PENERAPAN DAN PERANCANGAN APLIKASI *POINT OF SALE* DENGAN METODE *WATERFALL* DI OPTIK TAZMA

SKRIPSI

Diajukan sebagai salah satu syarat
untuk Menempuh Ujian Akhir Program Strata (S1)
Program Studi Teknik Informatika STMIK – Indonesia Mandiri

Oleh
Ryan Ramadhan Harahap
361762001

SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER
INDONESIA MANDIRI
2020

PENERAPAN DAN PERANCANGAN APLIKASI *POINT OF SALE* DENGAN METODE *WATERFALL* DI OPTIK TAZMA

SKRIPSI

Diajukan sebagai salah satu syarat
untuk Menempuh Ujian Akhir Program Strata (S1)
Program Studi Teknik Informatika STMIK – Indonesia Mandiri

Oleh
Ryan Ramadhan Harahap
361762001

SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER
INDONESIA MANDIRI
2020

LEMBAR PENGESAHAN

PENERAPAN DAN PERANCANGAN APLIKASI *POINT OF SALE* DENGAN *METODE WATERFALL* DI OPTIK TAZMA

Oleh :

RYAN RAMADHAN HARAHAP

361762001

Skripsi ini telah diterima dan disahkan untuk
Memenuhi persyaratan mencapai gelar

SARJANA TEKNIK INFORMATIKA

Pada

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
INDONESIA MANDIRI

Bandung, Oktober 2020
Disahkan oleh

Ketua Program Studi	Dosen Pembimbing
Chalifa Chazar, S.T., M.T.	Patah Herwanto, S.T., M.Kom.
NIDN. 0421098704	NIDN. 0027107501

LEMBAR PERSETUJUAN REVISI

PENERAPAN DAN PERANCANGAN APLIKASI *POINT OF SALE* DENGAN *METODE WATERFALL* DI OPTIK TAZMA

Oleh :

RYAN RAMADHAN HARAHAP

361762001

Telah melakukan sidang skripsi dan telah melakukan revisi sesuai dengan perubahan dan perbaikan yang diminta pada saat sidang skripsi.

Bandung, Oktober 2020

Menyetujui

No.	Nama Dosen	Keterangan	Tanda Tangan
1.	Patah Herwanto, S.T., M.Kom.	Pembimbing	
2.	Chairuddin, Ir., M.T., M.M., Dr.	Penguji 1	
3.	Moch. Ali Ramdhani, S.T., M.Kom.	Penguji 2	

Mengetahui
Ketua Program Studi Teknik Informatika,

Chalifa Chazar, S.T., M.T.
NIDN. 0421098704

SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa :

(1) Naskah skripsi adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik, baik di Sekolah Tinggi Manajemen Informatika dan komputer Indonesia Mandiri maupun perguruan tinggi lainnya.

(2) Skripsi ini murni merupakan karya penelitian saya sendiri dan tidak menjiplak karya pihak lain. Dalam hal ada bantuan atau arahan dari pihak lain maka telah saya sebutkan identitas dan jenis bantuannya di dalam lembar ucapan terima kasih.

(3) Seandainya ada karya pihak lain yang ternyata memiliki kemiripan dengan karya saya ini, maka hal ini adalah di luar pengetahuan saya dan terjadi tanpa kesengajaan dari pihak saya.

Pernyataan ini saya buat dengan sesungguhnya dan apabila di kemudian hari terbukti adanya kebohongan dalam pernyataan ini, maka saya bersedia menerima sanksi akademik sesuai norma yang berlaku di Sekolah Tinggi Manajemen Informatika dan Komputer Indonesia Mandiri.

Bandung, Oktober 2020
Yang membuat pernyataan

Ryan Ramadhan Harahap
361762001

ABSTRAK

PENERAPAN DAN PERANCANGAN APLIKASI *POINT OF SALE* DENGAN *METODE WATERFALL* DI OPTIK TAZMA

Oleh
Ryan Ramadhan Harahap
361762001

Tujuan penulisan Laporan Skripsi ini adalah untuk mempermudah dalam pendataan barang dan melakukan transaksi di Optik Tazma sebagai solusi untuk perbaikan jalan nya proses penjualan dengan lebih efektif dan efisien, serta memudahkan pekerjaan karyawan yang berhubungan dengan pelayanan (*service*) memudahkan kepada pemilik optik untuk melakukan pengecekan serta evaluasi terhadap pendapatan dari hasil penjualan. Metode penelitian yang dilakukan oleh penulis dalam melakukan penelitian meliputi studi lapangan dan studi Pustaka serta penggunaan metode sebagai penerapan dan perancangan aplikasi dengan bahasa pemrograman *Visual Basic*. Studi lapangan meliputi observasi dan analisa *system* yang berjalan. Studi pustaka di lakukan dengan penellitian kepustakaan yang relevan dengan masalah yang di hadapi penulis, metode perancangan serta pengembangan menggunakan metode *waterfall*. Dari analisa dan pengamatan yang dilakukan dapat diketahui bahwa pembuatan Aplikasi Penjulan *Point Of Sale* (POS) pada Optik Tazma dapat membantu memudahkan pekerjaan karyawan lainnya serta memudahkan dalam merencanakan target penjualan kedepan nya agar terus meningkat.

Kata kunci : *Point Of Sale, Waterfall, Visual Basic.*

ABSTRACT

APPLICATION AND DESIGN OF POINT OF SALE APPLICATION WITH WATERFALL METHOD IN TAZMA OPTIC

By
Ryan Ramadhan Harahap
361762001

The purpose of writing this thesis report is to facilitate the data collection of goods and make transactions at Optik Tazma as a solution to improve the road sales process more effectively and efficiently, as well as facilitate the work of employees related toservice, making it easier for optical owners to check and evaluation of revenue from sales. The research method used by the author in conducting research includes field studies and literature studies and the use of methods as application and application design with theprogramming language Visual Basic. Field studies include observation and analysis of the ongoing system. Literature study is carried out with literature research that is relevant to the problems faced by the author, the design and development method uses themethod waterfall. From the analysis and observations made it can be seen that the making of the Sales Application Point Of Sale (POS) on Optik Tazma can help facilitate the work of other employees and make it easier to plan future sales targets so that they continue to increase.

Keywords : Point Of Sale, Waterfall, Visual Basic.

KATA PENGANTAR

Puji dan syukur kita panjatkan kehadirat Allah SWT. yang telah memberikan penulis rahmat dan karunia untuk menyelesaikan skripsi yang berjudul "**Penerapan Dan Perancangan Aplikasi *Point Of Sale* Dengan Metode *Waterfall* Di Optik Tazma**". Penyusunan skripsi ini bertujuan untuk memenuhi persyaratan kelulusan pada Program Studi Teknik Informatika di STMIK-IM Bandung.

Tentunya dalam proses penyusunan skripsi ini, penulis mengalami beberapa kesulitan, baik dari situasi, kemampuan dan waktu yang dimiliki. Dikarenakan keterbatasan kemampuan yang dimiliki, penulis sangat mengapresiasi dan sangat berterima kasih kepada pihak-pihak yang membantu penulis dalam melancarkan skripsi ini. Bantuan yang penulis dapatkan dari pihak-pihak tersebut bermacam-macam, dimulai dari artikel internet, buku-buku, jurnal yang di perpustakaan maupun internet, dan referensi program yang ada.

Penulis sangatlah menyadari bahwa skripsi ini masih memiliki banyak kekurangan, serta masih jauh dari kata sempurna. Oleh karena itu, penulis sangatlah mengharapkan kritik dan saran yang dapat membangun penulis, guna pembelajaran di kemudian hari.

Penulis sangat berharap, semoga skripsi ini dapat memberikan wawasan lebih, serta manfaat untuk pembaca khususnya untuk adik tingkat yang rekan-rekan yang mencari referensi untuk menyusun skripsi dikemudian hari.

Bandung, Oktober 2020

Ryan Ramadhan Harahap
361762001

UCAPAN TERIMA KASIH

Pada kesempatan kali ini, penulis menyampaikan rasa berterima kasih kepada rekan–rekan, pembimbing, serta pihak yang ikut serta membantu penulis dalam penyusunan skripsi ini, antara lain :

1. Bapak Patah Herwanto, S.T., M.Kom. selaku pembimbing yang telah memberikan penulis ilmu dan masukan dalam penyusunan skripsi.
2. Pemilik Toko Tazma yang telah memberikan kesempatan penulis untuk membuat program system penjualan guna pendukung kelancaran penyusunan skripsi.
3. Bapak Dr. Chairuddin, Ir., M.M., M.T. selaku Ketua STMIK-IM Bandung yang telah memberikan ilmu kepada penulis.
4. Ibu Chalifa Chazar, S.T., M.T. selaku Ketua Program Studi Sistem Informasi.
5. Kedua orang tua penulis, Ibu dan Bapak yang selalu memberikan dukungan penuh untuk penulis.
6. Bapak & Ibu dosen yang telah memberikan penulis ilmu selama penulis melakukan kegiatan belajar mengajar di STMIK-IM Bandung.
7. Rekan–rekan di STMIK-IM Bandung maupun di Optik Tazma.
8. Kepala dan seluruh Star Administrasi, BAAK, BAUKe, Perpustakaan, dan Karyawan STMIK – Indonesia Mandiri Bandung.

9. Kedua orang tua tercinta, Ibu Ipih Sopiah dan Bapak Aslidana Harahap yang selalu mendoakan, memberikan semangat, dan selalu memberikan kasih sayangnya kepada penulis.
10. Anggi Fauziyah yang tidak pernah berhenti memberi semangat, selalu membantu, berkorban dan mendukung selama proses pengerjaan skripsi ini.
11. Teman teman satu perjuangan di STMIK – Indonesia Mandiri Bandung, yang selalu memberikan semangat dan saling membantu.
12. Bapak Rizki Kurnia yang selalu membantu dalam penyelesaian program aplikasi yang di buat oleh penulis.
13. Rekan – rekan di PT. Lintas Mediatama yang selalu membantu dan terus memberikan semangat.
14. Seluruh pihak yang tidak dapat disebutkan Namanya satu per satu, yang telah memberikan bantuan baik moril maupun material kepada penulis dalam penyusunan laporan skripsi ini.

Semoga semua amal baik yang telah diberikan akan menjadi pahala dan mendapat balasan berlipat dari Allah Subhanahu Wa Ta'ala. Aamiin.

Harapan penulis semoga skripsi ini dapat bermanfaat bagi para pembaca sekalian.

Bandung, Oktober 2020
Pembuat pernyataan,

Ryan Ramadhan Harahap
361762001

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1. Latar Belakang

Teknologi informasi dalam pemanfaatan teknologi komputer untuk pengolahan data juga menjadi perhatian di perusahaan besar. Berbagai kegiatan perusahaan bisa dikembangkan menjadi sistem yang kinerjanya menggunakan perangkat komputer, seperti pengolahan data penjualan barang, pembelian barang dari vendor, serta pengelolaan data barang di gudang (Cahyodi dan Arifin, 2017).

Pelayanan dengan sistem pencatatan manual membutuhkan waktu yang lama sehingga berjalan tidak efektif, selain itu juga ada kemungkinan terjadi kesalahan informasi, penulisan dalam penjualan, pencatatan barang dan rugi laba yang dihasilkan. Oleh karena itu dalam sebuah perusahaan harus memiliki sistem yang terotomatisasi sehingga akan menjadi lebih efektif dan mengurangi terjadi kesalahan informasi, pencatatan penjualan dan rugi laba perusahaan (Yuarita dan Marisa, 2017).

Aplikasi transaksi *Point Of Sale* (POS) adalah sebuah sistem aplikasi yang terdiri dari *hardware* dan *software* yang didesain sesuai dengan keperluan dan dapat diintegrasikan dengan berbagai alat pendukung agar dapat membantu mempercepat proses transaksi. Dalam lingkup *Point Of Sale* (POS), sebuah mesin kasir tidak berdiri sendiri, namun sudah termasuk *software* penunjang dan piranti lain. Sistem *Point Of Sale* (POS) melakukan lebih dari sekedar tugas transaksi jual beli,

didalamnya bisa terintegrasi juga perhitungan akuntansi, manajemen barang dan stok, laporan laba rugi dalam jangka waktu mingguan dan bulanan atau dalam jangka waktu tertentu sesuai dengan kebutuhan dari pemakai (Sugianto dan Tjandra, 2016).

Adapun tujuan dari penelitian ini yaitu melakukan analisis dan perancangan aplikasi *Point Of Sale* (POS) dengan metode *Waterfall* untuk mendukung sistem layanan pembelian yang dapat membantu para pengusaha kecil dan menengah dalam pengelolaan data. Pembuatan aplikasi POS ini dimulai dari pengumpulan seluruh data-data yang dibutuhkan dengan menggunakan metode observasi dan wawancara, perancangan model aplikasi dengan pendekatan diagram berbasis obyek dengan alat bantu perancangan aplikasi berupa diagram alur atau flowchart dan *Unified Modeling Language* (UML) serta penerapan metode *Waterfall*, hingga diimplementasikannya aplikasi POS ini. Dengan diterapkannya aplikasi *Point Of Sale* (POS) ini dapat membantu tugas pihak-pihak terkait atau seluruh stake-holder yang berhubungan langsung dengan aplikasi *Point Of Sale* (POS) ini (Permana dan Faisal, 2015).

Berdasarkan penjelasan latar belakang di atas maka peneliti tertarik untuk melakukan penelitian dengan judul "**Penerapan Dan Perancangan Aplikasi *Point Of Sale* Dengan Metode *Waterfall* Di Optik Tazma**".

1.2. Identifikasi Masalah

Untuk menyelesaikan masalah yang akan di bahas selanjutnya, maka diperlukan identifikasi masalah sehingga hasil penelitian dapat terarah dan sesuai dengan tujuan penelitian, dan latar belakang yang sebelumnya telah di paparkan dapat diidentifikasi permasalahan-permasalahan yang muncul dalam penelitian :

1. Bagaimana cara agar pendataan barang bisa dilakukan dengan pencatatan yang tepat dan lebih efektif ?
2. Bagaimana cara agar pelayanan terhadap konsumen bisa lebih cepat serta efsien ?
3. Bagaimana cara agar pendapatan serta pengeluaran uang bisa tercatat secara detail ?

1.3. Tujuan Penulisan

Tujuan dari penelitian ini sesuai dengan rumusan masalah sebagai berikut :

1. Untuk mengetahui barang apa saja yang masuk serta barang apa saja yang keluar dalam kurun waktu harian, mingguan, bulanan serta tahunan.
2. Untuk mempercepat proses pelayanan terhadap konsumen maka di butuhkan pencatatan serta pencetakan bukti pembayaran dengan cepat dan efisien.
3. Untuk bisa mengetahui pendatapan dari hasil penjualan dengan lebih mudah serta dapat melakukan evaluasi penjualan barang untuk kedepan nya.

1.4. Batasan Masalah

Mengingat begitu luasnya ruang lingkup penelitian ini maka penulis membatasi permasalahan tersebut pada :

1. Mengingat banyaknya program aplikasi yang di buat dengan berbagai bahasa pemrograman, maka penulis pada penelitian ini menggunakan bahasa pemrograman *Visual Basic* dan merancang program tersebut berbasis *Desktop*.
2. Banyak sekali aplikasi-aplikasi yang dapat berjalan dengan sangat efektif di beberapa *Operating System*, maka dalam hal ini penulis hanya menjalankan program tersebut di perangkat yang menggunakan *Operating System Windows.*
3. Alat bantu yang digunakan untuk bisa berjalan nya aplikasi ini serta dapat membuat suatu informasi yang dapat diolah, ditampilkan, dan dimanipulasi sehingga bisa menyajikan suatu informasi yang lebih detail, maka diperlukan suatu perangkat keras *Personal Computer* serta perangkat lunak *Operating System Windows.*
4. *Software* yang akan di gunakan dalam pembuatan aplikasi ini yaitu menggunakan *software Visual Studio* yang memungkinkan untuk merancang interface dengan lebih efektif dan mudah di operasikan oleh petugas-petugas di Optik Tazma.
5. Aplikasi ini bisa berjalan efektif hanya di Optik Tazma berdasarkan atas kebutuhan dan kondisi yang terjadi di Optik Tazma.

1.5. Metodologi Penelitian

1.5.1. Metode Pengumpulan Data

Metode penelitian yang dilakukan oleh penulis dalam pengumpulan data antara lain :

1. Observasi

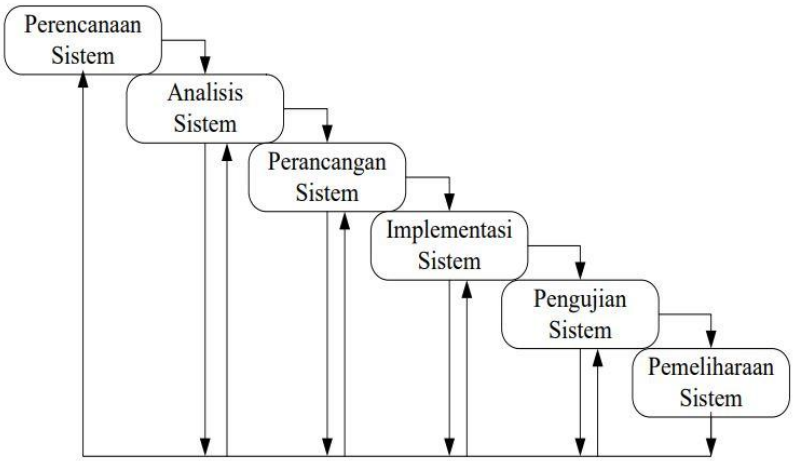
Observasi adalah pengamatan langsung bagaimana alur proses pendataan barang-barang di Optik Tazma, mulai dari data barang masuk, data barang terjual, *stock* barang serta rugi dan laba yang di dapat.

2. Penelitian Kepustakaan

Penelitian yang bertujuan untuk mengumpulkan data-data yang mendukung serta mempunyai kaitan dengan penelitian ini yang bersifat teoritis dengan cara membaca alur proses pengumpulan data yang sedang berjalan.

1.5.2. Metode Perancangan Sistem

Metode *Waterfall* adalah suatu proses pengembangan perangkat lunak berurutan, di mana kemajuan dipandang sebagai terus mengalir ke bawah (seperti air terjun) melewati fase-fase perencanaan, pemodelan, implementasi (konstruksi), dan pengujian. Dalam pengembangannya metode *waterfall* memiliki beberapa tahapan yang runtut: *requirement* (analisis kebutuhan), *design* sistem (*system design*), *Coding And Testing*, Penerapan Program, dan Pemeliharaan (Tristianto, 2018).



GAMBAR: 1. 1 Tahapan Metode *Waterfall* (Tristianto, 2018:13)

1. Perencanaan Sistem (*Project Definition*)

   Permodelan ini diawali dengan mencari kebutuhan dari keseluruhan sistem yang akan diaplikasikan ke dalam bentuk *software*. Hal ini sangat penting, mengingat *software* harus dapat berinteraksi dengan elemen-elemen yang lain seperti *hardware*, *database*, dan yang lainnya.

2. Analisis Kebutuhan (*Requirement*)

   Dalam langakah ini merupakan analisa terhadap kebutuhan sistem. Pengumpulan data dalam tahap ini bisa melakukan sebuah penelitian, wawancara atau *study* literatur. Seseorang *system* analisis akan menggali informasi sebanyak-banyaknya dari *user* sehingga akan tercipta sebuah sistem komputer yang bisa melakukan tugas-tugas yang diinginkan oleh *user* tersebut. Tahapan ini akan menghasilkan dokumen *user requirement* atau bisa dikatakan sebagai data yang berhubungan dengan keinginan user dalam

pembuatan sistem. Dokumen inilah yang akan menjadi acuan *system* analisis untuk menterjemahkan kedalam bahasa pemrograman.

3. Desain Sistem (*Design System*)

   Proses *design* akan menterjemahkan syarat kebutuhan kesebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat koding. Proses ini berfokus pada : struktur data, arsitektur perangkat lunak, representasi *interface*, dan detail (algoritma) prosedural. Tahapan ini akan menghasilkan dokumen yang disebut *software requirement*. Dokumen inilah yang akan digunakan programmer untuk melakukan aktivitas pembuatan sistemnya.

4. Penulisan Kode Program Dan Implementasi (*Coding And Testing*)

   Coding merupakan penerjemahan *design* dalam bahasa yang bisa dikenali oleh komputer. Dilakukan oleh *programmer* yang akan meterjemahkan transaksi yang diminta oleh *user*. Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu sistem. Dalam artian penggunaan *computer* akan dimaksimalkan dalam tahapan ini. Setelah pengkodean selesai maka akan dilakukan *testing* terhadap sistem yang telah dibuat tadi. Tujuan *testing* adalah menemukan kesalahan-kesalahan terhadap *system* tersebut dan kemudian bisa diperbaiki.

5. Penerapan Dan Pengujian Program (*Integration And Testing*)

   Tahapan ini bisa dikatakan final dalam pembuatan sebuah sistem. Setelah melakukan analisa, *design* dan pengkodean maka sistem yang sudah jadikan digunakan oleh *user*.

6. Pemeliharaan (*Operation And Maintenance*)

   Perangkat lunak yang susah disampaikan kepada pelanggan pasti akan mengalami perubahan. Perubahan tersebut bisa karena mengalami kesalahan karena perangkat lunak harus menyesuaikan dengan lingkungan (*Pheriperal* atau *system* operasi baru) baru, atau karena pelanggan membutuhkan perkembangan fungsional.

