikut kodingan untuk menampilkan menu *sign*, serta *button sign in* dan *button sign up*:

```
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;
```

```

public class Sign extends AppCompatActivity {
    Button btnSignIn,btnSignUp;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sign);

        btnSignIn = findViewById(R.id.btnSignIn);
        btnSignUp = findViewById(R.id.btnSignUp);

        btnSignIn.setOnClickListener(v) -> {
            Intent signIn = new Intent(Sign.this, SignInActivity.class);
            startActivity(signIn);
        });

        btnSignUp.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent Register = new Intent(Sign.this, SignUpActivity.class);
                startActivity(Register);
            }
        });
    }
}

```

4.2.2.1. Button Sign In

Button Sign In ditampilkan pada menu Sign setelah menu Loader. Apabila user menekan tombol ini, maka user akan segera diarahkan ke menu berikutnya, yaitu Menu Sign In yang berfungsi agar user dapat masuk ke akun Expedia miliknya melalui e-mail dan password yang sudah terdaftar sebelumnya.

4.2.2.2. Button Sign Up

Selain button Sign In, pada menu Sign juga terdapat button Sign Up. Button ini berfungsi mengarahkan user ke menu Sign Up agar user yang belum memiliki akun Expedia sebelumnya dapat melakukan registrasi (register) akun.

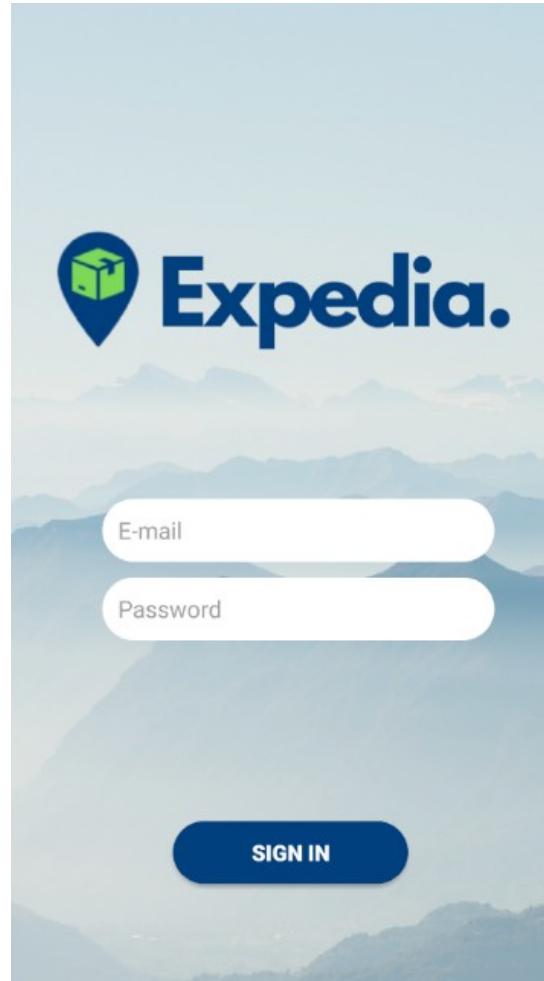


4.2.3. Menu Sign In

Menu Sign In merupakan menu yang tampil karena inisiatif user menekan tombol Sign In pada menu Sign sebelumnya. Menu ini terdiri dari dua kolom yang harus diisi oleh user agar dapat masuk ke akun Expedia milik user yang sudah terdaftar, yaitu Email dan Password. Kedua hal ini harus sesuai dengan apa yang

sudah diregister sebelumnya. Apabila user tidak mengisi salah satu atau kedua di antaranya dan langsung menekan tombol “Sign In”, maka akan muncul pop-up yang memberi tahu user bahwa kedua hal tersebut harus diisi dan benar.

Selanjutnya, apabila kedua hal ini sudah diisi dan benar, maka user akan menekan tombol Sign In yang mengarahkan user ke menu berikutnya, yaitu Main Menu.



Berikut kodingan untuk menampilkan menu *sign in*:

```
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;
```

```
public class SignInActivity extends AppCompatActivity {
    public static String userName;
    public static String userID;

    private EditText Login_Pass,Login_Email;
    private String email,password;
    private String urlcheckdata ="http://192.168.1.78/mobappbackend/router/login.php";
    private static final String TAG_USER="data";
    private JSONObject data;

    Button btnSignInMenu;
```

Berdasarkan kodingan di atas, terdapat pembuatan *class public* yang berhubungan penggunaan *method* yang ada, baik yang bersifat *public* maupun *private*. Selain itu, terdapat juga inisialisasi pada *button* menu *sign in*.

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_sign_in);

    Login_Email = (EditText) findViewById(R.id.Login_Email);
    Login_Pass = (EditText) findViewById(R.id.Login_Pass);

    btnSignInMenu = findViewById(R.id.btnSignInMenu);
    Button btnlogin = (Button) findViewById(R.id.btnSignInMenu);

```

Adapun pembuatan *widget* berupa *edit text* sebagai tempat bagi *user* untuk memasukkan teks berupa data *email* dan *password*, serta *button sign in*.

```

btnlogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        email = Login_Email.getText().toString();
        password = Login_Pass.getText().toString();
        RequestQueue queue = Volley.newRequestQueue(SignInActivity.this);
        StringRequest stringRequest = new StringRequest(Request.Method.POST, urlcheckdata, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    JSONObject jObj = new JSONObject(response);
                    int sukses = jObj.getInt("code");

                    if (sukses == 200) {
                        Intent i = new Intent(getApplicationContext(), MainMenuActivity.class);
                        startActivity(i);
                        data = jObj.getJSONObject("data");
                        String userRole = data.getString("role");
                        userName = data.getString("name");
                        userID = data.getString("id");

                        if(userRole.equals("user")){
                            Intent i = new Intent(getApplicationContext(), MainMenuActivity.class);
                            i.putExtra("userName" , userName);
                            startActivity(i);
                            finish();
                        }

                        if(userRole.equals("Kurir")){
                            Intent i = new Intent(getApplicationContext(), AdminTrackingUpdate.class);
                            i.putExtra("email",Uname);
                            startActivity(i);
                            finish();
                        }
                    } else {
                        Toast.makeText(SignInActivity.this, "Email dan Password yang anda masukkan salah", Toast.LENGTH_SHORT).show();
                    }
                } catch (Exception ex) {
                    Log.e("Error", ex.toString());
                }
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Log.e("Error", error.getMessage());
                Toast.makeText(SignInActivity.this, "silahkan cek koneksi internet anda", Toast.LENGTH_SHORT).show();
            }
        });
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Log.e("Error", error.getMessage());
        Toast.makeText(SignInActivity.this, "silahkan cek koneksi internet anda", Toast.LENGTH_SHORT).show();
    }
}); {

```

```
        @Override
        protected Map<String, String> getParams() {
            Map<String, String> params = new HashMap<>();
            params.put("email", email);
            params.put("password", password);
            return params;
        }

        @Override
        public Map<String, String> getHeaders() throws AuthFailureError {
            Map<String, String> params = new HashMap<>();
            params.put("Content-Type", "application/x-www-form-urlencoded");
            return params;
        }
    };
    queue.getCache().clear();
    queue.add(stringRequest);
}
});
```

4.2.3.1. Button Sign In

Setelah user memasukkan email dan password milik akunnya, maka user perlu menekan tombol Sign In yang tersedia pada menu Sign In. Ketika user menekan tombol itu, maka user segera diarahkan ke menu utama, yaitu Main Menu.



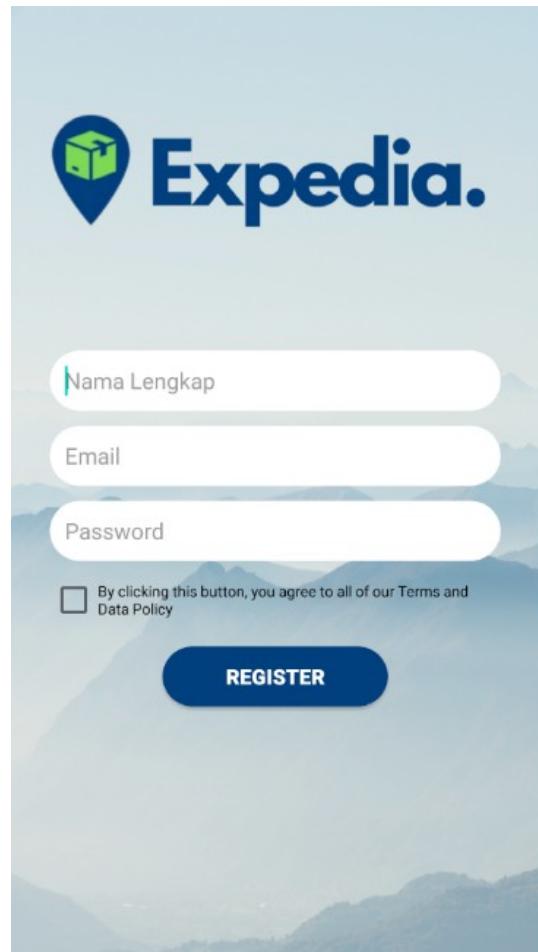
4.2.4. Menu Sign Up

Pada menu ini, user melakukan registrasi (sign up) untuk mendaftar akun yang baru agar dapat mengakses aplikasi Expedia. Beberapa hal yang esensial yang harus diisi oleh user antara lain:

- A. Nama lengkap
 - B. Email
 - C. Password

Setelah user mengisi ketiga hal di atas, maka user perlu memberi tanda centang pada checkbox yang tersedia pada menu ini. Checkbox yang sudah diberi tanda berarti user sudah menyetujui Term and Policy dari Expedia.

Sesudah user memberi tanda centang, maka user perlu menekan tombol “Register” pada bagian bawah untuk melakukan konfirmasi registrasi.



Berikut kodingan untuk menampilkan menu *sign up*:

```
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;
```

```

public class SignUpActivity extends AppCompatActivity {
    private EditText editName1, editEmail, editPass;
    CheckBox simpleCheckBox;
    Button btnSignUp, btn;

    private String txtname;
    private String txtemail;
    private String txtpassword;
    private CheckBox checkbox;
    private String urlregisterdata ="http://192.168.1.78/mobappbackend/router/registration.php";
    private static final String TAG_USER="data";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sign_up);

        editName1 = (EditText)findViewById(R.id.editName1);
        editEmail = (EditText)findViewById(R.id.editEmail);
        editPass = (EditText)findViewById(R.id.editPass);
        checkbox = (CheckBox)findViewById(R.id.simpleCheckBox);

        btnSignUp = (Button)findViewById(R.id.btnSignUp);
    }
}

```

Berdasarkan kodingan di atas, terdapat pembuatan *class public* yang berhubungan penggunaan *method* yang ada, baik yang bersifat *public* maupun *private*. Selain itu, terdapat juga inisialisasi pada *button* menu *sign up* dan *checkbox*. Adapun pembuatan *widget* berupa *edit text* sebagai tempat bagi *user* untuk memasukkan teks berupa data *name*, *email* dan *password*. Selain itu, adapun juga *widget checkbox* serta *button sign up* yang digunakan dengan kondisi tertentu di bawahnya.

```

btnSignUp.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        txtname = editName1.getText().toString();
        txtemail = editEmail.getText().toString();
        txtpassword = editPass.getText().toString();

        if(checkbox.isChecked() ){
            RequestQueue queue = Volley.newRequestQueue(SignUpActivity.this);
            StringRequest stringRequest = new StringRequest(Request.Method.POST, urlregisterdata, new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {
                    try {
                        JSONObject jObj = new JSONObject(response);
                        int sukses = jObj.getInt("code");
                        if (sukses == 200) {
                            Toast.makeText(SignUpActivity.this, "Register data berhasil! Silahkan Sign In", Toast.LENGTH_SHORT).show();
                            Intent i = new Intent(getApplicationContext(), SignInActivity.class);
                            startActivity(i);
                            finish();
                        } else {
                            Toast.makeText(SignUpActivity.this, "Register data gagal periksa informasi tertera", Toast.LENGTH_SHORT).show();
                        }
                    } catch (Exception ex) {
                        Log.e("Error", ex.toString());
                    }
                }
            });
            queue.add(stringRequest);
        }
    }
})

```

```

    }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Log.e("Error", error.getMessage());
            Toast.makeText(SignUpActivity.this, "silahkan cek koneksi internet anda", Toast.LENGTH_SHORT).show();
        }
    );
}

@Override
protected Map<String, String> getParams() {
    Map<String, String> params = new HashMap<>();
    params.put("name", txtname);
    params.put("email", txtemail);
    params.put("password", txtpassword);

    return params;
}

@Override
public Map<String, String> getHeaders() throws AuthFailureError {
    Map<String, String> params = new HashMap<>();
    params.put("Content-Type", "application/x-www-form-urlencoded");
    return params;
}
};

}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Log.e("Error", error.getMessage());
        Toast.makeText(SignUpActivity.this, "silahkan cek koneksi internet anda", Toast.LENGTH_SHORT).show();
    }
);

@Override
protected Map<String, String> getParams() {
    Map<String, String> params = new HashMap<>();
    params.put("name", txtname);
    params.put("email", txtemail);
    params.put("password", txtpassword);

    return params;
}

@Override
public Map<String, String> getHeaders() throws AuthFailureError {
    Map<String, String> params = new HashMap<>();
    params.put("Content-Type", "application/x-www-form-urlencoded");
    return params;
}
};

queue.getCache().clear();
queue.add(stringRequest);

else{
    Toast.makeText(SignUpActivity.this, "Anda Harus Setuju dengan Policy yang Ada", Toast.LENGTH_SHORT).show();
}

queue.getCache().clear();
queue.add(stringRequest);

else{
    Toast.makeText(SignUpActivity.this, "Anda Harus Setuju dengan Policy yang Ada", Toast.LENGTH_SHORT).show();
}

```

4.2.4.1. Button Register

Tombol ini berfungsi mensubmit data yang sudah user isi bagian menu Sign Up ke bagian database (back-end). Setelah user menekan tombol ini, maka user akan diarahkan ke menu utama, yaitu Main Menu dengan akun yang sudah terdaftar tersebut.



4.2.5. Main Menu

Pada menu ini, user dapat melihat tampilan menu utama dari aplikasi Expedia. Main Menu merupakan menu utama ketika user Log In atau Sign Up melalui akunnya. Pada menu ini, terdapat tiga menu utama, yaitu Menu Proses Paket, Menu Tracking dan History Transaction, dan Menu User. Ketiga menu ini dapat dipilih oleh user melalui menekan button yang sudah tersedia pada Main Menu.



Berikut kodingan untuk menampilkan menu *main menu*:

```
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;

import android.os.Bundle;
import android.view.MenuItem;

import com.google.android.material.bottomnavigation.BottomNavigationView;
```

```

public class MainMenuActivity extends AppCompatActivity {

    BottomNavigationView navigationView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_menu);

        navigationView = findViewById(R.id.bottom_nav);
        getSupportFragmentManager().beginTransaction().replace(R.id.framefragment, new MenuOrderService()).commit();
        navigationView.setSelectedItemId(R.id.nav_pesan);

        navigationView.setOnNavigationItemSelected(new BottomNavigationView.OnNavigationItemSelectedListener() {
            @Override
            public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
                Fragment fragment = null;
                switch (menuItem.getItemId()){
                    case R.id.nav_pesan:
                        fragment = new MenuOrderService();
                        break;

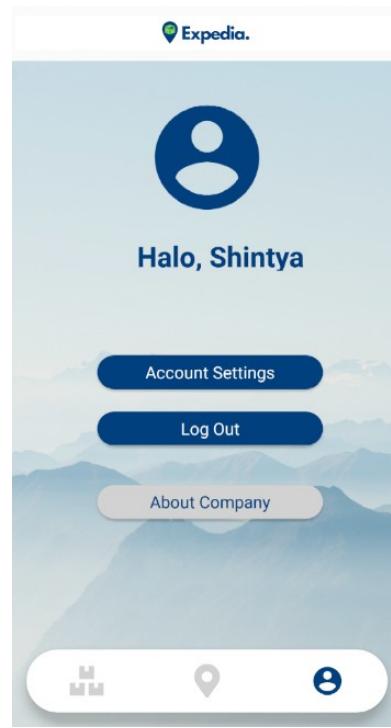
                    case R.id.nav_tracking:
                        fragment = new MenuTracking();
                        break;

                    case R.id.nav_user:
                        fragment = new MenuUser();
                        break;
                }
                getSupportFragmentManager().beginTransaction().replace(R.id.framefragment, fragment).commit();
                return true;
            }
        });
    }
}

```

4.2.6. Menu User

Menu user merupakan menu yang berisi Account Settings, pilihan untuk Log Out, dan tombol About Company. Menu ini menampilkan profil user secara sederhana yang terdiri dari ikon atau gambar dari profil dan nama user. Menu ini terdiri dari tiga button, di antaranya:



Berikut kodingan untuk menampilkan menu *user*:

```
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.TextView;
import static com.example.mobappproject.SignInActivity.userName;
```

```

public class MenuUser extends Fragment {
    TextView txtName;

    private Button btnAdmin, btnLogout, btnAboutComp;
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        View v = inflater.inflate(R.layout.activity_menu_user, container, false);
        txtName = v.findViewById(R.id.txtName);
        txtName.setText(userName);

