BAB IV

ANALISA DAN PERANCANGAN SISTEM

4.1. Analisa Permasalahan

Bisnis laundry adalah sebuah bisnis yang cukup menjanjikan di masa sekarang. Pada era modern yang serba cepat ini, banyak sekali yang membutuhkan usaha laundry untuk kepentingan kebersihan pakaian. Bukan hanya pakaian, segala jenis kain seperti sprei, *bed cover*, sepatu, dan lain lain juga sering mengandalkan usaha laundry untuk masalah kebersihanya.

Bisnis ini muncul dengan memberikan banyak sekali manfaat, terutama untuk yang tidak memiliki waktu untuk mengurus pakaian kotor dan menginginkan sebuah pelayanan kebersihan pakaian secara cepat dan praktis. Laundry ini hadir dengan menawarkan beberapa pelayanan seperti cuci cepat, cuci setrika, cuci setrika beserta *parfume*, dan lain lain. Harga dari pelayanan tersebut juga terbilang cukup terjangkau tergantung pelayanan apa saja yang dipilih dan berapa berat cucian yang masuk.

Dengan harga yang terjangkau dan pelayanan yang menjanjikan, bisnis ini menjadi salah satu bisnis yang diminati di masyarakat luas. Karena bisnis ini cukup besar dan banyak diminati, maka tidak menutup kemungkinan bahwa akan banyak sekali UMKM laundry yang buka dengan standar pelayanan yang masih kurang memadai, terutama pada bagian administrasi dan management bisnis mereka yang masih prematur dan kurang diperhatikan.

Salah satu masalah yang bisa diambil dari bisnis laundry ini adalah masalah management internal dari laundry tersebut. Resiko terbesar dari masalah management ini adalah kebangkrutan dari usaha laundry itu sendiri, yang dikarenakan oleh kesalahan management yang masih kurang efisien. Contoh dari proses mangement yang kurang efisien adalah proses transaksi yang masih dilakukan secara manual tanpa menggunakan sistem di dalamnya, sehingga proses ini rawan terjadi *Human Error* yang menyebabkan kerugian pada sisi management internal.

Oleh karena itu, tercipta sebuah ide untuk membuat sebuah aplikasi transaksi laundry digital yang terhubung dengan timbangan digital yang bisa melakukan kalkulasi berat pakaian dengan harga satuan secara otomatis, sehingga proses transaksi bisa dilakukan secara cepat dan efisien. Data transaksi dari aplikasi ini secara otomatis akan direkam di dalam *database* yang mana bisa disajikan sebagai data penjualan yang bisa diakses secara *online.* Dengan aplikasi ini, maka usaha laundry bisa meminimalisir kesalahan dalam melakukan transaksi hingga melakukan rekap data transaksi, sehingga management laundry bisa berjalan secara lancar dan sehat.

4.2. Analisa Kebutuhan Sistem

Analisa kebutuhan sistem dilakukan intuk mengetahui kebutuhan dan spesifikasi apa saja yang diperlukan dalam proses pembuatan aplikasi ini yang mencangkup kebutuhan *hardware* dan *Software.*

4.2.1. Analisa Kebutuhan Hardware

Perangkat keras yang digunakan dalam proses pembuatan aplikasi ini adalah :

1. Laptop dengan spesifikasi:
    a. Ryzen 5 7000 Series.
    b. AMD Radeon Graphic Card.
    c. RAM 16 GB.
    d. SSD 512 GB
2. *Smartphone* android Oppo Reno 4.

4.2.2. Analisa Kebutuhan Software

Perangkat lunak yang digunakan dalam proses pembuatan aplikasi ini adalah :

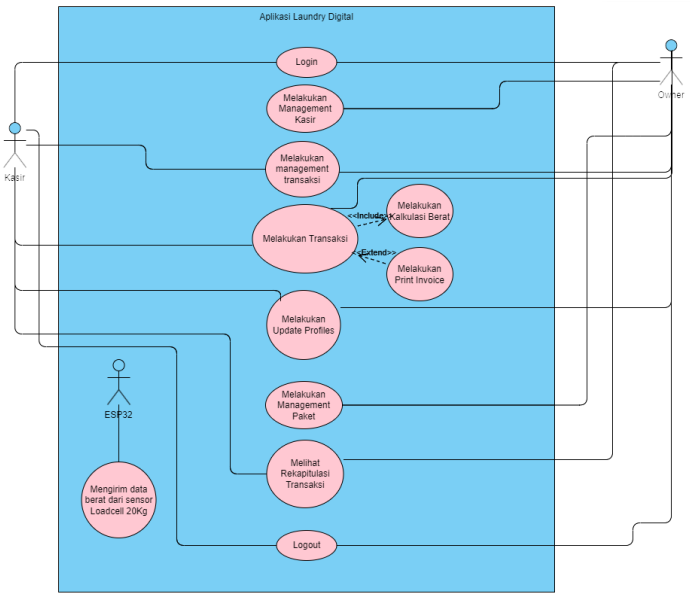
1. *OS Windows* 11.
2. *Android Studio.*
3. *Visual Studio Code.*
4. *Postman.*
5. *MySQL Database.*

4.3. Perancangan Sistem

Perencanaan sistem sangat penting dalam menyederhanakan proses pengembangan aplikasi ini. penerapan model pada perancangan sistem ini menggunakan UML (Unified Modeling Language) dan *Flowchart* agar sistem bisa dijabarkan secara rinci melalui *Usecase*, *Class*, dan, *Activity*.

4.3.1. *Usecase* Diagram

Usecase diagram yang dibuat pada aplikasi transaksi laundry digital berbasis mobile menggunakan flutter bisa dilihat pada Gambar 4.1.



Gambar 4. 1. *Usecase* Diagram Aplikasi Laundry Digital

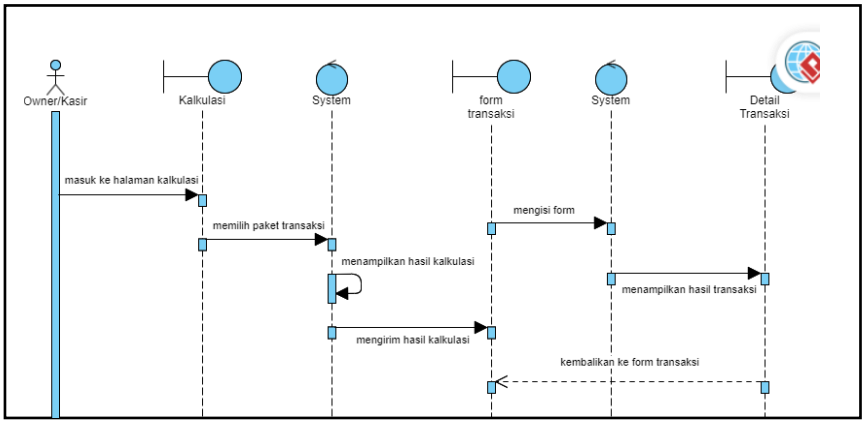
Pada Gambar 4.1 dijelaskan bahwa sistem memiliki 3 aktor, yaitu *owner*, kasir, dan *ESP32*. Pada bagian *owner*, dijelaskan bahwa actor owner memiliki akses untuk melakukan *login*, *logout*, management kasir, management paket, halaman rekapitulasi, *update profile*, dan transaksi.