1.6. Ruang Lingkup Penelitian

Ruang lingkup pada penlitian ini dilakukan di Optik Tazma yang terletak di Jalan Alkateri (ABC) Bandung, dengan melakukan *review* serta wawancara terhadap pemilik Optik Tazma serta peninjauan terhadap alur dan proses kerja yang berjalan di Optik Tazma.

1.7. Sistematika Penulisan

Sistematika penulisan menjelaskan secara ringkas isi dari setiap bagian-bagian yang ada pada penelitian ini. Adapun sistematika penulisan laporan tugas akhir ini, sebagai berikut :

BAB I PENDAHULUAN

Dalam bab ini terdiri dari latar belakang, identifikasi masalah, tujuan penelitian, batasan masalah, dan metodologi penelitian.

BAB II LANDASAN TEORI

Dalam bab ini menguraikan latar belakang Optik Tazma yang meliputi aktivitas, visi dan misi, serta berisi tentang teori-teori yang berkaitan dengan isi penelitian ini dan pendukung dalam pemecahan masalah yang di anggap relevan.

BAB III ANALISA MASALAH DAN PERANCANGAN PROGRAM

Pada bab ini penulis menguraikan tentang pengamatan dan pembahasan dari aplikasi yang dibuat serta menganalisa masalah yang terjadi baik dari segi pembuatan program serta alur proses dalam program tersebut.

BAB IV IMPLEMENTASI DAN UJI COBA

Pada bab ini penulis menjelaskan serta melakukan uji coba pada aplikasi yang sudah di buat, dengan mengimplementasikan langsung aplikasi tersebut serta melakukan uji coba kelayakan dalam mendata barang-barang yang akan di jual dan laporan rugi dan laba dari hasil pejualan.

BAB V PENUTUP

Pada bab ini penulis mengungkapkan kesimpulan dari topik yang penulis paparkan didalam penelitian ini dan saran-saran yang mungkin di perlukan dalam pengembangan ilmu pengetahuan.

BAB II

LANDASAN TEORI

2.1. Aplikasi (*Application*)

Aplikasi merupakan perangkat lunak yang berfungsi untuk melakukan berbagai pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan dan penambahan data. Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau *suite* aplikasi (*application suite*). *Visual Studio* dan *Notepad* sebagai sarana pembuatan suat aplikasi dengen memasukan suatu perintah atau *syntax*. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi. Sering kali, mereka memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. Contohnya, suatu lembar kerja dapat disimpan dalam suatu dokumen pengolah kata walaupun dibuat pada aplikasi lembar kerja yang terpisah. (Koyuko dkk, 2016).

Secara istilah pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanankan suatu fungsi bagi pengguna jasa aplikasi serta penggunaan aplikasi lain yang dapat digunakan oleh suatu sasaran yang akan dituju, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemrosesan data yang di harapkan. Pengertian aplikasi menurut Kamus Besar Bahasa Indonesia,

“Aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa program tertentu”. (Juansyah, 2015).

2.2. *Waterfall Method*

Metode *Waterfall* adalah suatu proses pengembangan perangkat lunak berurutan, di mana kemajuan dipandang sebagai terus mengalir ke bawah (seperti air terjun) melewati fase-fase perencanaan, pemodelan, implementasi (konstruksi), dan pengujian. Dalam pengembangannya metode *waterfall* memiliki beberapa tahapan yang runtut: *requirement* (analisis kebutuhan), *design* sistem (*system design*), *Coding And Testing*, Penerapan Program, dan Pemeliharaan (Tristianto, 2018).

2.3. *Point Of Sale* (POS)

Sistem *Point of Sale* (POS) merupakan sebuah sistem untuk menyinkronkan dan mengintegrasikan data reservasi, data pesanan, data *e-commerce*, data kartu hadiah, atau data poin loyalitas yang berada di perangkat *Point Of Sale* (POS) dengan perangkat situs web pedagang dan menyinkronkan data yang berada di basis data situs web ke perangkat *Point Of Sale* (POS) terkait. Sistem *Point Of Sale* (POS) digunakan untuk mengintegrasikan sistem reservasi, pemesanan, dan sistem merchant *e-commerce* lainya yang disediakan melalui situs web. Sitem *Point Of Sale* (POS) mencakup perangkat, *server website*, lapisan basis data, aplikasi situs web *Point Of Sale* (POS). Adapun publikasi ilmiah yang berjudul *Website Based Point of Sale System* atau disingkat WPOS berupa sistem berbasis *website* yang

memungkinkan manajemen laporan secara jarak jauh, dan memungkinkan pelanggan melakukan penjadwalan atau penjadwalan ulang waktu pengiriman. *Server* menyediakan semua data dan informasi penting yang dibutuhkan melalui *web browser* pelanggan. dalam sistem ini, *server* antar toko dapat berkomunikasi satu sama lain dengan *mainframe* kantor pusat. WPOS dapat diimplementasikan sebagai rangkaian terintegrasi untuk kolaborasi antar lokasi (Abidin dan Putro, 2020).

Analisa dan perancangan Aplikasi *Point Of Sale* (POS) yang baik adalah aplikasi yang mencakup semua s*cope* atau ruang lingkup yang ditentukan. Tentu saja aspek integrasi antara bagian dalam suatu aplikasi POS sangatlah penting, dimana data dalam aplikasi POS saling berinteraksi dalam meningkatkan kecepatan, akurasi dan kemudahan.

GAMBAR: 2. 1 Kerangka Konsep Aplikasi POS (Permana dan Faisal, 2015:22)

Gambar 2.2 diatas menujukkan kerangka konsep analisa dan perancangan aplikasi *Point Of Sale* (POS). Kerangka konsep ini menggambarkan bagaimana proses aplikasi POS ini dibuat. Pembuatan aplikasi *Point Of Sale* (POS) ini dimulai

dari pengumpulan data dengan observasi dan wawancara hingga diimplementasikannya aplikasi *Point Of Sale* (POS) ini. Dengan adanya proses yang baik inilah diharapkan aplikasi *Point Of Sale* (POS) yang dibangun tepat guna dan dapat mengakomodasi seluruh kegiatan atau proses bisnis yang terjadi (Permana dan Faisal, 2015).

2.3.1. Konsep *Point Of Sale*

Skema pemetaan hubungan *input*, *process* dan *output* dapat terlihat pada gambar dibawah ini. *Input* yaitu data yang ada yang akan dimasukkan kedalam aplikasi, *Process* yaitu hal yang sedang berlangsung dan berkaitan dengan bisnis proses yang ada dan *Output* yang meliputi laporan dari setiap kegiatan yang ada.

GAMBAR: 2. 2 Skema Aplikasi *Point Of Sale* (Permana dan Faisal, 2015:22)

Dari gambar diatas, dapat dilihat antara *input*, *process*, dan *output* saling berinteraksi. Hal tersebut didapatkan tentunya dalam pengumpulan kebutuhan sistem (*requirement gathering*) melalui hasil observasi dan wawancara (Permana dan Faisal, 2015).

2.4. *Visual Studio*

Merupakan sebuah perangkat lunak lengkap (*Suite*) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi *console*, aplikasi *Windows*, ataupun *Web*. *Visual Studio* mencakup kompiler, SDK, *Integrated Development Environment* (IDE), dan dokumentasi (umumnya berupa MSDN *Library*). Adapun beberapa kompiler yang dimasukkan ke dalam paket *Visual Studio* antara lain *Visual* C++, *Visual* C#, *Visual Basic*, *Visual Basic* NET, *Visual InterDev*, *Visual* J++, *Visual* J#, *Visual FoxPro*, dan *Visual SourceSafe*. *Microsoft Visual Studio* dapat digunakan untuk mengembangkan aplikasi dalam *native code* (dalam bentuk bahasa mesin yang berjalan di atas *Windows*) ataupun *managed code* (dalam bentuk *Microsoft Intermediate Language* di atas .NET *Framework*). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi *Silverlight*, aplikasi *Windows Mobile* (yang berjalan di atas .NET *Compact Framework*) (Wikipedia Indonesia, Diakses 10 Agustus 2020).

2.5. *Open Database Connectivity* (ODBC)

Open Database Connectivity yaitu seperangkat fungsi untuk melakukan koneksi database secara local maupun remote, ODBC mempermudah koneksi aplikasi kebeberapa database yang formatnya berbeda, misalnya format *Database* Ms. *FoxPro*, Ms. *Access*, Ms. SQL *Server*, My SQL ataupun *Oracle* (Nopriansyah, 2019).

Standar *Open Database Connectivity* (ODBC) adalah sebuah *interface* dimana program-program aplikasi dapat mengakses dan memproses *Database* SQL yang tidak bergantung pada aturan DBMS. Tujuan awal *Microsoft* dalam mendukung standar ODBC adalah untuk memungkinkan produk-produk seperti *Microsoft Excel* untuk mengakses *database* dari berbagai macam produk DBMS tanpa harus dikompilasi ulang (Saputra dan Riyadi, 2016).

2.6. *Database* (MySQL)

MySQL merupakan sebuah *Database Developer* yang juga bersifat *free*, MySQL banyak digunakan sebagai *database* karena mudah digunakan dan juga sangat banyak tersedia. MySQL sendiri menggunakan bahasa SQL yang saat ini sudah banyak digunakan. MySQL merupakan *software database* yang termasuk paling populer di lingkungan *Linux* atau *Unix*, kepopuleran ini ditunjang karena *query* dari basis data yang saat itu bisa dikatakan paling cepat dan juga memiliki sedikit permasalahan (Wiguna dkk., 2018).

Beberapa keunggulan MySQL dibandingkan database lain :

a. Kemudahan dalam penggunaan MySQL adalah *Simple Database System* dengan performa tinggi dan tidak kompleks untuk proses instalasi dan *administrator*-nya dibanding dengan sistem yang lebih besar.
b. Mendukung bahasa *query* MySQL dapat menggunakan SQL, juga dapat diakses dengan menggunakan aplikasi ODBC.
c. Kemampuan banyak *Client* dapat berhubungan dengan *Server* pada saat bersamaan. *Clients* dapat menggunakan *Multiple Database* secara bersamaan.

2.7. *Visual Basic*

Visual Basic merupakan salah satu dari sekian banyak bahasa pemrograman. Basis dari *Visual Basic* adalah pemrograman yang bersifat grafis. Perbedaan yang jelas antara program text dan grafis adalah pada program grafis, orientasinya pada obyek. Obyek bisa didefinisikan sebagai suatu benda yang mempunyai "properti/atribut" dan "kejadian atau *event*". Dalam *Visual Basic*, sama juga seperti benda lain, misalnya tombol mempunyai atribut : tinggi, lebar, warna, tulisan dan lain-lain. Kejadian yang berhubungan misalnya *click*, *gotfocus*, dan lain-lain. Kita nantinya membuat prosedur atau fungsi untuk tiap kejadian pada tiap obyek yang terlibat dalamaplikasi kita. Pembuatan program secara *visual* biasanya dibentuk dalam proyek. Proyek ialah kumpulan dari *form*, *module* dan kontrol-kontrol yang membentuk program aplikasi. Setiap membuka *visual basic* secara otomatis *visual basic* membuat obyek baru (Wahyudi dkk., 2018).

2.8. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma 'berorientasi objek" (Wati dan Kusumo, 2016).

Tujuan penggunaan UML adalah :

a. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemrograman dan proses rekayasa.
b. Menyatukan praktek-praktek terbaik yang terdapat dalam pemodelan.

c. Memberikan model yang siap pakai, bahsa pemodelan *visual* yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum.

Unified Modeling Language (UML) bisa juga berfungsi sebagai sebuah (*blueprint*) cetak biru karena sangat lengkap dan *detail*. Dengan cetak biru ini maka akan bias diketahui informasi secara *detail* tentang koding program atau bahkan membaca program dan menginterpretasikan kembali ke dalam bentuk diagram (*reserve enginering*).

Kesembilan jenis diagram dalam *Unified Modeling Language* (UML) yang dapat digunakan yaitu :

a. *Use case* Diagram

   Diagram ini bersifat statis yang memperlihatkan himpunan *use case* dan aktor-aktor.

b. *Activity* Diagram

   Diagram ini bersifat dinamis, tipe khusus dari diagram ini yaitu *state* yang memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dalam suatu sistem.

c. *Sequence* Diagram

   Diagram ini bersifat dinamis, diagram ini merupakan diagram urutan yang memperlihatkan interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu.

d. *Class* Diagram

Diagram ini bersifat statis yang memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi.

e. *Statechart* diagram

Diagram ini bersifat dinamis, pada diagram ini memperlihatkan *state-state* pada sistem yang memuat *state*, *transition*, *event*, serta aktifitas.

f. *Object* Diagram

Diagram ini bersifat statis yang memperlihatkan objek-objek serta relasi-relasi antar objek. *Object* diagram memperlihatkan instansi statis dari segala sesuatu yang dijumpai pada *class* diagram.

g. *Collaboration* Diagram

Diagram ini bersifat dinamis. Diagram ini diagram interaksi yang menekankan organisasi struktural dari objek-objek yang menerima serta mengirim pesan (*message*).

h. *Component* diagram

Diagram ini bersifat statis yang memperlihatkan organisasi serta kebergantungan pada komponen-komponen yang telah ada sebelumnya.

i. *Deployment* Diagram

Diagram ini bersifat statis yang memperlihatkan konfigurasi saat aplikasi dijalankan (saat *run-time*). Diagram ini memuat simpul-simpul (*node*) beserta komponen-komponen yang ada di dalamnya.

2.9. Studi Literatur Sejenis

Berikut ini adalah tabel studi literatur sejenis yang berkaitan dengan penelitian yang di lakukan oleh penulis.

TABEL: 2. 1 Tabel Penelitian Terkait

No.	Penulis	Judul	Masalah	Metode	Hasil
1.	Wahyudi dkk., (2018).	Aplikasi Penjualan *Point Of Sale* (POS) Menggunakan *Barcode* Pada Koperasi Bina Kasih Sejahtera Berbasis *Desktop* Dengan Metode *Firt In First Out* (FIFO)	1. Pencatatan manual. 2. Data kurang akurat.	1. Metode *Waterfall*. 2. Metode *Deskriptif*.	1. Aplikasi *Point Of Sale* Berbasis *Desktop*.
2.	Permana, S.D.H. dan Faisal, (2015).	Analisa Dan Perancangan Aplikasi *Point Of Sale* (POS) Untuk Mendukung Manajemen Hubungan Pelanggan.	1. Penyajian informasi yang kurang baik. 2. Informasi dan pnyajian terbatas.	1. Observasi. 2. Wawancara. 3. *Flowchart* dan *Unified Modeling Language* (UML).	1. Aplikasi *Point Of Sale* (POS)
3.	Wiguna dkk., (2018).	Rancang Bangun Aplikasi *Point of Sale Distro Management System* dengan Menggunakan *Framework React Native*.	1. Semua proses pendataan yang dilakukan masih manual dengan menggunakan buku tulis. 2. kesulitan dalam	1. Metode *Classic Life Cycle* (CLC). 2. *Metode Black Box*. 3. Pembuatan aplikasi dengan *Framework*	1. Aplikasi *Point Of Sale Distro Management System*.

TABEL: 2.1 Tabel Penelitian Terkait (Lanjutan)

No.	Penulis	Judul	Masalah	Metode	Hasil
			manajemen dan pengecekan stok produk. 3. kesulitan dalam pencatatan transaksi penjualan, kesulitan dalam manajemen pembayaran *clothing* dan lain-lainnya.	*React Native*.	
4.	Abidin dan Putro, (2020).	Penerapan MVC Dalam Pengembangan Sistem *Point Of Sale*.	1. Ketidaksa-maan dalam aturan penulisan kode.	1. *Model View Controller* (MVC).	1. Dokumen-tasi. 2. Model Dan Struktur Program Baru.
5.	Sri Rahayu dkk., (2017).	Aplikasi *Point Of Sale* Berbasis Web Menggunakan *Framework Codeigniter* Pada Martabak ABC.	1. Pencatanan nota secara manual. 2. Sistem Pembayaran belum terkomputasi.	1. Metode *Waterfall.* 2. Metode *Blackbox Testing*.	1. Kemudahan dalam bertransaksi. 2. Pencatatan penjualan yang efektif. 3. Terbentuk Aplikasi *Point Of Sale*.

BAB III

ANALISA MASALAH DAN PERANCANGAN PROGRAM

3.1. Perencanaan Sistem *(Project Definition)*

Proses ini ialah tahapan awal dari metode *Waterfall*. Pada tahap ini dilakukan analisa serta perencanaan terhadap proses-proses yang akan berjalan dalam Aplikasi Penjualan atau POS, yaitu :

1. *Login* akses.
2. Proses *input* barang.
3. Proses *input* data pelanggan.
4. Proses transaksi atau penjualan barang.
5. Proses cetak *report* data barang, data pelanggan dan data transaksi.

GAMBAR: 3. 1 *Mapping* Aplikasi POS

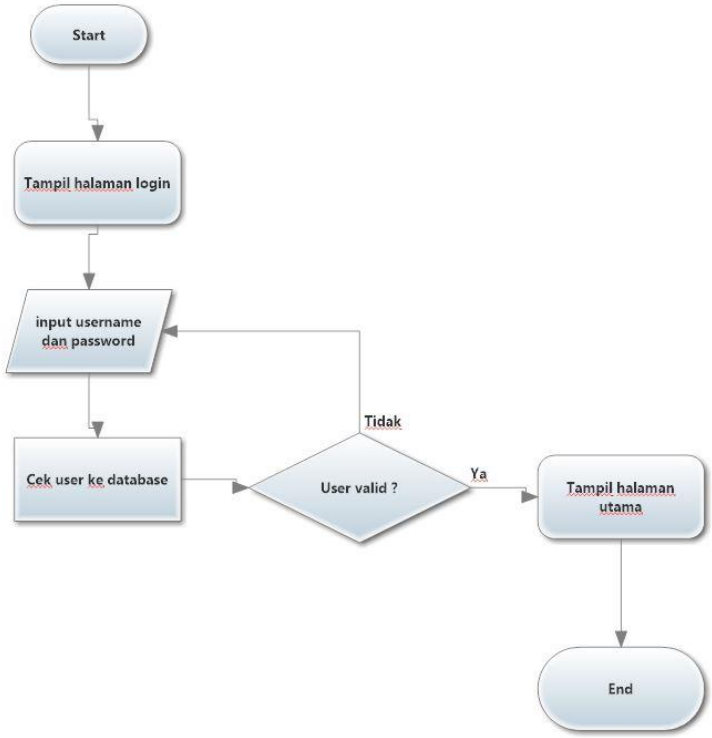
Berikut *detail* proses-proses yang akan berjalan dalam aplikasi penjualan (POS), adalah sebagai berikut :

1. *Login* akses.
    - *Input* username.
    - *Input* password.

2. Proses *input* barang
    - Id barang.
    - Nama barang.
    - Nama brand.
    - Tanggal masuk barang.
    - *Barcode*.
    - Jumlah barang.
    - Satuan barang.
    - Harga barang.
3. Proses *input* data pelanggan
    - *Id* pelanggan.
    - Nama pelanggan.
    - No telepon pelanggan.
    - Alamat *e-mail* pelanggan.
4. Proses transaksi atau penjualan barang.
    - Nomor transaksi.
    - Data pelanggan.
    - Tanggal jual barang.
    - Jam jual barang.
    - Total *item* atau total barang yang terjual.
    - Diskon harga barang.
    - Harga jual barang.
    - Jumlah uang yang dibayarkan.

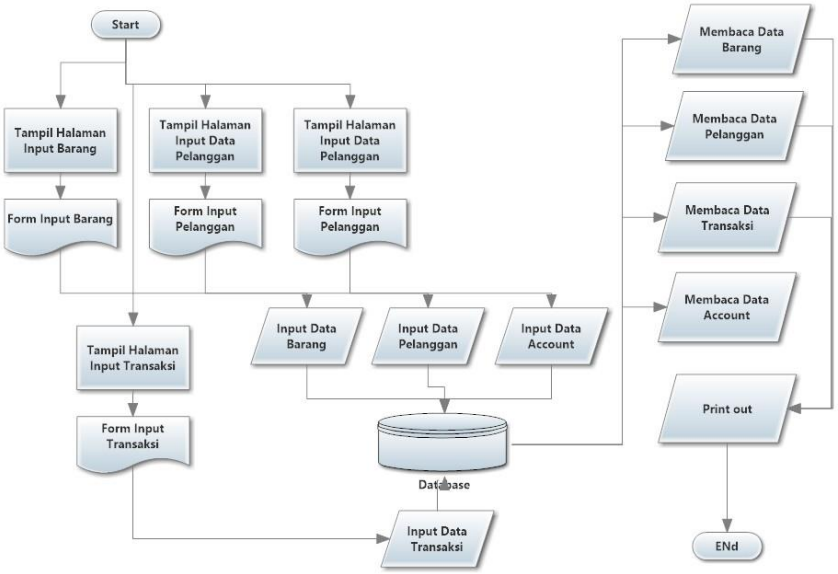
- Jumlah uang Kembali.

5. Proses cetak report data barang, data pelanggan, dan data transaksi.
    - *Print* jumlah data barang yang tersedia.
    - *Print* jumlah data pelanggan.
    - *Print* data transaksi secara berkala atau sesuai tanggal yang di tentukan.



