LAMPIRAN

Lampiran 1 Transkrip Wawancara

Berikut terlampir transkrip dari wawancara dengan 4 narasumber yang memilki peranan yang berbeda terhadap pengolahan sampah. yaitu staff di Dinas Lingkungan Hidup DKI Jakarta; Fona fetria, Ahli dan pendidik dibidang lingkungan hidup; Dicky Surya, staff instansi kebersihan di Universitas Bakrie; Indra, serta mahasiswa Universitas Bakrie yang merupakan konsumen dari produk yang menghasilkan sampah anorganik; Rizky Novriyedi.

Keterangan:

Wawancara dilakukan pada April 2017 dengan staff di Dinas Lingkungan Hidup DKI Jakarta; Fona fetria, bertempat di Kantor Dinas Lingkungan Hidup DKI Jakarta.

P: Peneliti

F: Fona Fetria (Staff di Dinas Lingkungan Hidup)

P: Siang Bu, Saya Ridho Mahasiswa Universitas Bakrie. Saya ingin melakukan wawancara mengenai pengolahan sampah di DKI Jakarta.

F: Siang Ridho, Saya Fona Fetria, silahkan.

P: Saya ingin bertanya mengenai data sampah di DKI Jakarta Bu. Kira – kira berapa jumlah sampah yang dihasilkan oleh DKI Jakarta setiap harinya ya bu?

F: Setiap harinya DKI Jakarta menghasilkan sampah sebanyak kurang lebih 7000 Ton.

P: Apa usaha pemerintah untuk mengurangi sampah tersebut bu?

F: Pemerintah DKI Jakarta telah berusaha untuk mengurangi jumlah sampah tersebut dengan mengeluarkan Peraturan Daerah No 3 Tahun 2013 tentang Pengelolaan Sampah pada Pasal 12 Ayat 1 yang mewajibkan setiap masyarakat untuk memilah sampah rumah tangga sebelum diangkut ke TPS, serta Instruksi Gubernur No. 157 Tahun 2016 tentang Pembinaan dan Pengembangan Bank Sampah yang bertujuan untuk mengoordinasikan program pembinaan dan pengembangan lokasi bank sampah ditiap Rukun Warga atau RW. Namun kurangnya kesadaran masyarakat menghambat proses pengolahan sampah di DKI Jakarta.

P: Apakah pemerintah DKI Jakarta menggunakan pendekatan IT untuk mengatasi permasalahan Sampah?

F: Pemerintah DKI Jakarta membuat sebuah aplikasi berbasis *Android* yang diperuntukan bagi pengelola bank sampah di wilayah DKI Jakarta. Namun, sistem informasi ini masih tidak bersifat terpusat dan tidak menjangkau individu masyarakat.

P: Baik Bu, terima kasih atas waktunya.

Interviewer,	Responden,
Ridho Gilang Fiesta	Fona Fetria

Keterangan:

Wawancara dilakukan pada April 2017 dengan Ahli dan pendidik dibidang lingkungan hidup ; Dicky Surya, bertempat di Universitas Bakrie.

P: Peneliti

D: Dicky Surya (Ahli dan Pendidik di Bidang Lingkungan Hidup)

P: Sore Pak, Saya Ridho mahasiswa Informatika Universitas Bakrie ingin melakukan wawancara mengenai pengolahan sampah.

F: Baik, silahkan mulai wawancaranya.

P: Saya ingin bertanya mengenai pengolahan sampah saat ini Pak, menurut bapak apakah sudah berjalan dengan efisien dan baik?

F: Menurut saya pengolahan sampah saat ini belum berjalan dengan baik dan efisien. Hal ini disebabkan oleh banyaknya masyarakat yang hanya menganggap sampah sebagai produk sisa. Banyak masyarakat tidak tahu bahwa sampah yang mereka buang memiliki value tertentu.

P: Menurut bapak apakah bank sampah yang ada saat ini dapat mengurangi jumlah sampah yang ada?

D: Menurut saya kebanyakan bank sampah yang ada tidak dapat mengurangi jumlah sampah yang ada, hal ini disebabkan oleh masih banyak bank sampah tidak memiliki sistem yang efisien serta produk dari bank sampah tersebut tidak dapat memberikan keuntungan bagi nasabah bank sampah.

P: Bagaiman dengan usaha pemerintah dalam mengatasi permasalahan sampah pak?

D: Pemerintah juga melakukan usaha untuk mengurangi jumlah sampah dengan membuat infrastruktur untuk pengolahan sampah, seperti Pembangkit Listrik Tenaga Sampah, namun proses perancangan dan pembangunan infrastruktur tersebut membutuhkan waktu yang lama dan dana yang besar sehingga belum dapat dapat dirasakan oleh masyarakat.

P: Bagaimana pendapat bapak apabila ada sistem informasi berbasis web untuk mengatasi sampah dengan cara memberikan value point terhadap sampah?

D: Menurut saya ini adalah ide yang menarik, karena saya belum mengetahui sistem informasi yang digunakan untuk mengatasi permasalahan sampah. Dan penggunaan point dapat meningkatkan minat pengguna dalam ikut serta mengatasi sampah.

P: Baik pak. Terima kasih atas waktunya.

Interviewer,	Responden,
Ridho Gilang Fiesta	Dicky Surya

Keterangan:

Wawancara dilakukan pada Maret 2017 dengan Staff Biro Bagian Umum Universitas Bakrie; Indra Lesmana, bertempat di Universitas Bakrie.

P: Peneliti

I: Indra Lesmana(Staff Biro Bagian Umum Universtias Bakrie)

P: Siang Mas, Saya Ridho mahasiswa Informatika ingin melakukan wawancara mengenai pengolahan sampah di Universitas Bakrie.

I: Baik mas, silahkan mulai wawancaranya.

P: Saya ingin bertanya mengenai pengolahan sampah di Universitas Bakrie. Kira – kira setiap harinya universitas bakrie menghasilkan berapa banyak sampah?

I: Setiap harinya Universitas Bakrie menghasilkan kurang lebih 100kg sampah.

P: dari jumlah tersebut kira-kira berapa persen jumlah sampah plastik dan anorganik?

I: Kurang lebih 50% dari sampah itu sampah botol plastik mas.

P: Apakah Universitas bakrie melakukan pemilahan sampah mas?

I: Sampahnya hanya dipilah pada beberapa tong sampah saja mas. Jumlah tong sampah kita terbatas. Dan hasil pemilahan tersebut juga disatukan kembali dan dibuang pada TPST yang sama.

P: apakah Universitas Bakrie melakukan proses daur ulang atau tidak mas?

I: Universitas bakrie tidak melakukan proses daur ulang mas.

P: Apabila ada sistem informasi yang memberikan value terhadap sampah anorganik apakah mas Indra mau ikut dan menggunakan sistem informasi tersebut?

I: saya tertarik untuk menggunakan sistem informasi tersebut mas.

P: baik mas terima kasih.

Interviewer, Responden,

Ridho Gilang Fiesta Indra Lesmana

Keterangan:

Wawancara dilakukan pada Maret 2017 dengan Mahasiswa Universitas Bakrie serta salah satu konsumen dari produk yang menghasilkan sampah anorganik, bertempat di Universitas Bakrie.

P: Peneliti

R: Rizky Novriyedi (mahasiswa Universitas Bakrie)

P: Siang Rizky, saya ingin mewawancarai anda mengenai sampah anorganik dari produk yang anda konsumsi.

R: Baik Ridho, silahkan mulai wawancaranya.

P: Apakah anda menghasilkan sampah anorganik setiap harinya?

R: Iya, saya menghasilkan 2-3 botol air mineral ukuran 1.5 liter setiap harinya.

P: Apakah anda pernah mendaur ulang sampah tersebut?

R: Saya tidak pernah melakukan proses daur ulang dan hanyak membuang sampah tersebut ke tempat pengumpulan sampah sementara.

P: Apakah anda tertarik untuk melakukan proses daur ulang?

R: Jika memungkinkan. Saya ingin melakukan proses daur ulang. Hanya saja saya tidak mengetahui informasi tentang tempat pengolahan kembali sampah anorganik.

P: Apabila ada sistem informasi yang memberikan value terhadap sampah anorganik. Apakah anda berniat menggunakannya?

R: Saya berminat, karena saya bisa memanfaatkan sampah yang menurut saya sudah tidak berharga.

P: Baik Rizky Terima kasih atas waktunya.

Interviewer, Responden,

Lampiran 2 Document Software Requirement Specification(SRS)

Software Requirement Specification

MULUNG: Aplikasi Pengelolaan Sampah Anorganik yang Dibangun Menggunakan *Progressive Web App (Front-End)* dan Diimplementasikan Menggunakan *Microservices)*

1. Introduction

Tahap introduction berfungsi untuk memberikan penjelasan ruang lingkup dan tujuan dari aplikasi Mulung beserta daftar istilah dan defenisi dari yang akan digunakan dalam dokumen SRS.