Bagian *actor* kasir dijelaskan bahwa kasir hanya bisa melakukan akses pada *login, logout, update profile, dan transaksi.* Untuk bagian *actor ESP32* hanya melakukan pengiriman data berat yang telah direkam oleh sensor *Loadcell* 20kg ke dalam sistem.

4.3.2. *Sequence* Diagram

Pada pada perancangan sistem ini, *sequence* diagram juga digunakan untuk mengetahui alur dari suatu proses, sehingga bisa mendapatkan gambaran tentang cara kerja dari suatu proses tersebut. Berikut beberapa contoh dari *Sequence* diagram.

1) *Sequence* Diagram Input Transaksi

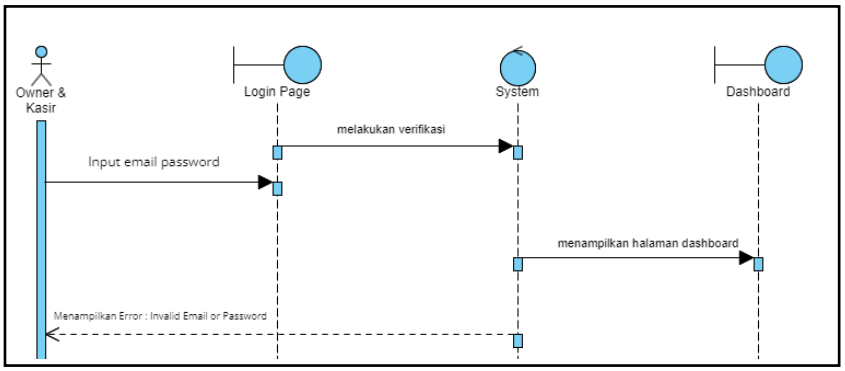


Gambar 4. 2. Proses Transaksi

Dalam Gambar 4.2. *Sequence* diagram dari proses transaksi dimulai dengan *Owner* atau kasir yang membuka halaman transaksi, yang kemudian menekan tombol kalkulasi berat. Lalu halaman transaksi akan membuka halaman kalkulasi berat dan memproses data berat.

Hasil dari kalkulasi berat ini akan mengahsilkan data berat yang telah dikalikan dengan data harga satuan. yang selanjutnya data tersebut dikirimkan kedalam halaman form transaksi. Pada halaman ini user akan diminta untuk melengkapi form data pelanggan. Setelah selesai form data akan dikirmkan ke sistem dan sistem akan mengembalikan pesan transaksi telah dibuat ke halaman transaksi.

2) *Sequence* Diagram *Login*

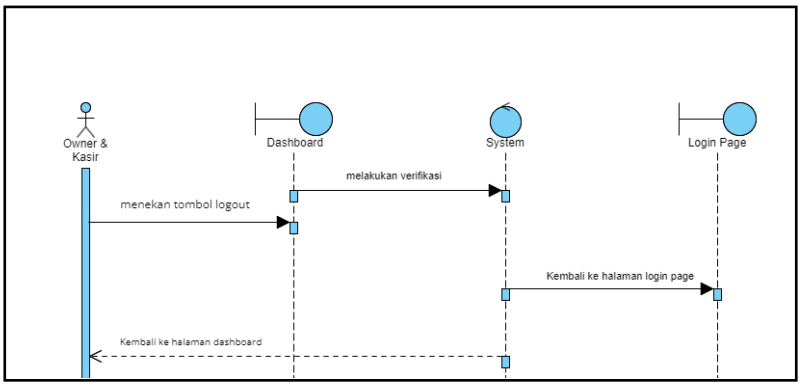


Gambar 4. 3. Proses *Login*

Proses *Login* dimulai dengan *user* yang mengakses halaman *login Page* dan *user* diminta untuk mengisi *email* dan *password*. Lalu sistem akan melakukan verifikasi untuk mengecek apakah data valid atau tidak, jika iya maka akan diarahkan ke

halaman *dashboard* dan jika tidak valid maka *user* akan diarahkan kehalaman *login* dengan menampilkan *error email or password* salah.

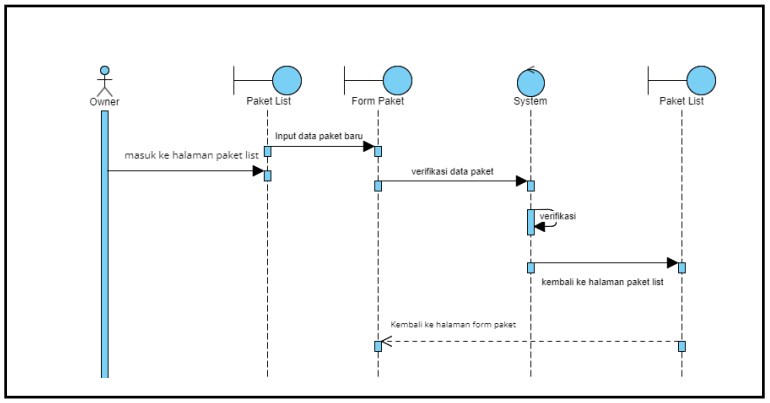
3) *Sequence* Diagram *Logout*



Gambar 4. 4. Proses *Logout*

Proses *Logout* dimulai dengan *user* yang menekan tombol *logout* pada halaman *dashboard*, lalu sistem akan melakukan verifikasi, apakah yakin akan melakukan *logout* atau tidak. Jika iya maka *user* akan diarahkan ke dalam halaman *login.* Dan jika tidak maka *user* akan tetap berada di halaman *dashboard.*