GAMBAR: 3. 2 *Flowchart Login*

Gambar diatas merupakan alur dari pengguna (*user*) aplikasi POS melakukan login, pada halaman *login*, *user* memasukan *username* dan *password* kemudian *system* akan membaca ke *database* jika data yang di *input* oleh *user* sesuai dengan data yang ada di *database* maka akan berpindah ke halaman utama jika data tidak sesuai maka *user* harus memasukan data *username* dan *password* yang *valid*.



GAMBAR: 3. 3 *Flowchart* Aplikasi Penjualan (POS)

Gambar diatas menunjukan *user* yang sudah melakukan proses *login* akan masuk ke halaman utama, dimana *user* bisa masuk ke halaman data barang, data pelanggan, data transaksi, dan data *account*, selain itu *user* bisa melakukan *input*, *edit*, dan *delete* terhadap data barang, data pelanggan, dan data *account* namun untuk data transaksi *user* hanya bisa melakukan *input* saja, dan *user* juga bisa melakukan *print report* dari data barang, data pelanggan dan data transaksi.

3.2. Analisa Kebutuhan (*Requirement*)

Proses ini dilakukan untuk mendapatkan daftar kebutuhan-kebutuhan yang akan digunakan baik perangkat keras (*hardware*) maupun perangkat lunak (*software*), diantaranya :

1. Komputer

   Komputer digunakan untuk jalannya proses pembuatan aplikasi POS.

2. Aplikasi XAMPP.

   Aplikasi XAMPP digunakan untuk merangcang basis data yang akan di terapkan.

3. Aplikasi *Visual Studio*.

   Aplikasi *Visual Studio* digunakan untuk merancang serta membuat *coding* terhadap aplikasi yang akan dibuat

4. Aplikasi *Crystal Report*.

   Aplikasi *Crystal Report* digunakan untuk *review* hasil *report* barang, pelanggan, dan hasil transaksi dan untuk proses *print report*.

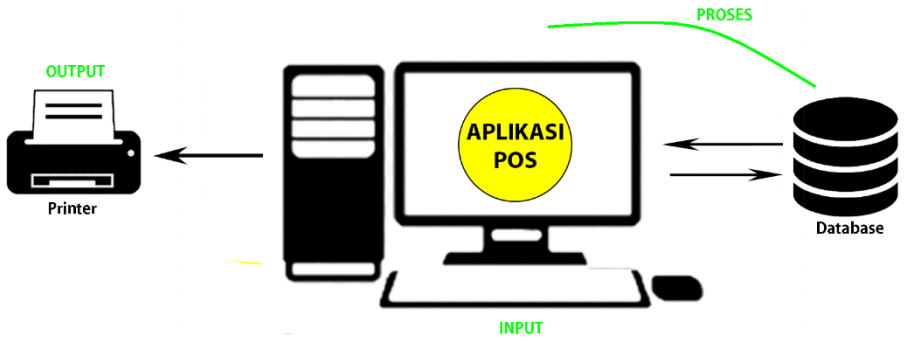
5. Aplikasi ODBC *Connector*

   Aplikasi ODBC *Connector* digunakan sebagai penghubung antar aplikasi *Visual Stuido* dengan basis data di aplikasi XAMPP.

6. Data-data hasil observasi

   Data-data serta parameter-parameter yang di dapatkan ketika observasi di masukan ke dalam sistem yang akan di buat

Permasalahan-permasalah yang terdapat pada tahap sebelumnya sudah dianalisa dan selanjutnya akan dilanjutkan dalam perancangan arsitektur aplikasi POS. Berikut adalah perancangan terhadap *system* yang akan di buat berdasarkan *input, proses* dan *output*.



GAMBAR: 3. 4 *Architecture* Aplikasi POS

Dimana user melakukan *input*-an data terhadap aplikasi penjualan (POS) kemudian system di aplikasi akan memproses ke *database*, dari *database* mengirimkan kembali data tersebut ke dalam aplikasi dan selanjutkan *system* akan memproses pencetakan ke *printer*.

3.3. Desain Sistem *(Design System)*

3.3.1. Desain Sistem Dengan *Unified Modeling Language (UML)*

Perancangan yang dilakukan oleh penulis terhadap alur sistem yang dibuat akan di implementasikan menggunakan UML yang terdiri dari use case diagram, activity diagram, sequence diagram dan class diagram.

3.3.1.1. *Use Case* Diagram

Pada use case diagram ini menjelaskan interaksi antara sistem dengan helpdesk atau manger yang terlibat dengan sistem.

1. Definisi Aktor

TABEL: 3. 1 Tabel Definisi Aktor

No.	Aktor	Deskripsi
1.	Penanggung Jawab Optik	*Input* barang, *edit* barang, *delete* barang, *input* pelanggan, *edit* pelanggan, *delete* pelanggan, *input* transaksi, *input account*, *edit account*, *delete account*, *view* dan *print report* barang, pelanggan, dan transaksi.
2.	*Admin*	*Input* barang, *input* pelanggan, *input* transaksi, *view* dan *print report* barang, pelanggan, dan transaksi.

2. Definisi *Use Case*

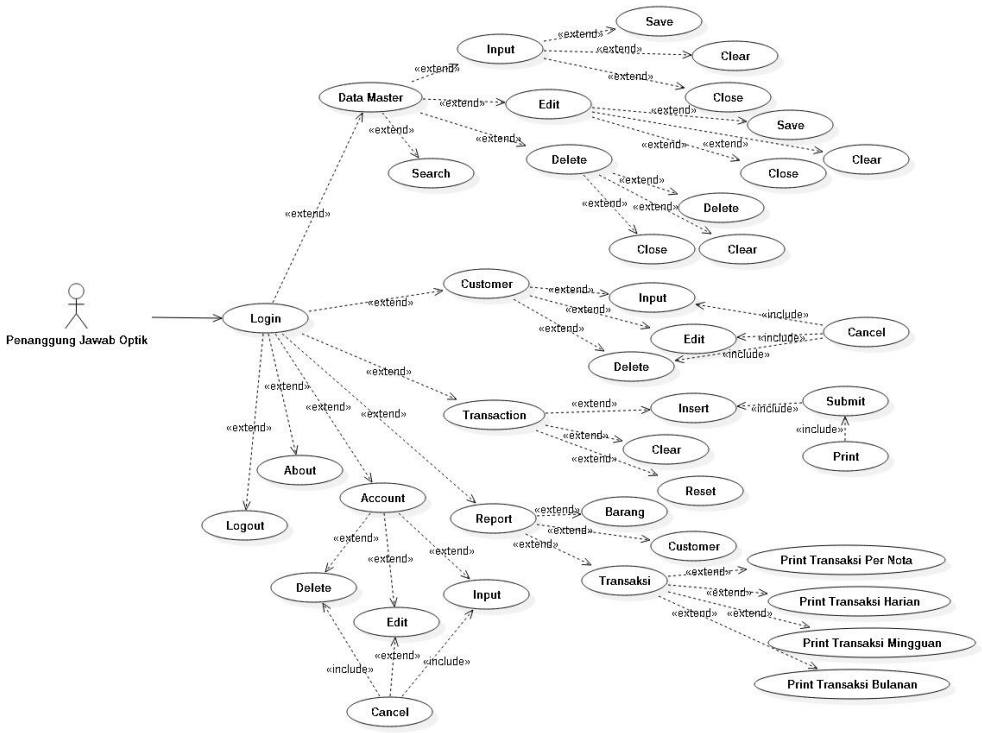
TABEL: 3. 2 Tabel Definisi *Use Case*

No.	Use Case	Deskripsi
1.	Melakukan *Login*	Proses masuknya *user* kedalam sistem.
2.	*Input* Barang	Proses *input* data barang ke dalam sistem
3.	*Edit* Barang	Proses *edit* data barang didalam sistem
4.	*Delete* Barang	Proses *delete* data barang didalam sistem
5.	Melihat Data Barang	Proses untuk melihat data barang secara *detail*.
6.	*Input* Data Pelanggan	Proses *input* data pelanggan ke dalam sistem
7.	*Edit* Data Pelanggan	Proses *edit* data pelanggan didalam sistem
8.	*Delete* Data Pelanggan	Proses *delete* data pelanggan didalam sistem
9.	Melihat Data Pelanggan	Proses untuk melihat data pelanggan secara *detail*.
10.	*Input* Data Transaksi	Proses *input* data transaksi ke dalam sistem
11.	Melihat Data Transaksi	Proses untuk melihat data transaksi secara *detail*.
12.	*Print Report* Barang	Proses untuk *print out* data barang yang ada di sistem.
13.	*Print Report* Data Pelanggan	Proses untuk *print out* data pelanggan yang ada di sistem.
14.	*Print Report* Data Transaksi	Proses untuk *print out* data transaksi yang ada di sistem.

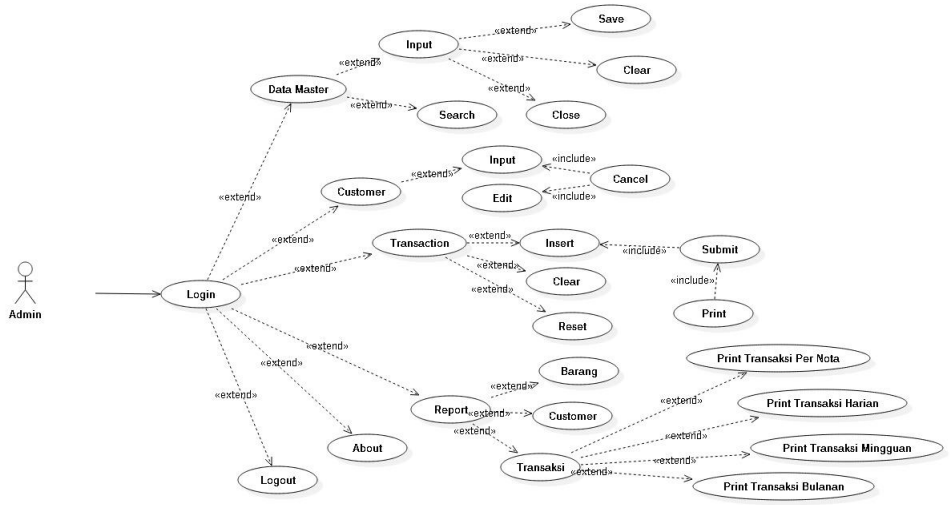
TABEL: 3. 2 Tabel Definisi *Use Case* (Lanjutan)

No.	Use Case	Deskripsi
15.	Input Account Baru	Proses *input* data *user* baru ke dalam sistem
16.	*Edit Account*	Proses *edit* data *user* didalam sistem
17.	*Delete Account*	Proses *delete* data *user* didalam sistem
18.	Melihat Data *Acoount*	Proses untuk melihat data akun secara detail.
19.	Melihat Tetang Aplikasi (*About*)	Proses untuk melihat tentang aplikasi.
20.	*Logout*	Proses untuk keluar dari *account user* yang digunakan

3. Use Case Diagram



GAMBAR: 3. 5 *Use Case* Diagram Penanggung Jawab Optik



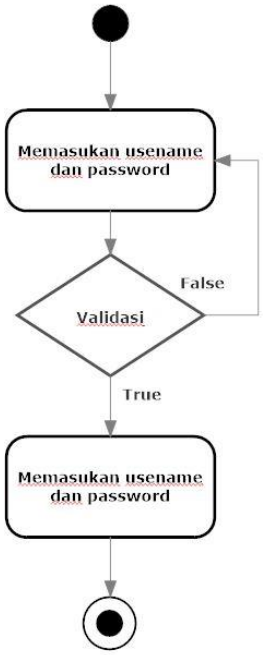
GAMBAR: 3. 6 *Use Case* Diagram *Admin*

3.3.1.2. *Activity* Diagram

Pada diagram aktivitas ini menjelaskan bagaimana proses-proses yang ada pada sistem berdasarkan dengan fungsi yang dijalankan oleh penanggung jawab optik maupun *admin*.

- Proses pada saat *user* melakukan *login*.

  Proses ini dimana aktivitas user saat melakukan *login*, mulai dari *input username* dan *password*, mengidentifikasi data *valid* atau tidak dan masuk ke halam utama jika data yang di masukan *valid*.



GAMBAR: 3. 7 *Activity User Login*

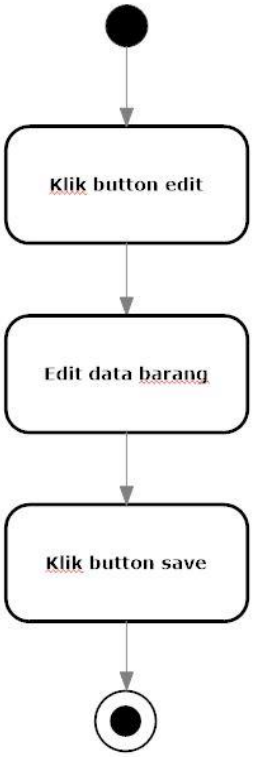
- Proses pada saat *user* melakukan *input* barang.

  Proses ini dilakukan dimana aktivitas *user* ketika melakukan input data barang di awali dengan melakukan klik pada *button input* di halaman data barang, kemudian *user* memasukan data barang sesuai dengan kriterianya, lalu *user* klik *button save* untuk menyimpan data barang yang sudah di masukan ke dalam *database*.

GAMBAR: 3. 8 *Activity Input* Data Barang

- Proses pada saat *user* melakukan *edit* barang.

  Proses ini dilakukan dimana aktivitas *user* ketika melakukan *edit* data barang di awali dengan melakukan klik pada *button edit* di halaman data barang, kemudian *user* melakukan *update* data barang, lalu *user* klik *button save* untuk menyimpan data barang ke dalam *database*.



GAMBAR: 3. 9 *Activity Edit* Data Barang

- Proses pada saat *user* melakukan *delete* barang

  Proses ini dilakukan dimana aktivitas *user* ketika melakukan *delete* data barang di awali dengan melakukan klik pada *button delete* di halaman data barang, kemudian *user* mencari data barang yang akan di hapus, lalu *user* klik *button delete* untuk menghapus data dari *database*.

GAMBAR: 3. 10 *Activity Delete* Data Barang

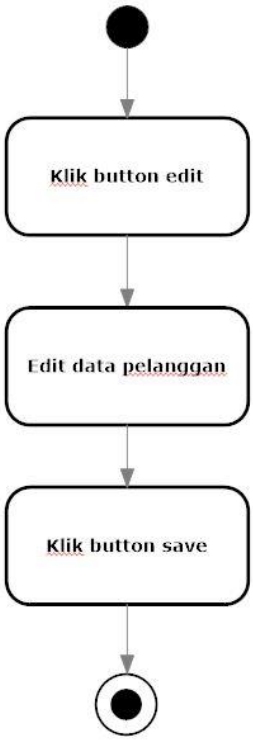
- Proses pada saat *user* melakukan *input* data pelanggan.

  Proses ini dilakukan dimana aktivitas *user* ketika melakukan *input* data pelanggan di awali dengan melakukan klik pada *button input* di halaman data pelanggan (*customer*), kemudian *user* memasukan data pelanggan, lalu *user* klik *button save* untuk menyimpan data pelanggan ke dalam *database*.

GAMBAR: 3. 11 *Activity Input* Data Pelanggan

- Proses pada saat *user* melakukan *edit* data pelanggan

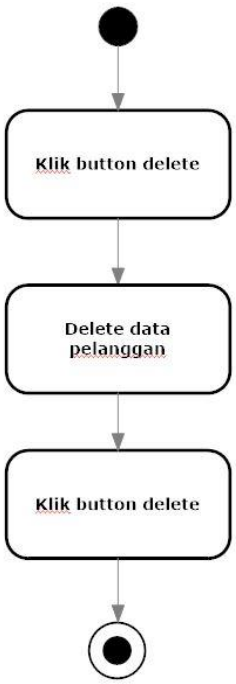
  Proses ini dilakukan dimana aktivitas *user* ketika melakukan *edit* data pelanggan (customer) di awali dengan melakukan klik pada *button edit* di halaman data pelanggan, kemudian *user* melakukan *update* data pelanggan, lalu *user* klik *button save* untuk menyimpan data barang ke dalam *database*.



GAMBAR: 3. 12 *Activity Edit* Data Pelanggan

- Proses pada saat *user* melakukan *delete* pelanggan.

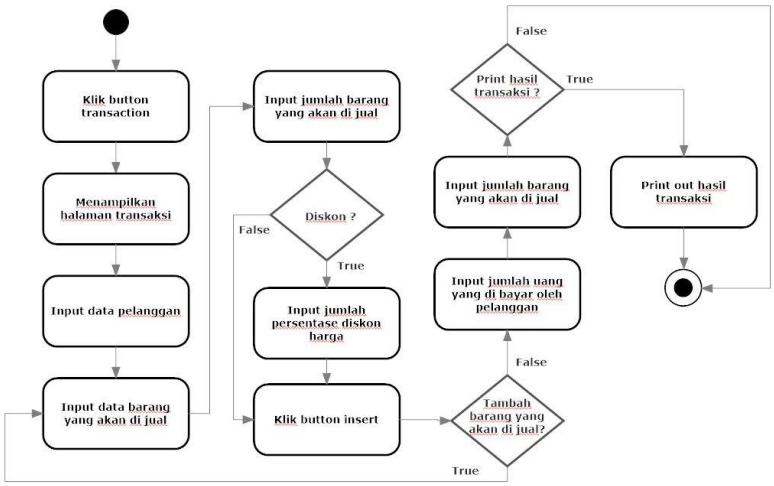
  Proses ini dilakukan dimana aktivitas *user* ketika melakukan *delete* data pelanggan (*customer*) di awali dengan melakukan klik pada *button delete* di halaman data pelanggan, kemudian *user* mencari data pelanggan yang akan di hapus, lalu *user* klik *button delete* untuk menghapus data dari *database*.



GAMBAR: 3. 13 *Activity Delete Data* Pelanggan

- Proses pada saat *user* melakukan transaksi

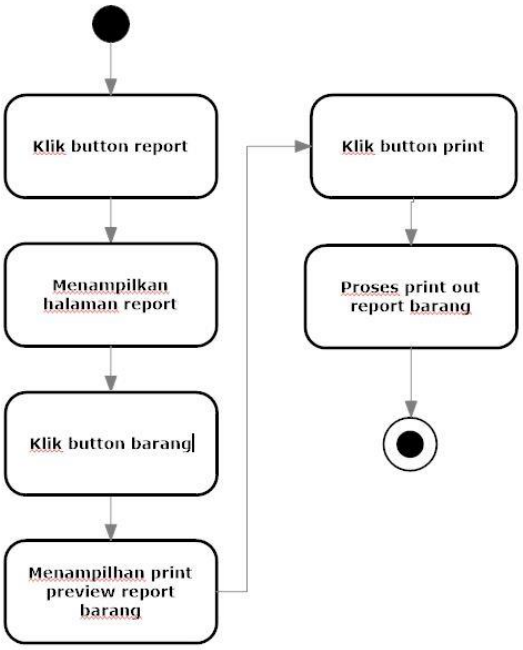
  Proses ini dilakukan dimana aktivitas user untuk melakukan transaksi di awali dengan klik *input* di halaman transaksi, kemudian *user* memasukan data pelanggan, data barang, jumlah barang, diskon (jika ada diskon), dan jumlah uang yang di bayarkan oleh pembeli, lalu klik *button insert* agar data tersebut terekam di dalam *DataGridView*, *user* akan mengulangi langkah tersebut sampai semua barang yang akan terjual sudah di masukan ke dalam *DataGrdiView*. Setelah proses tersebut selesai maka *user* melakukan klik *tombol submit* agar data barang di *database* bisa ter-*update*, Langkah terakhir user klik *button print* untuk mencetak hasil transaksi.



GAMBAR: 3. 14 *Activity* Transaksi

- Proses pada saat *user* melakukan *print out* data barang.

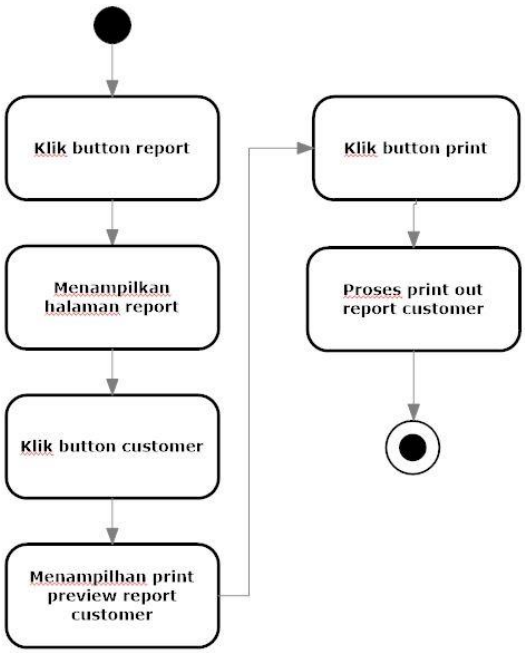
  Proses ini dilakukan ketika aktivitas *user* menekan *button barang* yang ada di halaman *report* maka akan muncul tampilan *print preview*, kemudian klik *icon print* untuk mencetak data tersebut.



GAMBAR: 3. 15 *Activity Print Report* Barang

- Proses pada saat *user* melakukan *print out* data pelanggan.