1.1 *Purpose*

Dokumen SRS ini menyajikan secara rinci mengenai aplikasi Mulung, meliputi kebutuhan dan fitur yang akan dikembangkan dalam aplikasi informasi Mulung. Dokumen SRS ini dijadikan sebagai referensi oleh *developer* dalam membangun dan mengembangkan aplikasi Mulung.

1.2 Scope of Project

Penelitian ini akan menghasilkan sebuah produk berupa aplikasi mulung yang berfungsi untuk mengatasi isu sampah anorganik, meliputi penukaran sampah anorganik, pengumpulan point secara otomatis oleh sistem, serta penukaran point dengan produk tertentu.

1.3 Glossary

Berikut daftar istilah beserta definisinya yang digunakan dalam dokumen SRS ini:

Tabel 1 Glossary

Term	***Definition***
PWA	*Progressive Web App* atau PWA adalah sebuah *framewor*k yang bertujuan untuk memaksimalkan *cache management* dari suatu *web-based application*.

Tabel 1 Glossary (lanjutan)

Term	***Definition***
Microservices Architecture	*Microservices architecture* adalah gaya arsitektur sistem yang menstrukturkan suatu sistem informasi sebagai kumpulan dari beberapa *services* yang tidak saling mengikat
Database	*database* adalah kumpulan data yang saling berhubungan dan berelasi
hardware	*Hardware* adalah komponen perangkat keras yang dapat disentuh dan digunakan oleh *users.*
software	*Software* adalah komponen perangkat lunak yang tidak terlihat secara fisik namun terdapat dalam sebuah sistem serta dapat dimanfaatkan oleh *users* untuk keperluan tertentu.
Services worker	*Service worker* adalah sebuah *script* yang berjalan di belakang *browser* pengguna dan tidak membutuhkan sebuah halaman ataupun interaksi dari pengguna untuk menjalankan fungsinya. *Service worker* akan tetap berjalan, walaupun halaman *web* tersebut tidak dibuka
Software testing	*Software Testing* adalah proses untuk mengaskses fungsionalitas dan kebeneran suatu perangkat lunak dalam melakukan suatu proses melalui analisis
Unified Model Language	UML adalah Notasi *grapichal* standar yang digunakan untuk mendeskripsikan *software* analisis dan *design*. Sehingga dapat mengurangi kesalah pahaman dan ambiguitas dari suatu model yang dibuat

Tabel 1 Glossary (lanjutan)

Term	***Definition***
Node.js	Node.js merupakan sebuah *framework Javascript* yang *dibuilt* pada *Chrome's v8 JavaScript engine*. Node.js menggunakan *event-driven, non-blocking I/O model* yang memaksimalkan efisiensi serta ukuran dari suatu sistem informasi.
User	Pengguna yang berinteraksi dengan sebuah sistem informasi.
Mulung	Mulung adalah nama aplikasi yang dibangun pada penelitian ini.

1.4 References

IEEE Std 830-1998, IEEE *Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

1.5 Overview of document

Bab pertama menjelaskan tentang ruang lingkup dan tujuan dari aplikasi yang akan dibangun, Bab kedua akan menjelaskan tentang keseluruan dan gambaran dari aplikasi Mulung. Sedangkan bab ketiga akan menjelaskan spesifikasi kebutuhan yang ditulis untuk *developer* serta detail teknis fungsional dari Mulung.

2. Overal Description

Aplikasi Mulung adalah aplikasi yang berfungsi untuk mengelola dan mengatasi isu sampah anorganik dengan cara memberikan value berupa point terhadap sampah anorganik yang dihasilkan oleh masyarakat. Sehingga dapat meningkatkan minat dan keinginan masyarakat untuk mengatasi permasalahan sampah.

2.1 *Product perspective*

Aplikasi Mulung merupakan sebuah aplikasi berbasis *web* yang memanfaatkan progressive web app pada sisi front-end serta microservices pada sisi *back-end*. Sehingga dapat meningkatkan *scalability* dari aplikasi mulung serta dapat meningkatkan *user engagement* dari pengguna aplikasi mulung. Aplikasi mulung dirancang untuk mengatasi permasalah pengelolaaan sampah anorganik dengan cara memberi value kembali terhadap sampah anorganik.

2.2 *Product functions*

Pada aplikasi mulung terdapat fungsi – fungsi sebagai berikut:

Tabel 2 Product Functions

No	Fungsi
1.	Fungsi Register
2.	Fungsi Login menggunakan Local Strategy
3.	Fungsi authorize menggunakan Google Strategy
4.	Fungsi authorize menggunakan Facebook Strategy
5.	Fungsi authorize menggunakan Twitter Strategy
6.	Fungsi tambahkan password dan alamat untuk pengguna baru dengan *social login*
7.	Fungsi lihat halaman home
8.	Fungsi lihat jumlah point

Tabel 2 Product Functions(lanjutan)

No	Fungsi
9.	Fungsi lihat halaman profile
10.	Fungsi edit profile
11.	Fungsi ubah password
12.	Fungsi ubah alamat
12.	Fungsi tukar point
13.	Fungsi tambahkan transaksi
15.	Fungsi logout

2.3 *Product perspective*

User atau pengguna dari aplikasi Mulung adalah masyarakat pribadi yang ingin ikut mengatasi permasalahan sampah anorganik serta ingin mendapatkan keuntungan pribadi dengan cara menukarkan sampah anorganik menjadi point. Serta pengelola apartemen dan indekos di Kecamatan Setiabudi.

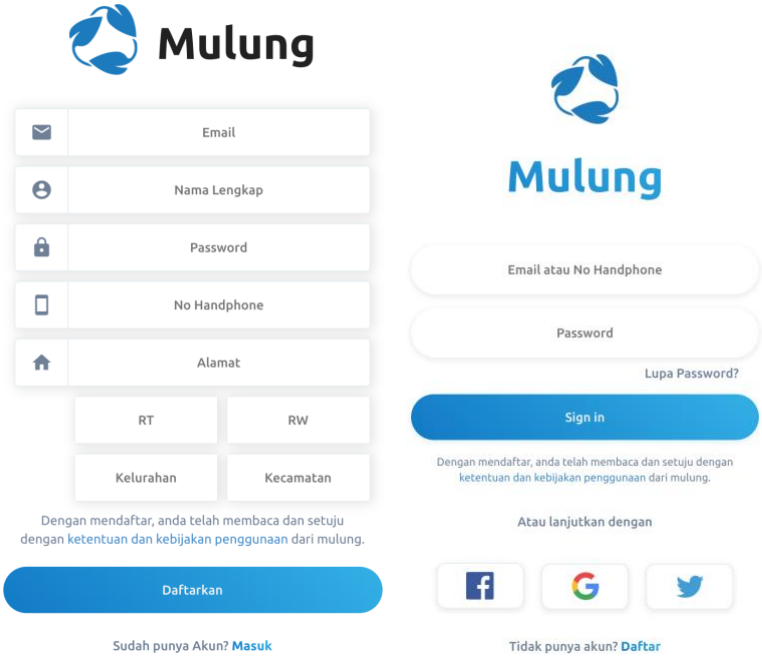
3. *Overal Description*

3.1 *External Interface Requirement*

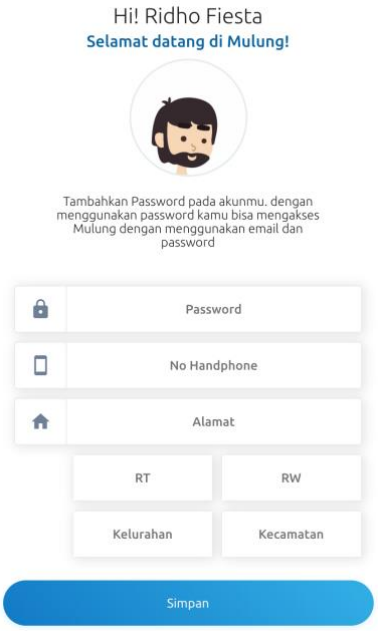
3.1.1 *User Interface*

Aplikasi mulung dirancang menggunakan *software* SketchApp serta proses *prototyping* dengan menggunakan aplikasi berbasis *web* MarvelApp. Sedangkan untuk pengembangannya *User Interface* aplikasi Mulung dikembangkan menggunakan HTML5 dan CSS3 serta menggunakan Javascript untuk keperluan tertentu.