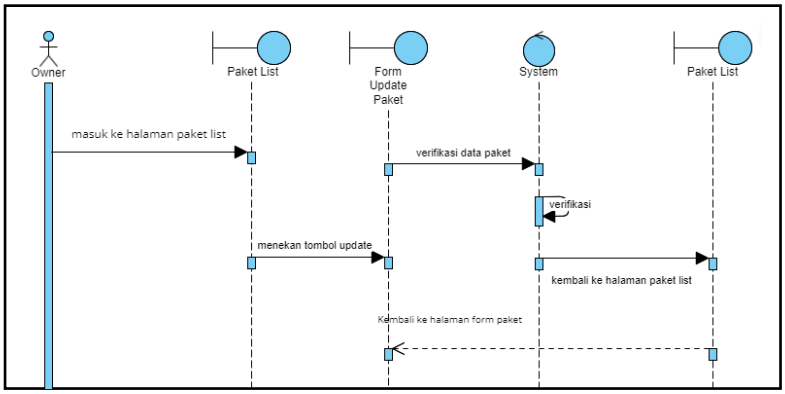
4) *S*equence Diagram Input Paket



Gambar 4. 5. Proses Input Paket

Proses input Paket dimulai dengan *user* yang masuk kedalam halaman Paket *List* dan akan diarahkan ke dalam halaman halaman *form* paket lalu *user* diminta untuk mengisi data paket yang akan ditambahkan. Setelah itu sistem akan melakukan proses verifikasi terhadap data paket apakah valid atau tidak. Jika valid maka paket akan disimpan dan *user* akan dikembalikan ke halaman paket *list*, dan jika tidak maka *user* akan dikembalikan ke halaman *form* pengisian.

5) *Sequence* Diagram *Update* Paket

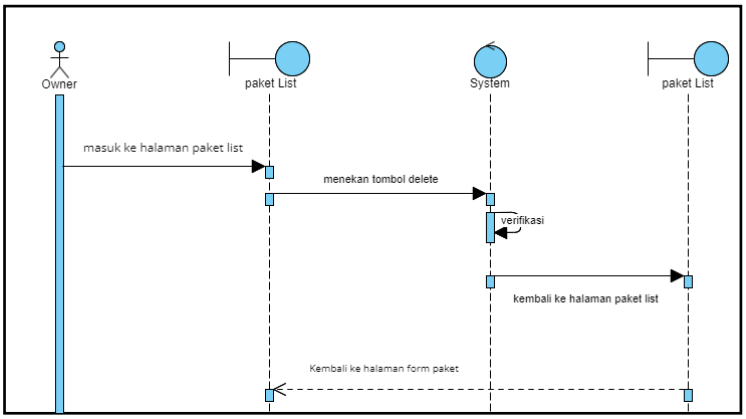


Gambar 4. 6. Proses *Update* Paket

Proses *Update* Paket dimulai dengan *user* yang masuk kedalam halaman Paket *List* dan *user* menekan tombol *update*, lalu *user* akan diarahkan ke dalam halaman halaman *form* paket lalu *user* diminta untuk mengisi data paket yang akan diperbarui. Setelah itu sistem akan melakukan proses verifikasi terhadap data paket apakah valid atau tidak. Jika valid maka paket akan diubah dan *user* akan dikembalikan ke halaman paket *list*, dan jika tidak maka *user* akan

dikembalikan ke halaman *form* pengisian.

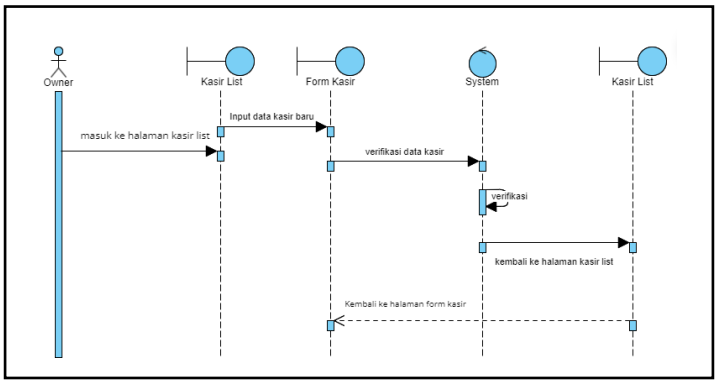
6) *Sequence* Diagram *Delete* Paket



Gambar 4. 7. Proses *Delete* Paket

Proses *Delete* Paket dimulai dengan *user* yang masuk kedalam halaman Paket *List* dan akan diminta untuk menekan tombol hapus. Setelah itu sistem akan melakukan proses verifikasi terhadap data paket apakah yakin akan dihapus atau tidak. Jika valid maka paket akan dihapus dan *user* akan dikembalikan ke halaman paket *list*, dan jika tidak maka *user* akan dikembalikan ke halaman paket *list.*

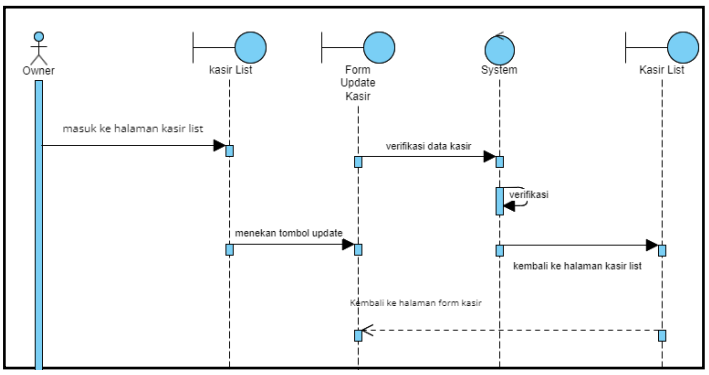
7) *Sequence* Diagram Input Kasir



Gambar 4. 8. Proses Input Kasir

Proses input Kasir dimulai dengan *user* yang masuk kedalam halaman Kasir *List* dan akan diarahkan ke dalam halaman halaman *form* kasir lalu *user* diminta untuk mengisi data kasir yang akan ditambahkan. Setelah itu sistem akan melakukan proses verifikasi terhadap data kasir apakah valid atau tidak. Jika valid maka data kasir akan disimpan dan *user* akan dikembalikan ke halaman kasir *list*, dan jika tidak maka *user* akan dikembalikan ke halaman *form* pengisian.