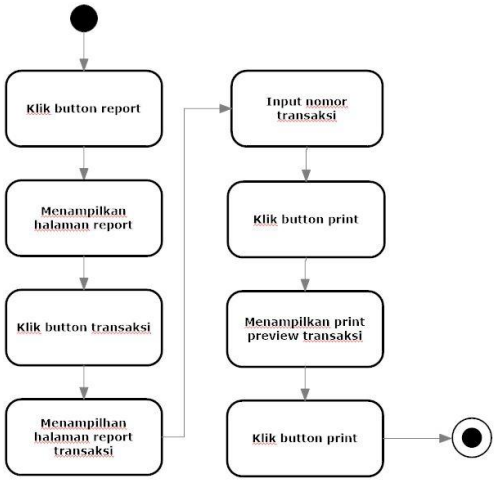
  Proses ini dilakukan ketika aktivitas *user* menekan *button customer* yang ada di halaman *report* maka akan muncul tampilan *print preview*, kemudian klik *icon print* untuk mencetak data tersebut.



GAMBAR: 3. 16 *Activity Print Report* Pelanggan

- Proses pada saat *user* melakukan *print out* data transaksi per nota.

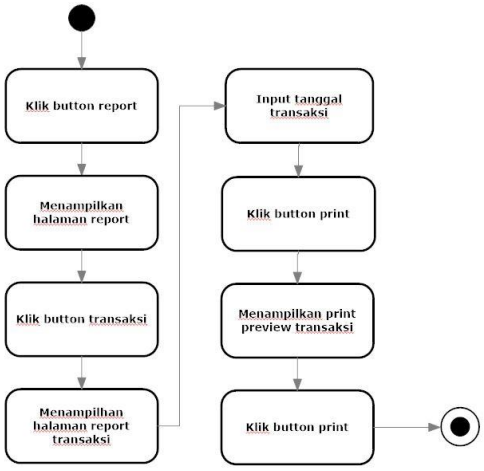
  Proses ini dilakukan ketika aktivitas *user* menekan *button* transaksi yang ada di halaman *report,* kemudian *user* memasukan nomor transaksi di kolom nomor transaksi dan klik *button print* untuk melihat data transaksi sesuai nomor transaksi yang sebelumnya di masukan, lalu klik *icon print* mencetak data tersebut.



GAMBAR: 3. 17 *Activity Print Report* Transaksi Per Nota

- Proses pada saat *user* melakukan *print out* data transaksi harian.

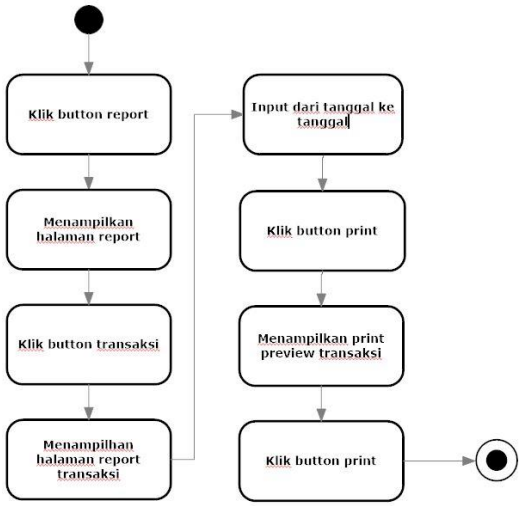
  Proses ini dilakukan ketika aktivitas *user* menekan *button* transaksi yang ada di halaman *report*, kemudian *user* memasukan tanggal transaksi di kolom tanggal transaksi dan klik *button print* untuk melihat data transaksi sesuai tanggal transaksi yang sebelumnya di masukan, lalu klik *icon print* mencetak data tersebut.



GAMBAR: 3. 18 *Activity Print Report* Transaksi Harian

- Proses pada saat *user* melakukan *print out* data transaksi mingguan.

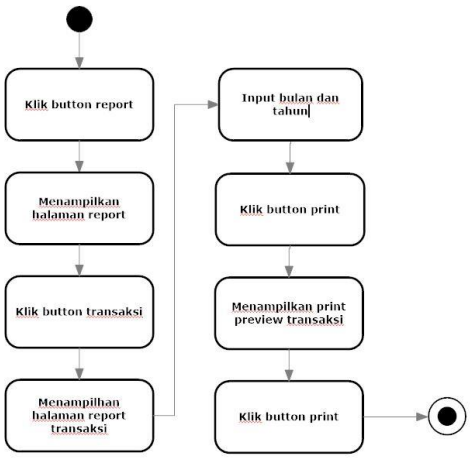
  Proses ini dilakukan ketika aktivitas *user* menekan *button* transaksi yang ada di halaman *report*, kemudian *user* memasukan dari tanggal transaksi ke tanggal transaksi di kolom tanggal transaksi dan klik *button print* untuk melihat data transaksi sesuai tanggal transaksi yang sebelumnya di masukan, lalu klik *icon print* mencetak data tersebut.



GAMBAR: 3. 19 *Activity Print Report* Transaksi Mingguan

- Proses pada saat *user* melakukan *print out* data transaksi bulanan.

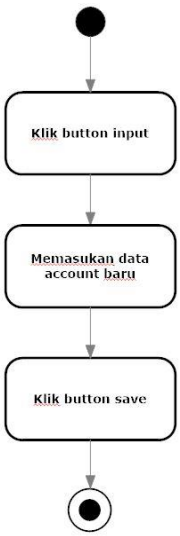
  Proses ini dilakukan ketika aktivitas *user* menekan *button* transaksi yang ada di halaman *report*, kemudian *user* memasukan bulan dan tahun transaksi di kolom bulan dan tahun transaksi dan klik *button print* untuk melihat data transaksi sesuai bulan dan tahun transaksi yang sebelumnya di masukan, lalu klik *icon print* mencetak data tersebut.



GAMBAR: 3. 20 *Activity Print Report* Transaksi Bulanan

- Proses pada saat *user* melakukan *input* account baru.

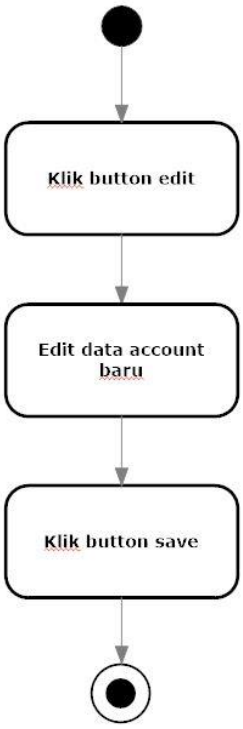
  Proses ini dilakukan dimana aktivitas *user* ketika melakukan input data *account* di awali dengan melakukan klik pada *button input* di halaman data *account*, kemudian *user* memasukan data *account* baru, lalu *user* klik *button save* untuk menyimpan data *account* baru ke dalam *database*.



GAMBAR: 3. 21 *Activity Input* Data *Account* Baru

- Proses pada saat *user* melakukan *edit* data account.

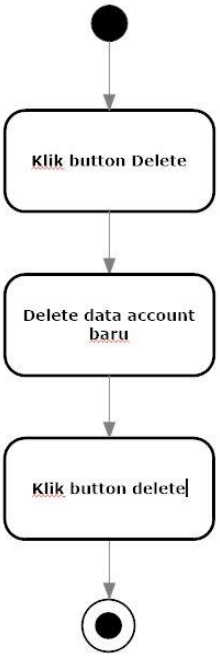
  Proses ini dilakukan dimana aktivitas *user* ketika melakukan *edit* data *account* di awali dengan melakukan klik pada *button edit* di halaman data *account*, kemudian *user* melakukan *update* data account, lalu *user* klik *button save* untuk menyimpan data barang ke dalam *database*.



GAMBAR: 3. 22 *Activity Edit* Data *Account*

- Proses pada saat *user* melakukan *delete* data *account*.

  Proses ini dilakukan dimana aktivitas *user* ketika melakukan *delete* data *account* di awali dengan melakukan klik pada *button delete* di halaman data *account*, kemudian *user* mencari data acconunt yang akan di hapus, lalu *user* klik *button delete* untuk menghapus data dari *database*.



GAMBAR: 3. 23 *Activity Delete* Data *Account*

- Proses pada saat *user* melihat tentang aplikasi.

  Proses ini dilakukan ketika *user* menekan *button about* di halaman utama maka akan muncul informasi tentang aplikasi penjualan atau aplikasi POS ini.

GAMBAR: 3. 24 *Activity* Melihat Tentang Aplikasi

- Proses pada saat *user* melakukan *logout*.

  Proses ini di lakukan ketika aktivitas *user* menekan *button logout* di halaman utama, maka akan memuncul kan halaman *login*.

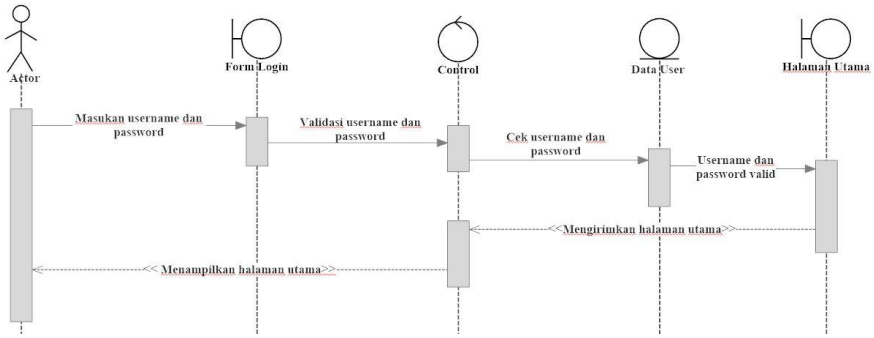
GAMBAR: 3. 25 *Activity User Logout*

3.3.1.3. *Sequence* Diagram

Pada diagram ini menjelaskan bagaimana interaksi antara objek pada sistem yang sedang berjalan berdasarkan urutan waktunya. Berikut adalah *sequence* diagram pada sistem yang dibuat :

- Interaksi yang terjadi pada saat aktor melakukan proses login ke sistem.

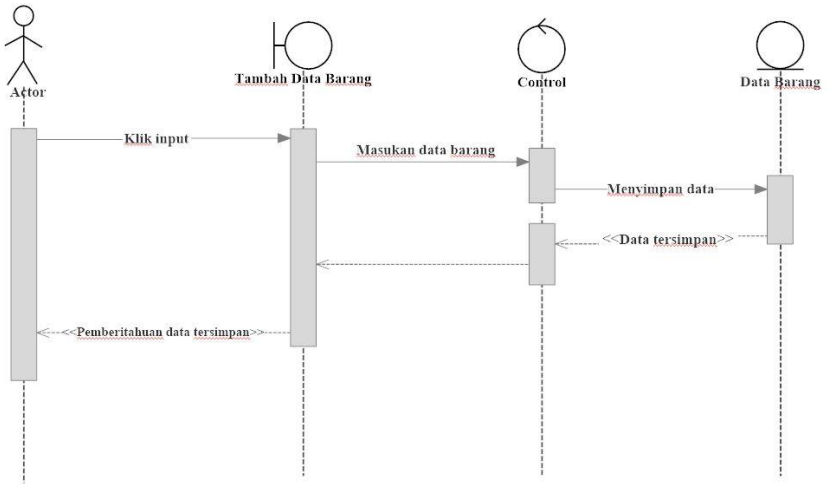
  Aktor (*user*) memasukan *username* dan *password* kemudian *system* mencari data kedalam *database*, jika data ditemukan maka akan muncul halaman utama.



GAMBAR: 3. 26 *Sequence Login User*

- Interaksi yang terjadi pada saat aktor melakukan *input* data barang ke sistem.

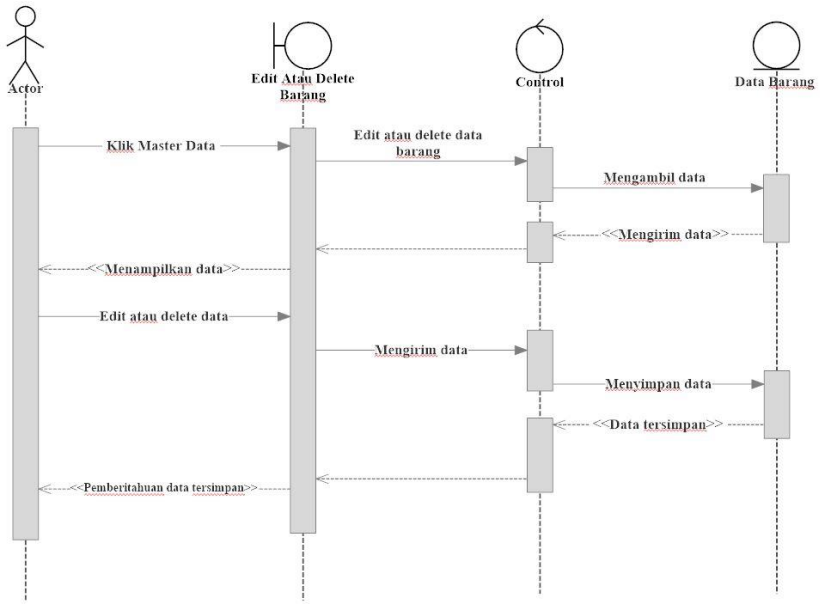
  Aktor (*user*) menekan tombol input, kemudian memasukan data barang ke dalam *form input* barang, lalu data barang di simpan ke *database* dan menampilkan pesan data berhasil di simpan.



GAMBAR: 3. 27 *Sequence Input* Data Barang

- Interaksi yang terjadi pada saat aktor melakukan *edit* atau *delete* data barang.

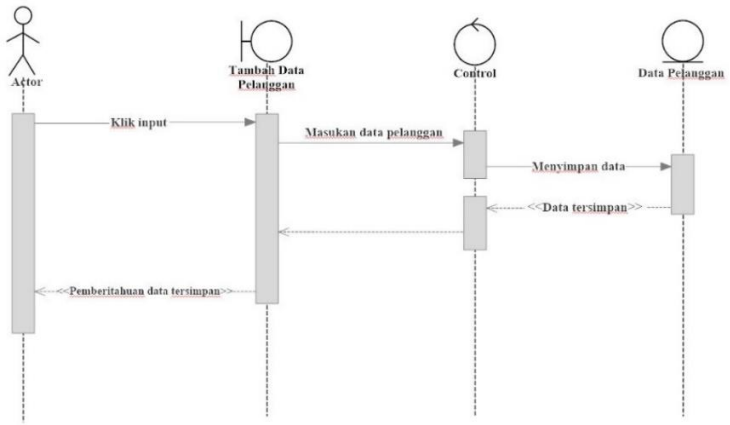
  Aktor (*user*) menekan *button* master data di halaman utama, kemudian mencari data barang yang akan di rubah atau di hapus, ketika barang yang di temukan sudah sesuai maka tekan *button edit* atau *delete*, *system* akan mengirimkan pesan data berhasil di simpan atau di hapus.



GAMBAR: 3. 28 *Sequence Edit Delete* Data Barang

- Interaksi yang terjadi pada saat aktor melakukan *input* data pelanggan ke sistem.

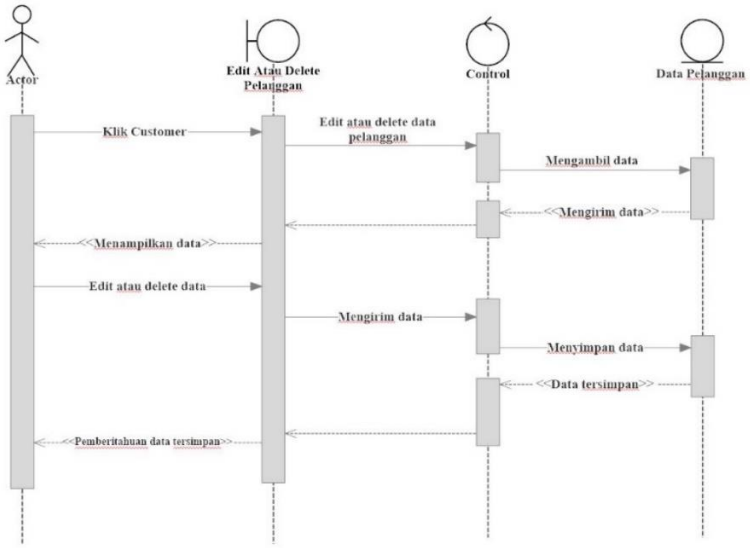
  Aktor (*user*) menekan tombol input, kemudian memasukan data pelanggan ke dalam *form input* pelanggan, lalu data pelanggan di simpan ke *database* dan menampilkan pesan data berhasil di simpan.



GAMBAR: 3. 29 *Sequence Input* Data Pelanggan

- Interaksi yang terjadi pada saat aktor melakukan *edit* atau *delete* data pelanggan.

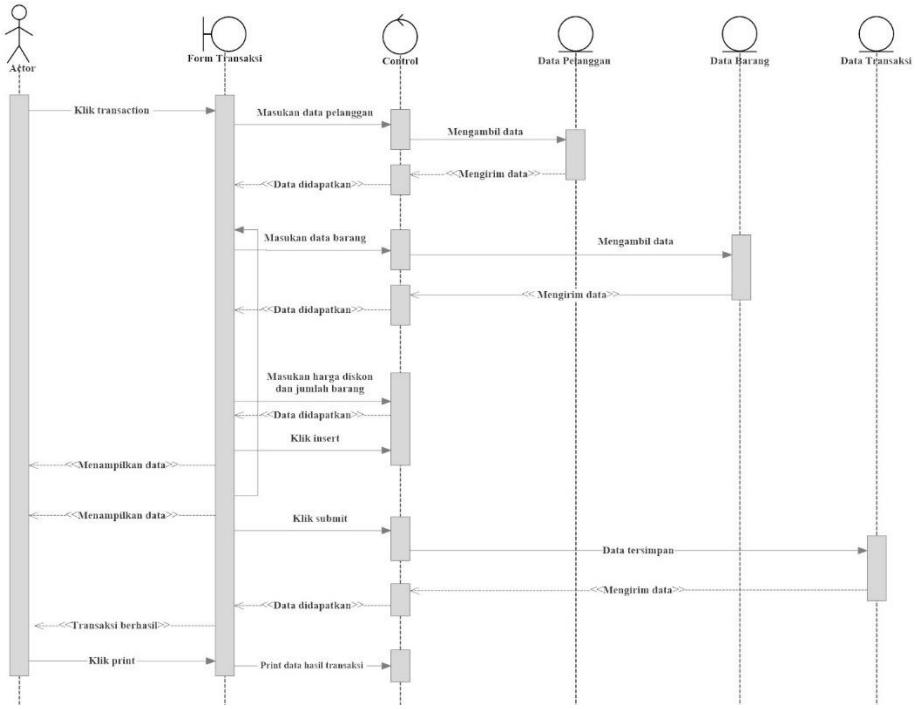
  Aktor (*user*) menekan *button* master data di halaman utama, kemudian mencari data pelanggan yang akan di rubah atau di hapus, ketika data tersebut di temukan sudah sesuai maka tekan *button edit* atau *delete*, *system* akan mengirimkan pesan data berhasil di simpan atau di hapus.



GAMBAR: 3. 30 *Sequence Edit Delete* Data Pelanggan

- Interaksi yang terjadi pada saat aktor melakukan transaksi ke system.

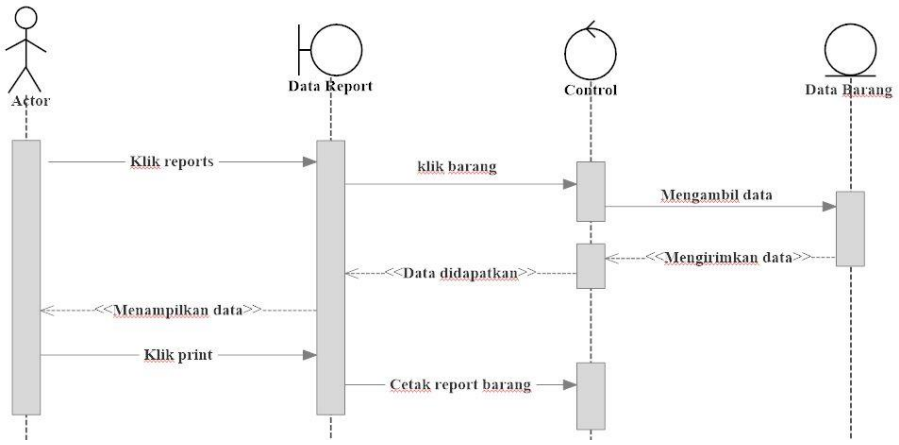
  Aktor (*user*) menekan button transaksi di halaman utama, kemudian memasukan data sesuai dengan *form* yang di sediakan di data transaksi, jika sudah lalu klik *button insert* untuk masukan data tersebut kedalam *list*, selanjutnya klik *button submit* untuk melakukan *update* ke *database*, dan klik *button print* untuk mencetak hasil transaksi.



GAMBAR: 3. 31 *Sequence* Transaksi

- Interaksi yang terjadi pada saat aktor melakukan *print report* data barang dari sistem.