        //btn acc settings
        Button btnAccountSetting = v.findViewById(R.id.btnAccountSetting);
        btnAccountSetting.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(getApplicationContext(), MenuUserSettings.class));
            }
        });

        //button logout
        btnLogout = v.findViewById(R.id.btnLogOut);
        btnLogout.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) { startActivity(new Intent(getActivity(), Sign.class)); }
        });
    }

    //button admin
    btnAdmin = v.findViewById(R.id.btnAdmin);
    btnAdmin.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(getActivity(), AdminLogin.class));
        }
    });

    //button admin
    btnAboutComp = v.findViewById(R.id.btnAboutComp);
    btnAboutComp.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(getActivity(), aboutcompany.class));
        }
    });
    return v;
}

```

4.2.6.1. Button Account Settings

Button ini merupakan button yang mengarahkan user ke menu berikutnya, yaitu menu Account Settings dimana user dapat mengubah dan/atau mengatur profil mereka sendiri.



4.2.6.2. Button Log Out

Button ini merupakan button yang mengarahkan user untuk keluar dari akun yang sedang aktif ke menu Sign yang sudah dijelaskan pada bagian 2.4.2.



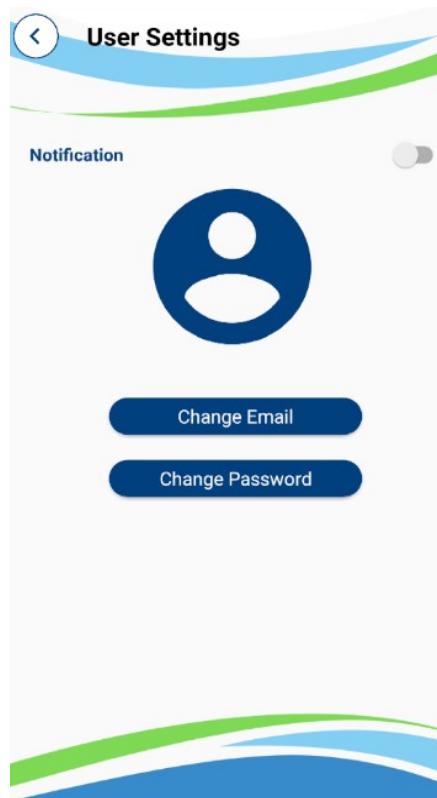
4.2.6.3. Button About Company

Button ini adalah button yang membawa user ke menu About Company yang berisi mengenai penjelasan mengenai Expedia Corp.



4.2.7. Menu User Settings

Menu User Settings adalah menu yang terdapat bagian menu User. Menu ini membantu user untuk mengubah atau memperbarui informasi user, seperti mengubah email atau mengubah password. Di samping itu, pada menu ini pun user dapat memilih apakah ingin menyalakan notifikasi atau tidak melalui tombol switch “Notification”.



Berikut kodingan untuk menampilkan menu *user settings*, tombol *switch notification*, serta *button change email* dan *change password*:

```
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MenuUserSettings extends AppCompatActivity {
    Button btnChangeemail,btnchangepass;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_menu_user_settings);

        btnChangeemail = findViewById(R.id.btnChangeemail);

        btnChangeemail.setOnClickListener(v) -> {
            Intent A = new Intent(MenuUserSettings.this, ChangeEmail.class);
            startActivity(A);
        };

        btnchangepass = findViewById(R.id.btnchangepass);

        btnchangepass.setOnClickListener(v) -> {
            Intent A = new Intent(MenuUserSettings.this, ChangePassword.class);
            startActivity(A);
        };
    }
}
```

4.2.7.1. Button Change Email

Button ini mengarahkan user untuk ke menu Change Email yang berfungsi membantu user untuk mengubah email akun user yang sudah terdaftar dengan email baru dan terverifikasi.



4.2.7.2. Button Change Password

Button ini mengarahkan user untuk ke menu Change Password yang bertujuan membantu user mengubah password akun user yang sudah terdaftar dengan password baru.



4.2.8. Menu Change Email

Menu ini merupakan menu yang membantu user untuk mengubah atau memperbarui email dari akun user baik admin maupun customer. Pada menu ini terdiri dari Current Email (email lama) dan New Email (email baru yang ingin dijadikan email user yang terbaru atau terupdate). Setelah user mengisi kedua kolom ini, maka user akan menekan tombol Change Email di mana akan muncul pop-up bahwa email sudah diubah dan terupdate. Di waktu berikutnya ketika user ingin log in kembali, maka user sudah dapat menggunakan email yang sudah diperbarui tersebut.

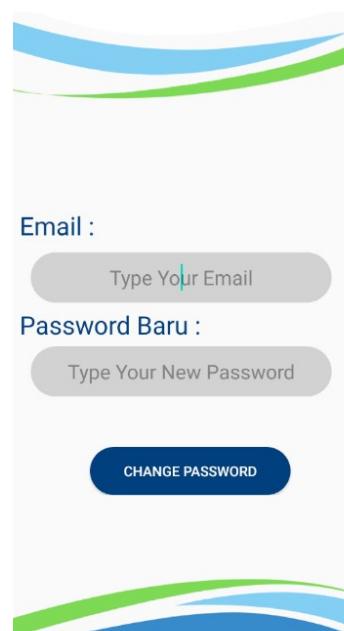
4.2.8.1. Button Change Email

Button ini berfungsi memverifikasi email yang sudah diganti oleh user. Ketika user menekan tombol ini, sistem dari Expedia mengirim update penggantian email ini ke sistem database untuk disimpan dan diperbarui. Oleh karena itu, ketika user sudah menekan tombol ini, maka user akan diarahkan ke Menu User Setting



4.2.9. Menu Change Password

Menu ini merupakan menu yang membantu user untuk mengubah atau memperbarui password (kata sandi) dari akun user baik admin maupun customer. Pada menu ini terdiri dari Current Password (password lama) dan New Password (password baru yang ingin dijadikan password user yang terbaru atau terupdate). Setelah user mengisi kedua kolom ini, maka user akan menekan tombol Change Password di mana akan muncul pop-up bahwa password sudah diubah dan terupdate. Di waktu berikutnya ketika user ingin log in kembali, maka user sudah dapat menggunakan password yang sudah diperbarui tersebut.



Berikut kodingan untuk menampilkan menu *change email*:

```
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;
```

```
public class ChangeEmail extends AppCompatActivity {

    private EditText txtemailbaru, txbemailganti;
    private String kirimemailbaru, kirimemailganti;
    Button btngantiemail;

    private String urlchangeemail ="http://192.168.1.78/mobappbackend/router/changeemail.php";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_change_email);

        txtemailbaru = (EditText)findViewById(R.id.txtemailbaru);
        txbemailganti = (EditText)findViewById(R.id.txbemailganti);

        // btngantiemail = (Button)findViewById(R.id.btnSignUp);

        btngantiemail = findViewById(R.id.btngantiemail);
        Button kirimbtxbemailganti = (Button)findViewById(R.id.btngantiemail);
```

```
kirimbtngantemail.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        kirimemailbaru = txtemailbaru.getText().toString();
        kirimemailganti = txbemailganti.getText().toString();

        RequestQueue queue = Volley.newRequestQueue(ChangeEmail.this);
        StringRequest stringRequest = new StringRequest(Request.Method.POST, urlchangeemail, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    JSONObject jObj = new JSONObject(response);
                    int sukses = jObj.getInt("code");
                    if (sukses == 200) {
                        Toast.makeText(ChangeEmail.this, "Email sukses terganti", Toast.LENGTH_SHORT).show();
                        Intent i = new Intent(getApplicationContext(),MenuUserSettings.class);
                        startActivity(i);
                        finish();
                    } else {
                        Toast.makeText(ChangeEmail.this, "Email gagal terganti, periksa pengisian data", Toast.LENGTH_SHORT).show();
                    }
                } catch (Exception ex) {
                    Log.e("Error", ex.toString());
                }
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Log.e("Error", error.getMessage());
                Toast.makeText(ChangeEmail.this, "silahkan cek koneksi internet anda", Toast.LENGTH_SHORT).show();
            }
        }) {
            @Override
            protected Map<String, String> getParams() {
                Map<String, String> params = new HashMap<>();
                params.put("email", kirimemailganti );
                params.put("emailbaru", kirimemailbaru);
                return params;
            }

            @Override
            public Map<String, String> getHeaders() throws AuthFailureError {
                Map<String, String> params = new HashMap<>();
                params.put("Content-Type", "application/x-www-form-urlencoded");
                return params;
            }
        };
    }

    queue.getCache().clear();
    queue.add(stringRequest);
}

});
```

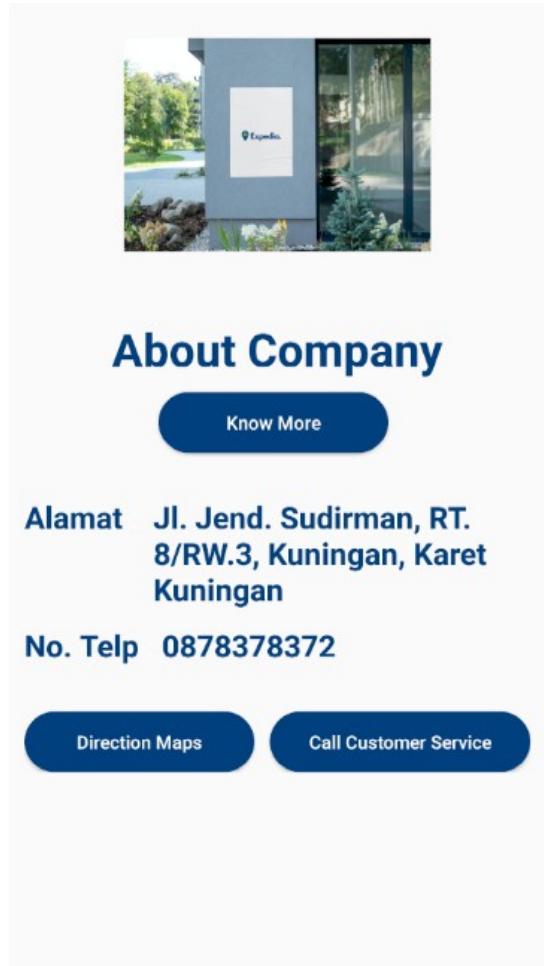
4.2.9.1. Button Change Password

Button ini berfungsi memverifikasi password yang sudah diganti oleh user. Ketika user menekan tombol ini, sistem dari Expedia mengirim update penggantian password ini ke sistem database untuk disimpan dan diperbarui. Oleh karena itu, ketika user sudah menekan tombol ini, akan muncul pop-up bahwa password sudah diperbarui.



4.2.10. Menu About Company

Pada menu ini, user akan melihat tampilan mengenai Expedia Corp. melalui alamat Expedia Corp., yaitu Jl. Jend. Sudirman, RT.8/RW.3, Kuningan, Karet Kuningan dengan nomor telepon 0878378372. Pada menu ini pun, user dapat menekan tombol Direction Maps untuk arah dari Google Maps dan tombol Call Customer Service untuk menghubungi customer service dari Expedia.



Berikut ini merupakan kodingan untuk menampilkan menu About Company, text view, button Know More, button Direction Maps, dan Call Customer Service:

```
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class aboutcompany extends AppCompatActivity {
    private Button btncall, btndirection;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_aboutcompany);

        btncall = (Button)aboutcompany.this.findViewById(R.id.btncall);
        btncall.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Intent.ACTION_DIAL, Uri.fromParts("tel", "08783783", null));
                startActivity(intent);
            }
        });
        btndirection = (Button)aboutcompany.this.findViewById(R.id.btndirection);
        btndirection.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String uri = "http://maps.google.co.in/maps?q=" + "Expedia";
                Intent i = new Intent(Intent.ACTION_VIEW, Uri.parse(uri));
                startActivity(i);
            }
        });
        btndirection = (Button)aboutcompany.this.findViewById(R.id.btndirection);
        btndirection.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String uri = "http://maps.google.co.in/maps?q=" + "Expedia";
                Intent i = new Intent( packageContext: aboutcompany.this, APIGoogleMaps.class);
                startActivity(i);
            }
        });
    }
}
```

```
        });

        btndirection = (Button)aboutcompany.findViewById(R.id.btndirection);
        btndirection.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String uri = "http://maps.google.co.in/maps?q=" + "Expedia";
                Intent i = new Intent( packageContext: aboutcompany.this, APIGoogleMaps.class);
                startActivity(i);
            }
        });
    }
}
```

4.2.10.1. Button Direction Maps

Button ini berfungsi untuk mengarahkan user ke menu API Google Maps dimana user dapat melihat letak perusahaan Expedia secara digital. Google Maps adalah platform pemetaan web dan aplikasi konsumen yang ditawarkan oleh Google. Melalui button ini, user dapat melihat citra satelit, aerial photography, peta jalan, pemandangan panorama jalan 360° yang interaktif (Street View), kondisi lalu lintas secara real-time, dan perencanaan rute untuk bepergian dengan berjalan kaki, mobil, udara (dalam versi beta), dan transportasi umum.

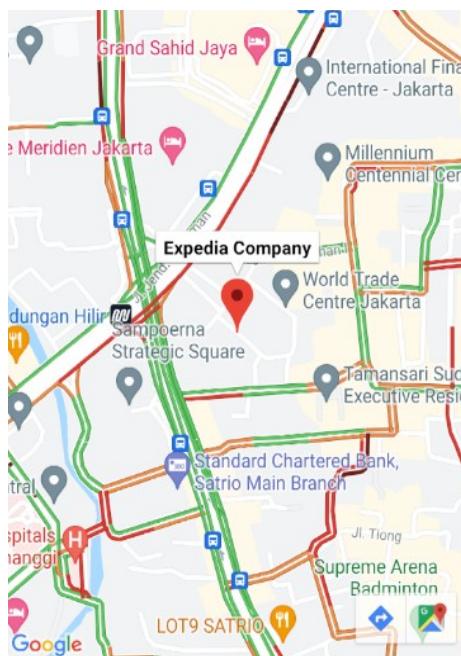
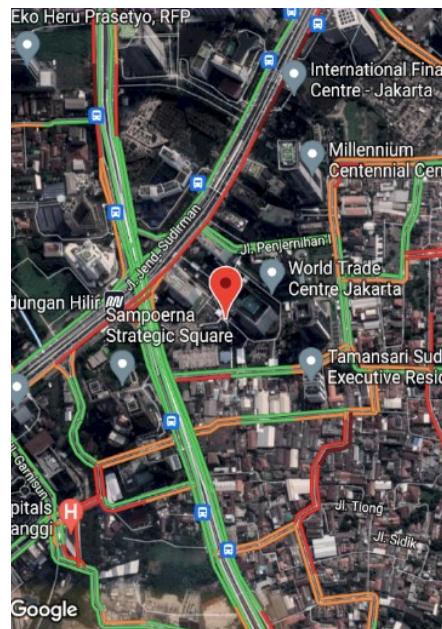


4.2.10.2. Button Call Customer Service

Button ini mengarahkan user untuk langsung menelepon sistem Customer Service dari Expedia dengan langsung directing ke Call Log atau Panggilan pada device masing-masing user.

4.2.11. Menu API Google Maps

Pada menu API Google Maps, user dapat melihat peta digital letak alamat kantor utama Expedia Corporation. Pada menu ini, user dihadapkan dengan dua pilihan layers map, yaitu mode normal dan mode hybrid.



Berikut ini merupakan kodingan untuk menampilkan menu API Google Maps dan buttonnya.