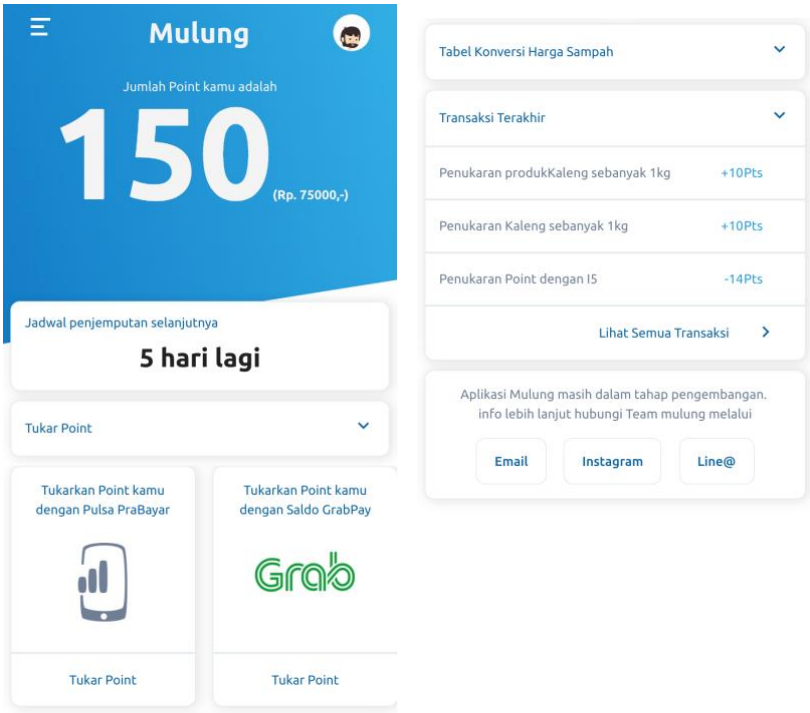
Berikut merupakan tampilan *user interface* dari aplikasi Mulung.



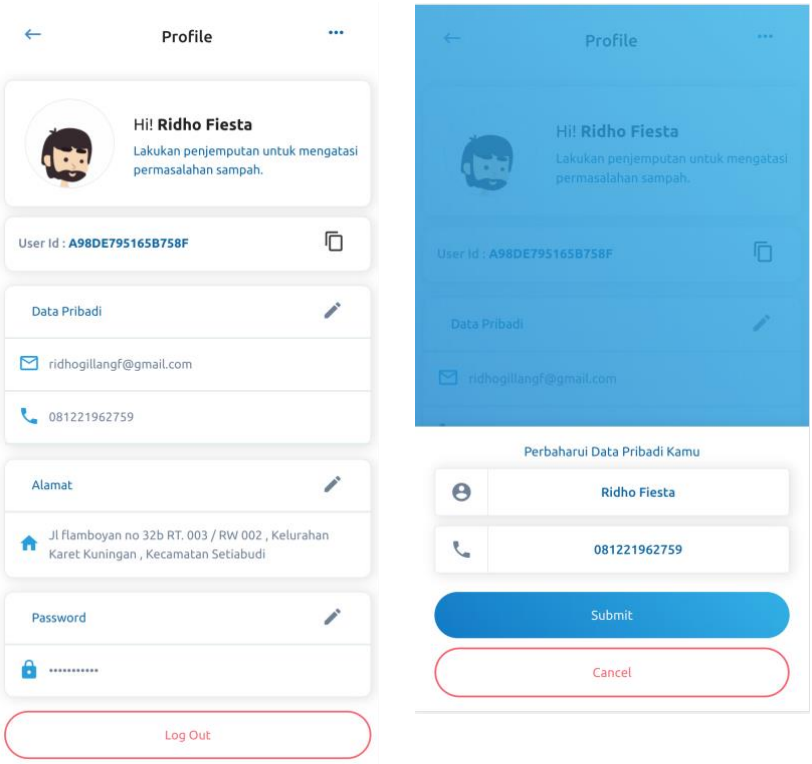
Gambar 1. Tampilan user interface register *(Kiri) dan* log in *(kanan).*



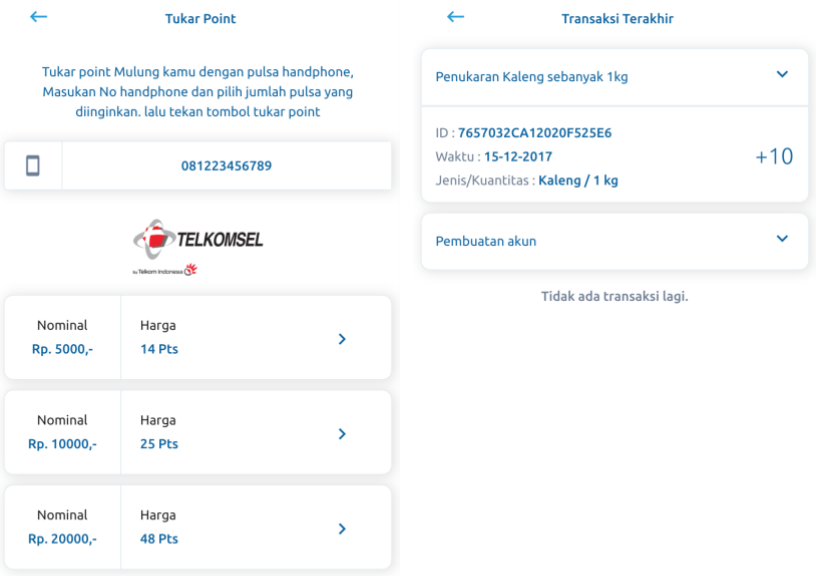
Gambar *2.* Tampilan *user interface Welcome*



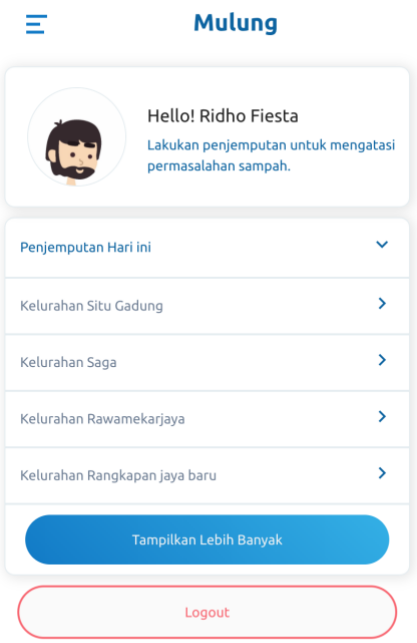
Gambar *3.* Tampilan *user interface home*



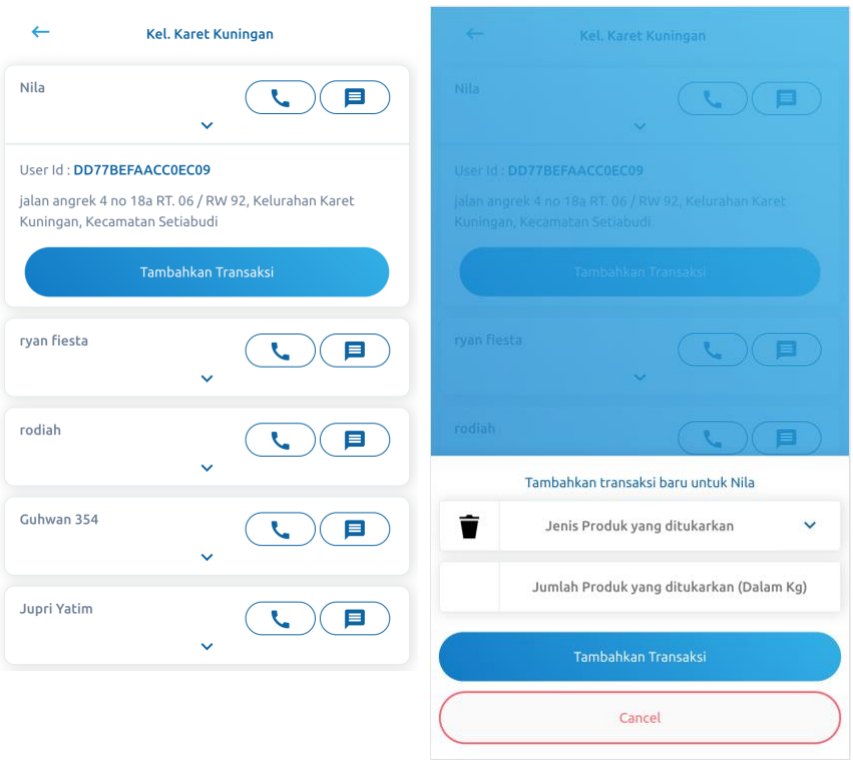
Gambar *4.* Tampilan *user interface profile dan edit profile*



Gambar *5.* Tampilan *user interface tukar point dan transaksi*



Gambar *6.* Tampilan *user interface admin*



Gambar *7.* Tampilan *user interface add transaction*

Gambar 1 sampai 7 merupakan user interface dari aplikasi mulung yang dikembangkan dengan HTML5 dan CSS3 serta Javascript.