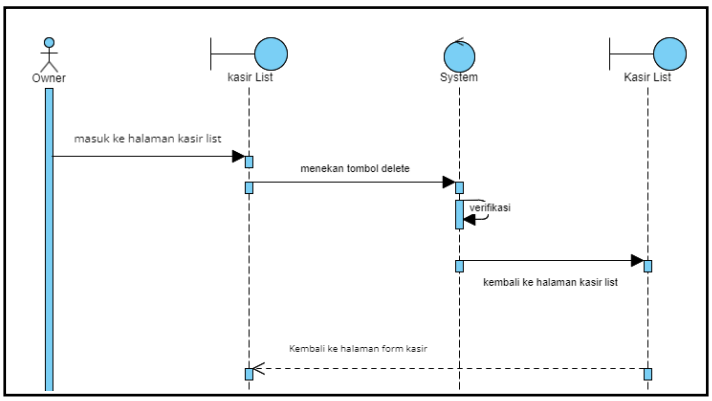
8) *Sequence* Diagram *Update* Kasir



Gambar 4. 9. Proses *Update* Kasir

Proses *Update* Kasir dimulai dengan *user* yang masuk kedalam halaman Kasir *List* dan *user* menekan tombol *update*, lalu *user* akan diarahkan ke dalam halaman halaman *form* kasir lalu *user* diminta untuk mengisi data kasir yang akan diperbarui. Setelah itu sistem akan melakukan proses verifikasi terhadap data kasir apakah valid atau tidak. Jika valid maka data kasir akan diubah dan *user* akan dikembalikan ke halaman kasir *list*, dan jika tidak maka *user* akan dikembalikan ke halaman *form* pengisian.

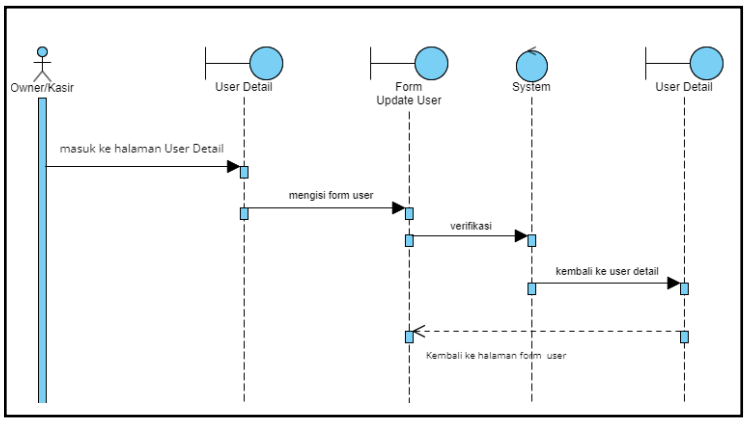
9) *Sequence* Diagram *Delete* Kasir



Gambar 4. 10. Proses *Delete* Kasir

Proses *Delete* Kasir dimulai dengan *user* yang masuk kedalam halaman Kasir *List* dan akan diminta untuk menekan tombol hapus. Setelah itu sistem akan melakukan proses verifikasi terhadap data kasir apakah yakin akan dihapus atau tidak. Jika valid maka kasir akan dihapus dan *user* akan dikembalikan ke halaman kasir *list*, dan jika tidak maka *user* akan dikembalikan ke halaman kasir *list.*

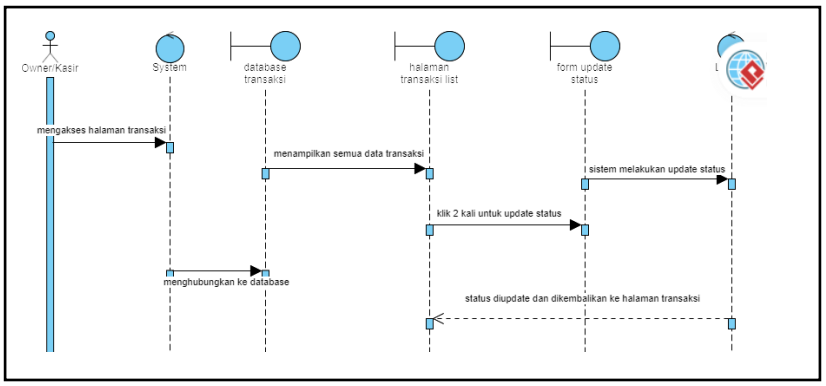
10) *Sequence* Diagram *Update Profile*



Gambar 4. 11. Proses Update Profile

Proses *Update* Profile dimulai dengan *user* yang masuk kedalam halaman *User Detail* dan *user* menekan tombol *update*, lalu *user* akan diarahkan ke dalam halaman halaman *form profiles* lalu *user* diminta untuk mengisi data *profile* yang akan diperbarui. Setelah itu sistem akan melakukan proses verifikasi terhadap data *profile* apakah valid atau tidak. Jika valid maka data *profile* akan diubah dan *user* akan dikembalikan ke halaman *Detail Profiles*, dan jika tidak maka *user* akan dikembalikan ke halaman *form* pengisian.

11) *Sequence* Diagram *Update* Transaksi



Gambar 4. 12. Proses *Update* Transaksi

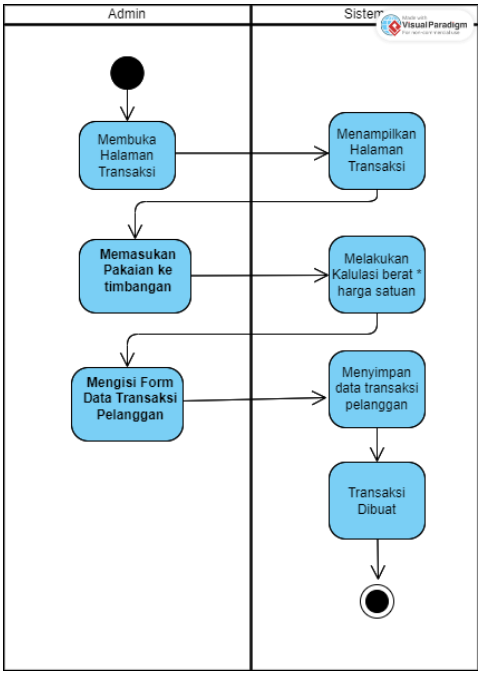
Proses *Update* Transaksi dimulai dengan *user* yang mengakses halaman Transaksi *List* dan menekan dua kali pada data yang akan diubah statusnya. Lalu sistem akan menampilkan *form* untuk mengubah status dari data transaksi yang dipilih dan *user* diminta untuk memilih salah satu.

Setelah *user* memilih salah satu, maka sistem akan melakukan verifikasi pada data status yang dipilih oleh *user* apakah valid atau tidak. Jika valid maka status pada data transaksi akan diubah dan *user* akan dikembalikan ke halaman transaksi *list*. Jika tidak valid maka *user* akan dikembalikan ke halaman *form input.*

4.3.3. *Activity* Diagram

Activity diagram pada sistem ini berfungsi untuk mengambarkan aktifitas proses transaksi pada sistem ini. Berikut beberapa contoh *activity* diagram.