```
import androidx.fragment.app.FragmentActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.example.mobappproject.databinding.ActivityApigooleMapsBinding;

public class APIGoogleMaps extends FragmentActivity implements OnMapReadyCallback {
    Button btndirection2;

    private GoogleMap mMap;
    private ActivityApigooleMapsBinding binding;

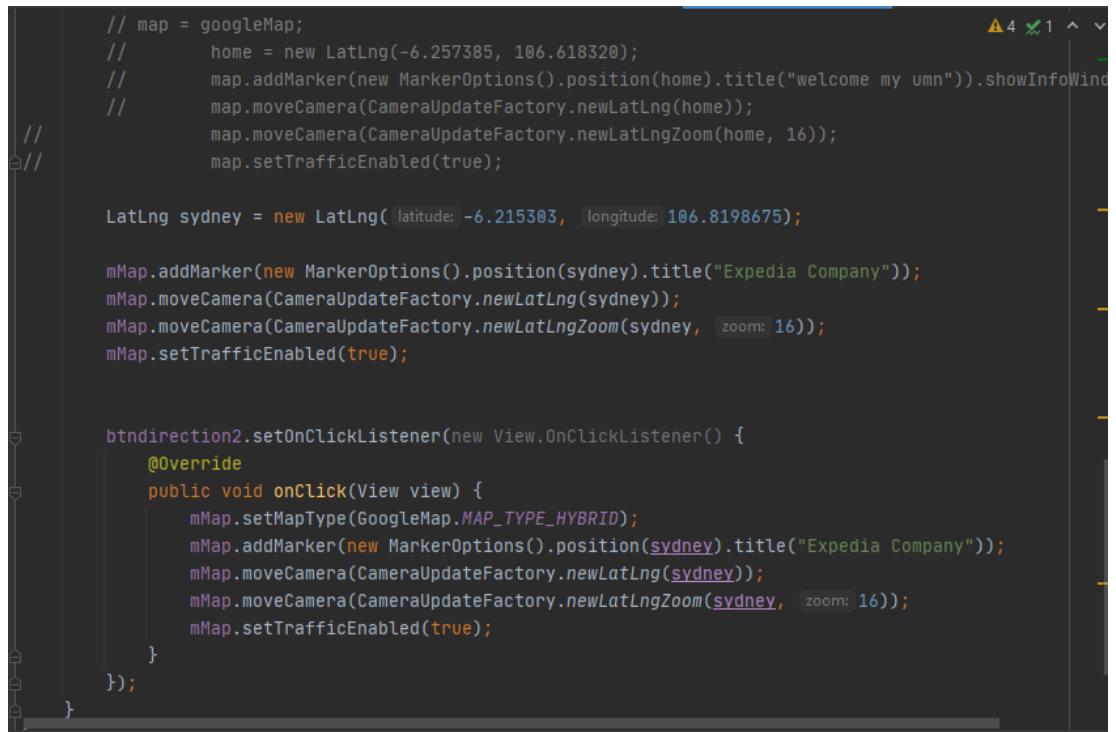
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityApigooleMapsBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
        mapFragment.getMapAsync(callback: this);
    }

    /**
     * Manipulates the map once available.
     * This callback is triggered when the map is ready to be used.
     * This is where we can add markers or lines, add listeners or move the camera. In this case,
     * we just add a marker near Sydney, Australia.
     * If Google Play services is not installed on the device, the user will be prompted to install
     * it inside the SupportMapFragment. This method will only be triggered once the user has
     * installed Google Play services and returned to the app.
     */
    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        btndirection2=(Button) findViewById(R.id.btndirection2);

        // Add a marker in Sydney and move the camera
        // map = googleMap;
        // home = new LatLng(-6.257385, 106.618320);
```



```
// map = googleMap;
//     home = new LatLng(-6.257385, 106.618320);
//     map.addMarker(new MarkerOptions().position(home).title("welcome my umn")).showInfoWindow();
//     map.moveCamera(CameraUpdateFactory.newLatLng(home));
//     map.moveCamera(CameraUpdateFactory.newLatLngZoom(home, 16));
//     map.setTrafficEnabled(true);

LatLng sydney = new LatLng( latitude: -6.215303, longitude: 106.8198675);

mMap.addMarker(new MarkerOptions().position(sydney).title("Expedia Company"));
mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(sydney, zoom: 16));
mMap.setTrafficEnabled(true);

btndirection2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        mMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
        mMap.addMarker(new MarkerOptions().position(sydney).title("Expedia Company"));
        mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(sydney, zoom: 16));
        mMap.setTrafficEnabled(true);
    }
});
```

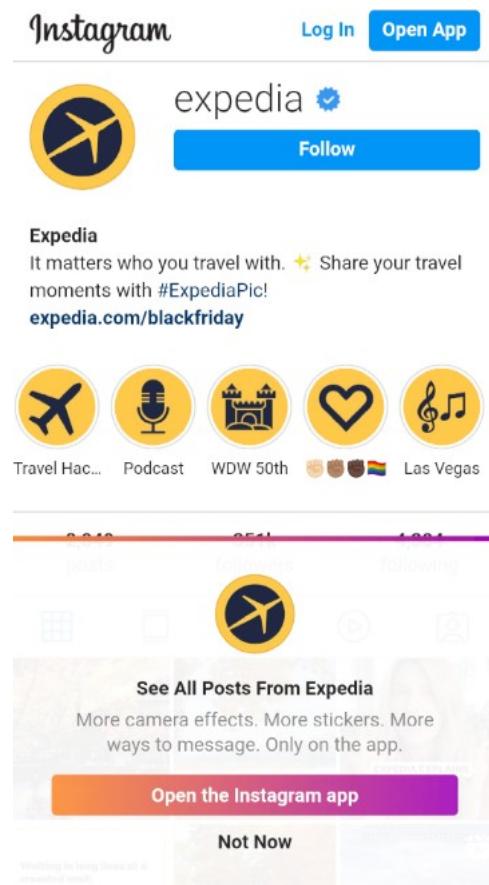
4.2.11.3. Button Know More

Pada button ini, user akan diarahkan ke menu berikutnya, yaitu penjelasan mengenai Expedia Corporation dan sosial media Expedia.

4.2.12. Menu Know More

Pada menu ini, user dapat membaca visi dari Expedia Corporation, yaitu menjadi perusahaan expedia terintegrasi yang dapat memberikan nilai kepuasan tertinggi bagi pelanggan.

Selain itu, pada menu ini juga user dapat swipe (mengusap layar) ke kanan untuk mengunjungi sosial media berupa Instagram milik Expedia seperti yang terlihat pada gambar di bawah ini. Pada Menu ini, kami menggunakan viewPager dan webView untuk masuk ke dalam profil penjelasan Expedia dan ketika digeser kekanan terdapat profil instagram Expedia.



Berikut ini merupakan kodingan untuk menampilkan Menu Know More, text view, dan viewpager.

```

public class ViewPager extends AppCompatActivity {
    androidx.viewpager.widget.ViewPager mViewPager;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_view_pager);

        mViewPager = (androidx.viewpager.widget.ViewPager) findViewById(R.id.pager);
        mViewPager.setAdapter(new ViewPagerAdapter(getSupportFragmentManager()));
    }

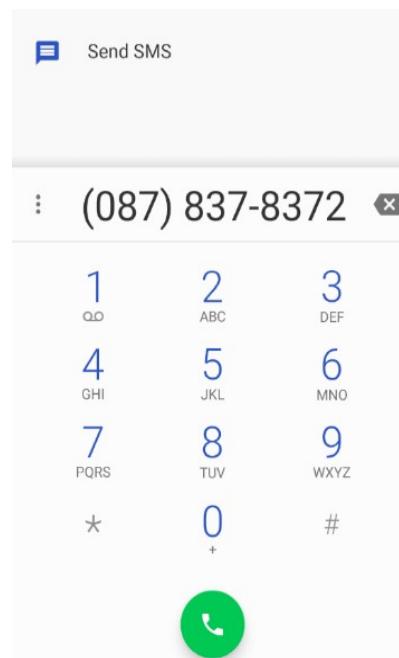
    public class ViewPagerAdapter extends FragmentPagerAdapter {
        public ViewPagerAdapter(FragmentManager fm) {
            super(fm, BEHAVIOR_RESUME_ONLY_CURRENT_FRAGMENT);
        }

        public Fragment getItem(int position) {
            if (position == 0) {
                return new CompanyInformation();
            } else {
                return new CompanyWeb();
            }
        }
    }
}

```

4.2.13. Call Customer Service

Setelah user menekan button “Call Customer Service”, maka user akan diarahkan user untuk langsung menelepon sistem Customer Service dari Expedia dengan langsung directing ke Call Log atau Panggilan pada device masing-masing user.



Berikut ini merupakan kodingan untuk menampilkan Menu Call Customer Service.

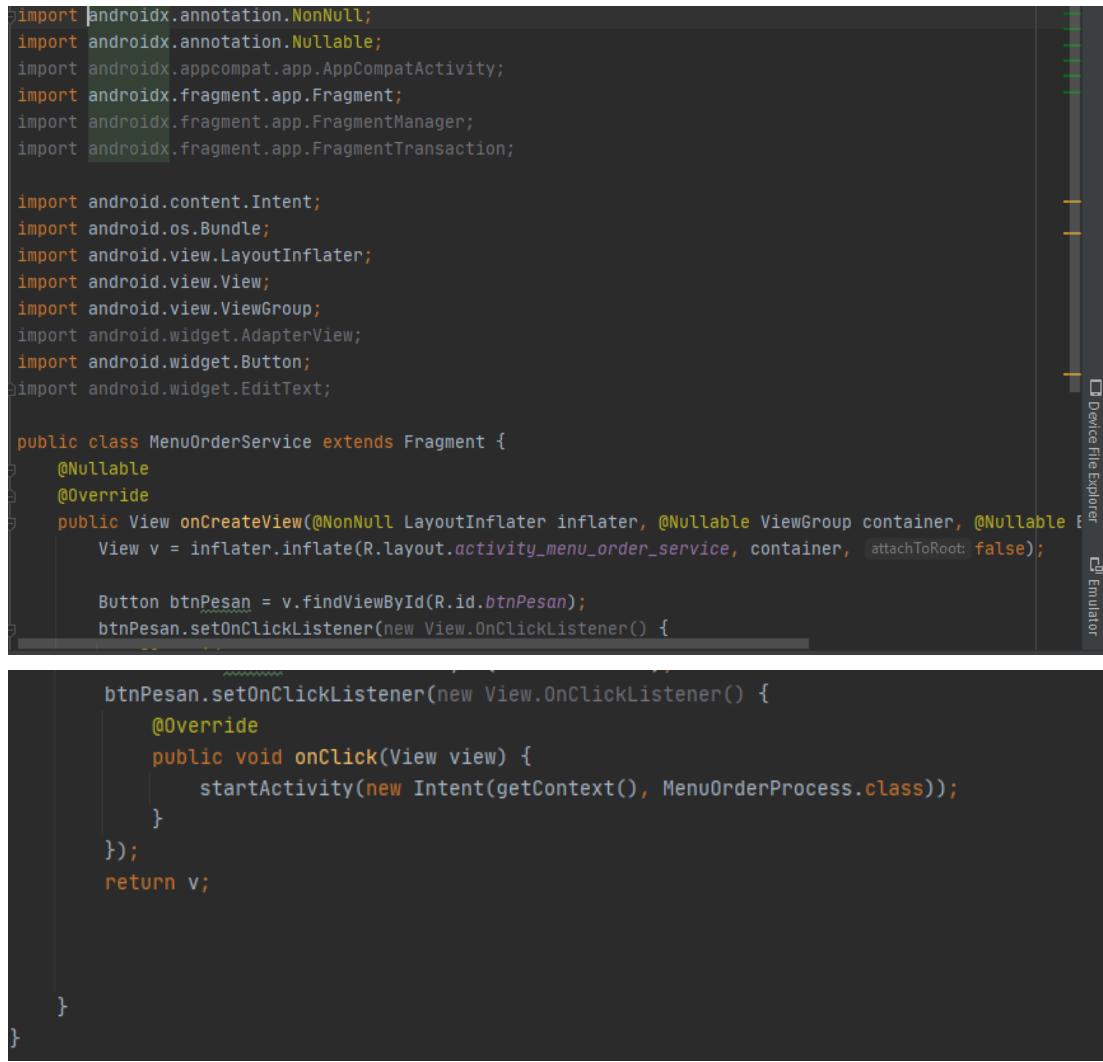
```
btnCall = (Button)aboutCompany.this.findViewById(R.id.btnCall);
btnCall.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(Intent.ACTION_DIAL, Uri.fromParts("tel", "0878378372", null));
        startActivity(intent);
    }
});
btnDirection = (Button)aboutCompany.this.findViewById(R.id.btnDirection);
btnDirection.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String uri = "http://maps.google.co.in/maps?q=" + "Expedia";
        Intent i = new Intent(Intent.ACTION_VIEW, Uri.parse(uri));
        startActivity(i);
    }
});
```

4.2.14. Menu Order Service

Menu ini merupakan menu di mana user baik customer maupun admin dapat mulai mengirim atau memesan (order) paket. Pada menu ini, terdapat tombol “Mulai Mengirim Paket” di mana tombol ini akan mengarahkan user ke Menu Order Process yang terdiri dari beberapa proses yang perlu diperhatikan dan diikuti oleh user untuk segera mengirim paket (untuk admin) dan memesan paket (untuk customer).



Berikut ini merupakan kodingan untuk menampilkan Menu Order Service dan button Mulai Mengirim Paket.



```
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;

import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;

public class MenuOrderService extends Fragment {
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        View v = inflater.inflate(R.layout.activity_menu_order_service, container, false);

        Button btnPesanan = v.findViewById(R.id.btnPesanan);
        btnPesanan.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(getContext(), MenuOrderProcess.class));
            }
        });
        return v;
    }
}
```

4.2.14.1. Button Mulai Mengirim Paket

Button pada Menu Order Service ini berfungsi mengarahkan user ke menu berikutnya, yaitu Menu Order Process.



4.2.15. Menu Order Process

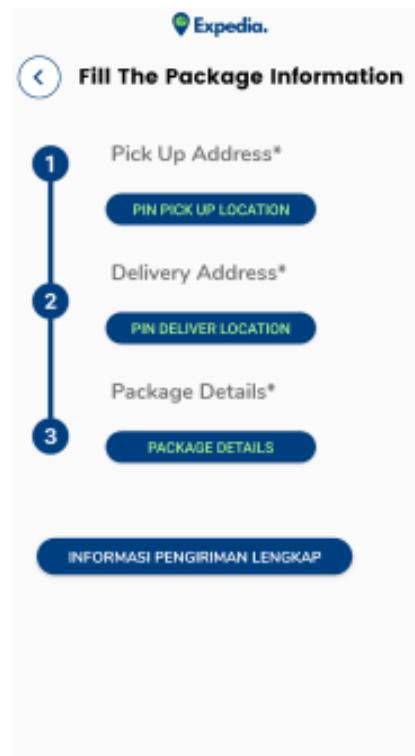
Menu ini merupakan menu yang tertampil setelah user menekan button “Mulai Mengirim Paket” pada menu sebelumnya, yaitu Menu Order Service. Menu Order Process ini merupakan menu yang menampilkan tahapan atau proses yang perlu dilaksanakan oleh user untuk dapat melakukan pemesanan atau pengiriman paket. Pada Menu Order Process, user dapat melihat pada bagian atas sebuah perintah yaitu “Fill The Package Information” yang berarti user perlu mengisi informasi yang berkaitan dengan paket.

Tahapan pertama yang perlu dilaksanakan oleh user adalah Pick Up Location, di mana user dapat melakukannya melalui menekan tombol “Pin Pick Up Location”. Tahapan ini merupakan tahapan yang wajib diisi dan diikuti ditandai dengan adanya tanda bintang (*) pada sebelah tulisan “Pick Up Location”

Tahapan kedua yang perlu dilewati oleh user adalah Delivery Address. Tahapan ini dapat dilakukan oleh user melalui menekan button “Pin Deliver Location”. Tahapan ini merupakan tahapan yang wajib diisi dan diikuti oleh user. Hal ini ditandai dengan adanya tanda bintang (*) pada sebelah tulisan “Delivery Address”.

Tahapan terakhir atau tahapan ketiga dari Menu Order Process adalah tahap Package Details. Seperti dua tahap di atasnya, tahap ketiga ini juga wajib dilaksanakan oleh user. Hal ini ditandai dengan adanya tanda bintang (*) pada sebelah tulisan “Package Details”. Tahapan ini dapat dilakukan oleh user dengan menekan tombol “Package Details”.

Setelah user melakukan ketiga tahapan di atas, maka user dapat menekan tombol “Informasi Pengiriman Lengkap” yang terletak di bawah ketiga tahapan.



Berikut ini merupakan kodingan untuk menampilkan Menu Order Process, text view, button Pin Pick Up Location, Pin Deliver Location, Package Details, dan Informasi Pengiriman Lengkap.