3.1.2 *Hardware Interface*

Berikut adalah informasi *harfware yang digunakna dalam pengembangan aplikasi Mulung:*

No	**Spesifikasi**	**Keterangan**
1.	*Device*	Macbook Pro Retina Display MGX72
	Operating System	macOS Sierra 10.12.5
	CPU	Intel® CoreTM i5 CPU @ 2.6GHz
	RAM	8.00 GB
No	**Spesifikasi**	**Keterangan**
2.	*Device*	Xiaomi Redmi Note 4
	Operating System	Android v.6.0 (Marshmallow)
	CPU	Octa-core 2.0 GHz Cortex-A53
	RAM	3.00 GB

3.1.3 *Software Interface*

Berikut informasi software yang digunakan dalam mengembangkan aplikasi Mulung

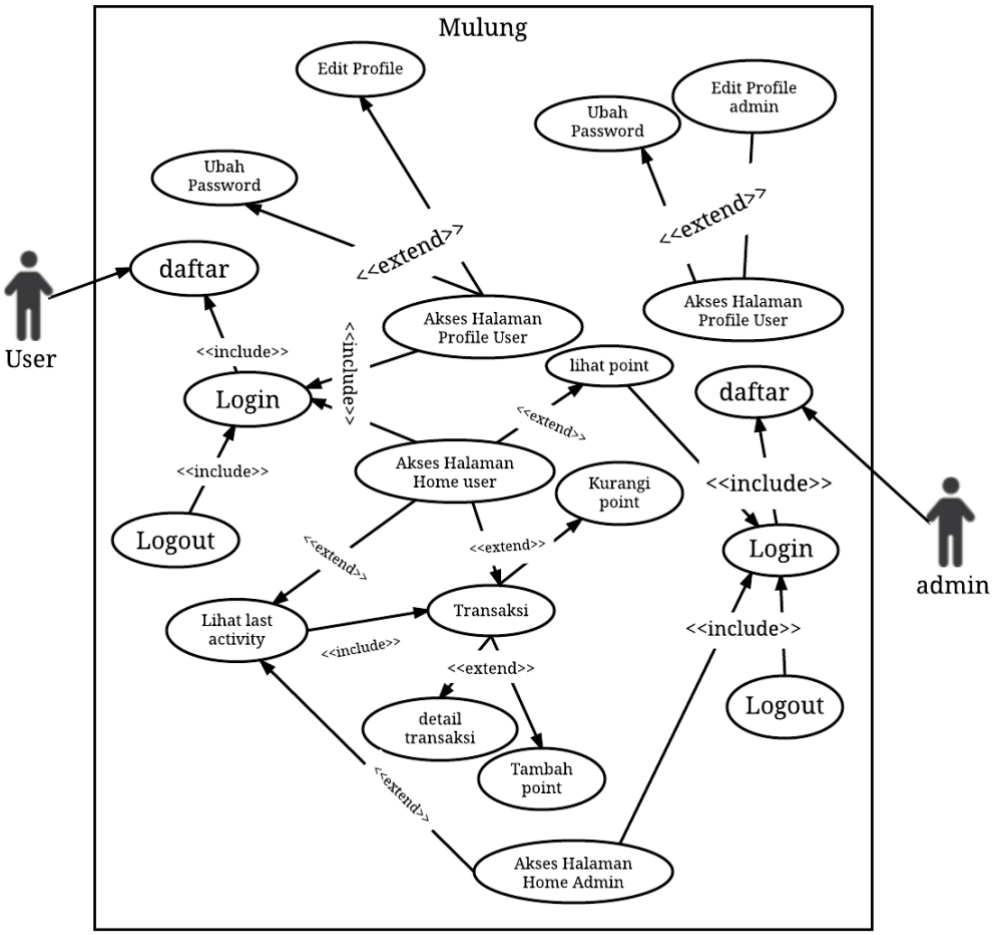
Tabel 3. **Software Interface**

No	Software	Version	Keterangan
1.	LucidChart	-	Digunakan untuk proses perancangan UML
2.	Microssoft Office	Pro 2016	Digunakan untuk membuat dokumentasi dari laporan tugas akhir.
3.	Adobe Illustrator CC	CC 2017 versi 21.0.0	Digunakan untuk membuat kebutuhan visual dari aplikasi mulung seperti logo dan icon
4.	SketchApp	4.0.0	Digunakan untuk merancang tampilan mockup dari sistem informai Mulung
5.	MarvelApp	-	Digunakan untuk merancang interaksi dan proses prototyping dari aplikasi Mulung.
6.	Sublime Text	Build 3126	Digunakan untuk pengkodean aplikasi mulung
7.	Google Chrome	Version 59.0.3071.115	Digunakan untuk menjalankan aplikasi Mulung.

3.2 *Functional requirement*

3.2.1 *Use Case Diagram*

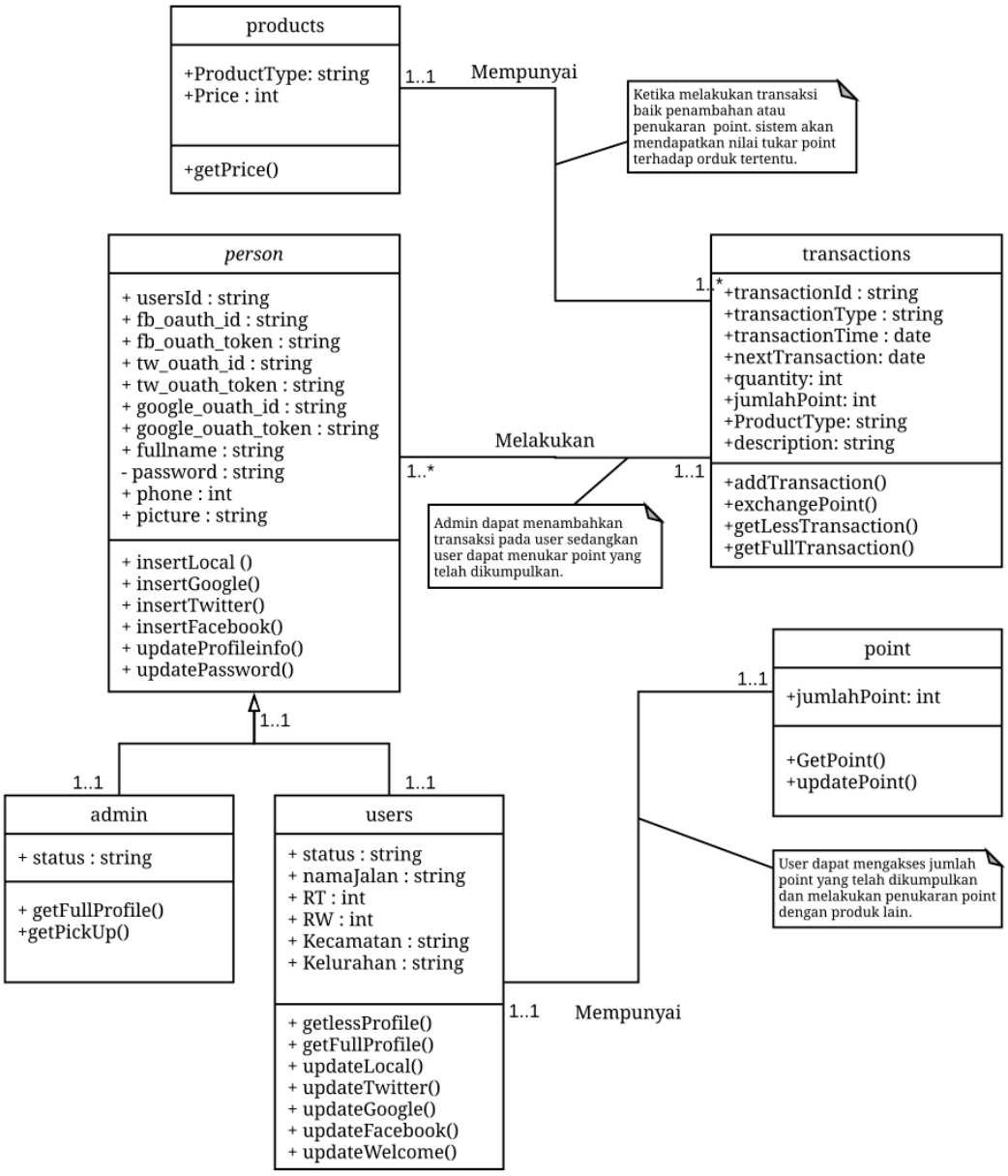
Use case diagram adalah *diagram* yang menjelaskan urutan dari suatu transaksi yang dilakukan oleh sistem yang menghasilkan suatu *output* yang nyata serta dapat memberikan *value* yang berbeda pada setiap *actor*. Berikut merupakan *Use Case diagram* dari aplikasi Mulung:



Gambar 8 Use Case dari aplikasi Mulung

3.2.2 *Class Diagram*

Class diagram adalah *diagram* yang berfungsi untuk menggambarkan *class* dan hubungan yang ada pada *class* tersebut.