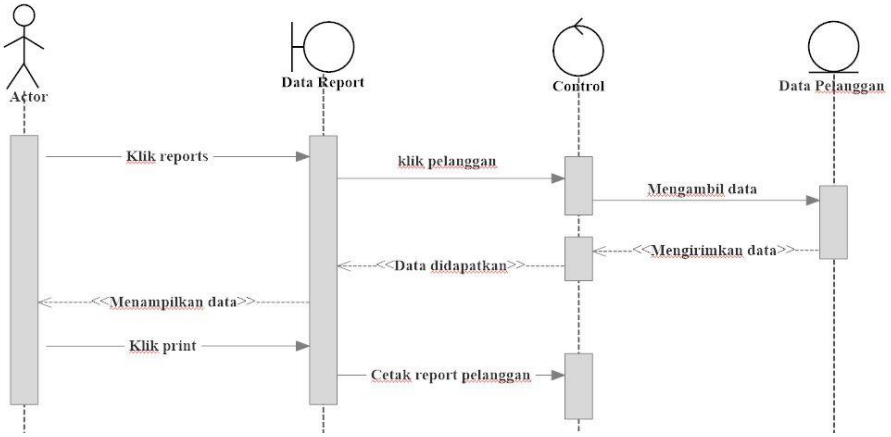
  Aktor (*user*) menekan *button report*, menekan *button* barang, maka *system* akan memunculkan data barang yang tersedia di *database*, klik *button print* untuk mencetak data terebut.



GAMBAR: 3. 32 *Sequence Print Out* Data Barang

- Interaksi yang terjadi pada saat aktor melakukan print report data pelanggan dari sistem.

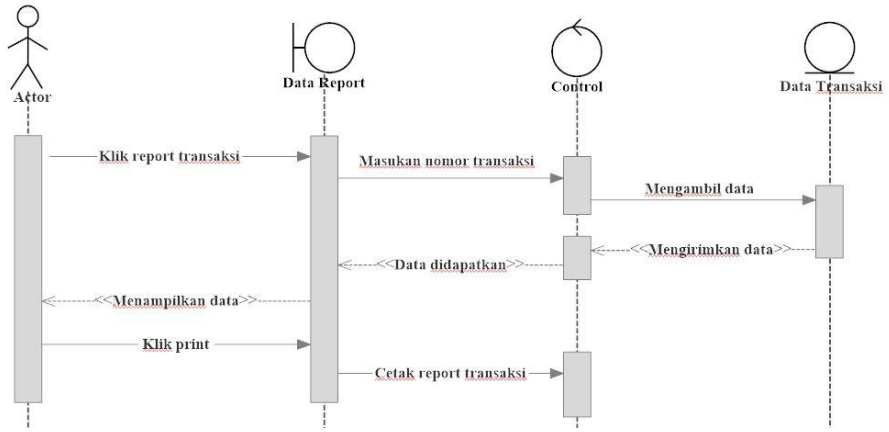
  Aktor (*user*) menekan *button report*, menekan *button* pelanggan (*customer*), maka *system* akan memunculkan data pelanggan yang tersedia di *database*, klik *button print* untuk mencetak data terebut.



GAMBAR: 3. 33 *Sequence Print Out* Data Pelanggan

- Interaksi yang terjadi pada saat aktor melakukan *print report* data transaksi per nota dari sistem.

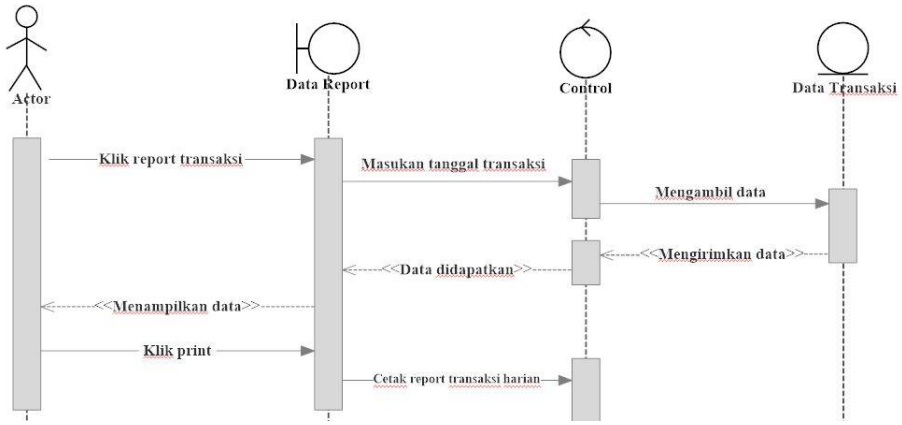
  Aktor (*user*) menekan *button report*, menekan *button* transaksi, kemudian masukan nomor transaksi maka *system* akan memunculkan data transaksi sesuai nomor transaksi yang di masukan dari *database*, klik *button print* untuk mencetak data terebut.



GAMBAR: 3. 34 *Sequence Print Out* Data Transaksi Per Nota

- Interaksi yang terjadi pada saat aktor melakukan *print report* data transaksi harian dari sistem.

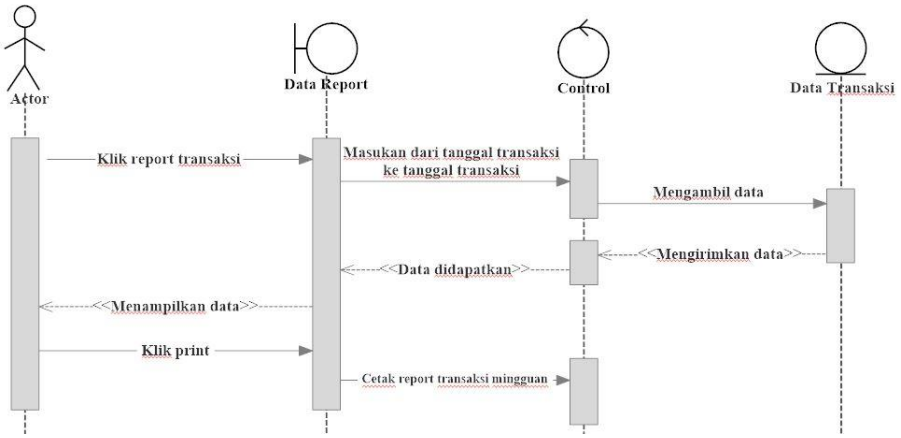
  Aktor (*user*) menekan *button report*, menekan *button* transaksi, kemudian masukan tanggal transaksi maka *system* akan memunculkan data transaksi sesuai tanggal transaksi yang di masukan dari *database*, klik *button print* untuk mencetak data terebut.



GAMBAR: 3. 35 *Sequence Print Out* Data Transaksi Harian

- Interaksi yang terjadi pada saat aktor melakukan *print report* data transaksi mingguan dari sistem.

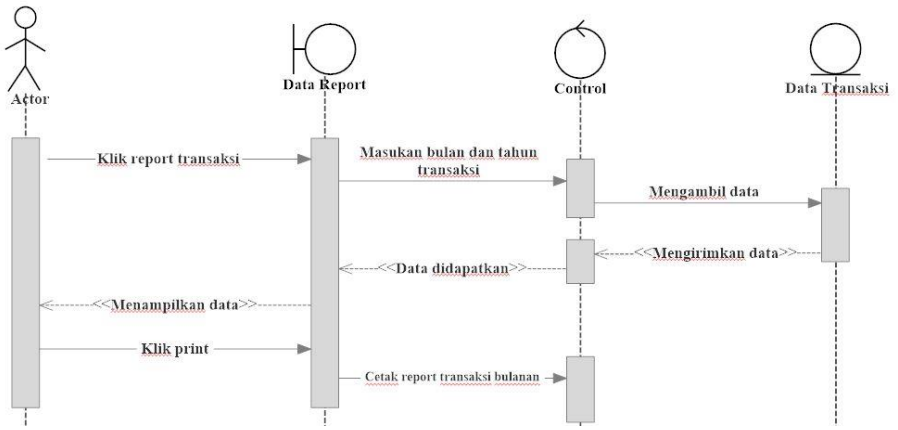
  Aktor (*user*) menekan *button report*, menekan *button* transaksi, kemudian masukan dari tanggal transaksi ke tanggal transaksi maka *system* akan memunculkan data transaksi sesuai tanggal transaksi yang di masukan dari *database*, klik *button print* untuk mencetak data terebut.



GAMBAR: 3. 36 *Sequence Print Out* Data Transaksi Mingguan

- Interaksi yang terjadi pada saat aktor melakukan *print* data transaksi bulanan dari sistem.

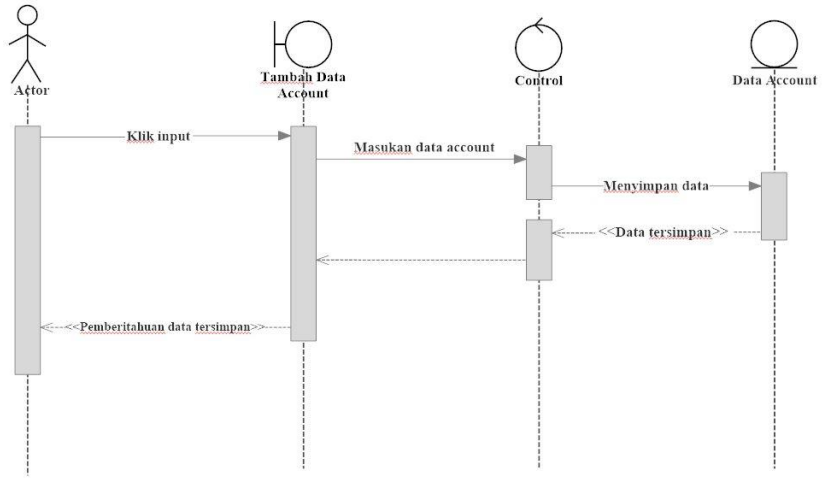
  Aktor (*user*) menekan *button report*, menekan *button* transaksi, kemudian masukan bulan dan tahun transaksi maka *system* akan memunculkan data transaksi sesuai bulan dan tahun transaksi yang di masukan dari *database*, klik *button print* untuk mencetak data terebut.



GAMBAR: 3. 37 *Sequence Print Out* Data Transaksi Bulanan

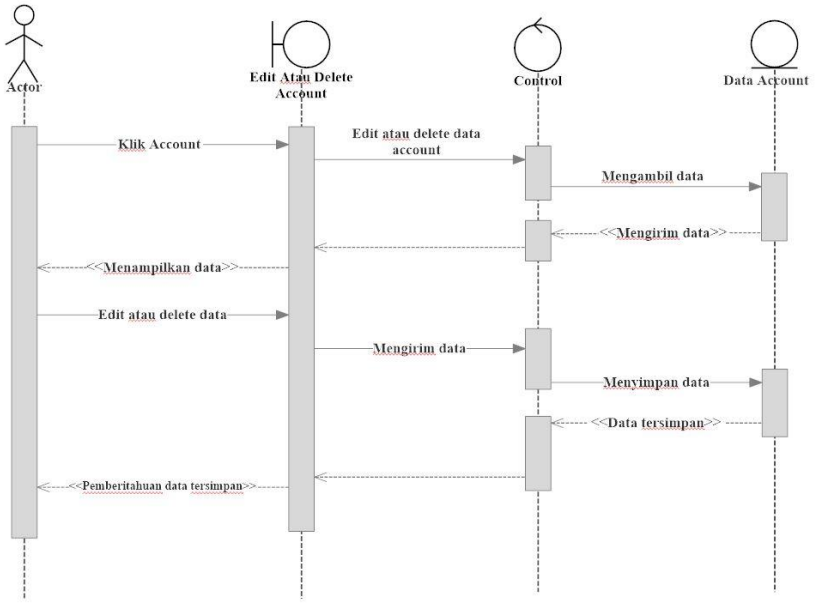
- Interaksi yang terjadi pada saat aktor melakukan *input* data *account* ke sistem.

  Aktor (*user*) menekan tombol input, kemudian memasukan data *account* baru ke dalam form input account, lalu data barang di simpan ke database dan menampilkan pesan data berhasil di inputkan.



GAMBAR: 3. 38 *Sequence Input* Data *Account*

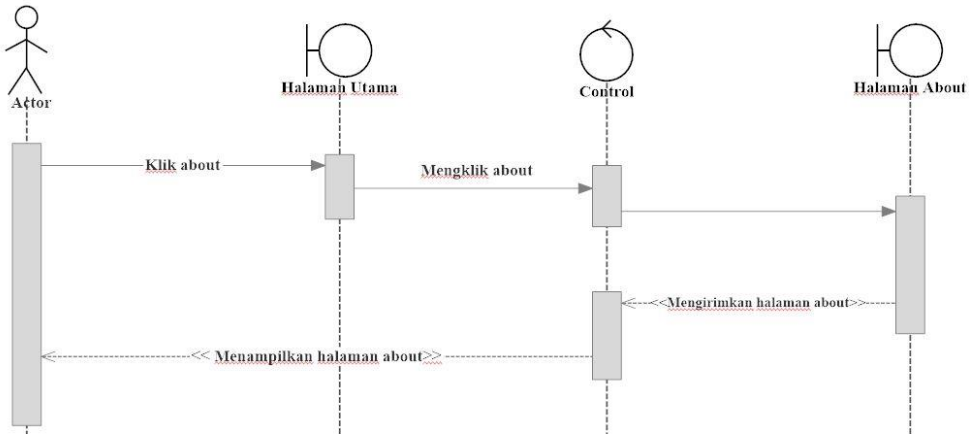
- Interaksi yang terjadi pada saat aktor melakukan *edit* atau *delete* data *account*.



GAMBAR: 3. 39 *Sequence Edit Delete* Data Account

- Interaksi yang terjadi pada saat aktor melihat tentang aplikasi.

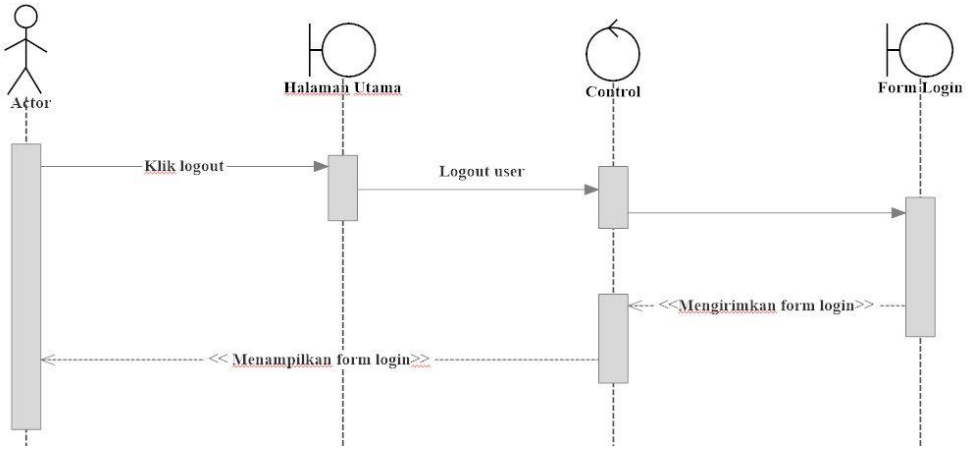
  Aktor (*user*) menekan *button about*, maka *system* akan menampilkan halaman informasi tentang aplikasi penjualan (POS) ini.



GAMBAR: 3. 40 *Sequence About*

- Interaksi yang terjadi pada saat aktor melakukan proses *login* dari sistem.

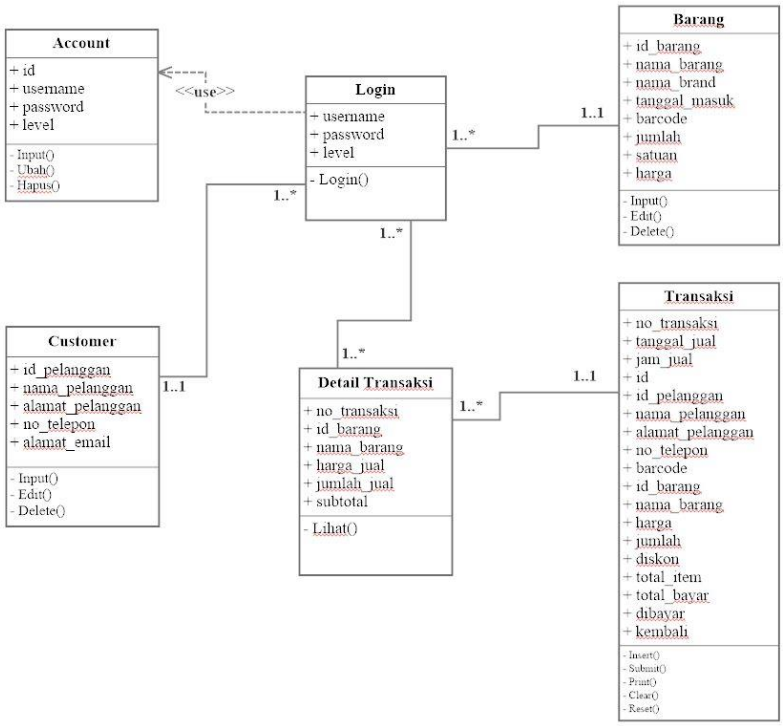
  Aktor (*user*) menekan *button logout*, maka *system* akan menampilkan halaman *login* aplikasi penjualan (POS) ini.



GAMBAR: 3. 41 *Sequence Logout User*

3.3.1.4. *Class* Diagram

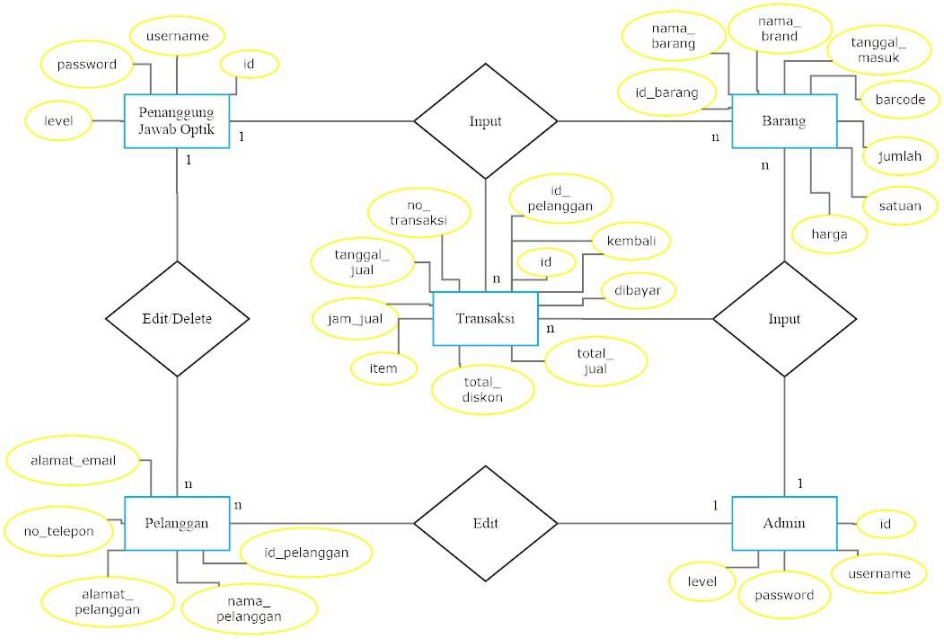
Pada gambar diagram ini menampilkan kelas-kelas yang ada pada sistem yang dibuat. Berikut adalah *class* diagramnya :



GAMBAR: 3. 42 *Class* Diagram Sistem

3.3.1.5. *Entity Relationship Database* (ERD)

Entity Relationship Database atau (ERD) yang digunakan dalam pembuatan Apliksi Penjualan (POS) ini adalah sebagai berikut :



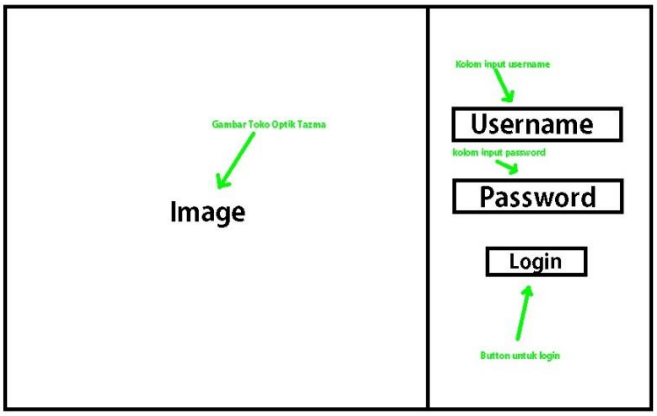
GAMBAR: 3. 43 ERD Sistem

3.3.1.6. *Design User Interface*

Berikut ini adalah desain antarmuka dari pengguna (*user*) terhadap aplikasi yang akan di buat.

- Desain Halaman Login

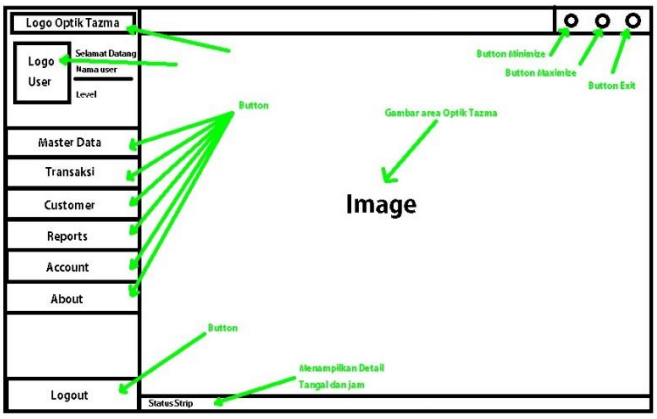
  Berikut adalah desain halam login, pada area kiri terdapat gambar yang menampilkan area Toko Optic Tazma, dan pada bagian kanan halaman terdiri dari 2 kolom *input*-an dan satu button untuk masuk ke halaman utama.