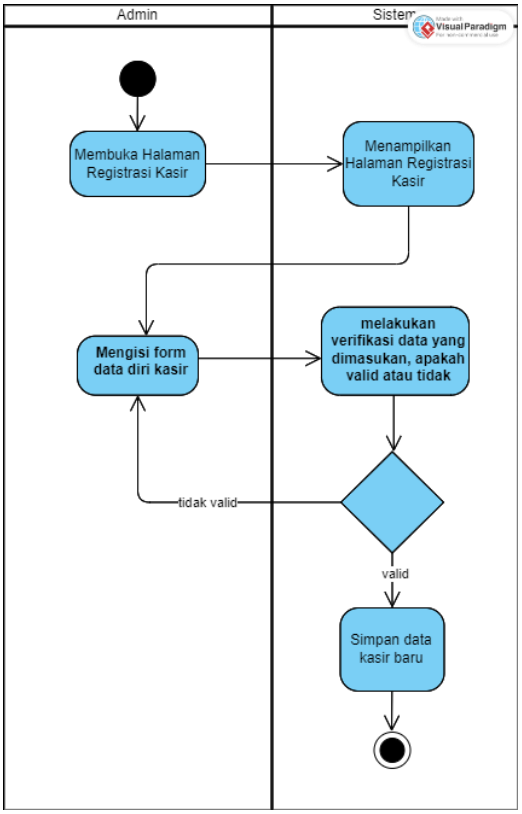
a) *Activity* Diagram halaman *dashboard*



Gambar 4. 13. Gambar *Activity* diagram

Proses yang digambarkan pada *Activity* diagram dimulai pada sisi admin atau *owner* yang dimulai dengan membuka halaman transaksi dan sistem akan menampilkannya.

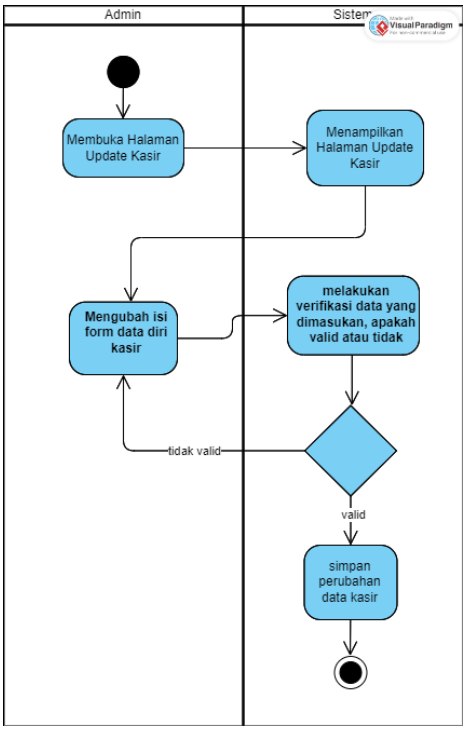
b) *Activity* Diagram Registrasi Kasir



Gambar 4. 14. *Activity* Diagram Registrasi

Gambar 4.11. dijelaskan bahwa proses dimulai dengan membuka halaman registrasi kasir lalu sistem akan menampilkannya. Dan *user* diminta untuk mengisi form data kasir. Setelah selesai sistem akan melakukan pemeriksaan apakah data yang dimasukan itu valid atau tidak, jika valid maka data disimpan, dan jika tidak valid maka akan dikembalikan ke halaman pengisian form.

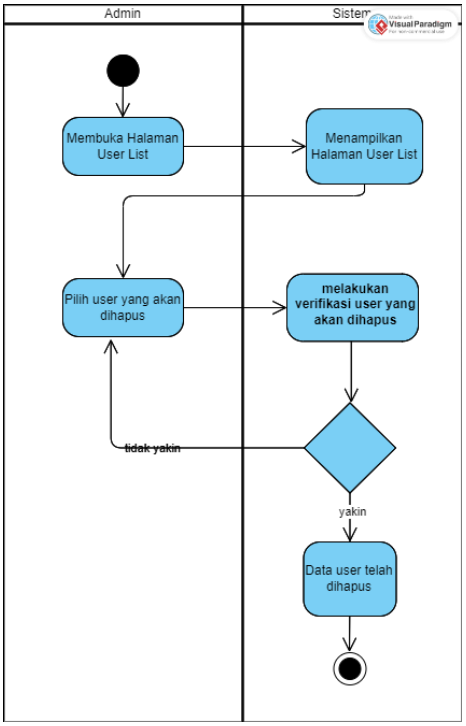
c) *Activity* Diagram *Update* Kasir



Gambar 4. 15. *Activity* Diagram *Update* Kasir

Pada Gambar 4.15. dijelaskan bahwa proses *update* kasir dimulai dengan membuka halaman update kasir dan sistem akan menampilkannya. Lalu user akan mengupdate data kasir dan sistem akan melakukan pemeriksaan pada data, apakah valid atau tidak. Jika valid maka kasir berhasil diupdate jika tidak valid maka akan dikembalikan ke dalam halaman form.

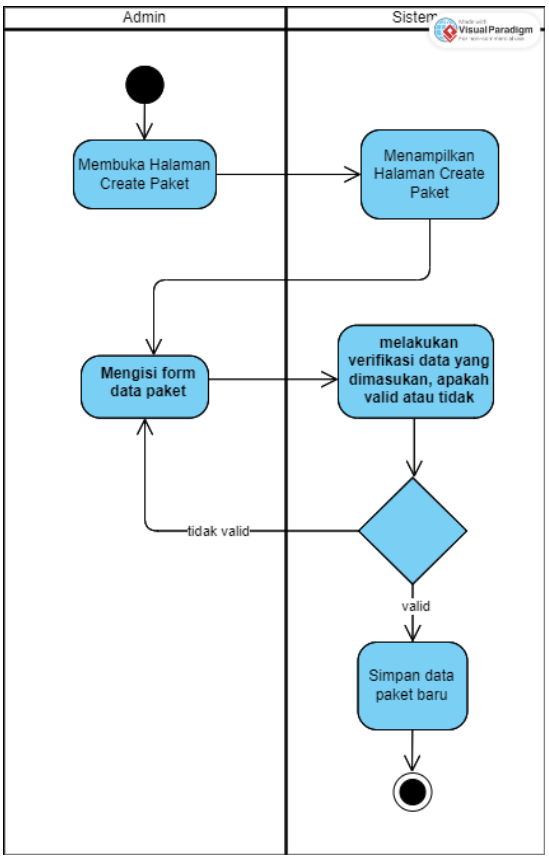
d) *Activity* Diagram *Delete User*



Gambar 4. 16. *Activity* Diagram *Delete User*

Pada Gambar 4.16. untuk menghapus user, diperlukan untuk mengakses halaman *User List* lalu pilih data *user* yang akan dihapus. Setelah itu sistem akan melakukan verifikasi, apakah yakin atau tidak. Jika iya maka data akan dihapus, jika tidak maka akan dikembalikan ke halaman *User List.*