```
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Adapter;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;

import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;
import static com.example.mobappproject.MenuPackageDetails.id_transaksi;

public class MenuOrderProcess extends AppCompatActivity implements AdapterView.OnItemSelectedListener {

    public static String namapengirim, alamatpengirim, namapenerima, alamatpenerima, namabarang, txtb
//    txtid_transaksi
    private JSONObject data;

    private String urlshowpackageinfo = "http://192.168.1.78/mobappproject/router/showpackagedetails.php";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_menu_order_process);

        Button passingdata =(Button)findViewById(R.id.buatconfirm);
        passingdata.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
public void onClick(View v) {
    RequestQueue queue = Volley.newRequestQueue( context: MenuOrderProcess.this);
    StringRequest stringRequest = new StringRequest(Request.Method.POST, urlshowpackageinfo, new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            try {
                JSONObject jObj = new JSONObject(response);
                int sukses = jObj.getInt( name: "code");

                if (sukses == 200) {
                    data = jObj.getJSONObject("data");
                    txtid_transaksi = data.getString("id_transaksi");
                    namapengirim = data.getString( name: "nama_pengirim");
                    alamatpengirim = data.getString( name: "alamat_pengirim");
                    namapenerima= data.getString( name: "nama");
                    alamatpenerima = data.getString( name: "alamat_lengkap");
                    namabarang= data.getString( name: "nama_barang");
                    txtberat = data.getString( name: "berat");
                    txtjarak = data.getString( name: "jarak");
                    asuransibarang = data.getString( name: "asuransibarang");

                    Intent i = new Intent(getApplicationContext(),MenuOrderConfirmOrder.class);
                    startActivity(i);
                    finish();
                } else {
                    Toast.makeText( context: MenuOrderProcess.this, text: "Email dan Password yang anda masukkan salah");
                }
            } catch (Exception ex) {
                Log.e( tag: "Error", ex.toString());
            }
        }
    }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Log.e( tag: "Error", error.getMessage());
            Toast.makeText( context: MenuOrderProcess.this, text: "Data harus di isi berurutan, Isi detail terlebih dahulu");
        }
    });
    @Override
    protected Map<String, String> getParams() {
        Map<String, String> params = new HashMap<>();
        params.put( k: "id_transaksi", id_transaksi);
    }
}
```

```
        return params;
    }

    @Override
    public Map<String, String> getHeaders() throws AuthFailureError {
        Map<String, String> params = new HashMap<>();
        params.put("Content-Type", "application/x-www-form-urlencoded");

        return params;
    }
};

queue.getCache().clear();
queue.add(stringRequest);
}

ton btnForm =(Button)findViewById(R.id.button4);
Form.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
    Intent i = new Intent(getApplicationContext(),FormAddress.class);
    startActivity(i);
}
}

public void onClick(View v) {
    Intent i = new Intent(getApplicationContext(),FormAddress.class);
    startActivity(i);
}

ton btnFormPen =(Button)findViewById(R.id.button5);
FormPen.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
    Toast.makeText(context: MenuOrderProcess.this, text: "Data harus di isi berurutan, Isi detail data m");
}
}

ton btnPackageDetails =(Button)findViewById(R.id.btnPackageDetails);
PackageDetails.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
    Toast.makeText(context: MenuOrderProcess.this, text: "Data harus di isi berurutan, Isi detail data m");
}
}
```

```

    //        buatconfirm =(Button)findViewById(R.id.buatconfirm);
    //        buatconfirm.setOnClickListener(new View.OnClickListener() {
    //            @Override
    //            public void onClick(View v) {
    //                Intent i = new Intent(getApplicationContext(),MenuOrderConfirmOrder.class);
    //                startActivity(i);
    //            }
    //        });
    }

    @Override
    public void onItemSelected(AdapterView<?> adapterView, View view, int position, long id) {

    }

    @Override
    public void onNothingSelected(AdapterView<?> adapterView) {

    }
}

```

4.2.15.1. Button Pin Pick Up Location

Button ini berfungsi mengarahkan user ke menu berikutnya yang merupakan tahapan pertama untuk membuat pesanan (order process), yaitu menu Form Address yang berisikan formulir informasi alamat pengiriman pesanan paket.



4.2.15.2. Button Pin Deliver Location

Button pada *stage* atau tahap kedua ini apabila ditekan oleh user maka akan menunjukkan toast dengan tulisan “Data harus di isi berurutan, Isi detail data mulai dari tombol proses 1”.



4.2.15.3. Button Package Details

Button pada tahap Package Details ini apabila ditekan oleh user maka akan memunculkan toast yaitu “Data harus diisi berurutan, Isi detail data mulai dari tombol proses 1”. Setelah kedua tahap sudah dilewati, maka user akan diarahkan ke Menu Package Details melalui menekan button ini.



4.2.15.4. Button Informasi Pengiriman Lengkap

Button ini akan ditekan user apabila semua tahap sudah dilaksanakan dan sistem sudah menyetujui dengan adanya toast-toast yang muncul. Apabila button ini ditekan oleh user, maka user akan diarahkan ke menu berikutnya, yaitu Menu Order Confirm Order. Menu ini merupakan menu untuk Check Out Order, lebih lanjutnya dapat dilihat pada bagian 2.4.16.



4.2.16. Menu Form Address

Menu ini merupakan menu yang tertampil saat user menekan tombol “Pin Pick Up Location” dan merupakan tahap pertama agar user dapat melakukan order paket. Pada menu ini terdapat beberapa kolom yang harus diisi oleh user (bersifat required). Kolom tersebut antara lain:

a. Nama

User mengisi kolom ini dengan menginput nama penerima, lebih baik apabila user menginput nama lengkap.

b. Nomor Handphone

User mengisi kolom ini dengan memasukkan nomor handphone penerima. Hal ini dapat mempermudah kurir untuk menghubungi penerima apabila kurir mengalami kendala dalam mengirim paket ke alamat yang dituju. Dengan adanya nomor telepon yang tersedia ini diharapkan kurir dapat mengatasi kendala yang muncul tersebut.

c. Alamat lengkap

Kolom ini wajib diisi oleh user dengan memasukkan alamat lengkap penerima. Diharapkan user mengisi selengkap mungkin, mulai dari nama jalan, nama komplek/perumahan, nomor rumah, blok tempat tinggal, dan RT/RW.

d. Provinsi

User wajib memilih provinsi alamat penerima melalui dropdown yang sudah tersedia. Provinsi yang dipilih hanya dapat dipilih satu provinsi saja.

e. Kota

User memasukkan kota tempat tinggal penerima sesuai dengan yang sebenarnya untuk mempermudah input pengiriman paket pemesanan.

f. Kode Pos

Kode pos ini wajib diisi oleh user dengan memasukkan kode pos wilayah penerima tinggal.

g. Notes Tambahan

Kolom ini merupakan kolom di mana user memberikan tambahan keterangan oleh user untuk menjadi catatan bagi pengirim atau kurir.

Pada menu ini juga terdapat menu Button Done di mana button ini akan mengarahkan user ke Menu Order Process apabila ditekan oleh user dengan tujuan user menyelesaikan proses yang selanjutnya.

The screenshot shows a mobile application interface for setting a pickup location. At the top, there's a back arrow icon and the word "Expedia". Below that is the title "Set Pickup Location". The form consists of several input fields:

- Name:** "Masukkan Nama Penerima" (Placeholder)
- No. Handphone:** "Masukkan Nomor Handphone Penerima" (Placeholder)
- Alamat Lengkap:** "Masukkan Alamat Lengkap Penerima" (Placeholder)
- Provinsi:** A dropdown menu labeled "Select Province".
- Kota:** "Masukkan Kota Penerima" (Placeholder)
- Kode Pos:** "Masukkan Kode Pos Penerima" (Placeholder)
- Notes Tambahan:** "Masukkan Notes Tambahan" (Placeholder)

At the bottom of the form, there is a note in red text: "Notes : Warna Merah Wajib di Isi". Below the note is a large blue "DONE" button.

Pada bagian bawah menu juga terdapat Notes yaitu “Warna merah wajib di isi” menandakan semua kolom kecuali kolom “Notes Tambahan” merupakan kolom yang wajib untuk diisi.

Berikut ini kodingan untuk menampilkan Menu Form Addres dan button Done.

```
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import android.view.textservice.SpellCheckerSession;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class FormAddress extends AppCompatActivity {
    public static String id_pengirimDB;
    private static final String TAG_ID = "data";

    private String txtNamaPengirim, txtNoTelpPengirim, txtAlamatLengkapPengirim, txtKotaPeng, txtKodePosPeng, txtProvinsi;
    private String txtSpinProvinsi;

    private Button btnAlamatPeng;

    private EditText namaPengirim, noTelpPengirim, alamatLengkapPengirim, kotaPeng, kodePosPeng, NotesPeng;
    private Spinner spinProvinsi;

    private String urlAddressPeng = "http://192.168.1.78/mobappbackend/router/inputalamatpeng.php";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_form_address);

        namaPengirim = (EditText)findViewById(R.id.namaPengirim);
        noTelpPengirim = (EditText)findViewById(R.id.noTelpPengirim);
        alamatLengkapPengirim = (EditText)findViewById(R.id.alamatLengkapPengirim);
        kotaPeng = (EditText)findViewById(R.id.kotaPeng);
        kodePosPeng = (EditText)findViewById(R.id.kodePosPeng);
        NotesPeng = (EditText)findViewById(R.id.NotesPeng);
        spinProvinsi = (Spinner)findViewById(R.id.spinProvinsi);
    }

    private void tambahData() {
        StringRequest stringRequest = new StringRequest(Request.Method.POST, urlAddressPeng, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                Log.d("Response", response);
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Log.d("Error", error.getMessage());
            }
        }) {
            @Override
            protected Map<String, String> getParams() throws AuthFailureError {
                Map<String, String> params = new HashMap<String, String>();
                params.put("id_pengirim", id_pengirimDB);
                params.put("data", TAG_ID);
                params.put("namaPengirim", namaPengirim.getText().toString());
                params.put("noTelpPengirim", noTelpPengirim.getText().toString());
                params.put("alamatLengkapPengirim", alamatLengkapPengirim.getText().toString());
                params.put("kotaPeng", kotaPeng.getText().toString());
                params.put("kodePosPeng", kodePosPeng.getText().toString());
                params.put("NotesPeng", NotesPeng.getText().toString());
                params.put("spinProvinsi", spinProvinsi.getSelectedItem().toString());
                return params;
            }
        };
        Volley.newRequestQueue(this).add(stringRequest);
    }
}
```

```
alamatLengkapPengirim = (EditText)findViewById(R.id.alamatLengkapPengirim);
kotaPeng = (EditText)findViewById(R.id.KotaPeng);
kodePosPeng = (EditText)findViewById(R.id.kodePosPeng);
NotesPeng=(EditText)findViewById(R.id.NotesPeng);
spinProvinsi = (Spinner) findViewById(R.id.spinProvinsi);

btnAlamatPeng = (Button)findViewById(R.id.btnAlamatPeng);
btnAlamatPeng.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        txtnamaPengirim = namaPengirim.getText().toString();
        txtnoTelpPengirim = noTelpPengirim.getText().toString();
        txtalamatLengkapPengirim = alamatLengkapPengirim.getText().toString();
        txtkotaPeng = kotaPeng.getText().toString();
        txtkodePosPeng = kodePosPeng.getText().toString();
        txtNotesPeng = NotesPeng.getText().toString();
        txtSpinProvinsi = spinProvinsi.getSelectedItem().toString();

        //Toast.makeText(getApplicationContext(),noplat + carmodel + caryear + carcolor + norang
        AlertDialog.Builder ad = new AlertDialog.Builder(context: FormAddress.this);
        ad.setTitle("Konfirmasi Alamat Pengirim");
        ad.setMessage("Apakah data yang anda sudah isi benar?");

        ad.setPositiveButton( text: "YES", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                RequestQueue queue = Volley.newRequestQueue( context: FormAddress.this);
                StringRequest stringRequest = new StringRequest(Request.Method.POST, urlAddressPeng, new Response.Listener<String>() {
                    @Override
                    public void onResponse(String response) {
                        try {
                            JSONObject jObj = new JSONObject(response);
                            int sukses = jObj.getInt( name: "code");
                            if (sukses == 200) {
                                Toast.makeText(FormAddress.this.getApplicationContext(), text: "Data alamat pengirim berhasil di simpan", duration: 2000).show();
                                data = jObj.getJSONObject("data");
                                id_pengirimDB = jObj.getString( name: "data");
                                Intent i = new Intent(getApplicationContext(),FormAddressPenerima.class);
                                startActivity(i);
                            }
                        } catch (JSONException e) {
                            e.printStackTrace();
                        }
                    }
                });
                queue.add(stringRequest);
            }
        });
    }
});
```

```
        finish();

        MyCarPage fragmycar = new MyCarPage();
        fragmycar.setArguments(getActivity().getIntent().getExtras());
        Bundle bundle = new Bundle();
        bundle.putString("username",username);
        fragmycar.setArguments(bundle);
        getActivity().getSupportFragmentManager().beginTransaction().replace(R.id.fragment_main, fragmycar).commit();
    } else {
        Toast.makeText(FormAddress.this.getApplicationContext(), "Data alamat pengirim berhasil diupdate", Toast.LENGTH_SHORT).show();
        Toast.makeText(FormAddress.this.getApplicationContext(), "Periksa pengisian", Toast.LENGTH_SHORT).show();
    }
}
catch (Exception ex) {
    Log.e(tag: "Error", ex.toString());
}
},
new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Log.e(tag: "Error", error.getMessage());
        Toast.makeText(context: FormAddress.this, error.getMessage(), Toast.LENGTH_SHORT).show();
    }
});
@Override
protected Map<String, String> getParams() {
    Map<String, String> params = new HashMap<>();
    params.put( k: "nama", txtnamaPengirim);
    params.put( k: "telpon", txtnoTelpPengirim);
    params.put( k: "alamat_lengkap", txtalamatLengkapPengirim);
    params.put( k: "provinsi", txtSpinProvinsi );
    params.put( k: "kota", txtkotaPeng);
    params.put( k: "kodepos", txtkodePosPeng);
    params.put( k: "notes_tambahan", txtNotesPeng);

    return params;
}

@Override
public Map<String, String> getHeaders() throws AuthFailureError {
    Map<String, String> params = new HashMap<>();
    params.put( k: "Content-Type", v: "application/x-www-form-urlencoded");
    return params;
}
};

queue.getCache().clear();
queue.add(stringRequest);
```

```
        }
        //pembatas

    }

};

ad.setNegativeButton( text: "NO", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {

    }
});
ad.show();
}
);
}

}
```

4.2.16.1. Button Done

Button Done pada menu Form Address ini merupakan button yang mensubmit atau mengirim data yang diisi oleh user ke sistem database. Terdapat kondisi yang ada dalam pengisian kolom ini. Apabila (if) semua kolom yang berwarna merah sudah terisi, maka ketika user menekan button Done, akan memunculkan toast atau pop-up dengan tulisan “Data alamat pengirim berhasil ditambahkan”. Tetapi (else) apabila kolom berwarna merah ada yang tidak terisi, maka ketika user menekan button Done, akan memunculkan toast atau pop-up dengan tulisan “Data alamat pengirim gagal ditambahkan. Periksa pengisian data sesuai keterangan”. Dengan demikian user akan mengetahui apabila data yang diisi sudah benar atau belum.



4.2.17. Menu Package Details

Menu ini merupakan menu pada tahap ketiga dalam Order Process. Apabila user menekan button Package Details dan kedua tahap sudah dilewati sebelumnya, maka user akan mendapatkan tampilan Menu Package Details seperti lampiran di bawah ini.

Pada menu ini, user akan diminta untuk mengisi detail paket (Fill The Package Details) melalui beberapa kolom yang harus diisi oleh user. Kolom itu antara lain:

a. Nama Barang

User mengisi nama barang seperti jenis barang, contoh : aksesoris, kesehatan, buku, alat masak, dan lain-lain.

b. Quantity

Diisi oleh dropdown unit berapa pcs tergantung oleh berapa barang yang ingin dikirim oleh user.

c. Weight

Diisi dengan menginput angka dalam satuan kilogram.

d. Checkbox Fragile

Checkbox ini berfungsi memberitahu sistem Expedia dan kurir bahwa barang yang dikirim merupakan barang pecah belah yang mudah rapuh atau pecah. Oleh karena itu, dengan adanya checkbox ini, diharapkan pihak kurir atau pengirim paket dapat lebih hati-hati dalam mengirimkan barang tersebut, sehingga terhindar dari adanya kerusakan pada barang. Barang dengan sifat *fragile* akan dikenakan biaya tambahan (*additional cost*) per barang karena barang dengan sifat ini akan diberi *packaging* atau kemasan dari kayu yang mana merupakan packaging dengan harga yang lebih daripada *packaging* reguler yang hanya menggunakan plastik atau kardus.

e. Checkbox Insurance

Checkbox ini memiliki tujuan agar pihak Expedia dapat memberikan asuransi terhadap adanya kerusakan atau hilangnya paket. Oleh karena itu, seperti halnya barang dengan sifat fragile, barang dengan tambahan asuransi akan diberikan *additional cost*.

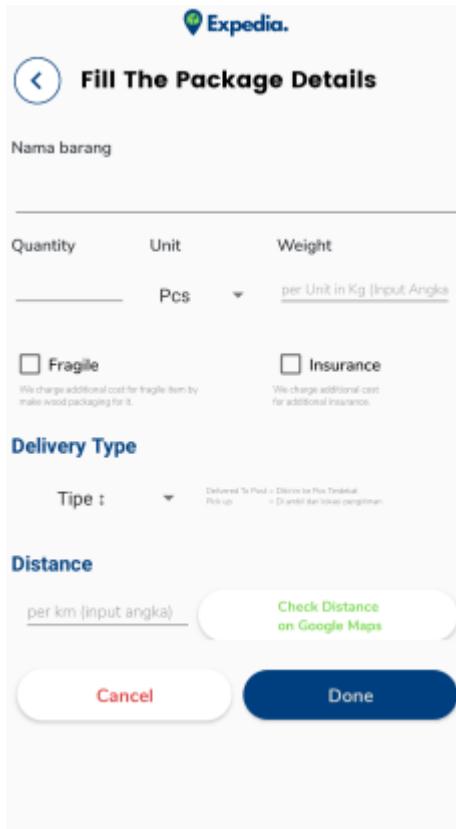
f. Delivery Type

Spinner ini berisi pilihan tipe pengiriman yang ingin user pilih. Beberapa tipe yang tersedia adalah “Delivered to Post” dan “Pick Up”. Pilihan “delivered to post” berarti user perlu mengirimkan paket ke pos terdekat. Sedangkan pilihan “Pick Up” berarti paket diambil dari lokasi pengiriman (alamat pengirim).

g. Distance

Merupakan kolom untuk mengisi berapa jarak dari alamat pengirim ke alamat penerima melalui Google Maps yang tersedia pada button di sebelahnya yang bertuliskan “Check Distance on Google Maps”.

Pada menu ini terdapat dua pilihan button yaitu, Button Cancel dan Button Done.



Berikut ini merupakan kodingan untuk menampilkan Menu Package Details, spinner, kolom, dan button.