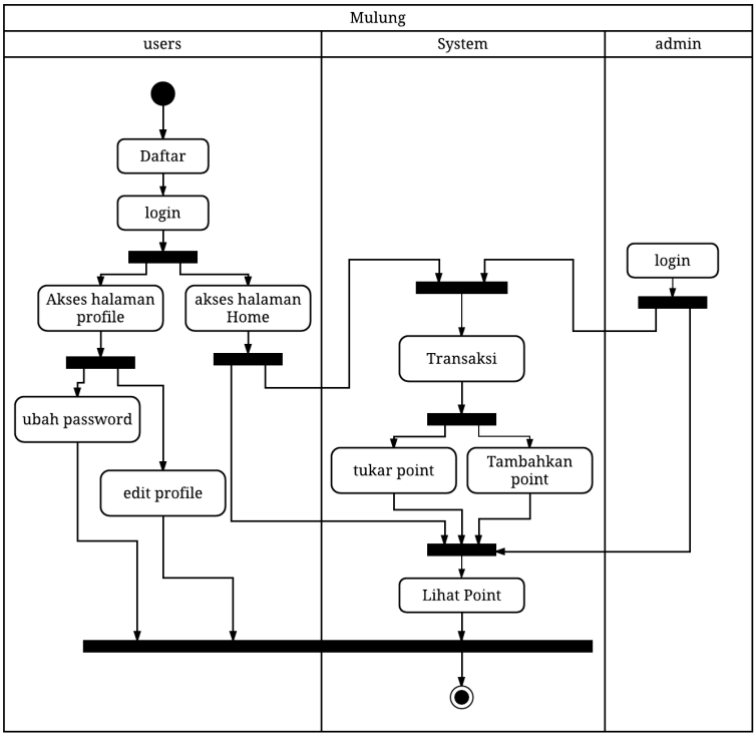
***Gambar 9 Class Diagram* Aplikasi Mulung**

3.2.3 *Sequence Diagram*

Sequence diagrams adalah *diagram* yang berfungsi untuk memvisualisasikan semua *scenario* yang mungkin terjadi pada *use case*.

3.2.4 *Activity Diagram*

Activity diagram adalah dapat mempermudah *developer* dalam fase design dari sebuah aplikasi.



Gambar 10 Activity dari aplikasi Mulung

3.3 *Non-functionality*

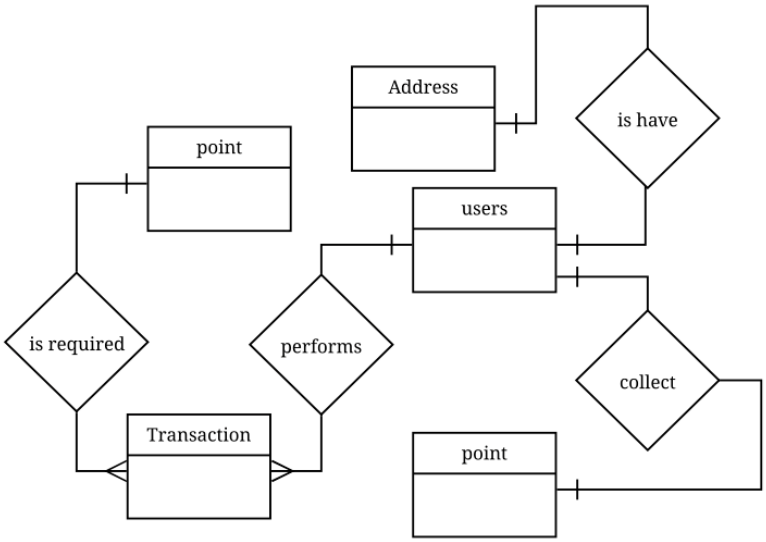
Kebutuhan *non-functional* dari aplikasi mulung adalah sebagai berikut :

1. *Aplikasi* mulung dirancang dengan sederhana dan *user friendly.* Informasi yang ada pada aplikasi mulung ditampilkan dengan lebih jelas dan mudah dimengerti, serta tidak membutuhkan banyak aksi dari *user,* sehingga dapat meningkatkan *user engagement.*
2. Aplikasi mulung digunakan oleh masyarakat luas yang berfungsi untuk memberikan *value* atau nilai kembali terhadap sampah anorganik.
3. Aplikasi mulung dapat digunakan oleh *user* menggunakan *mobile device* yang memiliki koneksi dengan jaringan Internet. Serta pemanfaatan *progressive web app* dalam perancangannya memungkinkan aplikasi mulung untuk diakses dalam keadaan *offline.*

3.4 *Other Non-Functional requiremenz*

3.3.1 *Conceptual Database Design*

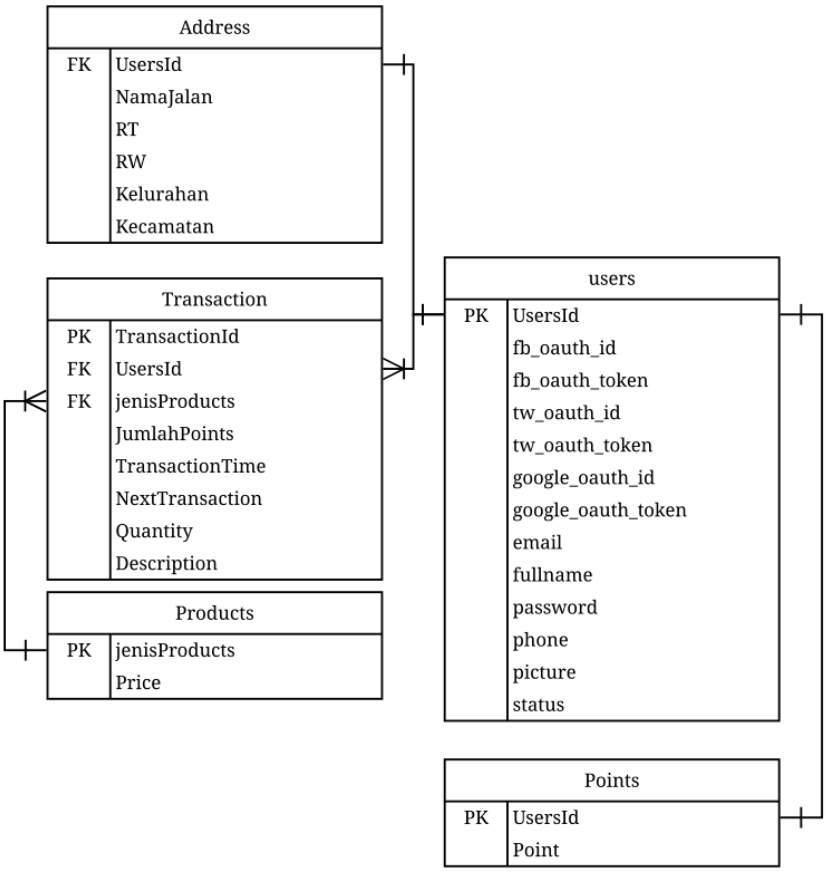
Conceptual database design dari aplikasi Mulung adalah sebagai berikut:



Gambar 11 Conceptual Database Design

3.3.2 *Logical Database Design*

Logical database design dari aplikasi Mulung adalah sebagai berikut:



Gambar 12 Logical Database Design

1) Users Entity

Tabel 4 Entity Users

Attribut	**Tipe**	**Deskripsi**	**Ket**
usersId	Varchar	Id dari pengguna	PK
fb_oauth_id	Varchar	Oauth Id pengguna dari facebook oauth.	-
fb_oauth_token	Varchar	Oauth Token pengguna dari facebook oauth.	-
tw_oauth_id	Varchar	Oauth Id pengguna dari twitter oauth.	-
tw_oauth_token	Varchar	Oauth Token pengguna dari twitter oauth.	-

Tabel 4 *Users Entity (Lanjutan)*

Attribut	Tipe	Deskripsi	Ket
email	Varchar	Email pengguna	-
fullname	Text	Nama lengkap pengguna	-
password	Varchar	Password pengguna	-
phone	Varchar	Nomor telpon pengguna	-
status	Varchar	Status dari Users	-