GAMBAR: 3. 44 *Design* Halaman Login

- *Design* Halaman Utama

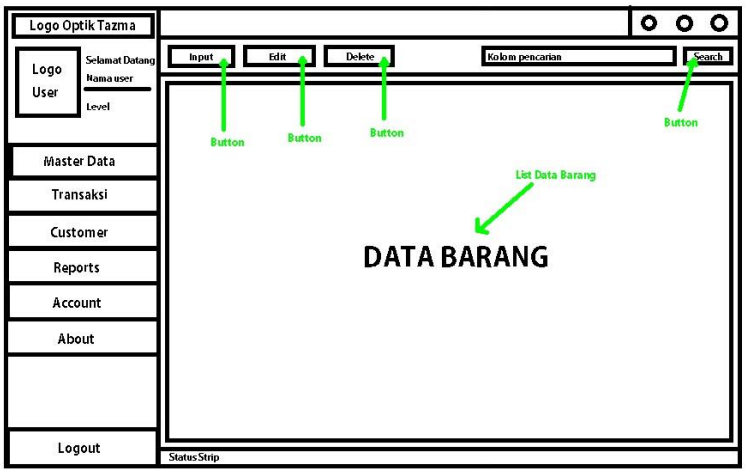
  Pada halaman ini menampilkan semua menu yang tersedia di Aplikasi Penjualan (POS) ini, mulai data data barang, data pelanggan, melakukan transaksi, menambahkan account, sampai pencetakan laporan dari hasil penjualan.



GAMBAR: 3. 45 *Design* Halaman Utama

- *Design* Halam Master Data

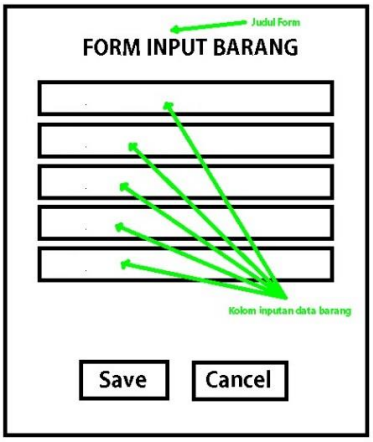
  Pada halaman ini menampilkan data *list* barang serta button untuk melakukan, *input* barang, *edit* barang dan *delete* barang serta adanya fitur pencarian barang.



GAMBAR: 3. 46 *Design* Halaman *Master* Data

- *Design* Form Input Edit Delete Barang

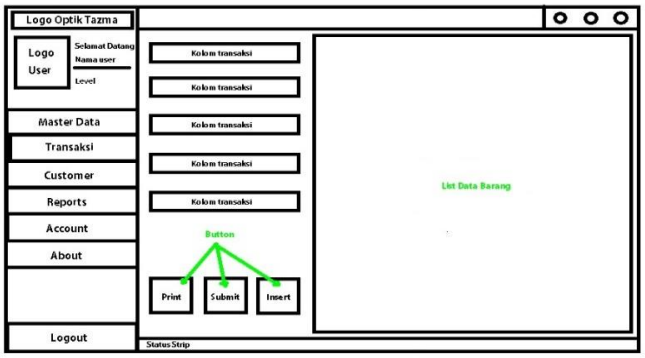
  Form ini merupakan *form* untuk melakukan *input* data barang, *edit* data barang dan *delete* data barang.



GAMBAR: 3. 47 *Design Form* Barang

- *Design* Halaman Transaksi

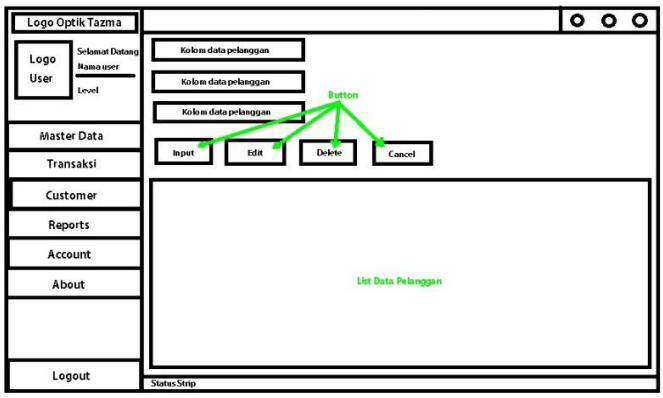
  Halaman ini merupakan halama untuk melakukan transaksi, terdiri dari kolom-kolom yang harus di masukan diantaranya data pelanggan, data barang, dan data lainnya sampai di akhir yaitu melakukan cetak data transaksi.



GAMBAR: 3. 48 *Design* Halaman Transaksi

- *Design* Halaman Pelanggan (*Customer*)

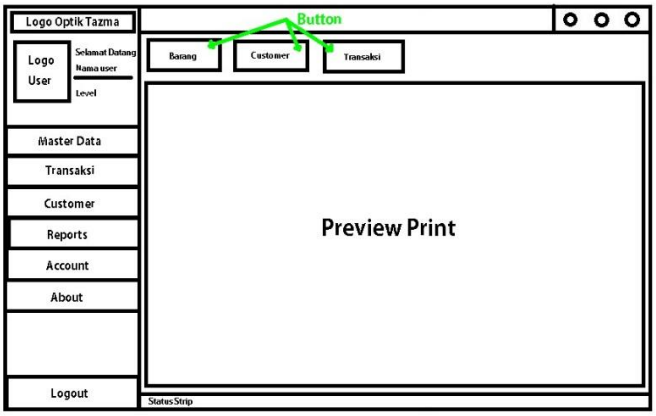
  Halaman ini merupakan halaman data pelanggan, yang terdiri dari list data pelanggan, serta button dan kolomg untuk melakukan *input* data pelanggan, *edit* dan *delete*.



GAMBAR: 3. 49 *Design* Halaman Pelanggan (*Customer*)

- *Design* Halaman *Reports*

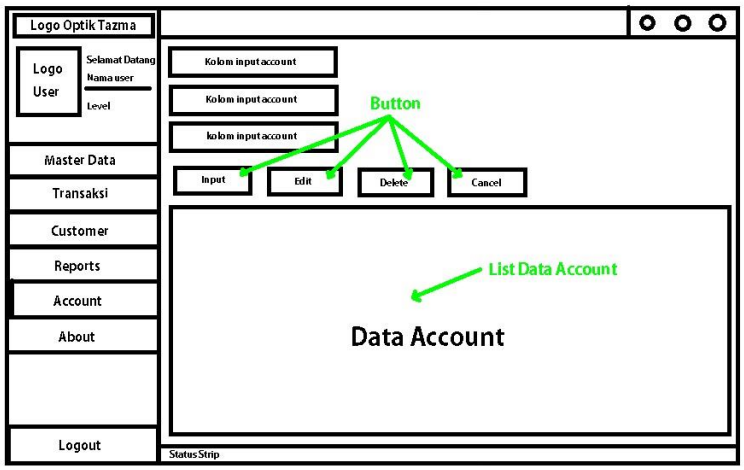
  Berikut ini adalah desain halaman *report*, dimana pada halaman ini *user* bisa melakukan proses pencetakan data barang, pelanggan serta data transaksi.



GAMBAR: 3. 50 *Design* Halaman *Report*

- *Design* Halaman *Account*

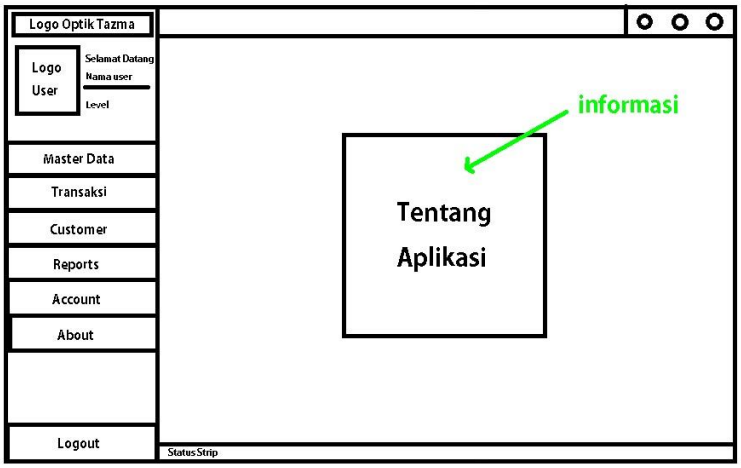
  Halaman ini merupakan halaman untuk mengelola *account*, dimana *user* bisa melakukan penambahan *account*, *update account* dan *delete*.



GAMBAR: 3. 51 *Design* Halaman *Account*

- *Design* Halaman *About*

  Halaman ini merupakan halaman dimana informasi dari Aplikasi Penjualan (POS) ini.



GAMBAR: 3. 52 *Design* Halaman *About*

BAB IV

IMPLEMENTASI DAN UJI COBA

4.1. Penulisan Kode Program Dan Implementasi (*Coding And Testing*)

Tahap ini dilakukan nya pembuatan program atau penulisan bahasa program terhadap aplikasi penjualan (POS) serta implementasi sebagai uji coba terhadap user interface dan fitur-fitur yang ada di aplikasi tersebut.

4.1.1. *Coding Program*

4.1.1.1. *Coding Form Login*

```
Imports System.Data.Odbc
Imports Aplikasi_Penjualan
Public Class Login_Form
    Private Sub TBUsername_Leave(sender As Object, e As EventArgs) Handles
TBUsername.Leave
        If TBUsername.Text = "username" Or TBUsername.Text = "" Then
            TBUsername.ForeColor = Color.Silver
            TBUsername.Text = "username"
        Else
            TBUsername.ForeColor = Color.Black
        End If
    End Sub

    Private Sub TBUsername_MouseClick(sender As Object, e As
MouseEventArgs) Handles TBUsername.MouseClick
        If TBUsername.Text = "username" Then
            TBUsername.Clear()
            TBUsername.ForeColor = Color.Black
        End If
    End Sub

    Private Sub TBUsername_TextChanged(sender As Object, e As EventArgs)
Handles TBUsername.TextChanged

    End Sub
```

```
    Private Sub TBPassword_Leave(sender As Object, e As EventArgs) Handles
TBPassword.Leave
        If TBPassword.Text = "********" Or TBPassword.Text = "" Then
            TBPassword.ForeColor = Color.Silver
            TBPassword.Text = "********"
        Else
            TBPassword.ForeColor = Color.Black
        End If
    End Sub

    Private Sub TBPassword_MouseClick(sender As Object, e As
MouseEventArgs) Handles TBPassword.MouseClick
        If TBPassword.Text = "********" Then
            TBPassword.Clear()
            TBPassword.ForeColor = Color.Black
        End If
    End Sub

    Private Sub TBPassword_TextChanged(sender As Object, e As EventArgs)
Handles TBPassword.TextChanged

    End Sub

    Private Sub BTNClose_Click(sender As Object, e As EventArgs) Handles
BTNClose.Click
        End
    End Sub

    Private Sub BTNMasuk_Click(sender As Object, e As EventArgs) Handles
BTNMasuk.Click
        If TBUsername.Text = "username" Or TBPassword.Text = "password" Then
            MsgBox("Username and Password Required!")
        Else
            Call koneksi()
            Cmd = New OdbcCommand("Select * From tb_Login where username='"
& TBUsername.Text & "' and password ='" & TBPassword.Text & "'", Conn)
            Rd = Cmd.ExecuteReader
            Rd.Read()
            If Rd.HasRows Then
                Me.Hide()
                Home.Show()
                Home.STLabel2.Text = Rd!id
                Home.STLabel4.Text = Rd!username
                Home.STLabel6.Text = Rd!level
                Home.LblUser.Text = Rd!id
```

```
            Home.LblStatus.Text = Rd!Level
        Else
            MsgBox("Username or Password is Wrong!")
        End If
    End If

    If Home.STLabel6.Text = "User" Then
        Home.BtnAccount.Visible = False
        Home.BtnDeletePelanggan.Visible = False
        Home.BtnEditBarang.Visible = False
        Home.BtnDeleteBarang.Visible = False
    End If
  End Sub
  Private Sub Login_Form_Load(sender As Object, e As EventArgs) Handles
MyBase.Load

  End Sub
End Class.
```

4.1.1.2. *Coding Input* Data Barang

```
Private Sub BTNInput_Click(sender As Object, e As EventArgs) Handles
BTNInput.Click
    If BTNInput.Text = "Input" Then
        BTNInput.Text = "Save"
        BTNEdit.Enabled = False
        BTNDelete.Enabled = False
        BTNBatal.Enabled = True
        Call Siapisi()
        Call NomorOtomatis2()
        TBUsername.Focus()
    Else
        Call koneksi()
        Cmd = New OdbcCommand("Select * from tb_login where username = '"
& TBUsername.Text & "'", Conn)
        Rd = Cmd.ExecuteReader
        Rd.Read()
        If Rd.HasRows = True Then
            MsgBox("This username already available")
        Else
            If TBId.Text = "" Or TBUsername.Text = "" Or TBPassword.Text = ""
Or CBLevel.Text = "" Then
                MsgBox("Columns cannot be empty!")
            Else
                Call koneksi()
```

```
            Dim InputData As String = "Insert into tb_login values('" &
TBId.Text & "','" & TBUsername.Text & "','" & TBPassword.Text & "','" &
CBLevel.Text & "')"
            Cmd = New OdbcCommand(InputData, Conn)
            Cmd.ExecuteNonQuery()
            MsgBox("Account input successfully")
            Call kondisiAwal()
          End If
        End If
      End If
    End Sub.
```

4.1.1.3. *Coding Edit* Data Barang

```
Private Sub BTNEdit_Click(sender As Object, e As EventArgs) Handles
BTNEdit.Click
      If BTNEdit.Text = "Edit" Then
        BTNEdit.Text = "Save"
        TBId.Enabled = True
        BTNInput.Enabled = False
        BTNDelete.Enabled = False
        BTNBatal.Enabled = True
        Call Siapisi()
      Else
        If TBId.Text = "" Or TBUsername.Text = "" Or TBPassword.Text = "" Or
CBLevel.Text = "" Then
          MsgBox("Columns cannot be empty!")
        Else
          Call koneksi()
          Dim UpdateData As String = "Update tb_login set id='" & TBId.Text &
"',Username='" & TBUsername.Text & "',Password='" & TBPassword.Text &
"',Level='" & CBLevel.Text & "' where id='" & TBId.Text & "' "
          Cmd = New OdbcCommand(UpdateData, Conn)
          Cmd.ExecuteNonQuery()
          MsgBox("Update Account successfully")
          Call kondisiAwal()
        End If
      End If
    End Sub.
```

4.1.1.4. *Coding Delete* Data Barang

```
Private Sub BTNDelete_Click(sender As Object, e As EventArgs) Handles
BTNDelete.Click
    If BTNDelete.Text = "Delete" Then
        BTNDelete.Text = "Remove"
        TBId.Enabled = True
        BTNInput.Enabled = False
        BTNEdit.Enabled = False
        BTNBatal.Enabled = True
        Call Siapisi()
    Else
        If TBId.Text = "" Or TBUsername.Text = "" Or TBPassword.Text = "" Or
CBLevel.Text = "" Then
            MsgBox("Columns cannot be empty!")
        Else
            Call koneksi()
            Dim HapusData As String = "Delete from tb_login where id='" &
TBId.Text & "'"
            Cmd = New OdbcCommand(HapusData, Conn)
            Cmd.ExecuteNonQuery()
            MsgBox("Delete Account successfully")
            Call kondisiAwal()
        End If
    End If
  End Sub.
```

4.1.1.5. *Coding Input* Data Pelanggan

```
Private Sub BtnInputPelanggan_Click(sender As Object, e As EventArgs)
Handles BtnInputPelanggan.Click
    If BtnInputPelanggan.Text = "Input" Then
        BtnInputPelanggan.Text = "Save"
        BtnEditPelanggan.Enabled = False
        BtnDeletePelanggan.Enabled = False
        BtnBatalPelanggan.Enabled = True
        Call Siapisi1()
        Call NomorOtomatis3()
        Call koneksi()
        TBNamaPelanggan.Focus()
    Else
        Call koneksi()
        Cmd = New OdbcCommand("Select * from tbl_pelanggan where
no_telepon = '" & TBTelepon.Text & "' Or alamat_email = '" &
TBAlamatEmail.Text & "' ", Conn)
```

```
        Rd = Cmd.ExecuteReader
        Rd.Read()
        If Rd.HasRows = True Then
            MsgBox("This username already available")

        Else
            If TBIdPelanggan.Text = "" Or TBNamaPelanggan.Text = "" Or
TBAlamat.Text = "" Or TBTelepon.Text = "" Or TBAlamatEmail.Text = "" Then
                MsgBox("Columns cannot be empty!")
            Else
                Call koneksi()
                Dim InputDatapelanggan As String = "Insert into tbl_pelanggan
values('" & TBIdPelanggan.Text & "','" & TBNamaPelanggan.Text & "','" &
TBAlamat.Text & "','" & TBTelepon.Text & "','" & TBAlamatEmail.Text & "')"
                Cmd = New OdbcCommand(InputDatapelanggan, Conn)
                Cmd.ExecuteNonQuery()
                MsgBox("Customer input successfully")
                Call kondisiAwal2()
            End If
        End If
    End If
  End Sub.
```

4.1.1.6. *Coding Edit* Data Pelanggan

```
Private Sub BtnEditPelanggan_Click(sender As Object, e As EventArgs) Handles
BtnEditPelanggan.Click
    If BtnEditPelanggan.Text = "Edit" Then
        BtnEditPelanggan.Text = "Save"
        TBIdPelanggan.Enabled = True
        BtnInputPelanggan.Enabled = False
        BtnDeletePelanggan.Enabled = False
        BtnBatalPelanggan.Enabled = True
        Call Siapisi1()
    Else
        If TBIdPelanggan.Text = "" Or TBNamaPelanggan.Text = "" Or
TBAlamatEmail.Text = "" Or TBTelepon.Text = "" Or TBAlamatEmail.Text = ""
Then
            MsgBox("Columns cannot be empty!")
        Else
            Call koneksi()
            Dim UpdateDataPelanggan As String = "Update tbl_pelanggan set
id_pelanggan='" & TBIdPelanggan.Text & "',nama_pelanggan='" &
TBNamaPelanggan.Text & "',alamat_pelanggan='" & TBAlamat.Text &
```

```
"',no_telepon='" & TBTelepon.Text & "',alamat_email='" & TBAlamatEmail.Text
& "' where id_pelanggan='" & TBIdPelanggan.Text & "' "
            Cmd = New OdbcCommand(UpdateDataPelanggan, Conn)
            Cmd.ExecuteNonQuery()
            MsgBox("Update Customer successfully")
            Call kondisiAwal2()
        End If
    End If
End Sub.
```

4.1.1.7. *Coding Delete* Data *Pelanggan*

```
Private Sub BtnDeletePelanggan_Click(sender As Object, e As EventArgs)
Handles BtnDeletePelanggan.Click
    If BtnDeletePelanggan.Text = "Delete" Then
        BtnDeletePelanggan.Text = "Remove"
        TBIdPelanggan.Enabled = True
        BtnInputPelanggan.Enabled = False
        BtnEditPelanggan.Enabled = False
        BtnBatalPelanggan.Enabled = True
        Call Siapisi1()
    Else
        If TBIdPelanggan.Text = "" Or TBNamaPelanggan.Text = "" Or
TBAlamatEmail.Text = "" Or TBTelepon.Text = "" Or TBAlamatEmail.Text = ""
Then
            MsgBox("Columns cannot be empty!")
        Else
            Call koneksi()
            Dim HapusData As String = "Delete from tbl_pelanggan where
id_pelanggan='" & TBIdPelanggan.Text & "'"
            Cmd = New OdbcCommand(HapusData, Conn)
            Cmd.ExecuteNonQuery()
            MsgBox("Delete Customer successfully")
            Call kondisiAwal2()
        End If
    End If
End Sub.
```