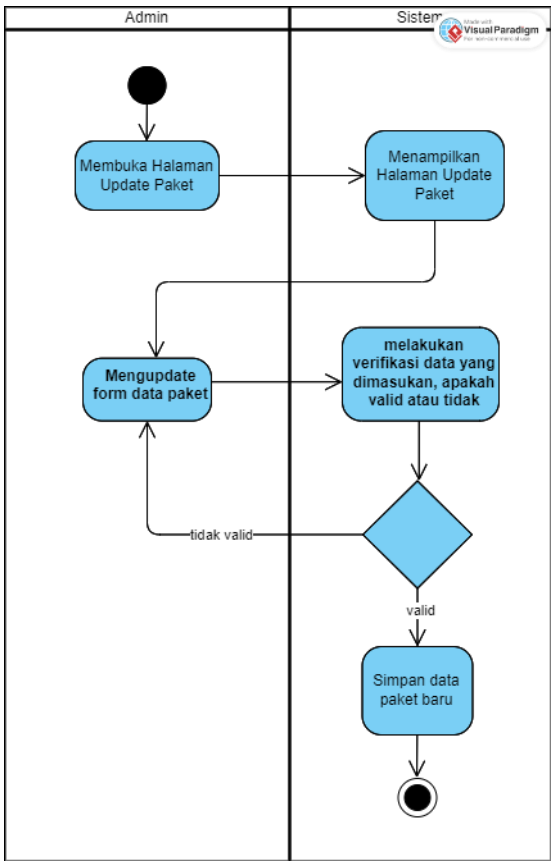
e) *Activity* Diagram *Create* Paket



Gambar 4. 17. *Activiy* Diagram *Create* Paket

Pada Gambar 4.17. proses dimulai dengan membuka halaman *Create* Paket dan *user* akan diminta mengisi form data paket yang akan dibuat. Lalu sistem akan memverifikasi data paket, apakah valid atau tidak. Jika valid maka data paket disimpan, dan jika tidak akan dikembalikan ke halaman halaman form pengisian.

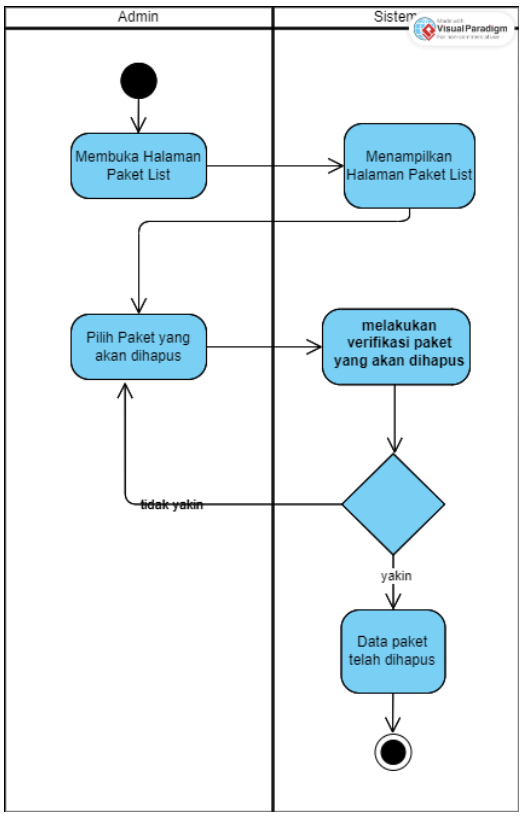
f) *Activity* Diagram *Update* Paket



Gambar 4. 18. *Activity* Diagram *Update* Paket

Pada Gambar 4.18. dijelaskan bahwa proses *update* paket dimulai dengan membuka halaman update paket dan sistem akan menampilkannya. Lalu user akan mengupdate data paket dan sistem akan melakukan pemeriksaan pada data, apakah valid atau tidak. Jika valid maka paket berhasil diupdate jika tidak valid maka akan dikembalikan ke dalam halaman form.

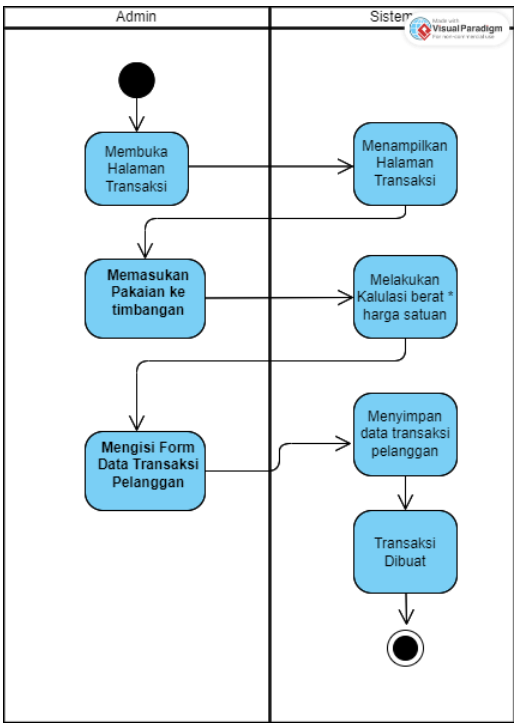
g) *Activity* Diagram *Delete* paket



Gambar 4. 19. Activity Diagram *Delete* Paket

Pada Gambar 4.19. dijelaskan bahwa proses *delete* paket dimulai dengan membuka halaman paket *List* lalu pilih data paket yang akan dihapus. Dan sistem akan memverifikasi data, apakah yakin untuk dihapus atau tidak. Jika iya maka data akan dihapus, dan jika tidak maka akan dikembalikan ke halaman paket *List.*