2) Points Entity

Tabel 5 Point Entity

Attribut	Tipe	Deskripsi	Ket
usersId	Varchar	Id dari pengguna	FK
Jumlahpoint	Int	Jumlah point dari pengguna	-

3) Transaction Entity

Tabel 6 Entity Transaction

Attribut	Tipe	Deskripsi	Ket
transactionId	Varchar	Id dari transaksi yang dilakukan.	PK
usersId	Varchar	Id dari pengguna	FK
ProductType	Varchar	Id dari product yang dijual atau pun dibeli.	FK
TransactionType	Enum(add, redeem)	Jenis dari transaksi yang dilakukan	-
JumlahPoint	Int	Jumlah point yang didapatkan pada transaksi	-

Tabel 6 ***Entity Transaction(Lanjutan)***

Attribut	**Tipe**	**Deskripsi**	**Ket**
TransactionTime	DateTime	Waktu transaksi yang dilakukan	-
NextTransaction	DateTime	Waktu jadwal transaksi selanjutnya	-
Quantity	Int	Jumlah produk yang dilakukan pada transaksi.	-
Description	LongText	Deskripsi dari transaksi yang dilakukan.	-

4) Address Entity

Tabel 7 Address Entity

Attribut	**Tipe**	**Deskripsi**	**Ket**
usersId	Varchar	Id dari pengguna	FK
NamaJalan	Varchar	Nama Jalan dari rumah pengguna	-
RT	Varchar	Nomor RT dari rumah pengguna.	-
RW	Varchar	Nomor RW dari rumah pengguna.	-
Kelurahan	Varchar	Nama Kelurahan dari rumah pengguna.	-
Kecamatan	Varchar	Nama Kecamatan dari rumah pengguna.	-

5) Product Entity

Tabel 8 Products Entity

Attribut	**Tipe**	**Deskripsi**	**Ket**
ProductType	Varchar	Jenis dari product yang diperjual belikan/	PK
ProductPrice	Varchar	harga dari product yang diperjual belikan.	-

3.3.3 *Availability*

Aplikasi mulung tersedia dan dapat digunakan selama pengguna terhubung dengan Internet dan server dari aplikasi Mulung dalam keadaan normal.

3.3.4 *Security*

Server aplikasi mulung menggunakan layanan dari google yaitu *google cloud platform. Password* yang digunakan oleh *user* akan disimpan dalam bentuk enkripsi dengan menggunakan *framework* bcrypt dengan menggunakna 10 rounds dalam proses hashingnya

Lampiran 3 White-Box Testing Aplikasi Mulung

Pengujian *White-box* dilakukan bertujuan untuk mengecek apakah aplikasi Mulung telah berjalan dengan benar atau tidak.

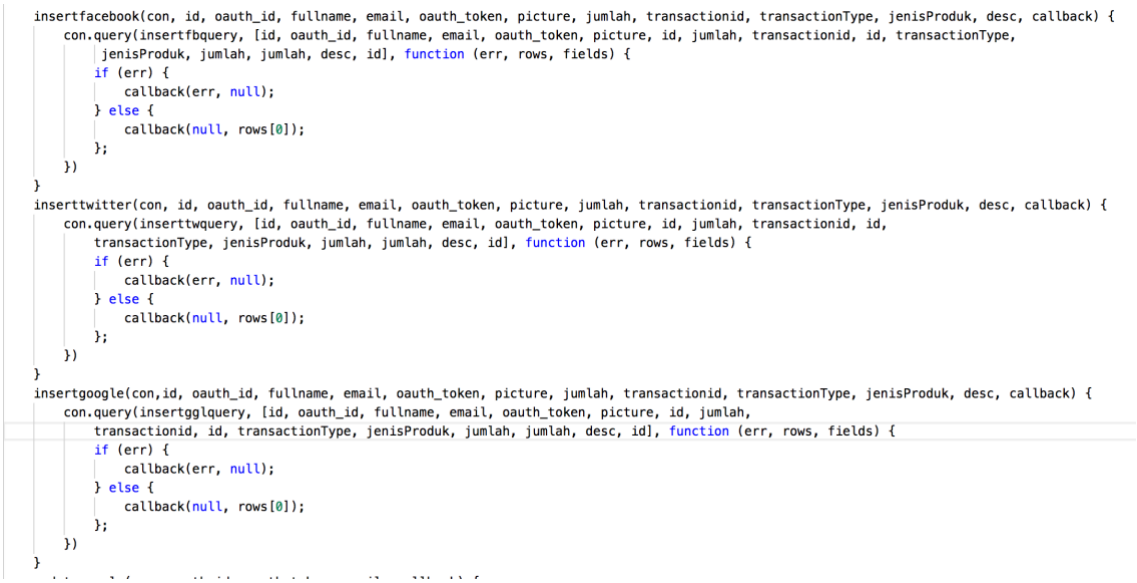
1) Class person.js

```
insertlocal(con, id, password, fullname, email, phone, Long, Lat, jumlah,
    transactionid, transactionType, jenisProduk, desc, NamaJalan,
    RT, RW, Kel, Kec, callback) {
    con.query(insertlocalquery, [id, password, fullname, email, phone, Long, Lat, id,
        jumlah, transactionid, id, transactionType, jenisProduk,
        jumlah, jumlah, desc, id, NamaJalan, RT, RW, Kel, Kec
    ], function (err, rows, field) {
        if (err) {
            callback(err, null)
        } else {
            callback(null, rows[0]);
        }
    })
}
```

Gambar 13 Function insertlocal pada class person

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
1.	insertLocal	Ridho@mail.com, Ridho Fiesta, Password, 08123456789, Jl Manggis No.1, 003, 004, Karet, Setiabudi Kondisi:User sudah ada pada database	Aplikasi Mulung menampilkan alert bahwa No handphone dan email yang didaftarkan sudah terdaftar dan redirect ke halaman login	Benar

Test Case	*Function*	*Input*	*Expected Output*	*Actual Output*
2.	insertLocal	Ridho@mail.com, Ridho Fiesta, Password, 08123456789, Jl Manggis No.1, 003, 004, Karet Kondisi:Data tidak lengkap	Aplikasi Mulung menampilkan alert yang menyatakan bahwa user belum mengisi data lengkap pada form register	Benar



```
insertfacebook(con, id, oauth_id, fullname, email, oauth_token, picture, jumlah, transactionid, transactionType, jenisProduk, desc, callback) {
    con.query(insertfbquery, [id, oauth_id, fullname, email, oauth_token, picture, id, jumlah, transactionid, id, transactionType,
        jenisProduk, jumlah, jumlah, desc, id], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        };
    })
}
inserttwitter(con, id, oauth_id, fullname, email, oauth_token, picture, jumlah, transactionid, transactionType, jenisProduk, desc, callback) {
    con.query(inserttwquery, [id, oauth_id, fullname, email, oauth_token, picture, id, jumlah, transactionid, id,
        transactionType, jenisProduk, jumlah, jumlah, desc, id], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        };
    })
}
insertgoogle(con,id, oauth_id, fullname, email, oauth_token, picture, jumlah, transactionid, transactionType, jenisProduk, desc, callback) {
    con.query(insertgglquery, [id, oauth_id, fullname, email, oauth_token, picture, id, jumlah,
        transactionid, id, transactionType, jenisProduk, jumlah, jumlah, desc, id], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        };
    })
}
```

Gambar 14 **Function insertfacebook, insertgoogle, inserttwitter *pada* class person**

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
3.	InsertGoogle	User menekan tombol login dengan Google *Kondisi: User tidak ada pada database*	Aplikasi Mulung akan menyimpan data pada Database dan redirect Ke halaman welcome	Benar
4.	InsertTwitter	User menekan tombol login dengan Twitter Kondisi: User tidak ada pada database	Aplikasi Mulung akan menyimpan data pada Database dan redirect Ke halaman welcome	Benar
5.	InsertFacebook	User menekan tombol login dengan Facebook Kondisi: User tidak ada pada database	Aplikasi Mulung akan menyimpan data pada Database dan redirect Ke halaman welcome	Benar

```
updategoogle(con, oauth_id, oauth_token, email, callback) {
    con.query(updategglquery, [oauth_id, oauth_token, email], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            console.log(rows)
            callback(null, rows[0]);
        }
    })
}
 updatefacebook(con,oauth_id, oauth_token, email, callback) {
    con.query(updatefbquery, [oauth_id, oauth_token, email], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        };
    })
}

 updatetwitter(con, oauth_id, oauth_token, email, callback) {
    con.query(updatetwquery, [oauth_id, oauth_token, email], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        };
    })
}
```