```
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentActivity;

import android.content.DialogInterface;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
```

```
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

import static com.example.mobappproject.FormAddress.id_pengirimDB;
import static com.example.mobappproject.FormAddressPenerima.id_penerimaDB;
import static com.example.mobappproject.SignInActivity.userID;

public class MenuPackageDetails extends AppCompatActivity {
    public static String txtHarga, id_transaksi;

    private String txtNamaBarang,txtQuantity,txtWeight,txtJarak;

    private String txtDeliveryType, txtspinUnit;
    private String fnl_berat, fnl_jarak;

    private Spinner deliveryType, spinUnit;
    private Button btnDone,btnMaps;
    private EditText edtNamaBarang,edtQuantity,edtWeight,edtJarak;
    private CheckBox cbAsuransi,cbFragile;

    private String urlPackageDetails = "http://192.168.1.78/mobappbackend/router/inputpackagedetail.php";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_menu_package_details);

        edtNamaBarang = (EditText)findViewById(R.id.edtNamaBarang);
        edtQuantity = (EditText)findViewById(R.id.edtQuantity);
        edtWeight = (EditText)findViewById(R.id.edtWeight);
        edtJarak = (EditText)findViewById(R.id.edtJarak);
        deliveryType = (Spinner)findViewById(R.id.deliveryType);
        spinUnit = (Spinner) findViewById(R.id.spinUnit);
        cbAsuransi = (CheckBox)findViewById(R.id.cbAsuransi);
        cbFragile = (CheckBox) findViewById(R.id.cbFragile);

        btnDone = (Button)findViewById(R.id.btnDone);
        btnDone.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
@Override
public void onClick(View v) {

    txtWeight = edtWeight.getText().toString();
    txtJarak = edtJarak.getText().toString();

    int itungberat = Integer.parseInt(txtWeight);
    int itungjarak = Integer.parseInt(txtJarak);

    int harga = 10000* itungberat * itungjarak;

    Integer converthasil = new Integer(harga);
    txtHarga= converthasil.toString();

    Integer berattt = new Integer(itungberat);
    fnl_berat = berattt.toString();

    Integer jarakkk = new Integer(itungjarak);
    fnl_jarak = jarakkk.toString();

}

txtNamaBarang = edtNamaBarang.getText().toString();
txtQuantity = edtQuantity.getText().toString();
txtDeliveryType = deliveryType.getSelectedItem().toString();
txtspinUnit = spinUnit.getSelectedItem().toString();

String txtAsuransi ;
if(cbAsuransi.isChecked() ){
    txtAsuransi = "1";
}else {
    txtAsuransi = "0";
}
String finalTxtAsuransi = txtAsuransi;

String txtFragile;
if(cbFragile.isChecked() ){
    txtFragile = "1";
}else {
    txtFragile = "0";
}
String finaltxtFragile = txtFragile;
```

```

//Toast.makeText(getApplicationContext(),noplant + carmodel + caryear + carcolor + nora
AlertDialog.Builder ad = new AlertDialog.Builder( context: MenuPackageDetails.this);
ad.setTitle("Konfirmasi Package Details");
ad.setMessage("Apakah data yang anda sudah isi benar?");

ad.setPositiveButton( text: "YES", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        RequestQueue queue = Volley.newRequestQueue( context: MenuPackageDetails.this);
        StringRequest stringRequest = new StringRequest(Request.Method.POST, urlPackageDetails, new
            @Override
            public void onResponse(String response) {
                try {
                    System.out.print(response);
                    JSONObject jObj = new JSONObject(response);

                    int sukses = jObj.getInt( name: "code");
                    if (sukses == 200) {
                        id_transaksi = jObj.getString( name: "data");
                        Toast.makeText(MenuPackageDetails.this.getApplicationContext(), text: "Data p

```



```

Intent i = new Intent(getApplicationContext(),MenuOrderProcess.class);
Toast.makeText(MenuPackageDetails.this.getApplicationContext(), text: "Klik I
startActivity(i);
finish();

MyCarPage fragmycar = new MyCarPage();
fragmycar.setArguments(getActivity().getIntent().getExtras());
Bundle bundle = new Bundle();
bundle.putString("username",username);
fragmycar.setArguments(bundle);
getActivity().getSupportFragmentManager().beginTransaction().replace(R.id.
} else {
    Toast.makeText(MenuPackageDetails.this.getApplicationContext(), text: "Data p
    Toast.makeText(MenuPackageDetails.this.getApplicationContext(), text: "Periks
}

}
catch (Exception ex) {
    Log.e( tag: "Error", ex.toString());
}
}
}, new Response.ErrorListener() {
    @Override

```

```
        @Override
        public void onErrorResponse(VolleyError error) {
            Log.e( tag: "Error", error.getMessage());
            Toast.makeText( context: MenuPackageDetails.this, error.getMessage(), Toast.LENGTH_SHORT
        } }) {

        @Override
        protected Map<String, String> getParams() {
            Map<String, String> params = new HashMap<>();
            params.put( k: "nama_barang", txtNamaBarang);
            params.put( k: "kuantitas", txtQuantity);
            params.put( k: "unit_paket", txtspinUnit);
            params.put( k: "berat", fnl_berat );
            params.put( k: "jarak", fnl_jarak);
            params.put( k: "fragile", finaltxtFragile);
            params.put( k: "asuransibarang", finalTxtAsuransi);
            params.put( k: "tipe_pengambilan", txtdeliveryType);
            params.put( k: "harga", txtHarga);
            params.put( k: "id_user", userID);
            params.put( k: "id_penerima", id_penerimaDB);
            params.put( k: "id_pengirim", id_pengirimDB);

            return params;
        }

        @Override
        public Map<String, String> getHeaders() throws AuthFailureError {
            Map<String, String> params = new HashMap<>();
            params.put( k: "Content-Type", v: "application/x-www-form-urlencoded");
            return params;
        }
    };
    queue.getCache().clear();
    queue.add(stringRequest);
}

});

ad.setNegativeButton( text: "NO", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {

    }
});
ad.show();

btnMaps = (Button)findViewById(R.id.btnMaps);
btnMaps.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String uri = "http://maps.google.co.in/maps?q=" + "Jakarta Pusat";
        Intent i = new Intent(Intent.ACTION_VIEW, Uri.parse(uri));
        startActivity(i);
    }
});
```

4.2.17.1. Button Done

Button Done pada menu ini mengarahkan user ke menu berikutnya untuk menyelesaikan pemesanan, yaitu ke Menu Order Process.

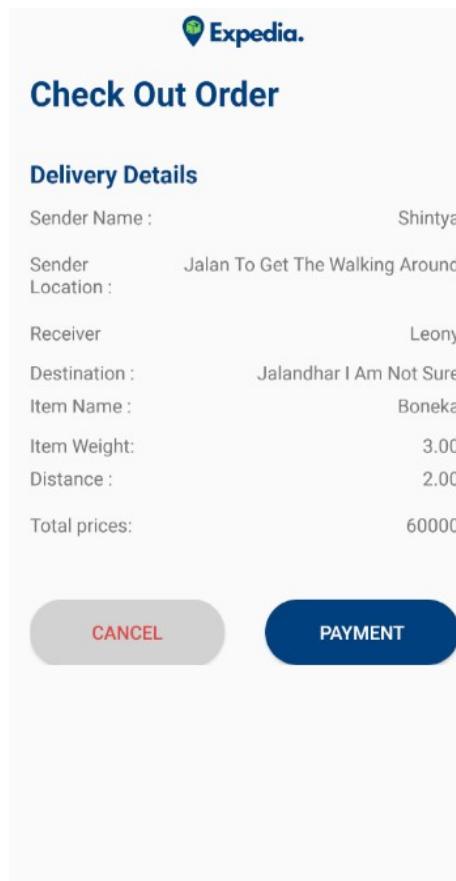


4.2.17.2. Button Cancel

Button Cancel ini menandakan user tidak bersedia melanjutkan ke tahap penyelesaian pemesanan, oleh karena itu apabila user menekan button ini, maka user akan tetap *stay* di Menu Package Details.



4.2.18. Menu Order Confirm Order



Pada menu ini, user akan melihat sebuah tampilan Check Out Order yang akan muncul apabila user menekan tombol pada menu sebelumnya, yaitu Button

Informasi Pengiriman Lengkap. Pada menu ini, user akan melihat rincian pesanan yang sudah berhasil dibuat dimulai dari nama pengirim, alamat pengirim, nama penerima, destinasi (alamat penerima), nama produk, berat produk, dan jarak, beserta total harga yang perlu dibayarkan.

Informasi nama pengirim, alamat pengirim, nama penerima, dan destinasi atau alamat penerima, serta nama dan berat produk diperoleh dari form address yang sudah diisi oleh user sebelumnya. Sedangkan distance diperoleh dari perhitungan sistem database yang diperoleh dari informasi pada form address. Total price atau harga total yang harus dibayarkan merupakan hasil kalkulasi Item Weight x Distance x 10.000. Pada contoh di atas, dapat kita lihat item weight sebesar 3 kilogram dengan distance yaitu 2 kilometer. Sehingga kalkulasinya adalah: $3 \times 2 \times 10.000 = 60.000$

Berikut penjelasan kodingan mengenai perhitungan tersebut :

```
txtWeight = edtWeight.getText().toString();
txtJarak = edtJarak.getText().toString();

int itungberat = Integer.parseInt(txtWeight);
int itungjarak = Integer.parseInt(txtJarak);

int harga = 10000* itungberat * itungjarak;

Integer converthasil = new Integer(harga);
txtHarga= converthasil.toString();

Integer berattt = new Integer(itungberat);
fnl_berat = berattt.toString();

Integer jarakkk = new Integer(itungjarak);
fnl_jarak = jarakkk.toString();
```

Pada menu ini juga terdapat dua button yang dapat dipilih oleh user yaitu, button Payment dan button Cancel.

Seperti yang tertera diatas, seluruh data di ambil dari menu pengisian form dengan menggunakan *passing public* data, lalu di import dengan kodingan sebagai berikut :

```
import static com.example.mobappproject.MenuPackageDetails.txtHarga;
import static com.example.mobappproject.MenuOrderProcess.namapengirim;
import static com.example.mobappproject.MenuOrderProcess.alamatpengirim;
import static com.example.mobappproject.MenuOrderProcess.namapenerima;
import static com.example.mobappproject.MenuOrderProcess.alamatpenerima;
import static com.example.mobappproject.MenuOrderProcess.namabarang;
import static com.example.mobappproject.MenuOrderProcess.txtberat;
import static com.example.mobappproject.MenuOrderProcess.txtjarak;
```

Lalu setelah data telah di import, data di olah dana di atur sesuai dengan letak yang telah diberikan seperti gambar diatas, dengan kodingan sebagai berikut :

```
txtTotalPrice = findViewById(R.id.txtTotalPrices);
txtTotalPrice.setText(txtHarga);

txtjarakganti = findViewById(R.id.txtjarakganti);
txtjarakganti.setText(txtjarak);

txtbarang = findViewById(R.id.txtbarang);
txtbarang.setText(namabarang);

txtItemWeight = findViewById(R.id.txtItemWeight);
txtItemWeight.setText(txtberat);

txtnamapengirim = findViewById(R.id.txtnamapengirim);
txtnamapengirim.setText(namapengirim);

txtpengirimAddress = findViewById(R.id.txtpengirimAddress);
txtpengirimAddress.setText(alamatpengirim);

txtnamapenerima = findViewById(R.id.txtnamapenerima);
txtnamapenerima.setText(namapenerima);

txtalamatpenerima = findViewById(R.id.txtalamatpenerima);
txtalamatpenerima.setText(alamatpenerima);
```

4.2.18.1. Button Payment

Apabila user menekan button Payment pada Menu Order Confirm Order, maka user akan diarahkan ke Menu Order Payment, di mana user akan diberi penjelasan lebih detail mengenai proses pembayaran (*payment*). Pada menu berikutnya user akan diarahkan untuk melakukan pembayaran melalui metode yang sudah ditetapkan.



4.2.18.2. Button Cancel

Button cancel disediakan agar user yang ingin membatalkan proses pemesanan. Button ini mengarahkan user ke menu sebelumnya, yaitu Menu Order Process.



4.2.19. Menu Order Payment

Pada menu ini, user akan melihat tampilan tulisan yang menandakan bahwa pemesanan berhasil melalui tulisan “Transaction Success”. Di menu ini pun, user akan melihat petunjuk dari pihak Expedia dalam melakukan pembayaran (*payment*). Pada contoh lampiran di bawah, kita dapat melihat salah satu metode pembayaran yaitu melalui mobile banking dengan virtual account yang sudah tertera. Selain nomor virtual account pembayaran, tertera juga kode pemesanan beserta nominal yang perlu dibayarkan. Di samping itu, Expedia juga memberikan catatan kepada *payer* agar tetap menyimpan bukti *screenshot* pembayaran dan mengirimnya ke nomor Whatsapp yang sudah tertera. Pada bagian bawah proses payment, terdapat button Done.



Transaction Success!

For now, you can make a payment
only by Bank Transfer to:

62378987

a.n. EXPEDIA INDONESIA

Rp 60000

Take a photo of the evidence of transfer
and send it to Whatsapp
+62 1390 4616 4842

Done

Berikut kodingan untuk menampilkan harga total dan button done :

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_menu_order_payment);
    txtTotalPrice = findViewById(R.id.txtTotalPrice);
    txtTotalPrice.setText(txtHarga);

    Button btnTransactionDone =(Button)findViewById(R.id.btnTransactionDone);
    btnTransactionDone.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent i = new Intent(getApplicationContext(), MainMenuActivity.class);
            startActivity(i);
        }
    });
}
```

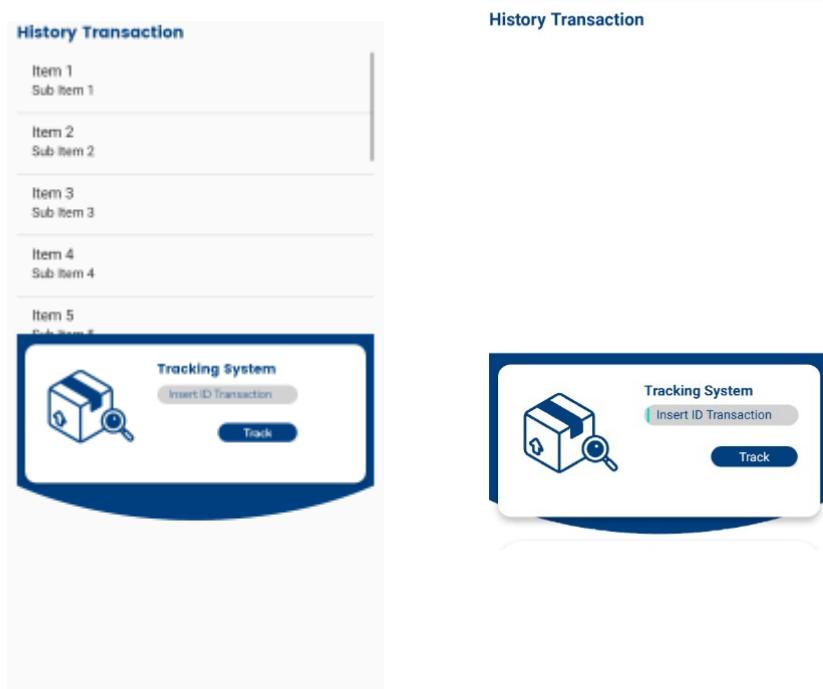
4.2.19.1. Button Done

Apabila user menekan tombol ini, maka user akan diarahkan kembali ke menu utama, yaitu Main Menu.

Done

4.2.20. Menu History Transaction

Menu ini merupakan menu agar user dapat melihat riwayat atau history dari transaksi yang pernah dibuat. Menu ini berfungsi agar user dapat kembali melihat detail pesanan, kurir, jenis barang, kode pemesanan, alamat penerima, dan lain sebagainya. Pada menu ini, terlihat beberapa list riwayat transaksi yang sudah dilakukan oleh user. Untuk melihat salah satu transaksi dan untuk mempermudah user melakukan pencarian, maka terdapat Tracking System di bagian paling bawah. Tracking system dilakukan dengan memasukkan ID Transaction atau kode pemesanan dan user dapat langsung menekan button “Track”. Apabila data yang dicari sudah ditemukan, maka akan muncul tampilan seperti di gambar ketiga. Menu ini merupakan menu yang berisi list tracking pesanan yang sedang diproses atau bahkan yang sudah selesai diproses. Pada menu ini terlihat beberapa informasi seperti ID Transaction, Sender Address, dan Receiver Address. Menu ini merupakan menu yang dapat dilihat oleh customer atau pelanggan menggunakan listView.