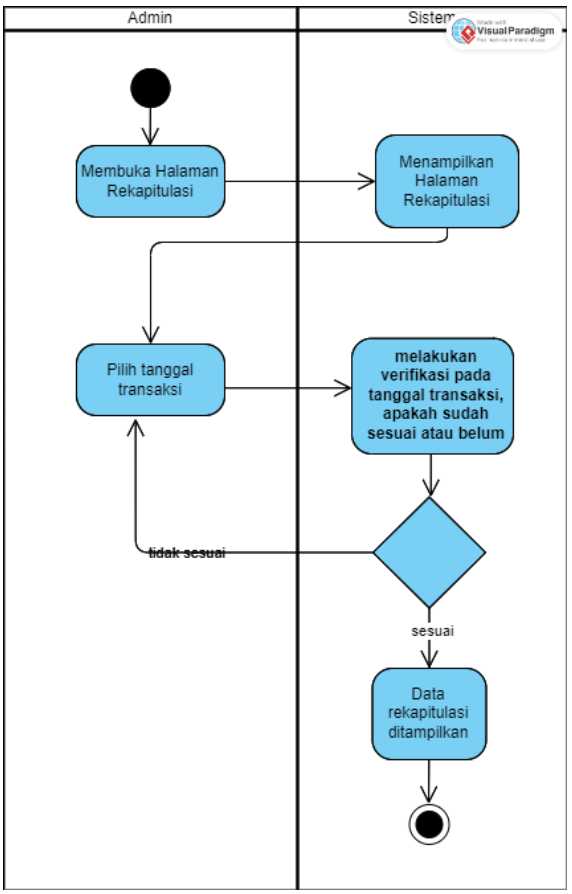
h) *Activity* Diagram Transaksi



Gambar 4. 20. *Activity* Diagram Transaksi

Pada Gambar 4.20. dijelaskan bahwa proses *update* kasir dimulai dengan membuka halaman update kasir dan sistem akan menampilkannya. Lalu user akan mengupdate data kasir dan sistem akan melakukan pemeriksaan pada data, apakah valid atau tidak. Jika valid maka kasir berhasil diupdate jika tidak valid maka akan dikembalikan ke dalam halaman form.

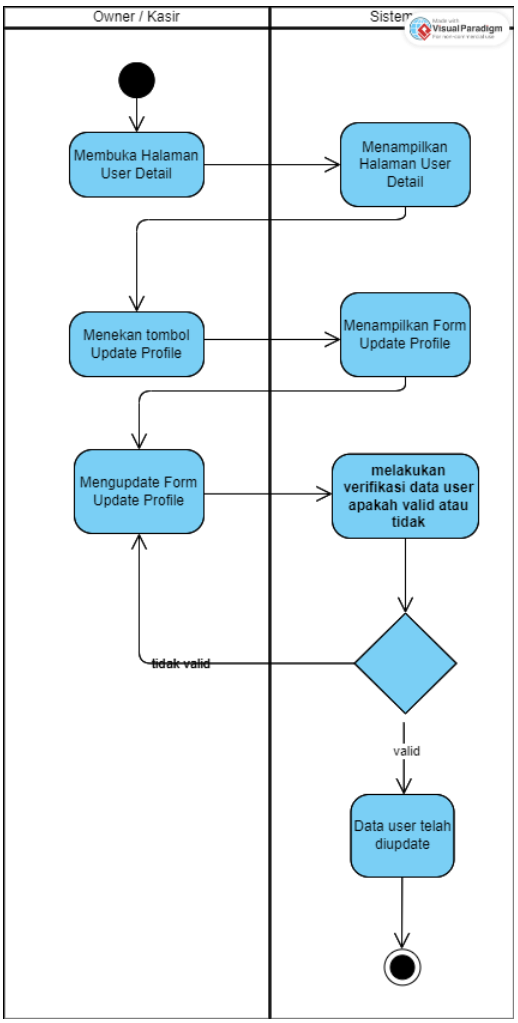
i) *Activity* Diagram Rekapitulasi



Gambar 4. 21. *Activity* Diagram Rekapitulasi

Pada Gambar 4.21. dijelaskan bahwa proses rekapitulasi dimulai dengan membuka halaman rekapitulasi dan halaman akan meminta tanggal transaksi, setelah tanggal transaksi dipilih, maka sistem akan memverifikasi apakah tanggal transaksi sesuai atau tidak. Jika iya maka data rekapitulasi transaksi akan ditampilkan. Jika tidak maka akan dikembalikan ke halaman pemilihan tanggal.

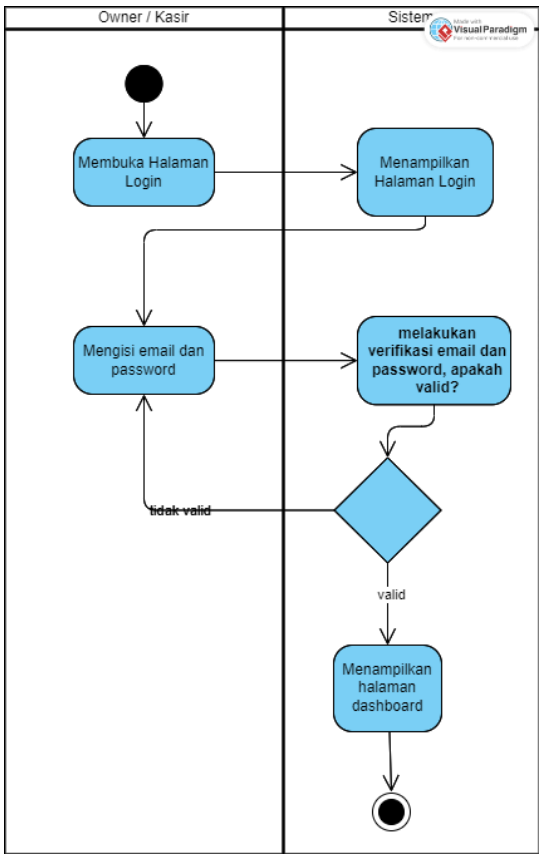
j) *Activity* Diagram *Update Profile*



Gambar 4. 22. *Activity* Diagram *Update Profile*

Pada Gambar 4.22. dijelaskan bahwa proses update profile dimulai dengan membuka halaman *user detail* lalu menekan tombol *update profile*, maka sistem akan menampilkan halaman form *update profile* dan *user* akan diminta untuk mengupdate data pada form tersebut. Lalu sistem akan memverifikasi data yang telah diupdate, apakah valid atau tidak. Jika valid maka data berhasil diupdate, jika tidak valid maka akan dikembalikan ke halaman form.