Gambar 15 **Function updatefacebook, updategoogle, updatetwitter *pada* class person**

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
6.	updateGoogle	User menekan tombol login dengan Google Kondisi: User telah melakukan register dengan menggunakan form register	Aplikasi Mulung akan mengupdate data tertentu meliputi ouathid dan oauthtoken Google dari pengguna.	Benar

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
7.	updateTwitter	User menekan tombol login dengan Twitter Kondisi: User telah melakukan register dengan menggunakan form register	Aplikasi Mulung akan mengupdate data tertentu meliputi ouathid dan oauthtoken twitter dari pengguna.	Benar
	updateFacebook	User menekan tombol login dengan Facebook Kondisi: User telah melakukan register dengan menggunakan form register	Aplikasi Mulung akan mengupdate data tertentu meliputi ouathid dan oauthtoken Facebook dari pengguna.	Benar

```
    updateprofileinfo(con, nama, phone, usersId, namaJalan, rt, rw, kel, kec, callback) {
        con.query(updateprofilequery, [nama, phone, usersId, namaJalan, rt, rw, kel, kec, usersId], function (err, rows, fields) {
            if (err) {
                callback(err, null);
            } else {
                callback(null, rows[0]);
            };
        })
    }
    updatepassword(con, newpassword, usersId, callback) {
        con.query(updatepwquery, [newpassword, usersId], function (err, rows, fields) {
            if (err) {
                callback(err, null);
            } else {
                callback(null, rows[0]);
            };
        })
    }

}
```

Gambar 16 **Function updateprofile dan update password *pada* class person**

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
8.	updateProfilePage	Mengisi data baru user yang terdapat pada form yang ada pada halaman profile	Aplikasi Mulung akan mengupdate data tertentu meliputi informasi yang diperbaiki	Benar
9.	updatePassword	Password lama, password baru *Kondisi:Password Beda*	Aplikasi Mulung akan di redirect ke ahlaman profile dan menampilkan alert bahwa penukaran password gagal karena password lama tidak sama	Benar
10.	updatePassword	Password lama, password baru *Kondisi:Password*	Redirect Kelahaman profile dan mengirimkan bahwa proses	Benar

		sama	penukan password berhasil	

```
getlessProfile(con, email, phone, callback) {
    con.query(selectemailorphonequery, [email, phone], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        };
    })

};
getlessProfileusersId(con, usersId, phone, callback) {
    con.query(selectusersIDorphonequery, [usersId, phone], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        };
    })

};
getFullProfile(con, usersId, callback) {
    con.query(selectAllquery, usersId, function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        };
    });
}
```

***Gambar 17* Function getlessProfile, getlessProfileusersId, getFullProfile *pada* class person**

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
11.	getLessProfile untuk proses login	Memasukkan email/no hanpdhone dan password Kondisi: email/no handphone tidak ada pada database	Redirect ulang kehalaman login dan mengirimkan alert	Benar

Test Case	*Function*	*Input*	*Expected Output*	*Actual Output*
12.	getLessProfile untuk proses login	Memasukkan email/no hanpdhone dan password Kondisi: email/no handphone ada pada database tetapi password salah.	Aplikasi Mulung akan meredirect ulang ke halaman login dan mengirimkan alert bahwa password yang dimassukan oleh user salah.	Benar
13.	getLessProfile untuk proses login	Memasukkan email/no hanpdhone dan password Kondisi: email/no handphone ada pada database dan password bennar	Aplikasi Mulung akan membuat session dan meredirect user ke halaman home serta memberikan alert bahwa proses login telah berhasil dilakukan	Benar
14.	getLessProfile untuk proses login	Memasukkan email/no hanpdhone dan password Kondisi: email/no handphone ada pada database tetapi password salah.	Aplikasi Mulung akan meredirect ulang ke halaman login dan mengirimkan alert bahwa password yang dimassukan oleh user salah.	Benar

Test Case	*Function*	*Input*	*Expected Output*	*Actual Output*
15.	getLessProfile untuk welcome	Mengakses halaman welcome	menampilkan data yang dapat diakses apabila user melakukan login menggunakan akun media social.	Benar
16.	getFullProfile untuk home	Mengakses halaman home	menampilkan data lengkap user. apabila user telah melakukan proses login.	Benar

2) Class users.js

```
class users extends person{

    updatewelcome(con, password, phone, long, lat, alamat, rt, rw, kel, kec, id, callback) {
       con.query(updatewelcomequery, [password, phone, long, lat, id, alamat, rt, rw, kel, kec, id], function (err, rows, fields) {
           if (err) {
               callback(err, null);
           } else {
               callback(null, rows[0]);
           };
       })
   }

}
    ridhogillang, 23 days ago • add new pickup filtering system
```

Gambar 18 *Function updatewelcome* pada *class users*

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
17.	updateWelcome	Password, 08123456789, Jl Manggis No 1, 003, 004, Karet, Setiabudi *Kondisi:Data lengkap*	Aplikasi Mulung akan mengupdate data user meliputi password, nomor handphone dan alamat.	Benar
18.	updateWelcome	Password, 08123456789, Jl Manggis No 1, 003, 0*04*, Karet, Kondisi:Data tidak lengkap	Redirect Ke halaman welcome dan mengeluarkan alert yang menyatakan bahwa data ynag dimasukkan tidak lengkap	Benar

3) Class admin.js

```
getpickupall(con, ArraySort, ArrayRows, callback) {
    con.query(getpickupquery, function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            for (var i = 0; i < rows.length; i++) {
                if (rows[i].NextTransaction === 'PickUp') {
                    ArrayRows.push(rows[i]);

                    if (ArraySort.indexOf(rows[i].Kelurahan) == -1) {
                        ArraySort.push(rows[i].Kelurahan);
                    }
                }
            }      ridhogillang, 23 days ago • add new pickup filtering syste

            callback(null, ArraySort, ArrayRows);
        };
    })
}
```

Gambar 19 *Function* getpickupall pada *class admin*

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
19.	GetPickupAll pada halaman admin	Mengakses halaman admin Kondisi:user adalah admin	Aplikasi Mulung akan mengakses data dari tabel transaction dan membandingkan tanggal hari ini dengan waktu next transaction apabila waktu hari ini lebih kecil dari nextTransaction maka akan diberi label Pickup yang kemudian akan digunakan untuk melakukan proses sort.	Benar

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
20.	GetPickupAll pada halaman admin	Mengakses halaman admin Kondisi:user bukan admin	Aplikasi Mulung akan meredirect user kembali kehalaman home sehingga halaman admin tidak akan dapat diakses kecuali user yang mengakses adalah admin.	Benar

```
getpickupsortbykel(con, Kel, rw, rt, ArraySortRWArraySort, ArrayRows, ArraySortRW, ArraySortRT
    if (rw == null && rt == null) {
        pickupquery = getpickupsortbykelquery
    } else if (rt == null) {
        pickupquery = getpickupsortbyrwquery
    } else {
        pickupquery = getpickupsortbyrtquery
    }

    con.query(pickupquery, [Kel, rw, rt], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {

            for (var i = 0; i < rows.length; i++) {
                if (rows[i].NextTransaction === 'PickUp') {
                    ArrayRows.push(rows[i]);

                    if (ArraySort.indexOf(rows[i].Kelurahan) == -1) {
                        ArraySort.push(rows[i].Kelurahan);

                    }
                    if (ArraySortRW.filter(e => e.RW === rows[i].RW).length == 0) {

                        ArraySortRW.push({
                            RW: rows[i].RW,
                            Kel: rows[i].Kelurahan
                        });

                    }
                    if (ArraySortRT.filter(e => e.RT === rows[i].RT).length == 0) {

                        ArraySortRT.push({
                            RT: rows[i].RT,
                            RW: rows[i].RW,
                            Kel: rows[i].Kelurahan
                        });

                    }

                }
            }

            callback(null, ArraySort, ArrayRows, ArraySortRW, ArraySortRT);
        };
    })
}
```

Gambar 20 *Function* getpickupsortbyKel pada *class admin*

Test Case	*Function*	*Input*	*Expected Output*	*Actual Output*
21.	getPickupsortbyKel pada halaman addtransaction	Menekan *Button* Kelurahan	Sama halnya dengan getPickupall namun pada function ini akan di sort berdasarkan kelurahan yang terdapat pada halaman url.	Benar