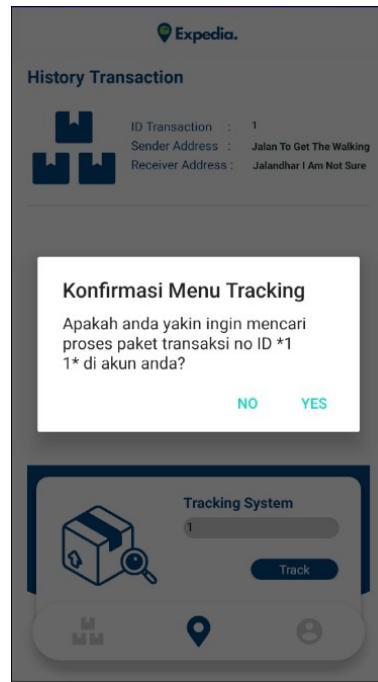
Berikut penjelasan kodingan diatas menggunakan listview untuk *history transaction* dengan 3 data :

```
@Nullable
@Override
public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable E
    username = getArguments().getString("username");
    return inflater.inflate(R.layout.activity_menu_tracking, container, attachToRoot: false);
}

@Override
public void onStart() {
    super.onStart();
    final String[] from={ "id_transaksi", "alamat_pengirim", "alamat_penerima"};//string array
    final int[] to={R.id.list_content2,R.id.list_content4,R.id.list_content6};//int array of views id

    list_anggota =new ArrayList<>();
    lv = getActivity().findViewById(R.id.listView);
```

Apabila user memasukkan kode transaksi pada kolom Tracking System yang tersedia lalu menekan tombol “Track”, maka tampilan akan menunjukkan toast atau pop-up seperti berikut:



Lalu setelah tampilan seperti ini, maka user yang ingin mentrack pesanan dengan kode tersebut perlu menekan pilihan “Yes”. Lalu akan muncul tampilan menu *Transaction Details*. Dengan mengambil data menggunakan id transaksi yang di input oleh user itu sendiri.

```
@Override
public void onErrorResponse(VolleyError error) {
    Log.e("Error", error.getMessage());
    Toast.makeText(getApplicationContext(), error.getMessage(), Toast.LENGTH_SHORT).show();
}

@Override
protected Map<String, String> getParams() {
    Map<String, String> params = new HashMap<>();
    params.put("id_transaksi", strTracking);

    return params;
}

@Override
public Map<String, String> getHeaders() throws AuthFailureError {
    Map<String, String> params = new HashMap<>();
    params.put("Content-Type", "application/x-www-form-urlencoded");
    return params;
}
};

queue.getCache().clear();
queue.add(stringRequest);
```

4.2.21. Menu Transaction Details

Menu ini merupakan menu yang tampil apabila user ingin melihat dan menelusuri riwayat pemesanan yang sudah terjadi melalui ID Transaction yang sudah diinput sebelumnya. Menu ini menampilkan detail transaksi seperti yang terlihat pada gambar di bawah ini.

Menu Transaction Details ini berfungsi agar user baik customer maupun admin dapat kembali melihat data dan informasi yang sudah tercatat untuk dijadikan sebuah bahan pencatatan atau hanya sekadar melihat dna mengetahui saja. Terkhusus bagi perusahaan, hal ini dapat mencegah adanya penipuan yang dilakukan oleh karyawan. Begitupun sebaliknya, apabila karyawan dicurigai atau dituduh melakukan kesalahan, maka karyawan dapat menunjukkan data dan informasi yang sebenarnya melalui Menu Transaction Details ini.

Pada menu ini, user dapat melihat beberapa informasi utama seperti:

a. Transaction ID

Menunjukkan kode transaksi atau ID Transaction dari pemesanan.

b. Sender Information (Informasi Pengirim)

Bagian kedua ini berisi nama pengirim, alamat pengirim, dan nomor telepon pengirim.

c. Receiver Information (Informasi Penerima)

Pada bagian ini, terdapat rincian mengenai informasi penerima, mulai dari nama penerima, alamat penerima, dan nomor telepon penerima.

d. Package Detail (Detail Paket)

Bagian ini berisi detail paket pemesanan yang terdiri dari nama paket (nama barang), jumlah atau kuantitas, unit package, bersifat fragile atau tidak, diberi tambahan asuransi atau tidak, berat paket, jarak tempuh pengiriman, total harga, dan delivery type, dan status paket.

e. Status Date Track

Di bagian ini, user dapat melihat status pemesanan paket, di antaranya apakah status tersebut sudah di pick up, delivery to post, warehouse transit, accepted by kurir, on going, atau failed.

Berikut kodingan menu tracking dengan menggunakan *passing public data* yang diambil melalui database dengan Json Object :

```

@Override
public void onClick(DialogInterface dialog, int which) {
    RequestQueue queue = Volley.newRequestQueue(getApplicationContext());
    StringRequest stringRequest = new StringRequest(Request.Method.POST, url_track_user, new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            try {
                JSONObject jObj = new JSONObject(response);
                int sukses = jObj.getInt("code");
                if (sukses == 200) {
                    Toast.makeText(getApplicationContext(), "Paket anda sedang dalam proses! Berikut detail yang
                    data = jObj.getJSONObject("data");
                    System.out.print(data);
                    txtid_transaksi = data.getString("id_transaksi");
                    //o
                    System.out.print(txtid_transaksi);
                    txtnamapengirim = data.getString("nama_pengirim");
                    txtalamatpengirim = data.getString("alamat_pengirim");
                    txttelp_peng= data.getString("telp_peng");
                    txtnama_penerima = data.getString("nama_penerima");
                    txtalamat_penerima= data.getString("alamat_penerima");
                    txttelp_pen = data.getString("telp_pen");
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
    });
    queue.add(stringRequest);
}

```

```

//10
txtnama_barang = data.getString( name: "nama_barang");
txtkuantitas = data.getString( name: "kuantitas");
txtunit_paket = data.getString( name: "unit_paket");
txtfragile = data.getString( name: "fragile");
txtasuransibarang = data.getString( name: "asuransibarang");
txtberat = data.getString( name: "berat");
txtjarak = data.getString( name: "jarak");
txtharga = data.getString( name: "harga");
txttipe_pengambilan = data.getString( name: "tipe_pengambilan");
txtstatus_paket = data.getString( name: "status_paket");
//7
txttanggal_pickup = data.getString( name: "tanggal_pickup");
txttanggal_deliveredtopost = data.getString( name: "tanggal_deliveredtopost");
txttanggal_warehousetransit = data.getString( name: "tanggal_warehousetransit");
txttanggal_acceptedbykurir = data.getString( name: "tanggal_acceptedbykurir");
txttanggal_ongoing = data.getString( name: "tanggal_ongoing");
txttanggal_arrived = data.getString( name: "tanggal_arrived");
txttanggal_failed = data.getString( name: "tanggal_failed");

Intent i = new Intent(getApplicationContext(), TransactionDetails.class);
startActivity(i);

```

4.2.21.1. Button Done Tracking

Apabila user menekan tombol ini, maka user akan diarahkan ke menu utama, yaitu Main Menu Activity.



4.2.22. Menu Kurir Tracking Update

Menu ini merupakan menu untuk melacak (*track*) paket dari sisi Kurir. Pada layar akan terlihat beberapa list item paket yang tercatat pada sistem. Pada bagian bawah, Kurir dapat memperbarui atau update status paket melalui memasukkan ID Transaction dan memilih status paket melalui spinner yang tersedia. Kurir dapat memilih untuk terlebih dahulu menambahkan catatan pada button Notes atau langsung memilih button Update Done.

Adapun pada menu ini, user dapat mengetahui bahwa terdapat tujuh status tracking dalam sistem, di antaranya sebagai berikut:

a. Status “Not Yet Process”

Status ini menyatakan bahwa pesanan belum diproses sama sekali oleh pihak ekspedisi.

b. Status “Pick Up”

Status ini berarti bahwa paket atau pesanan sudah *dipick-up* atau dijemput dari lokasi pengirim oleh pihak ekspedisi.

c. Status “Delivered to Post”

Status ini menyatakan bahwa paket sudah sampai di lokasi ekspedisi untuk dikirim ke lokasi gudang atau *warehouse* ekspedisi yang dituju.

d. Status “Warehouse Transit”

Status ini berarti bahwa paket sudah sampai di gudang atau *warehouse* yang terdapat di daerah lokasi penerima.

e. Status “Accepted by Courier”

Dari status ini, user dapat mengetahui bahwa paket sudah berada di tangan kurir dan siap untuk dikirim oleh kurir perorangan.

f. Status “On Going Delivery”

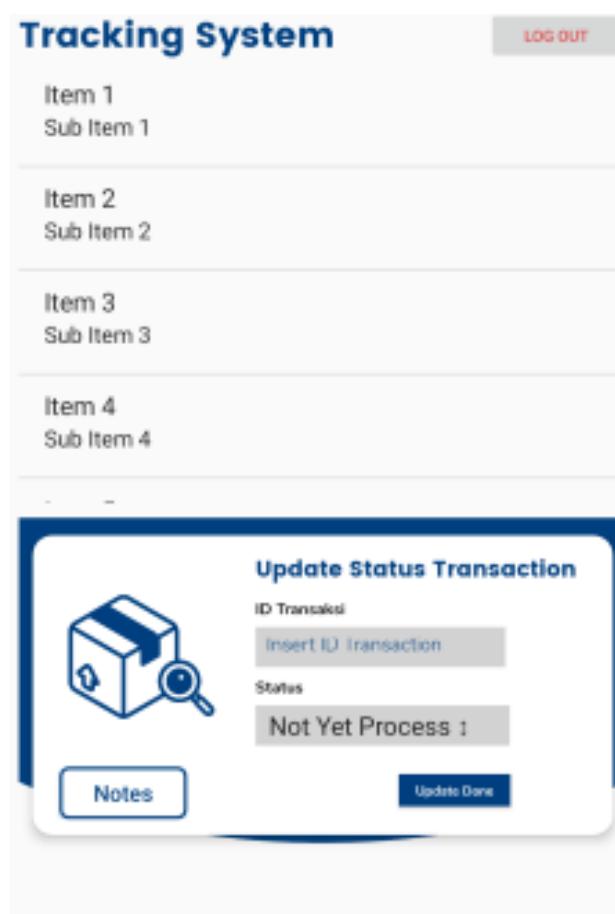
Status ini menyatakan bahwa paket yang berada di kurir sedang dalam perjalanan menuju lokasi alamat penerima secara langsung.

g. Status “Package Arrived”

Status ini berarti paket atau pesanan yang dikirim oleh kurir sudah sampai di lokasi alamat penerima.

h. Status “Failed to Delivered”

Status ini menyatakan bahwa paket atau pesanan tidak berhasil dikirimkan yang bisa terjadi karena berbagai kendala, misalnya keterlambatan pengiriman, kesalahan alamat penerima, dan hal lainnya.



Berikut kodingan yang akan diinput oleh kurir yaitu ID Transaction dan status yang dipilih :

```
@Override
public void onErrorResponse(VolleyError error) {
    Log.e("tag: Error", error.getMessage());
    Toast.makeText(context: AdminTrackingUpdate.this, error.getMessage(), Toas
} }

@Override
protected Map<String, String> getParams() {
    Map<String, String> params = new HashMap<>();
    params.put( k: "id_transaksi", txt_idupdatestatus);
    params.put( k: "status_paket", txt_updatestatus);

    return params;
}

@Override
public Map<String, String> getHeaders() throws AuthFailureError {
    Map<String, String> params = new HashMap<>();
    params.put( k: "Content-Type", v: "application/x-www-form-urlencoded");
    return params;
}
};
```

Dengan menggunakan PHP, kami juga menginput tanggal yang akan sama dengan waktu kurir mengupdate status yang ada :

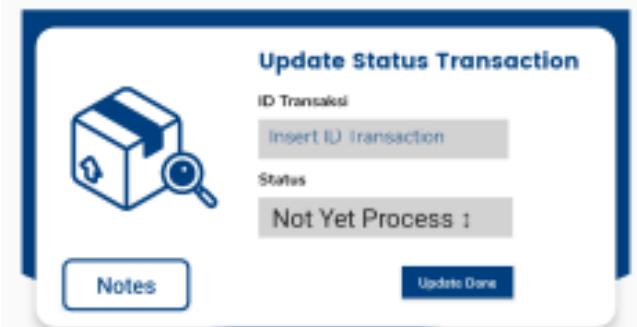
```
<?php
require "../util/connection.php";
/**
 * @return object
 */
function UpdatestatusinDB($status_paket, $id_transaksi, $kolom_tanggal) {
    $op = "database/UpdateStatusinDB";

    try {
        $con = GetConnection();

        $query = "UPDATE transaksi_paket SET status_paket = ?, ". $kolom_tanggal . " = now() WHERE id_transaksi = ?";
        $result = $con->prepare($query);
        $result->execute([
            $status_paket,
            $id_transaksi
        ]);
    } catch (\Exception $e) {
        throw new Exception("[$op] $e");
    }
}
?>
```

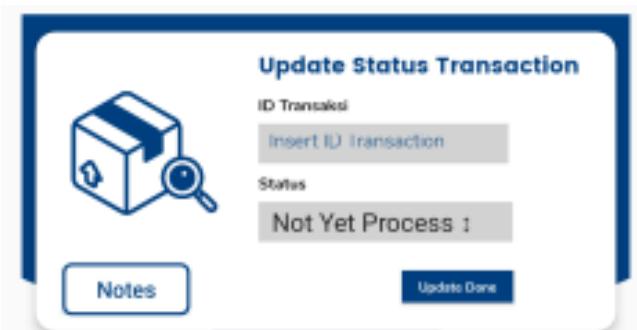
4.2.22.1. Button Notes

Button ini mengarahkan user ke menu Notes Driver.



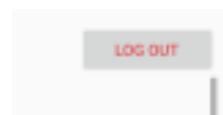
4.2.22.2. Button Update Done

Apabila user menekan button ini, maka secara otomatis data yang diupdate akan dikirim ke sistem database untuk diperbarui.



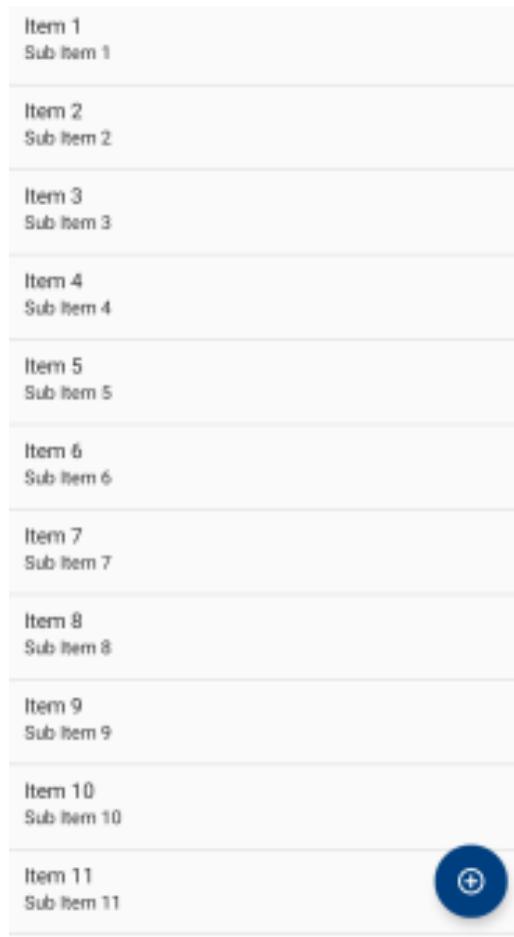
4.2.22.4 Button Log Out

Apabila user menekan button ini, maka admin akan diarahkan ke menu Sign yang mana berarti user melakukan sign out dan keluar dari akunnya.



4.2.23. Menu Notes Driver

Menu ini berisi notes atau catatan untuk driver atau kurir, terdiri dari list item paket yang sedang berjalan atau berproses. Menu ini menggunakan SQLite untuk menampung data yang ada.