4.1.1.8. *Coding* Transaksi

```
Private Sub BtnInsert_Click(sender As Object, e As EventArgs) Handles
BtnInsert.Click
```

```
    If LblNamaBarang.Text = "" Or TBJumlah.Text = "" Or
LblNamaPelanggan.Text = "" Or LblAlamatPelanggan.Text = "" Or
TBTeleponPelanggan.Text = "" Then
        MsgBox("Error! Please Input All")
    Else
        If Val(LblJumlahBarang.Text) < Val(TBJumlah.Text) Then
            MsgBox(" Stok The amount of stock of goods is less!")
        Else
            If Val(LblJumlahBarang.Text) < TBJumlah.Text Then
                MsgBox(" Stok The amount of stock of goods is less!")
            Else
                DataGridViewTransaksi.Rows.Add(New String() {LblIdBarang.Text,
LblNamaBarang.Text, LblHarga.Text, TBJumlah.Text, Val(LblTotalDiskon.Text)
* Val(TBJumlah.Text), Val(LblTotalHarga.Text) * Val(TBJumlah.Text)})
                Call RumusSubTotal()
                Call RumusSubTotalDiskon()
                TBBarcode.Text = ""
                LblIdBarang.Text = ""
                LblNamaBarang.Text = ""
                LblHarga.Text = ""
                TBdiskon.Text = ""
                TBJumlah.Text = ""
                LblTotalDiskon.Text = ""
                LblHargajual.Text = ""
                TBJumlah.Enabled = False
                CheckBox1.Checked = False
                TBDibayar.Enabled = True
                BtnSubmit.Enabled = False
                BtnPrint.Enabled = False
                Call RumusJumlahItem()
                TBBarcode.Focus()
            End If
        End If
    End If
  End Sub

Private Sub BtnSubmit_Click(sender As Object, e As EventArgs) Handles
BtnSubmit.Click
    If LblKembali.Text = "" Or LblNamaPelanggan.Text = "" Or LblTotal.Text
= "" Then
        MsgBox("No Transaction! Pelase input Data All Column")
    Else
        TglMySQL = Format(Today, "yyyy-MM-dd ")
        Dim SimpanTransaksi As String = "insert into tbl_transaksi value ('" &
LblNoTransaction.Text & "','" & TglMySQL & "','" & LblJam.Text & "','" &
LblItem.Text & "','" & LblGrandTotalDiskon.Text & "','" & LblTotal.Text & "','"
```

```
& TBDibayar.Text & "','" & LblKembali.Text & "','" & TBidPelangganTrs.Text &
"','" & LblAdmin.Text & "')"
        Cmd = New OdbcCommand(SimpanTransaksi, Conn)
        Cmd.ExecuteNonQuery()

        For Baris As Integer = 0 To DataGridViewTransaksi.Rows.Count - 1
            Dim SimpanDetail As String = "insert into tbl_detail_transaksi
values('" & LblNoTransaction.Text & "', '" &
DataGridViewTransaksi.Rows(Baris).Cells(0).Value & "', '" &
DataGridViewTransaksi.Rows(Baris).Cells(1).Value & "', '" &
DataGridViewTransaksi.Rows(Baris).Cells(2).Value & "', '" &
DataGridViewTransaksi.Rows(Baris).Cells(3).Value & "', '" &
DataGridViewTransaksi.Rows(Baris).Cells(5).Value & "')"
            Cmd = New OdbcCommand(SimpanDetail, Conn)
            Cmd.ExecuteNonQuery()

            Cmd = New OdbcCommand("select * from tbl_barang where
id_barang='" & DataGridViewTransaksi.Rows(Baris).Cells(0).Value & "'",
Conn)
            Rd = Cmd.ExecuteReader
            Rd.Read()
            Dim KurangiStok As String = "Update tbl_barang set jumlah ='" &
Rd.Item("Jumlah") - DataGridViewTransaksi.Rows(Baris).Cells(3).Value & "'
Where id_barang='" & DataGridViewTransaksi.Rows(Baris).Cells(0).Value &
"'"
            Cmd = New OdbcCommand(KurangiStok, Conn)
            Cmd.ExecuteReader()
        Next

        Call kondisiAwal3()
        Call BuatKolom()
        Call kondisiawal4()
        Call kondisiAwal5()
        BtnInsert.Enabled = False
        BtnClearPelanggan.Enabled = False
        TBidPelangganTrs.Enabled = False
        TBTeleponPelanggan.Enabled = False
        MsgBox("Transaction Successfully")
        BtnPrint.Focus()
    End If
End Sub.
```

4.1.1.9. *Coding Reports*

```
Private Sub BtnReportBarang_Click(sender As Object, e As EventArgs) Handles
BtnReportBarang.Click
        PnlReportIsi.Visible = True
        PnlReportIsi2.Visible = False
        If MessageBox.Show("Do you want view report barang ?", "",
MessageBoxButtons.YesNo) = Windows.Forms.DialogResult.Yes Then
            CRV1.ReportSource = "ReportBarang.rpt"
            CRV1.RefreshReport()
        End If
    End Sub.

Private Sub BtnReportPelanggan_Click(sender As Object, e As EventArgs)
Handles BtnReportPelanggan.Click
        PnlReportIsi.Visible = True
        PnlReportIsi2.Visible = False
        If MessageBox.Show("Do you want view report customer ?", "",
MessageBoxButtons.YesNo) = Windows.Forms.DialogResult.Yes Then
            CRV1.ReportSource = "ReportCustomer.rpt"
            CRV1.RefreshReport()
        End If
    End Sub.

Private Sub BtnPrintPerNota_Click(sender As Object, e As EventArgs) Handles
BtnPrintPerNota.Click
        If TBTransaksiPerNota.Text = "" Then
            MsgBox("This column not empty!")
        Else
            AxCrystalReport2.SelectionFormula =
"totext({tbl_transaksi.no_transaksi})='" & TBTransaksiPerNota.Text & "'"
            AxCrystalReport2.ReportFileName = "ReportTransaction.rpt"
            AxCrystalReport2.WindowState =
Crystal.WindowStateConstants.crptMaximized
            AxCrystalReport2.RetrieveDataFiles()
            AxCrystalReport2.Action = 1
        End If
    End Sub.

Private Sub BtnPrintHarian_Click(sender As Object, e As EventArgs) Handles
BtnPrintHarian.Click
        AxCrystalReport3.SelectionFormula =
"totext({tbl_transaksi.tanggal_jual})='" & DateTimePickerHarian.Value & "'"
        AxCrystalReport3.ReportFileName = "ReportTransactionHarian.rpt"
        AxCrystalReport3.WindowState =
Crystal.WindowStateConstants.crptMaximized
```

```
        AxCrystalReport3.RetrieveDataFiles()
        AxCrystalReport3.Action = 1
    End Sub.

Private Sub BtnPrintMingguan_Click(sender As Object, e As EventArgs) Handles
BtnPrintMingguan.Click
        AxCrystalReport4.SelectionFormula = "{tbl_transaksi.tanggal_jual} in date
(" & LblInDate.Text & ") to date (" & LblToDate.Text & ")"
        AxCrystalReport4.ReportFileName = "ReportTransactionMingguan.rpt"
        AxCrystalReport4.WindowState =
Crystal.WindowStateConstants.crptMaximized
        AxCrystalReport4.RetrieveDataFiles()
        AxCrystalReport4.Action = 1
    End Sub.

Private Sub BtnPrintBulanan_Click(sender As Object, e As EventArgs) Handles
BtnPrintBulanan.Click
        If CBReportBulanan.Text = "" Or CBReportTahunan.Text = "" Then
            MsgBox("Pleaser insert month and year")
        Else
            AxCrystalReport5.SelectionFormula =
"Month({tbl_transaksi.tanggal_jual})=" & Val(CBReportBulanan.Text) & " and
year({tbl_transaksi.tanggal_jual})=" & Val(CBReportTahunan.Text)
            AxCrystalReport5.ReportFileName = "ReportTransactionBulanan.rpt"
            AxCrystalReport5.WindowState =
Crystal.WindowStateConstants.crptMaximized
            AxCrystalReport5.RetrieveDataFiles()
            AxCrystalReport5.Action = 1
        End If
    End Sub.
```

4.1.1.10. *Coding Input* Data *Account*

```
Private Sub BTNInput_Click(sender As Object, e As EventArgs) Handles
BTNInput.Click
        If BTNInput.Text = "Input" Then
            BTNInput.Text = "Save"
            BTNEdit.Enabled = False
            BTNDelete.Enabled = False
            BTNBatal.Enabled = True
            Call Siapisi()
            Call NomorOtomatis2()
            TBUsername.Focus()
        Else
            Call koneksi()
```

```
        Cmd = New OdbcCommand("Select * from tb_login where username = '"
& TBUsername.Text & "'", Conn)
        Rd = Cmd.ExecuteReader
        Rd.Read()
        If Rd.HasRows = True Then
            MsgBox("This username already available")
        Else
            If TBId.Text = "" Or TBUsername.Text = "" Or TBPassword.Text = ""
Or CBLevel.Text = "" Then
                MsgBox("Columns cannot be empty!")
            Else
                Call koneksi()
                Dim InputData As String = "Insert into tb_login values('" &
TBId.Text & "','" & TBUsername.Text & "','" & TBPassword.Text & "','" &
CBLevel.Text & "')"
                Cmd = New OdbcCommand(InputData, Conn)
                Cmd.ExecuteNonQuery()
                MsgBox("Account input successfully")
                Call kondisiAwal()
            End If
        End If
    End If
  End Sub.
```

4.1.1.11. *Coding Edit* Data *Account*

```
Private Sub BTNEdit_Click(sender As Object, e As EventArgs) Handles
BTNEdit.Click
    If BTNEdit.Text = "Edit" Then
        BTNEdit.Text = "Save"
        TBId.Enabled = True
        BTNInput.Enabled = False
        BTNDelete.Enabled = False
        BTNBatal.Enabled = True
        Call Siapisi()
    Else
        If TBId.Text = "" Or TBUsername.Text = "" Or TBPassword.Text = "" Or
CBLevel.Text = "" Then
            MsgBox("Columns cannot be empty!")
        Else
            Call koneksi()
            Dim UpdateData As String = "Update tb_login set id='" & TBId.Text &
"',Username='" & TBUsername.Text & "',Password='" & TBPassword.Text &
"',Level='" & CBLevel.Text & "' where id='" & TBId.Text & "' "
            Cmd = New OdbcCommand(UpdateData, Conn)
```

```
            Cmd.ExecuteNonQuery()
            MsgBox("Update Account successfully")
            Call kondisiAwal()
        End If
    End If
End Sub.
```

4.1.1.12. *Coding Delete* Data *Account*

```
Private Sub BTNDelete_Click(sender As Object, e As EventArgs) Handles
BTNDelete.Click
    If BTNDelete.Text = "Delete" Then
        BTNDelete.Text = "Remove"
        TBId.Enabled = True
        BTNInput.Enabled = False
        BTNEdit.Enabled = False
        BTNBatal.Enabled = True
        Call Siapisi()
    Else
        If TBId.Text = "" Or TBUsername.Text = "" Or TBPassword.Text = "" Or
CBLevel.Text = "" Then
            MsgBox("Columns cannot be empty!")
        Else
            Call koneksi()
            Dim HapusData As String = "Delete from tb_login where id='" &
TBId.Text & "'"
            Cmd = New OdbcCommand(HapusData, Conn)
            Cmd.ExecuteNonQuery()
            MsgBox("Delete Account successfully")
            Call kondisiAwal()
        End If
    End If
End Sub.
```

4.1.1.13. *Coding Database*

- Membuat Tabel Login

```
CREATE TABLE `tb_login` (
 `id` varchar(6) not null primary key,
 `username` varchar(14),
 `password` varchar(14),
 `level` varchar(5)
);
```

- Membuat Tabel Barang

```
CREATE TABLE `tbl_barang` (
 `id_barang` varchar(13) not null primary key,
 `nama_barang` varchar(25),
 `nama_brand` varchar(25),
 `tanggal_masuk` date,
 `barcode` varchar(30),
 `jumlah` int(4),
 `satuan` varchar(5),
 `harga` int(9)
);
```

- Membuat Tabel Pelanggan

```
CREATE TABLE `tbl_pelanggan` (
  `id_pelanggan` varchar(13) not null primary key,
  `nama_pelanggan` varchar(25),
  `alamat_pelanggan` varchar(40),
  `no_telepon` varchar(14),
  `alamat_email` varchar(30)
);
```

- Membuat Tabel Transaksi

```
CREATE TABLE `tbl_transaksi` (
 `no_transaksi` varchar(13) not null primary key,
 `tanggal_jual` date,
 `jam_jual` time,
 `item` int(3),
 `total_diskon` int(9),
 `total_jual` int(9),
 `dibayar` int(9),
 `kembali` int(9),
 `foreign key (id_pelanggan) references tbl_pelanggan (id_pelanggan),
 `foreign key (id) references tb_login (id)
);
```

- Membuat Tabel Detail Transaksi

```
CREATE TABLE `tbl_detail_transaksi` (
 `no_transaksi` varchar(13),
 `id_barang` varchar(13),
 `nama_barang` varchar(25),
 `harga_jual` int(9),
 `jumlah_jual` int(3),
 `subtotal` int(9)
 `foreign key (no_transaksi) references tbl_transaksi (no_transaksi),
 `foreign key (id_barang) references tbl_barang (id_barang)
);
```

4.2. Penerapan Dan Pengujian Program (*Integrated And Testing*)

Pada tahap ini bisa di bilang tahap akhir dimana pembuatan aplikasi dengan metode *Waterfall* dapat di implementasikan langsung oleh user.

4.2.1. Pengguna *(User)*

Dalam program ini terdapat akun-akun yang terlibat dalam pengoprasian aplikasi penjualan (POS) ini, berikut ini adalah pengguna yang terlibat dalam Apllikasi, yaitu :

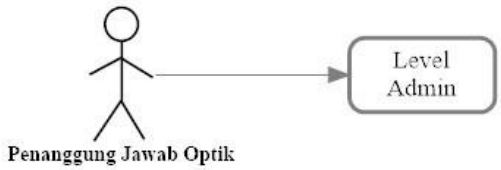
1. Penanggung Jawab Optik.
2. Admin.

4.2.2. Hak Akses

Pada tahap ini dilakukan identifikasi kebutuhan pengguna *user* dengan hak akses tertentu.

1. Penanggung Jawab Optik

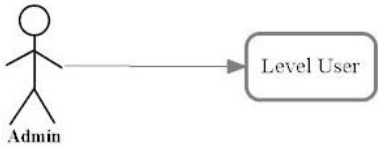
   Penanggung jawab optik berada di posisi level *admin* dimana mendapatkan hak akses full terhadap pengoprasian Aplikasi Penjual (POS).



GAMBAR: 4. 1 Hak Akses *Level Admin*

2. Admin

   Admin berada di posisi *level user*, dimana *level user* terdapat batasan-batasang yang tidak bisa di lakukan, diantaranya tidak bisa melakukan *edit* data barang, *delete* data barang, hapus data pelanggan dan tidak bisa mengelola *account*.



GAMBAR: 4. 2 Hak Akses *Level User*

4.2.3. Implementasi Program

4.2.3.1. Implementasi *Database*

Sesuai dengan apa yang dijabarkan pada bab 3, berikut adalah implementasi pembuatan tabel pada database yang digunakan oleh aplikasi :

1. Stuktur tabel *login* pada *database* yang digunakan.

#	Name	Type	Collation	Attributes	Null	Default
1	id	varchar(6)	utf8mb4_general_ci		No	*None*
2	username	varchar(14)	utf8mb4_general_ci		No	*None*
3	password	varchar(14)	utf8mb4_general_ci		No	*None*
4	level	varchar(5)	utf8mb4_general_ci		No	*None*

GAMBAR: 4. 3 Tabel *Login*

2. Stuktur tabel barang pada *database* yang digunakan.

#	Name	Type	Collation	Attributes	Null	Default
1	id_barang	varchar(13)	utf8mb4_general_ci		No	*None*
2	nama_barang	varchar(25)	utf8mb4_general_ci		No	*None*
3	nama_brand	varchar(25)	utf8mb4_general_ci		No	*None*
4	tanggal_masuk	date			No	*None*
5	barcode	varchar(30)	utf8mb4_general_ci		No	*None*
6	jumlah	int(4)			No	*None*
7	satuan	varchar(5)	utf8mb4_general_ci		No	*None*
8	harga	int(9)			No	*None*

GAMBAR: 4. 4 Tabel Barang

3. Stuktur tabel pelanggan pada *database* yang digunakan.

#	Name	Type	Collation	Attributes	Null	Default
1	id_pelanggan	varchar(13)	utf8mb4_general_ci		No	*None*
2	nama_pelanggan	varchar(25)	utf8mb4_general_ci		No	*None*
3	alamat_pelanggan	varchar(40)	utf8mb4_general_ci		No	*None*
4	no_telepon	varchar(14)	utf8mb4_general_ci		No	*None*
5	alamat_email	varchar(30)	utf8mb4_general_ci		No	*None*

GAMBAR: 4. 5 Tabel Pelanggan

4. Stuktur tabel transaksi pada *database* yang digunakan.

#	Name	Type	Collation	Attributes	Null	Default
1	no_transaksi	varchar(13)	utf8mb4_general_ci		No	None
2	tanggal_jual	date			No	None
3	jam_jual	time			No	None
4	item	int(3)			No	None
5	total_diskon	int(9)			No	None
6	total_jual	int(9)			No	None
7	dibayar	int(9)			No	None
8	kembali	int(9)			No	None
9	id_pelanggan	varchar(13)	utf8mb4_general_ci		No	None
10	id	varchar(6)	utf8mb4_general_ci		No	None

GAMBAR: 4. 6 Tabel Transaksi

5. Stuktur tabel *detail*_transaksi pada *database* yang digunakan.

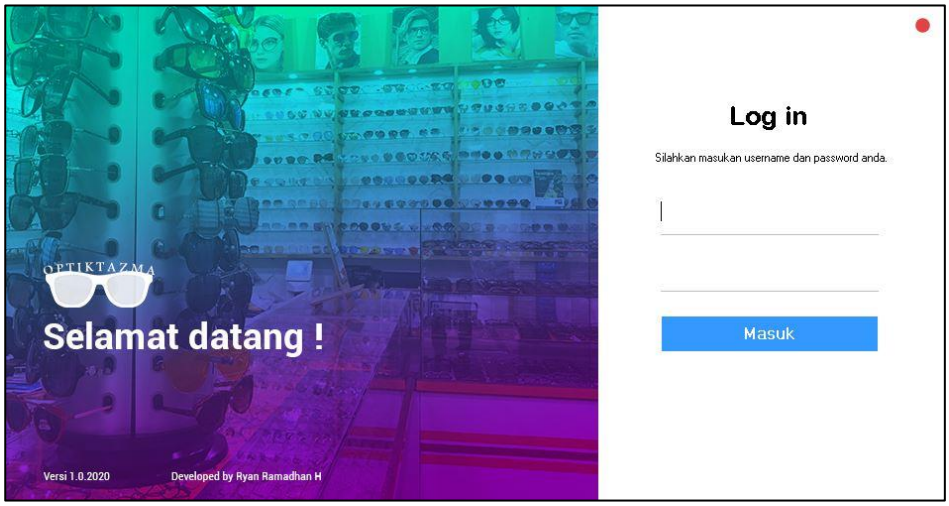
#	Name	Type	Collation	Attributes	Null	Default
1	no_transaksi	varchar(13)	utf8mb4_general_ci		No	None
2	id_barang	varchar(13)	utf8mb4_general_ci		No	None
3	nama_barang	varchar(25)	utf8mb4_general_ci		No	None
4	harga_jual	int(9)			No	None
5	jumlah_jual	int(3)			No	None
6	subtotal	int(9)			No	None

GAMBAR: 4. 7 Tabel *Detail* Transaksi

4.2.3.2. Implementasi Antarmuka (*User Interface*)

Pada implementasi ini penulis menyesuaikan tampilan yang ada pada aplikasi dengan rancangan tampilan yang ada pada bab 3. Berikut adalah tampilan yang ada pada aplikasi penjualan (POS) yang telah dibuat :

1. Menu yang di tampilkan oleh sistem pada saat *user* akan melakukan *login* dan mencoba untuk mengakses sistem.



GAMBAR: 4. 8 Tampilan Halaman *Login*

2. Pada saat *user* dengan *level Admin* berhasil *login*, sistem akan menampilkan halaman menu utama, dimana *level admin* bisa mengkases *full* menu serta fasilitas yang ada pada aplikasi POS ini.

GAMBAR: 4. 9 Tampilan *Home* (Menu Utama *Level Admin*)

3. Pada saat *user* dengan *level User* berhasil *login*, sistem akan menampilkan halaman menu utama, dimana *level user* tidak bisa mengkases *full* menu pada aplikasi POS ini, yaitu tidak bisa di akses menu *Account*.

GAMBAR: 4. 10 Tampilan *Home* (Menu Utama *Level User*)

4. Pada menu *Master* Data, *user* dengan *level Admin*, sistem akan menampilkan semua data barang yang ada berikut dengan *detail Id* Barang, Harga, Jumlah dan yang lainnya, serta dengan *level Admin* bisa mengkases *full* pada menu *Master* Data pada aplikasi POS ini.

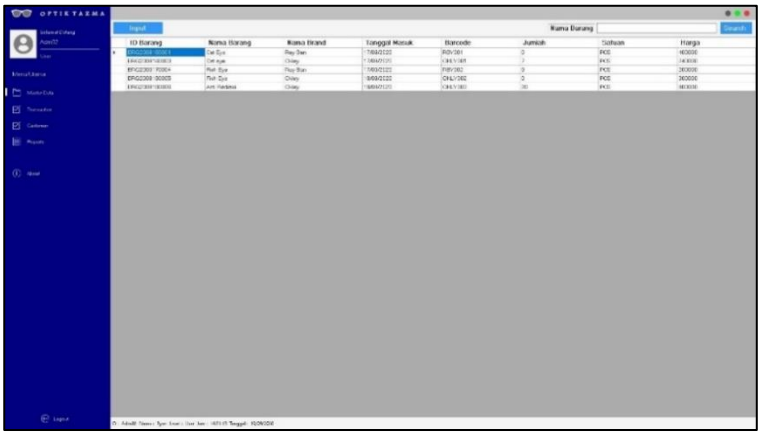
GAMBAR: 4. 11 Tampilan Menu *Master* Data (*Level Admin*)

GAMBAR: 4. 12 Tampilan *Form Input* Barang (*Level Admin* dan *User*)

GAMBAR: 4. 13 Tampilan *Form Edit* Barang (*Level Admin*)

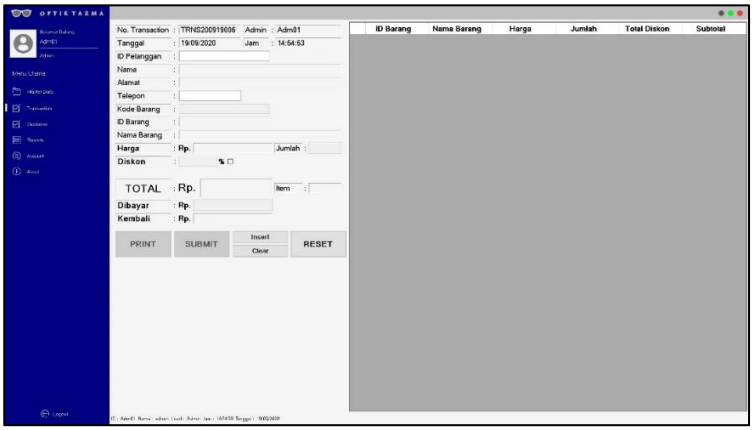
GAMBAR: 4. 14 Tampilan *Form Delete* Barang (*Level Admin*)

5. Pada menu *Master* Data, *user* dengan *level User*, sistem akan menampilkan semua data barang yang ada berikut dengan *detail Id* Barang, Harga, Jumlah dan yang lainnya, serta dengan level *User* tidak bisa mengkases *full* pada menu *Master* Data pada aplikasi POS ini, diantaranya tidak bisa melakukan *Edit* dan *Delete*.