4) Class transaction.js

```
gettransaction(con, usersId, transactionlist, days, callback) {
    con.query(gettransactionquery,usersId,  function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            var now = new Date();
            var target = new Date(rows[0].nextTransaction);
            var distance = target - now;
            var days = (Math.floor(distance / (1000 * 60 * 60 * 24)));
            if (days == 1) {
                days = 'Besok';
            } else if (days == 0) {
                days = 'Hari ini';
            } else if (days >= 1) {
                days = (days) + ' hari lagi';
            } else if (days <= -1) {
                days = 'Hubungi Team Mulung';
            }      ridhogillang, 23 days ago • add new pickup filtering system

            for (var i = 0; i < rows.length; i++) {
                rows[i].JumlahPoint = parseInt(rows[i].JumlahPoint)
                var tipe = rows[i].TransactionType;
                if (tipe == 'add') {
                    rows[i].JumlahPoint = '+' + rows[i].JumlahPoint;
                } else if (tipe == 'Redeem') {
                    rows[i].JumlahPoint = '-' + rows[i].JumlahPoint;
                }
                transactionlist.push(rows[i])
            }
            callback(null, transactionlist, days);

        };

    });
}
```

Gambar 21 *Function* gettransaction pada *class transaction*

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
22.	getTransaction untuk halaman home	Mengakses halaman home Kondisi:tidak ada session.	Redirect ke halaman login dan mengharuskan user untuk melakukan proses login ulang	Benar
23.	getTransaction untuk halaman home	Mengakses halaman home Kondisi:ada session.	Mengambil 3 transaksi terakhir untuk ditambil pada halaman home dan mengolah data nextTransaction untuk menentukan jadwal penjemputan selanjutnya	Benar

```
getfulltransaction(con, transactionlist, usersId, callback) {
    con.query(getalltransactionquery, usersId, function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            for (var i = 0; i < rows.length; i++) {
                var tipe = rows[i].TransactionType;
                var jenisProduk = rows[i].ProductType;
                var date = new Date(rows[i].TransactionTime).getDate();
                var month = new Date(rows[i].TransactionTime).getMonth();
                var year = new Date(rows[i].TransactionTime).getFullYear();
                rows[i].TransactionTime = date + "-" + (month + 1) + "-" + year;

                if (jenisProduk == "ProdukKaleng" || jenisProduk == "ProdukKardus" || jenisProduk ==
                "ProdukKarton" || jenisProduk == "ProdukPlastik"|| jenisProduk == "produkKaleng" ||
                jenisProduk == "produkKardus" || jenisProduk == "produkKarton" || jenisProduk == "produkPlastik") {
                    rows[i].Quantity = rows[i].Quantity + ' kg';
                    rows[i].ProductType = jenisProduk.toString().slice(6);
                } else if (jenisProduk == 'buatakun') {
                    rows[i].Quantity = rows[i].Quantity;
                } else {
                    rows[i].Quantity = rows[i].Quantity + ' buah';
                }
                rows[i].JumlahPoint = parseInt(rows[i].JumlahPoint)

                if (tipe == 'add') {
                    rows[i].JumlahPoint = '+' + rows[i].JumlahPoint;
                } else if (tipe == 'Redeem') {
                    rows[i].JumlahPoint = '-' + rows[i].JumlahPoint;
                }

                transactionlist.push(rows[i])
            }
            callback(null, transactionlist);
        };

    });
}
```

Gambar 22 *Function getfulltransaction* pada *class transaction*

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
24.	getFullTransaction untuk Transaction	Mengakases halaman Transaction Kondisi: tidak ada session	Redirect ke halaman login dan mengharuskan user untuk melakukan proses login ulang	Benar

Test Case	*Function*	*Input*	*Expected Output*	*Actual Output*
25.	getFullTransaction untuk Transaction.	Mengakses halaman *Transaction* *Kondisi:ada session*	Mengambil semua data transaction dari database meliputi informasi transaksi, jumlah produk, jumlah point waktu transaksi dan lain – lain.	Benar

```
addtransaction(con, transactionId, usersId, transactionType, productType, totalHarga, jumlah, desc,long, lat, callback) {
    con.query(addtransactionquery, [transactionId, usersId, transactionType,
        productType, totalHarga, jumlah, desc,long, lat, usersId], function (err, rows, fields) {      You, a few seconds ago
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        }
    })
}
```

Gambar 23 *Function addtransaction* pada *class transaction*

Test Case	*Function*	*Input*	*Expected Output*	*Actual Output*
26.	Addtransaction untuk addtransaction	Plastik, 1 dan menekan *button* Kondisi:Data Lengkap.	Aplikasi mulung akan menambahakan transaksi baru pada tabel transaksi meliputi Id transaksi, id user, jenis produk serta mengupdate jumlah point user.	Benar

Test Case	*Function*	*Input*	*Expected Output*	*Actual Output*
27.	Addtransaction untuk addtransaction	Plastik, dan menekan button Kondisi:Data Tidak Lengkap.	Aplikasi Mulung akan meredirect kembali user ke halaman add transaction dan memberikan alert bahwa user harus mengisi data dengan lengkap.	Benar

```
exchangepoint(con, transactionId, usersId, transactionType, productType, harga, quantity, desc, callback) {
    con.query(exchangepointquery, [transactionId, usersId,
        transactionType, productType, harga, quantity, desc], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            callback(null, rows[0]);
        }
    })
}
```

Gambar 24 *Function exchangepoint* pada *class transaction*

Test Case	*Function*	*Input*	*Expected Output*	*Actual Output*
28.	exchangepoint untuk Tukarpulsa, isiGopay, IsiGrabPay	081234567890 dan menekan button tukar point Kondisi:Point pengguna tidak cukup	Aplikasi Mulung akan me –redirect user ke halaman tukar pulsa dan memberikan alert yang menyatakan bahwa point dari user tidak cukup.	Benar

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
29.	exchangepoint untuk Tukarpulsa, isiGopay, IsiGrabPay	081234567890 dan menekan button tukar point Kondisi:Point pengguna cukup	Aplikasi Mulung akan me –redirect user halaman home dan mengirimkan post request ke penyedia jasa penukaran point dan mengurangi jumlah point user serta memasuk kantransaksi baru ke database.	Benar

5) Class Point.js

```
updatepoint(con, TotalPoint, usersId, callback) {
    con.query(updatepointquery, [TotalPoint, usersId], function (err, rows, fields) {
        if (err) {
            callback(err, null)
        } else {
            callback(null, rows[0]);
        }
    })
}
getpoint(con, usersId, callback) {
    con.query(getpointquery, [usersId], function (err, rows, fields) {
        if (err) {
            callback(err, null);
        } else {
            console.log(rows)
            var point = rows[0].TotalPoint
            callback(null, point)
        }
    })
}
```

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
30.	getpoint untuk Tukarpulsa, isiGopay, dan isiGrabPay	081234567890 dan menekan button produk yang diinginkan.	Aplikasi Mulung kan mengakses jumlah point dari user	Benar
31.	updatepoint untuk Tukarpulsa, isiGopay, dan isiGrabPay	081234567890 dan menekan button produk yang diinginkan.	Aplikasi mulung akan mengurangi jumlah point yang didapatkan pada function getpoint dengan hasil dari function getprice pada class product dan menyimpan point baru pada database.	Benar

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
32.	getpoint untuk add transaction pada admin	Menekan button tambahkan transaksi	Aplikasi Mulung kan mengakses point dari user yang akan ditambahkan transaksinya.	Benar
33.	updatepoint untuk add transaction pada admin	Menekan button tambahkan transaksi	Aplikasi mulung akan menambahkan jumlah point yang didapatkan pada function *getpoint* dengan hasil dari function *getprice* pada class *product* dan menyimpan point baru pada database.	Benar

6) Class Product.js

```
class products {
    getprice(con, ProductType, callback){
        con.query(getpricequery, [ProductType], function (err, rows, fields) {
            if (err) {
                callback(err, null)
            } else {
                var price = rows[0].Price;
                callback(null, price);
            }
        })
    }
}
```

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
34.	getprice untuk addtransaction	Plastik	Aplikasi Mulung akan mengakses jumlah point dan nilai tukar dari sampah dengan point.	Benar
35.	getprice untuk Tukarpulsa, isiGopay, dan isiGrabPay	081234567890 dan menekan button produk yang diinginkan.	Aplikasi Mulung akan mengakses jumlah point dan nilai tukar dari product yang diinginkan dengan point.	Benar

7) Function Logout

```
router.get('/logout', function (req, res) {
  req.logout();
  res.cookie('connect.sid', '', {
    expires: new Date(1),
    path: '/'
  });
  res.cookie('mycookie', '', {
    expires: new Date(1),
    path: '/',
    httpOnly: false
  });
  res.redirect('/login');
});      ridhogillang, 3 months ago • First commit
```

Test Case	***Function***	***Input***	***Expected Output***	***Actual Output***
36.	LogOut	Menekan button logout	Aplikasi Mulung akan menghapus session serta cookies dan meredirect user ke halaman login.	Benar