Berikut kodingan SQLite yang ada untuk menampung notes driver :

```
@Override  
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {  
    sqLiteDatabase.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);  
    onCreate(sqLiteDatabase);  
}  
  
private SQLiteDatabase database;  
  
public void open() throws SQLException {  
    database = this.getWritableDatabase();  
}  
  
private String[] allColumns =  
{COLUMN_ID, COLUMN_JUDULNOTES, COLUMN_KONTENNOTES};  
  
private Notes cursorToNotes(Cursor cursor){  
    Notes notes = new Notes();  
  
    notes.setID(cursor.getLong(0));  
    notes.setJudulNotes(cursor.getString(1));  
    notes.setKontenNotes(cursor.getString(2));  
    return notes;  
}
```

```

public void createNotes(String JudulNotes, String KontenNotes) {
    ContentValues values = new ContentValues();
    values.put(COLUMN_JUDULNOTES, JudulNotes);
    values.put(COLUMN_KONTENNOTES, KontenNotes);

    database.insert(TABLE_NAME, nullColumnHack, values);
}

public Notes getNotes(Long id){
    Notes notes = new Notes();

    Cursor cursor = database.query(TABLE_NAME, allColumns, selection: "_id="+id, selectionArgs: null, groupBy: null, having: null, cursor.moveToFirst();
    cursor.close();
    return notes;
}

public ArrayList<Notes> getAllNotes(){
    ArrayList<Notes> daftarnotes = new ArrayList<~>();
    Cursor cursor = database.query(TABLE_NAME, allColumns, selection: null, selectionArgs: null, groupBy: null, having: null, cursor.moveToFirst();
    while (!cursor.isAfterLast()){
        Notes notes = cursorToNotes(cursor);
        daftarnotes.add(notes);
        cursor.moveToNext();
    }

    Cursor cursor = database.query(TABLE_NAME, allColumns, selection: null, selectionArgs: null, groupBy: null, having: null, cursor.moveToFirst();
    while (!cursor.isAfterLast()){
        Notes notes = cursorToNotes(cursor);
        daftarnotes.add(notes);
        cursor.moveToNext();
    }

    cursor.close();
    return daftarnotes;
}

public void updateNotes(NOTES notes){
    String filter = "_id=" + notes.getID();
    ContentValues args = new ContentValues();
    args.put(COLUMN_JUDULNOTES, notes.getJudulNotes());
    args.put(COLUMN_KONTENNOTES, notes.getKontenNotes());

    database.update(TABLE_NAME, args, filter, whereArgs: null);
}

public void deleteNotes(Long id){
    String filter = "_id=" + id;

    database.delete(TABLE_NAME, filter, whereArgs: null);
}

```

Berikut kodingan Java untuk menginput data notes driver ke dalam SQLite di atas :

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_notes_driver);  
  
    dbHandler = new SQLite(context: this);  
    try {  
        dbHandler.open();  
    } catch (SQLException e){  
        e.printStackTrace();  
    }  
  
    values = dbHandler.getAllNotes();  
  
    ArrayAdapter<Notes> adapter = new ArrayAdapter<~> (context: this, android.R.layout.simple_list_item_1, values)  
    setListAdapter(adapter);  
  
    list = (ListView) findViewById(android.R.id.list);  
    list.setOnItemLongClickListener(this);  
  
    FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);  
    fab.setOnClickListener(new View.OnClickListener() {  
        @Override  
        public void onClick(View view) {...}  
    });|
```

4.2.24. Menu Notes Tambah

Menu ini tertampil apabila user menekan tombol tanda “+” pada menu sebelumnya, yaitu Menu Notes Driver. Di menu inilah, admin dapat menambahkan catatan bagi para driver.

ID Transaction	:	1
Notes	:	ternyata rumah deket man gitu dll
ID Transaction	:	1
Notes	:	many ditaruh dipagar

ID Transaction	
Notes	
RESET	SIMPAN



Berikut kodingan yang digunakan untuk menambah data yang ada:

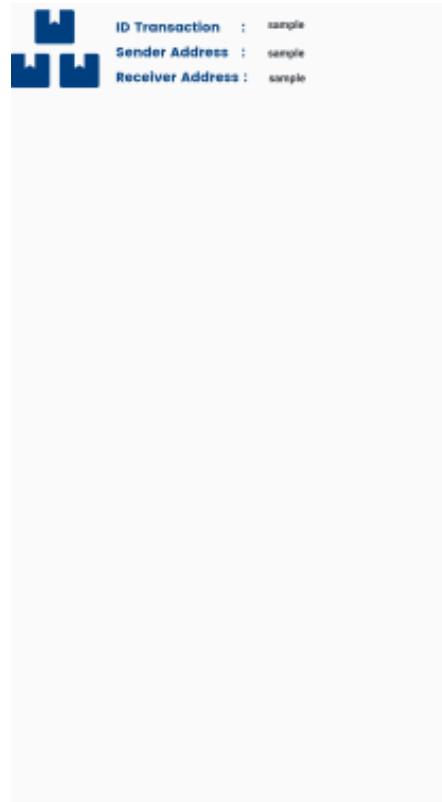
```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_notes_tambah);  
  
    final EditText edtNama = (EditText) findViewById(R.id.edtNama);  
    final EditText edtKatagori = findViewById(R.id.edtKatagori);  
  
    Button btnReset = (Button) findViewById(R.id.btnReset);  
    Button btnSimpan = findViewById(R.id.btnSimpan);  
  
    final SQLite dbHandler = new SQLite(context: this);  
  
    try {  
        dbHandler.open();  
    } catch (SQLException e) {  
        e.printStackTrace();  
    }  
}
```

```
btnSimpan.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        Notes notes = new Notes();  
        String JudulNotes = edtNama.getText().toString();  
        String KontenNotes = edtKatagori.getText().toString();  
  
        dbHandler.createNotes(JudulNotes, KontenNotes);  
  
        Toast.makeText(context: NotesTambah.this, text: "Note berhasil ditambahkan", Toast.LENGTH_LONG).show()  
        edtNama.setText("");  
        edtKatagori.setText("");  
  
        edtNama.requestFocus();  
  
        Intent i = new Intent(packageContext: NotesTambah.this, NotesDriver.class);  
        startActivity(i);  
        NotesTambah.this.finish();  
        dbHandler.close();  
    }  
});
```

```
btnReset.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        edtNama.setText("");  
        edtKatagori.setText("");  
        edtNama.requestFocus();  
  
        Intent i = new Intent(packageContext: NotesTambah.this, AdminTrackingUpdate.class);  
        startActivity(i);  
        NotesTambah.this.finish();  
        dbHandler.close();  
    }  
});
```

4.2.25. Menu List Tracking Kurir

Sama seperti menu list transaction user, menu ini merupakan menu yang berisi list tracking pesanan yang sedang diproses atau bahkan yang sudah selesai diproses. Pada menu ini terlihat beberapa informasi seperti ID Transaction, Sender Address, dan Receiver Address. Yang membedakan menu ini dengan menu sebelumnya adalah bahwa menu ini merupakan menu yang hanya bisa dilihat oleh kurir untuk melihat seluruh pesanan dan bukan pelanggan.



Berikut kodingan untuk melihat seluruhnya :

```
final String[] from={"id_transaksi","alamat_pengirim","alamat_penerima"};//string array
final int[] to={R.id.list_content8,R.id.list_content9,R.id.list_content10};//int array of views id's

RequestQueue queue = Volley.newRequestQueue( context: AdminTrackingUpdate.this);
StringRequest stringRequest = new StringRequest(Request.Method.POST, url_get_mahasiswa, new Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        try {
            JSONObject jobj = new JSONObject(response);
            JSONArray member= jObj.getJSONArray(TAG_MAHASISWA);
            //Toast.makeText(SemuaMahasiswaActivity.this, "sesudah tag", Toast.LENGTH_SHORT).show();

            for (int i=0; i< member.length();i++) {
                JSONObject a=member.getJSONObject(i);
                String id=a.getString(TAG_ID);
                String nama=a.getString(TAG_NAMA);
                String alamat = a.getString(TAG_ALAMAT);

                HashMap<String,String> map=new HashMap<>();
                map.put("id_transaksi",id);
                map.put("alamat_pengirim",nama);
                map.put("alamat_penerima",alamat);
                //Toast.makeText(SemuaMahasiswaActivity.this, "nama"+nama, Toast.LENGTH_SHORT).show();
                list_anggota.add(map);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
});
```

4.3. Query Pembuatan Tabel Basis Data

4.3.1. Create dan Use Database

```
CREATE DATABASE "Table_Expedia"  
USE "Table_Expedia"
```

4.3.2. Create Tabel users

```
CREATE TABLE users(  
id VARCHAR(11) PRIMARY KEY,  
name VARCHAR(50) NOT NULL,  
password VARCHAR(255) NOT NULL,  
email VARCHAR(50) NOT NULL,  
role CHAR(10) NOT NULL,  
join_date DATE);
```

4.3.3. Create Tabel daftar_alamat_pengirim

```
CREATE TABLE daftar_alamat_pengirim(  
id_alamatpengirim VARCHAR(11) PRIMARY KEY,  
nama VARCHAR(50) NOT NULL,  
telpon VARCHAR(20) NOT NULL,  
alamat_lengkap VARCHAR(250) NOT NULL,  
provinsi VARCHAR(30) NOT NULL,  
kota VARCHAR(30) NOT NULL,  
kodepos INT NOT NULL,  
latitude FLOAT NOT NULL,  
longitude FLOAT NOT NULL,  
notes_tambahan VARCHAR(250) NOT NULL);
```

4.3.3. Create Tabel daftar_alamat_penerima

```
CREATE TABLE daftar_alamat_penerima(  
id_alamatpenerima VARCHAR(11) PRIMARY KEY,  
nama VARCHAR(50) NOT NULL,  
telpon VARCHAR(20) NOT NULL,  
alamat_lengkap VARCHAR(250) NOT NULL,  
provinsi VARCHAR(30) NOT NULL,  
kota VARCHAR(30) NOT NULL,  
kodepos INT NOT NULL,  
latitude FLOAT NOT NULL,
```

```
longitude FLOAT NOT NULL,  
notes_tambahan VARCHAR(250) NOT NULL);
```

4.3.4. Create Tabel Tabel transaksi_paket

```
CREATE TABLE transaksi_paket(  
id_transaksi VARCHAR(11) PRIMARY KEY,  
id_pengirim VARCHAR(11) NOT NULL  
CONSTRAINT FK_transaksi_paket_id_pengirim REFERENCES  
daftar_alamat_pengirim(id_alamatpengirim),  
id_penerima VARCHAR(11) NOT NULL  
CONSTRAINT FK_transaksi_paket_id_penerima REFERENCES  
daftar_alamat_penerima(id_alamatpenerima),  
id_user VARCHAR(11) NOT NULL  
CONSTRAINT FK_transaksi_paket_id_user REFERENCES users(id),  
nama_barang VARCHAR(30) NOT NULL,  
kuantitas INT NOT NULL,  
unit_paket INT NOT NULL,  
berat DECIMAL(15,2) NOT NULL,  
fragile INT NOT NULL,  
asuransibarang INT NOT NULL,  
tipe_pengambilan VARCHAR(20) NOT NULL,  
jarak DECIMAL(15,2) NOT NULL,  
harga DECIMAL(15,2) NOT NULL,  
status_paket VARCHAR(50) NOT NULL,  
tanggal_pickup DATE,  
tanggal_deliveredtopost DATE,  
tanggal_warehousetransit DATE,  
tanggal_acceptedbykurir DATE,  
tanggal_ongoing DATE,  
tanggal_arrived DATE,  
tanggal_failed DATE);
```

4.4. Advanced Query Basis Data

4.4.1. View

View merupakan sebuah tabel virtual yang dibuat berdasarkan hasil set dari pernyataan di SQL. View terdiri dari kolom dan baris yang ditampilkan seperti tabel sesungguhnya, namun view memiliki bidang yang terdiri atas satu atau beberapa tabel nyata dalam suatu basis data. Query di bawah merupakan aktivitas pembuatan view untuk menampilkan info data *user*, berupa id transaksi, nama, email, nomor telepon, id alamat penerima, nama penerima, dan nomor telepon penerima yang

diperoleh dari tabel *user* yang digabung dengan tabel daftar alamat pengirim pada id alamat pengirim, serta tabel alamat penerima pada id alamat penerima. Dalam hal ini, hasil query yang akan ditampilkan berupa data *user* dimana id transaksi merupakan ‘TR001’.

Create View

```
CREATE VIEW show_info AS
SELECT id_transaksi, id_user, u.name, u.email, ad.telpon "Telepon Pengirim",
ap.id_alamatpenerima, ap.nama, ap.telpon "Telepon Penerima"
FROM transaksi_paket tp
INNER JOIN users u ON tp.id_user = u.id
INNER JOIN daftar_alamat_pengirim ad ON tp.id_pengirim = ad.id_alamatpengirim
INNER JOIN daftar_alamat_penerima ap ON tp.id_penerima =
ap.id_alamatpenerima
WHERE id_transaksi = (SELECT id_transaksi FROM transaksi_paket
WHERE id_transaksi = 'TR001');
```

Display View

```
SELECT * FROM show_info;
```

4.4.2. Procedure

Procedure merupakan salah satu sebuah program tersimpan yang berisi kode SQL dengan tujuan memudahkan dalam menjalankan atau menggunakan kode yang sama secara berulang kali. Query di bawah merupakan aktivitas pembuatan procedure yang digunakan untuk menampilkan informasi transaksi yang telah dilakukan berupa id transaksi, nama pengirim dan penerima, alamat lengkap pengirim dan penerima, nomor telepon pengirim dan penerima, nama barang, berat barang, jarak tempuh, harga barang, dan asuransi barang dari transaksi paket yang dihubungkan dengan tabel daftar alamat pengirim pada id alamat pengirim, dan tabel daftar alamat penerima pada id alamat penerima. Dalam hal ini, id transaksi diperoleh dari id transaksi dari transaksi paket dimana id transaksi merupakan ‘TR003’.

Create Procedure

```
CREATE PROCEDURE ShowInfoTrans @id_transaksi VARCHAR(11)
AS
SELECT id_transaksi, id_user, u.name, u.email, ad.telpon 'Telepon Pengirim',
ap.id_alamatpenerima, ap.nama, ap.telpon 'Telepon Penerima'
FROM transaksi_paket tp
INNER JOIN users u ON tp.id_user = u.id
INNER JOIN daftar_alamat_pengirim ad ON tp.id_pengirim = ad.id_alamatpengirim
INNER JOIN daftar_alamat_penerima ap ON tp.id_penerima =
ap.id_alamatpenerima
```

```
WHERE id_transaksi = @id_transaksi;
```

Display Data

```
SELECT * FROM transaksi_paket;
```

Execute Procedure

```
EXEC ShowInfoTrans @id_transaksi = 'TR003';
```

4.4.3. Cursor

Cursor merupakan sebuah objek dalam suatu basis data yang memungkinkan untuk mengambil setiap baris dan melakukan manipulasi data yang ada. Query di bawah merupakan aktivitas pembuatan cursor yang digunakan untuk mengambil dan menampilkan data berupa nama barang pada setiap baris dalam tabel transaksi yang dilakukan secara bertahap pada waktu yang bersamaan.

Declare Variable

```
DECLARE @nama_barang VARCHAR(30);
```

Declare Cursor

```
DECLARE cursor_barang CURSOR  
FOR SELECT nama_barang FROM transaksi_paket;
```

Open Cursor and Fetch Rows

```
OPEN cursor_barang;
```

```
FETCH NEXT FROM cursor_barang INTO  
@nama_barang;
```

```
WHILE @@FETCH_STATUS = 0  
BEGIN  
    PRINT @nama_barang;  
    FETCH NEXT FROM cursor_barang INTO  
        @nama_barang  
END;
```

Close Cursor

```
CLOSE cursor_barang;
```

Deallocate Cursor

```
DEALLOCATE cursor_barang;
```

4.4.4. Trigger

Trigger merupakan salah satu dari bagian dari kode prosedural yang berfungsi sebagai pemicu yang akan dijalankan ketika suatu peristiwa tertentu terjadi. Query di bawah merupakan aktivitas pembuatan trigger yang digunakan untuk memicu terjadinya proses pemasukkan data pengirim baru pada tabel daftar alamat pengirim, seperti id alamat pengirim, nama, telpon, alamat lengkap, provinsi, kota, kodepos, latitude, longitude, dan notes tambahan. Dalam hal ini, penambahan data tersebut akan memicu munculnya data baru pada tabel daftar alamat pengirim dan juga pada tabel daftar alamat penerima sebagai hasil dari proses trigger yang dieksekusi.

Create Trigger

```
CREATE TRIGGER insert_alamat
ON daftar_alamat_pengirim
AFTER INSERT
AS
    INSERT INTO daftar_alamat_penerima
        SELECT * FROM inserted;
```

Select Data for Trigger

```
SELECT * FROM daftar_alamat_pengirim;
SELECT * FROM daftar_alamat_penerima;
```

Insert Data

```
INSERT INTO daftar_alamat_pengirim VALUES ('AD004','Manila
Mania','082163995391','Jalan Tukad Musi No. 1, Renon, Denpasar Selatan,Denpasar,
Bali - Indonesia 80226','Bali','Denpasar','80226',-8.686110,115.240090,'Seberang
Gereja Paroki Roh Kudus Katedral Denpasar');
```

Display Data

```
SELECT * FROM daftar_alamat_pengirim;
SELECT * FROM daftar_alamat_penerima;
```

Drop Trigger

```
DROP TRIGGER insert_alamat;
```

4.4.5. Index

Index merupakan suatu tabel yang digunakan untuk tujuan pencarian khusus yang memudahkan pengambilan data dalam waktu yang singkat. Query di bawah merupakan aktivitas pembuatan index untuk id transaksi dalam mencari dan mengambil data secara keseluruhan pada tabel transaksi paket dimana id transaksi yang dicari adalah ‘TR004’.

Create Index

```
CREATE INDEX ix_transaction_id  
ON transaksi_paket(id_transaksi);
```

Display Data

```
SELECT * FROM transaksi_paket  
WHERE id_transaksi = 'TR004';
```

Drop Index