k) *Activity* Diagram *Login*



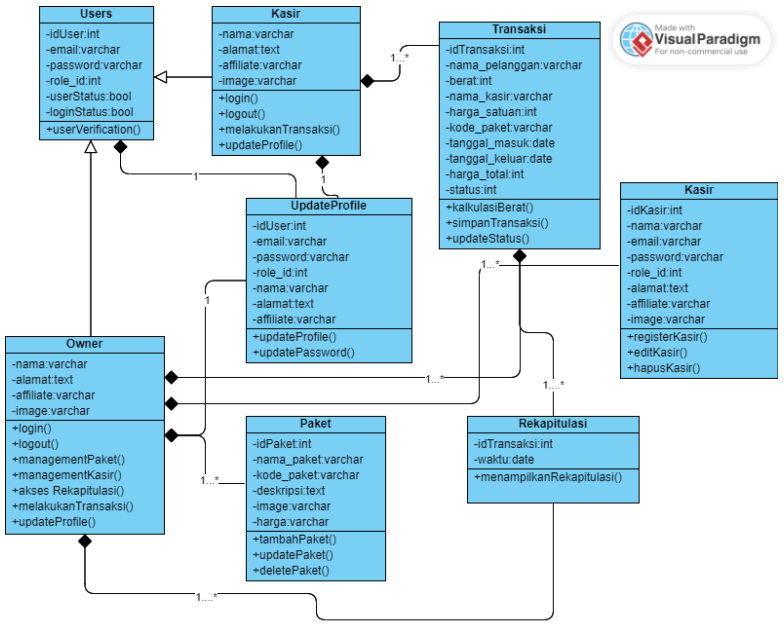
Gambar 4. 23. *Sequence* Diagram *Login*

Pada Gambar 4.23. dijelaskan bahwa proses *login* dimulai dengan *User* yang membuka halaman *login* dan sistem akan menampilkan halaman *login*. Setelah itu *user* akan diminta untuk mengisi email dan password yang dibutuhkan, dan sistem akan melakukan verifikasi pada data *user* apakah valid atau tidak. Jika iya maka *user* akan diarahkan ke halaman *dashboard*, dan jika tidak maka user akan dikembalikan ke halaman *login.*

4.3.4. *Class* Diagram

Pada sistem ini dibuat sebuah *Class* diagram yang berguna untuk mengetahui gambaran sistem secara rinci beserta dengan fungsi yang bisa diakses dalam sistem ini.

Gambar dari *Class* diagram ini bisa dilihat pada Gambar 4.24.



Gambar 4. 24. *Class* diagram aplikasi laundry

Pada Gambar 4.24. ditunjukan bahwa ada 8 objek class yang masing masing memiliki hubungan relasi mereka masing masing. Dimulai dengan objek users yang memiliki attribut berupa data user dan memiliki method userVerification yang berfungsi sebagai validasi user saat melakukan autentikasi.

Objek user memiliki 2 turunan yaitu *Owner* dan kasir yang masing masing mewarisi attribut milik user dengan beberapa penyesuaian attribut dan method mereka untuk kepentingan *authorization*.

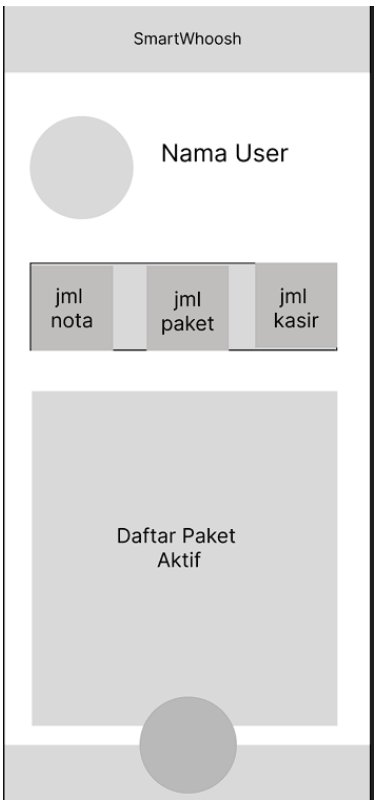
Objek owner bisa mengakses hampir semua objek *class* yang ada, seperti *class* transaksi, updateProfile, managementKasir, managementPaket, dan rekapitulasi. Sedangkan objek *class* kasir hanya bisa mengakses objek *class* transaksi dan updateProfile.

4.3.5. Rancangan Tampilan Aplikasi

Perancangan tampilan disini adalah rancangan menu pada aplikasi yang akan dibuat. Berikut beberapa contoh tampilan menu pada aplikasi transaksi laundry ini.

1. *Homepage* menu

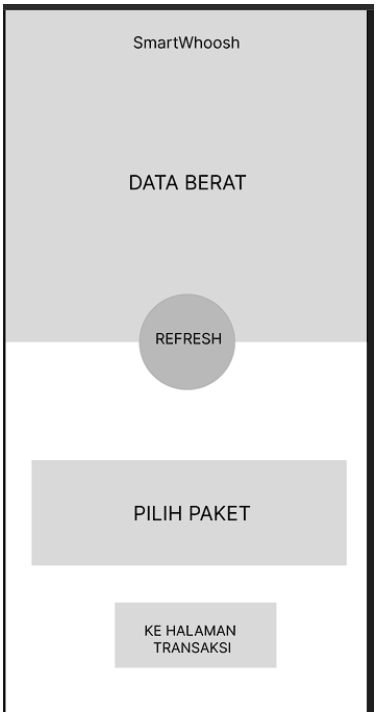
Gambar *Homepage* menu bisa dilihat pada Gambar 4.25.



Gambar 4. 25. Tampilan menu *Homepage*

2. Menu kalkulasi berat

Gambar menu kalkulasi berat bisa dilihat pada Gambar 4.26.



Gambar 4. 26. tampilan menu kalkulasi berat

3. Menu transaksi

Gambar menu transaksi bisa dilihat pada Gambar 4.27.

Gambar 4. 27. Tampilan menu transaksi

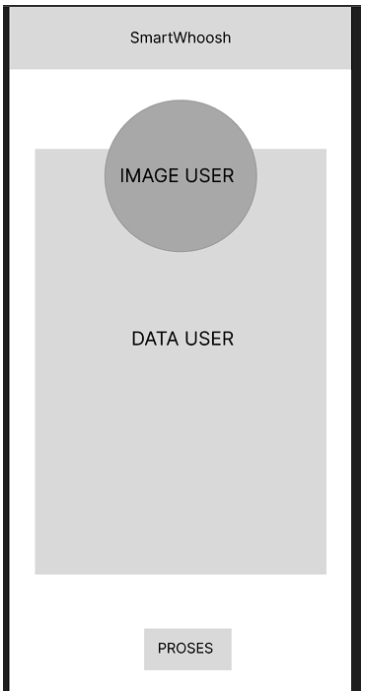
4. Menu Rekapitulasi

Gambar menu rekapitulasi bisa dilihat pada Gambar 4.28.

Gambar 4. 28. Tampilan menu rekapitulasi

5. Menu *Update Profiles*

Gambar tampilan menu *Update Profiles* bisa dilihat pada Gambar 4.29.



Gambar 4. 29. Tampilan menu *Update Profiles*