```
DROP INDEX transaksi_paket.ix_transaction_id;
```

4.5. Kesimpulan

Proyek ini diharapkan dapat memberikan berbagai manfaat bagi perusahaan Expedia dan bagi pengguna, yaitu masyarakat sebagai pengirim, penerima, dan jasa kurir. Berikut terdapat manfaat yang diharapkan dapat diperoleh dari sistem ini, antara lain:

- Menghindari adanya ancaman yang dapat mempengaruhi basis data perusahaan oleh karena adanya sistem.
- Menciptakan proses bisnis lebih efisien dan efektif.
- Mewujudkan segala aspek dalam sistem informasi, terutama dalam sistem basis data kehidupan nyata.
- Membantu meminimalisir kesalahan manusia (*human error*) yang dapat merugikan perusahaan dan pengguna.
- Memudahkan pelanggan dalam mendapatkan informasi terkait paket pengiriman yang dipesan.
- Meningkatkan efektivitas dan efisiensi kinerja karyawan dalam membuat laporan lebih lanjut kepada perusahaan.
- Memudahkan pengguna dalam mencatat informasi paket pemesanan dibandingkan mencatat melalui kertas formulir.
- Memudahkan pengguna dalam melacak lokasi paket pemesanan.

PERANAN ANGGOTA

Proyek ini bertujuan untuk dikerjakan bersama dengan anggota kelompok kami. Tanpa adanya sinergi, solidaritas, dan kerja sama dari seluruh anggota kelompok, maka proyek ini tidak dapat terselesaikan. Setiap anggota kelompok berperan penting bagi keberlangsungan proyek ini. Oleh sebab itu, berikut kami jabarkan seluruh *jobdesk* yang dimiliki oleh setiap anggota kelompok:

Pembagian Tugas Query (Tidak termasuk Subquery) :

1. Irene Maria Joseph: Query di menu profil dan menu *insert transaction*

a. Aktivitas Registrasi

Query di bawah merupakan aktivitas registrasi yang dilakukan oleh user

- Berikut query penjelasan di atas:

```
INSERT INTO users VALUES('US001','Abbey  
Heracles','abbeyh2804','abbeyhera@gmail.com','user','2021-10-03');
```

b. Aktivitas Pengecekan Validasi Email

Query di bawah merupakan aktivitas pengecekan validasi email, apabila user pernah mendaftarkan email tersebut ke dalam akun yang ada pada database.

- Berikut query penjelasan di atas:

```
SELECT * FROM users WHERE email like 'abbeyhera@gmail.com'
```

c. Aktivitas Login

Query di bawah merupakan aktivitas login yang dilakukan oleh user. Maka dari itu email dan password harus selaras dengan data yang ada di database.

- Berikut query penjelasan di atas:

```
SELECT * FROM users WHERE email like 'abbeyhera@gmail.com'  
and password like 'abbeyh2804'
```

2. Kelly Mae: Query di menu form pemesanan paket

a. Aktivitas Penampungan Daftar (*List*) Alamat Pengirim

Query di bawah merupakan aktivitas yang berguna untuk menampung daftar (*list*) alamat dari pengirim.

- Berikut query penjelasan di atas:

```
INSERT INTO daftar_alamat_pengirim VALUES ('AD001','Abbey Heracles','081287390467','Jalan Taman Patra Kuningan No. 4, Kuningan Timur, Kuningan, Jakarta Selatan, DKI Jakarta - Indonesia 12114','DKI Jakarta', 'Jakarta Selatan', 12114, -6.261493, 106.834570, 'Lokasi tepi jalan');
```

b. Aktivitas Penampungan Daftar (*List*) Alamat Penerima

Query di bawah merupakan aktivitas yang berguna untuk menampung daftar (*list*) alamat dari penerima.

- Berikut query penjelasan di atas:

```
INSERT INTO daftar_alamat_penerima VALUES ('AP001','Klemens Rosalina','08119733086','Jalan H. M. Kasim No. 9, Baji Pamai, Maros Baru, Maros, Sulawesi - Indonesia 90516','Sulawesi Selatan','Maros',90516,-5.005840,119.574310,'Seberang Museum Maros');
```

c. Aktivitas Penampungan Data Transaksi

Query di bawah merupakan aktivitas untuk menampung data transaksi yang dilakukan, berisi kode transaksi, status transaksi, dan tanggal dari masing-masing *update* transaksi.

- Berikut query penjelasan di atas:

```
INSERT INTO transaksi_paket VALUES ('TR001','AD001','AP001','US001','Baju Y2K',2,1,1.783,0,1,'Delivered to Post', 2.689, 349.000,'Arrived','','2021-08-13','2021-08-13','2021-08-13','2021-08-14','2021-08-19','');
```

3. **Leony Hana Noah Zebua:** Query di menu *show transaction*

a. Aktivitas Tampilan Awal

Query di bawah merupakan aktivitas tampilan awal saat user ingin melihat transaction yang telah dibuat.

- Berikut query penjelasan di atas:

```
SELECT id_transaksi, nama_barang, status_paket FROM  
transaksi_paket WHERE id_transaksi = 'TR001';
```

b. Aktivitas Tampilan Transaksi Paket

Query di bawah merupakan aktivitas tampilan transaksi paket yang sudah dilakukan, alamat penerima, id penerima, dan kode transaksi.

- Berikut query penjelasan di atas:

```
SELECT a.id_transaksi, a.nama_barang, b.nama FROM  
transaksi_paket a JOIN daftar_alamat_penerima b ON a.id_penerima  
= b.id_alamatpenerima where id_transaksi = 'TR001';
```

4. **Reuben Ryan Peter:** Query di pembayaran

a. Aktivitas untuk Menjelaskan Pembayaran

Query di bawah merupakan aktivitas yang berfungsi untuk menjelaskan mengenai pembayaran yang sudah dilakukan, berisi id atau kode transaksi, nama barang, berat, jarak, dan harga barang.

- Berikut query penjelasan di atas:

```
SELECT id_transaksi, nama_barang, berat, jarak, harga FROM  
transaksi_paket WHERE id_transaksi = 'TR001';
```

5. **Fareza Ananda Putra:** Query di *update* status

a. Aktivitas Pembaruan (*Update*) Status Transaksi Paket

Query di bawah merupakan aktivitas yang berguna untuk memperbarui (*update*) status transaksi paket yang berisi id atau kode transaksi, dan status paket, beserta tanggal transit di gudang (*warehouse*).

- Berikut query penjelasan di atas :

```
UPDATE transaksi_paket SET status_paket = 'Ware House Transit',  
tanggal_warehousetransit = '2021-08-13' WHERE id_transaksi =  
'TR001';
```

Pembagian Tugas Subquery :

1. Irene Maria Joseph:

a. Aktivitas untuk Menampilkan Data Transaksi

Query di bawah merupakan aktivitas untuk menampilkan id transaksi, nama pengirim dan penerima, alamat lengkap pengirim dan penerima, nomor telepon pengirim dan penerima, nama barang, berat barang, jarak tempuh, harga barang, dan asuransi barang dari tabel transaksi paket yang dihubungkan dengan tabel daftar alamat pengirim pada id alamat pengirim, dan tabel daftar alamat penerima pada id alamat penerima. Dalam hal ini, id transaksi diperoleh dari id transaksi dari tabel transaksi paket dimana id transaksi merupakan ‘TR001’.

- Berikut query penjelasan di atas :

```
SELECT tp.id_transaksi, ap.nama, ap.alamat_lengkap, ap.telpon,
ad.nama, ad.alamat_lengkap, ad.telpon, tp.nama_barang, tp.berat,
tp.jarak, tp.harga, tp.asuransibarang FROM transaksi_paket tp INNER
JOIN daftar_alamat_penerima ap ON tp.id_penerima =
ap.id_alamatpenerima INNER JOIN daftar_alamat_pengirim ad ON
tp.id_pengirim = ad.id_alamatpengirim WHERE(id_transaksi) =
(SELECT id_transaksi FROM transaksi_paket tp WHERE
id_transaksi = 'TR001');
```

b. Aktivitas Update Penghitungan Harga Barang

Query di bawah merupakan aktivitas untuk menampilkan harga yang ingin dibayarkan oleh pembayar dengan penghitungan yang disediakan dalam query. Query yang diperoleh dari *update* tabel transaksi paket diperoleh dari kolom harga untuk dispesifikasi di mana harga merupakan hasil dari penjumlahan harga dikalikan pajak, yaitu 0.1. Pada akhirnya, aktivitas ini akan menampilkan kolom id transaksi, id *user*, nama pengirim, nama barang, serta harga barang dari tabel transaksi paket yang digabung dengan tabel users pada id *user*. Dengan demikian, diperoleh harga yang telah *diupdate* dari tabel transaksi paket dimana id transaksinya adalah ‘TR003’.

- Berikut query penjelasan diatas :

```
UPDATE transaksi_paket SET harga = harga + harga * 0.1 SELECT
id_transaksi, id_user, u.name, nama_barang, harga FROM
```

```
transaksi_paket tp JOIN users u ON tp.id_user = u.id WHERE harga  
IN(SELECT harga FROM transaksi_paket WHERE id_transaksi =  
'TR003');
```

c. Aktivitas untuk Menampilkan Asuransi Barang

Query di bawah merupakan aktivitas untuk menampilkan apabila barang yang dikirim telah disertai dengan asuransi atau tidak. Query ini menampilkan id transaksi, asuransi barang, nama *user* yang diperoleh dari tabel transaksi paket yang digabung dengan tabel *users* pada id *user* dimana telah ditampilkan asuransi barang dari tabel transaksi paket dengan kondisi asuransi barang tersebut bernilai ‘1’, dengan kata lain barang tersebut memiliki asuransi..

- Berikut query penjelasan di atas :

```
SELECT id_transaksi, asuransibarang, u.name 'Nama User' FROM  
transaksi_paket p JOIN users u ON p.id_user = u.id WHERE  
asuransibarang IN (SELECT asuransibarang FROM transaksi_paket  
WHERE asuransibarang IN ('1'));
```

2. Kelly Mae:

a. Aktivitas Menampilkan Data User

Query di bawah merupakan aktivitas yang menampilkan id transaksi, nama, email, nomor telepon, id alamat penerima, nama penerima, dan nomor telepon penerima yang diperoleh dari tabel *user* yang digabung dengan tabel daftar alamat pengirim pada id alamat pengirim, serta tabel alamat penerima pada id alamat penerima. Dalam hal ini, hasil query yang akan ditampilkan berupa data *user* dimana id transaksi merupakan ‘TR001’.

- Berikut query penjelasan di atas :

```
SELECT id_transaksi, id_user, u.name, u.email, ad.telpon,  
ap.id_alamatpenerima, ap.nama, ap.telpon FROM transaksi_paket tp  
INNER JOIN users u ON tp.id_user = u.id INNER JOIN  
daftar_alamat_pengirim ad ON tp.id_pengirim =  
ad.id_alamatpengirim INNER JOIN daftar_alamat_penerima ap ON  
tp.id_penerima = ap.id_alamatpenerima WHERE id_transaksi =  
(SELECT id_transaksi FROM transaksi_paket WHERE id_transaksi
```

```
= 'TR001');
```

b. Aktivitas Menampilkan Data Transaksi yang Berhasil Dilakukan

Query di bawah merupakan aktivitas untuk menampilkan data transaksi yang berhasil dilakukan yang menampilkan id transaksi, id *user*, nama pengirim, nama barang, serta harga barang yang disertai dengan karakter ‘Rp’ sebelum harga tersebut dari tabel transaksi paket yang digabung dengan tabel *users* pada id *user* dan daftar alamat pengirim pada id pengirim. Dalam hal ini, query ini akan menampilkan data transaksi dengan status barang yang tidak bersifat ‘Failed’, dengan kata lain status barang tersebut sudah berhasil sampai pada tempat penerima..

- Berikut query penjelasan di atas :

```
SELECT id_transaksi, u.id, ad.nama, nama_barang, 'Rp. ' +
CAST(harga AS VARCHAR(20)) 'Harga Barang' FROM
transaksi_paket tp JOIN users u ON id_user = u.id JOIN
daftar_alamat_pengirim ad ON tp.id_pengirim =
ad.id_alamatpengirim WHERE NOT EXISTS(SELECT
tp.id_transaksi WHERE tp.status_paket = 'Failed');
```

c. Aktivitas Menampilkan User Terlama

Query di bawah merupakan aktivitas untuk menampilkan id, nama, *user join date* (tanggal bergabung) yang ditampilkan dengan format mm-dd-yyyy, dari tabel *users*, serta jumlah transaksi yang didapatkan dari banyaknya pemesanan yang dilakukan oleh pengirim dan ditambahkan dengan kata ‘transaksi’ di bagian akhirnya, dengan kondisi dimana user tersebut merupakan *user* terlama atau sudah lama bergabung.

- Berikut query penjelasan di atas :

```
SELECT id, name, CONVERT(VARCHAR(11), join_date, 0) 'User
Join Date', CAST((SELECT COUNT(id_transaksi) FROM
transaksi_paket tp WHERE tp.id_user = u.id) AS VARCHAR) + 'Transaksi'
'Jumlah Transaksi' FROM users u WHERE
YEAR(join_date) = (SELECT MIN(YEAR(join_date)) FROM users
u);
```

3. Leony Hana Noah Zebua:

a. Aktivitas Menampilkan Jumlah Transaksi

Query ini menampilkan id pengirim, nama pengirim, serta inisialnya yang didapatkan dari 2 karakter pertama nama pengirim, serta jumlah transaksi yang terdiri dari banyaknya pemesanan yang dilakukan oleh pengirim dengan menambahkan kata ‘transaksi’ di bagian akhirnya.

- Berikut query penjelasan di atas :

```
SELECT id_alamatpengirim 'ID Pengirim', ad.nama 'Nama Pengirim',
       UPPER(LEFT(ad.nama,      2))      'Inisial',      CAST((SELECT
       COUNT(id_transaksi)   FROM   transaksi_paket tp WHERE
       tp.id_pengirim = ad.id_alamatpengirim) AS VARCHAR) + 'Transaksi'
       'Jumlah Transaksi' FROM daftar_alamat_pengirim ad;
```

b. Aktivitas Menghapus ID User

Query di bawah merupakan aktivitas tampilan menghapus id *user* dari tabel users dimana id dari tabel transaksi paket adalah 'US004'. Aktivitas ini berfungsi untuk menghapus suatu data yang diperlukan untuk dihapus. Oleh karena itu, digunakan query “DELETE” untuk menghapus secara spesifik data dengan id ‘US004’ dengan menggunakan query ‘WHERE’ dari tabel transaksi paket, kemudian ditampilkan kembali seluruh data dari tabel users. Data yang dihapus tergantung dari data secara terkhusus apa yang ingin dihapus dengan menggunakan query ‘WHERE’ tersebut.

- Berikut query penjelasan di atas:

```
DELETE  FROM users WHERE id IN(SELECT id FROM
transaksi_paket WHERE id = 'US004');
SELECT * FROM users;
```

4. Reuben Ryan Peter:

a. Aktivitas Menampilkan Harga Barang di Atas 800

Query di bawah merupakan aktivitas untuk menampilkan id transaksi, id *user*, nama *user*, nama barang, jumlah barang (kuantitas), unit paket, berat barang dan harga barang yang diperoleh dari tabel transaksi paket. Dalam hal ini, tabel transaksi paket tersebut telah digabung dengan tabel *users* pada id

user, serta dimana id transaksi diperoleh dari id transaksi dari transaksi paket dimana harga barang di atas 800.

- Berikut query penjelasan di atas :

```
SELECT id_transaksi, u.id, u.name, nama_barang, kuantitas, unit_paket, berat, harga FROM transaksi_paket tp JOIN users u ON id_user = u.id JOIN daftar_alamat_pengirim ad ON tp.id_pengirim = ad.id_alamatpengirim WHERE id_transaksi IN(SELECT id_transaksi FROM transaksi_paket tp WHERE harga > 800);
```

5. **Fareza Ananda Putra:**

a. Aktivitas Menampilkan Tanggal Transaksi

Query di bawah merupakan query untuk menampilkan id transaksi, id *user*, nama *user*, serta tanggal transaksi berupa tanggal *pickup*, tanggal *deliveredtopost* (pengiriman menuju pos), tanggal paket sampai di warehouse (transit), tanggal *acceptedbykurir* (paket diterima oleh kurir), tanggal *ongoing* (tanggal berlangsungnya proses pengiriman), tanggal *arrived* (sampai di alamat penerima yang diterima), serta tanggal *failed* (tanggal paket gagal dikirimkan) dari tabel transaksi paket. Dalam hal ini, tabel transaksi paket digabung dengan tabel *users* pada id *user*, serta tabel daftar alamat pengirim pada id alamat pengirim. Dalam hal ini, hasil query yang akan ditampilkan berupa data tanggal transaksi dengan suatu kondisi dimana id transaksi merupakan ‘TR001’.

- Berikut query penjelasan di atas :

```
SELECT id_transaksi, u.id, ad.nama, tanggal_pickup, tanggal_deliveredtopost,tanggal_warehousetransit, tanggal_acceptedbykurir, tanggal_ongoing, tanggal_arrived, tanggal_failed FROM transaksi_paket tp JOIN users u ON tp.id_user = u.id JOIN daftar_alamat_pengirim ad ON tp.id_pengirim = ad.id_alamatpengirim WHERE(id_transaksi) = (SELECT id_transaksi FROM transaksi_paket tp WHERE id_transaksi = 'TR001');
```

LAMPIRAN

Resi