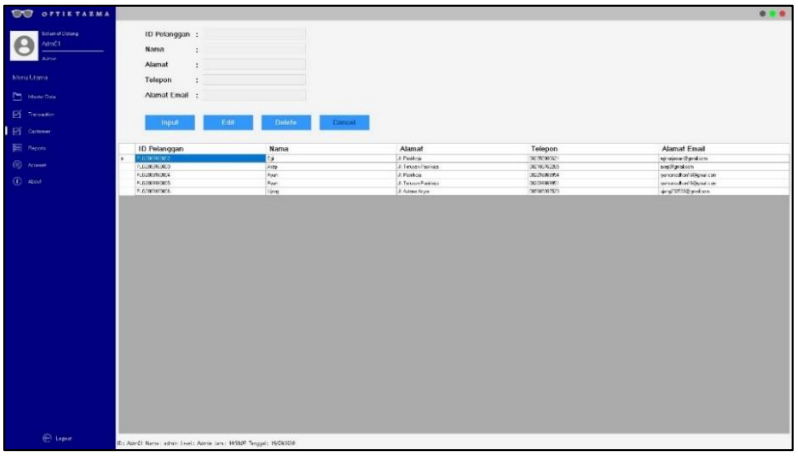
GAMBAR: 4. 15 Tampilan Menu Master Data (*Level User*)

6. Pada menu *Transaction*, sistem akan menampilkan kolom-kolom yang harus di isi ketika transaksi di lakukan, dan data yang di inputkan akan muncul pada tabel di samping form pengisian transaksi.



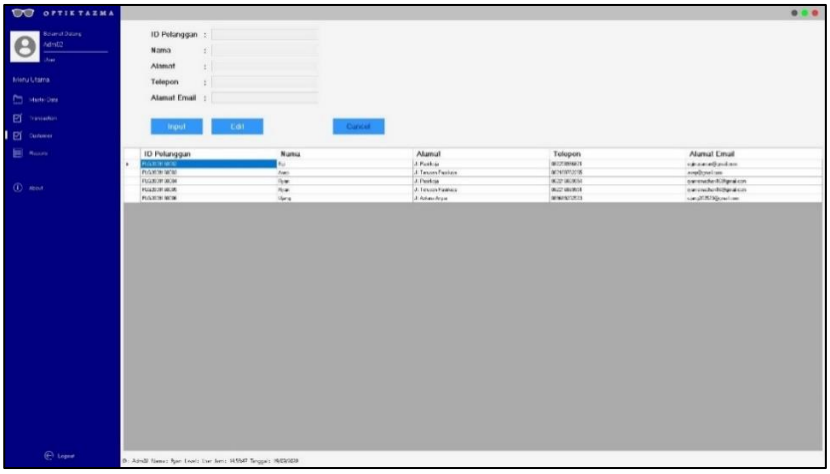
GAMBAR: 4. 16 Tampilan Menu *Transaction*

7. Pada menu *Customer*, *user* dengan *level Admin*, sistem akan menampilkan data pelanggan yang sebelumnya pernah berbelanja ataupun bisa melakukan penambahan, *update* dan *delete* terhadap data pelanggan.



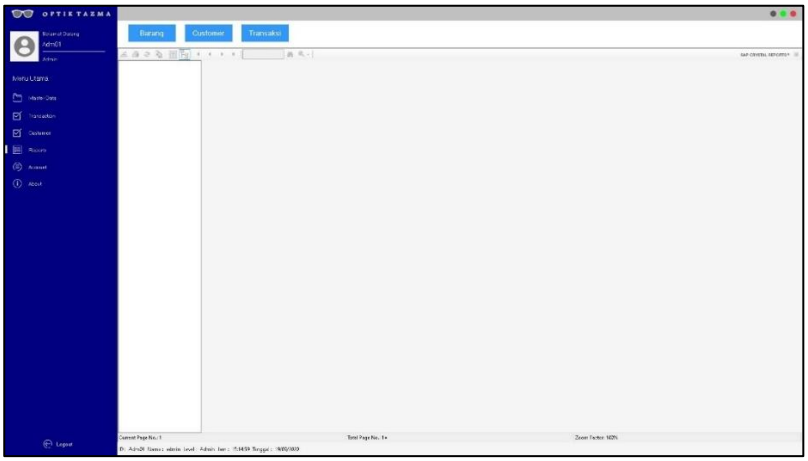
GAMBAR: 4. 17 Tampilan Menu *Customer* (*Level Admin*)

8. Pada menu *Customer*, *user* dengan *level User*, sistem akan menampilkan data pelanggan yang sebelumnya pernah berbelanja ataupun bisa melakukan penambahan dan *update* namun tidak bisa melakukan *delete* terhadap data pelanggan.



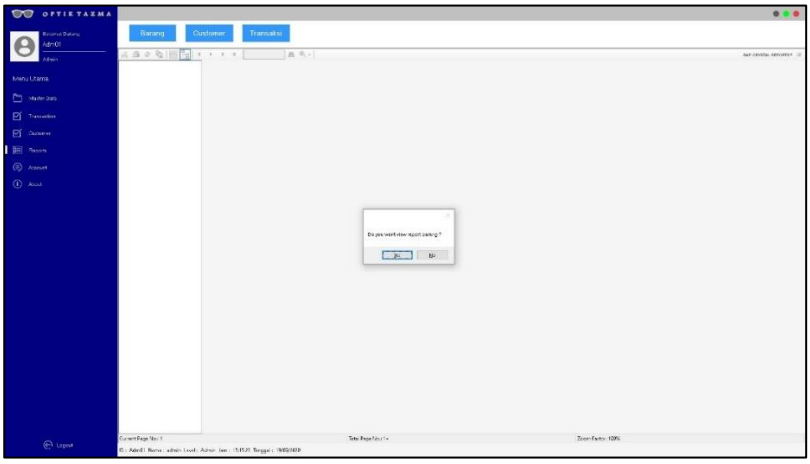
GAMBAR: 4. 18 Tampilan Menu *Customer* (*Level User*)

9. Pada menu *Report*, sistem akan menampilkan 3 (tiga) *button* serta tampilan kanvas untuk proses cetak (*print*) report.

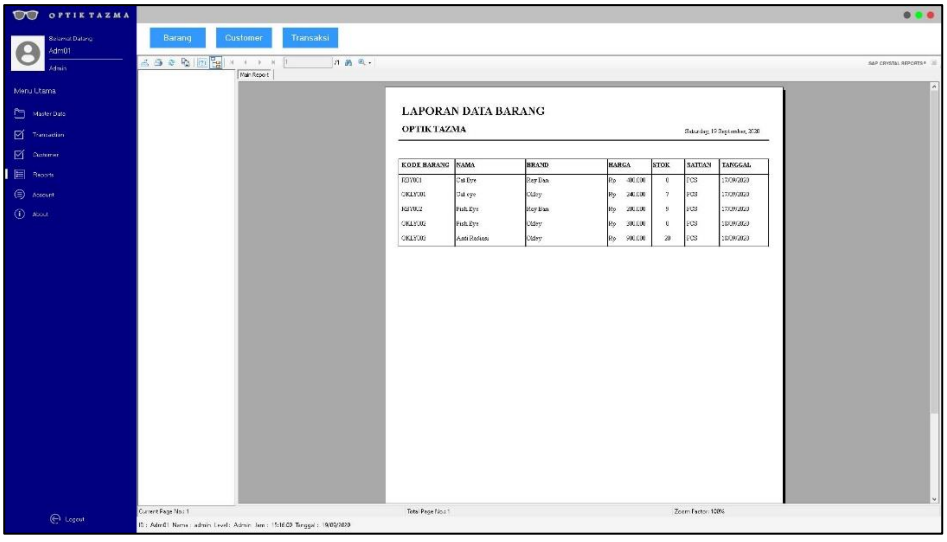


GAMBAR: 4. 19 Tampilan Menu *Report*

10. Pada menu *Report*, jika *user* menekan klik pada *button* Barang, sistem akan menampilkan validasi dengan menanyakan “Apakah ingin melihat *report* barang ?” jika pilih iya (*yes*) maka akan menampilkan *report* barang yang saat ini tersedia di menu master barang dan bisa melakukan *print out report* tersebut.

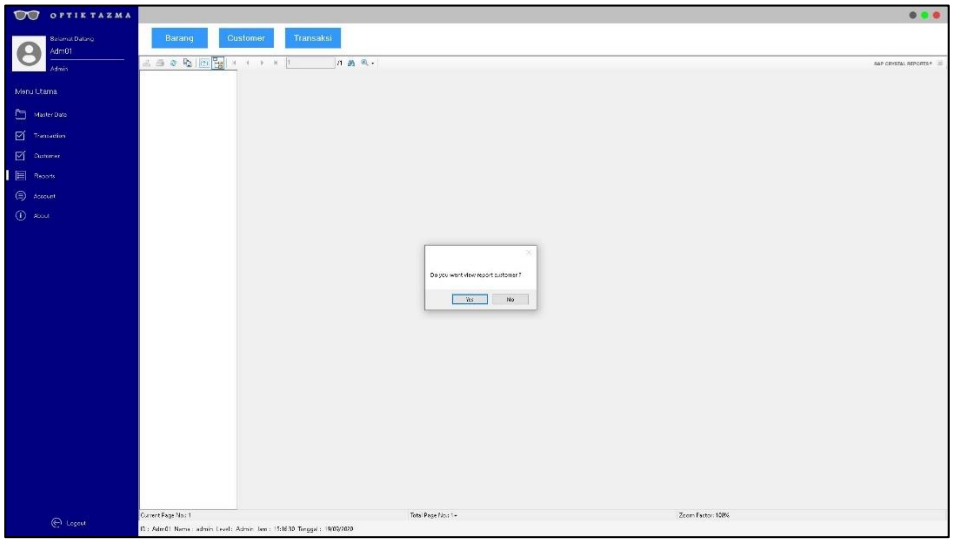


GAMBAR: 4. 20 Tampilan Menu *Report* Barang

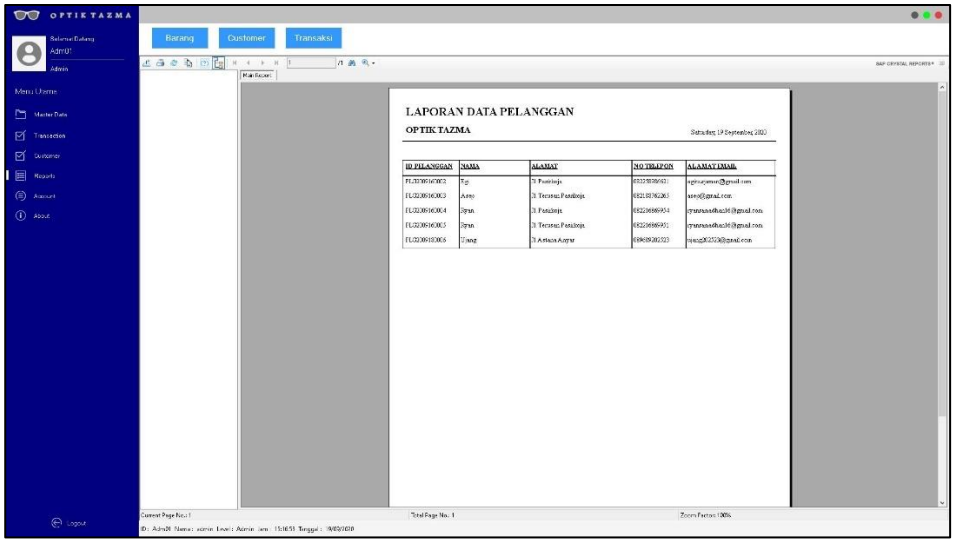


GAMBAR: 4. 21 Tampilan Menu *Report* Barang *Print*

11. Pada menu *Report*, jika *user* menekan klik pada *button Customer*, sistem akan menampilkan validasi dengan menanyakan "Apakah ingin melihat *report customer* ?" jika pilih iya (*yes*) maka akan menampilkan *report customer* yang saat ini tersedia di menu *customer* dan bisa melakukan *print out report* tersebut.

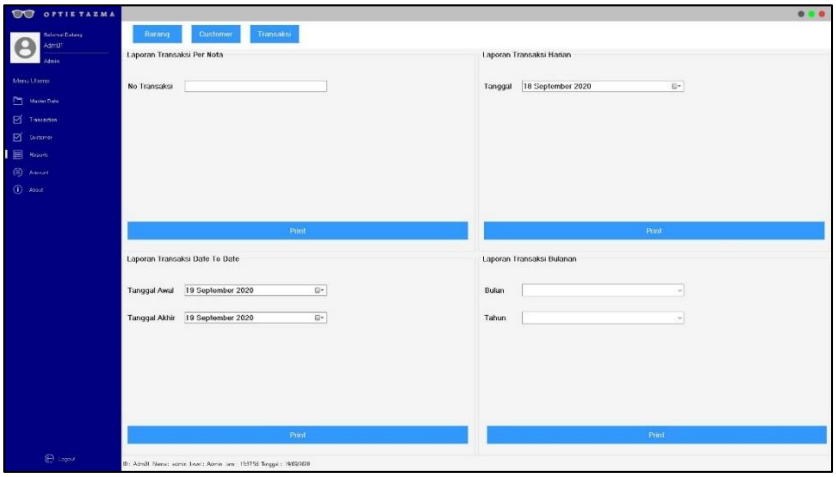


GAMBAR: 4. 22 Tampilan Menu *Report Customer*



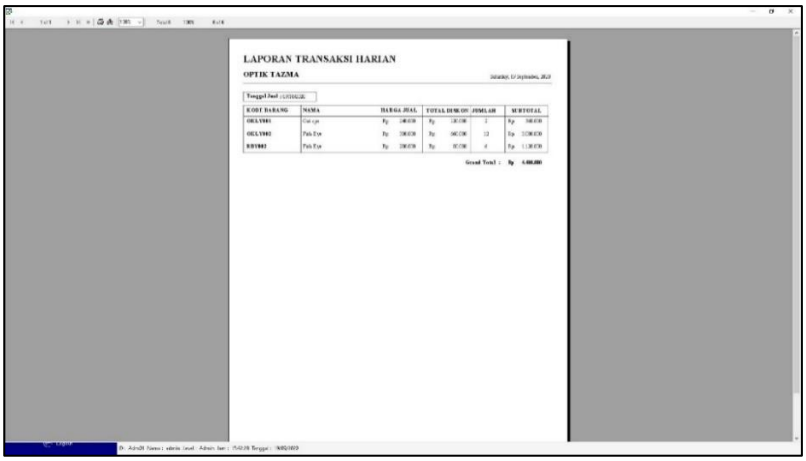
GAMBAR: 4. 23 Tampilan Menu *Report Customer Print*

12. Pada menu *Report*, jika *user* menekan klik pada *button* Transaksi, sistem akan menampilkan halaman untuk mencari report transaksi yang sebelumnya pernah dilakukan.



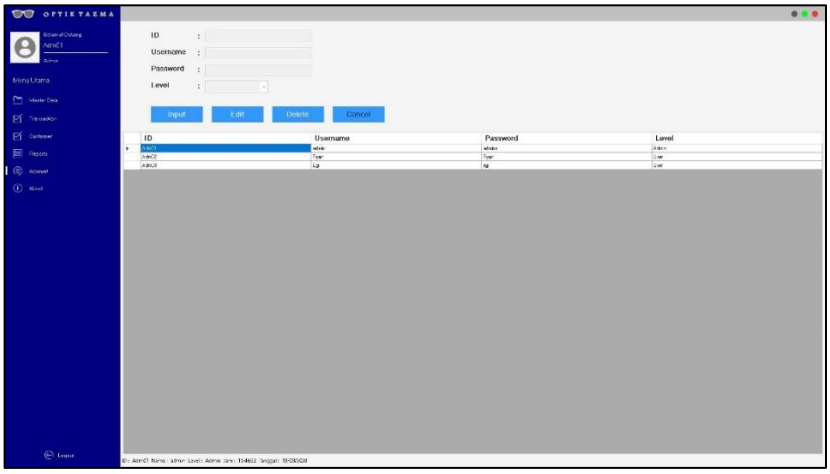
GAMBAR: 4. 24 Tampilan Menu *Report* Transaksi

13. Pada menu *Report*, jika *user* menekan klik pada *button* Transaksi, dan melakukan pencarian transaksi berdasarkan no transaksi atau berdasarkan harian, mingguan atau bulanan maka akan muncul tampilan *view report* transaksi yang akan di *print out*.



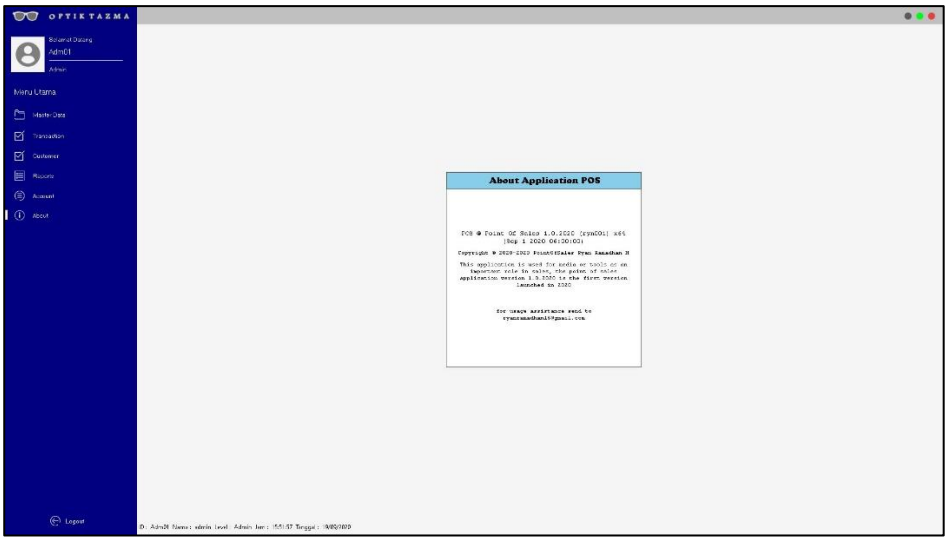
GAMBAR: 4. 25 Tampilan Menu *Report* Transaksi *Print*

14. Pada menu *Account*, *user* dengan *level Admin*, sistem akan menampilkan data akun-akun yang ada serta bisa melakukan penambahan, *update*, dan *delete* terhadap data akun.



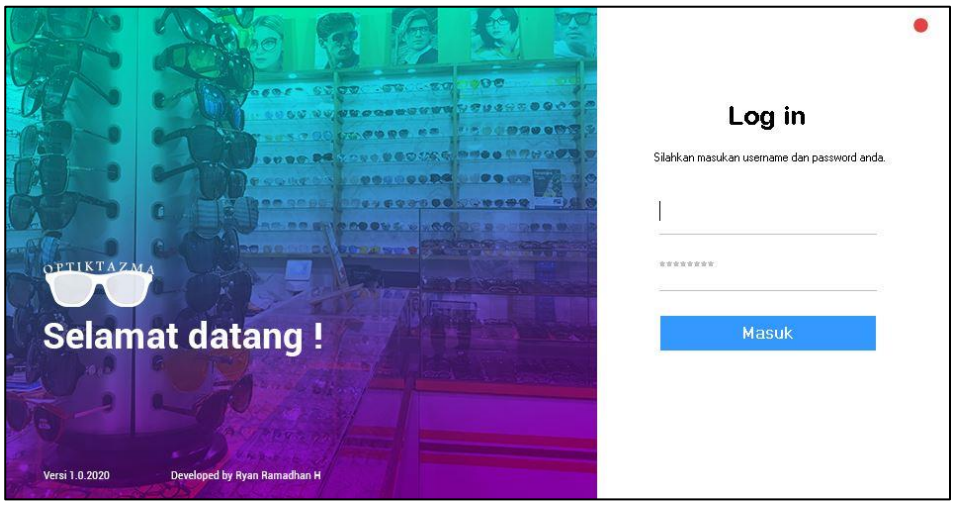
GAMBAR: 4. 26 Tampilan Menu *Account*

15. Pada menu *About*, sistem akan menampilkan seputar *developer* yang membuat aplikasi tersebut.



GAMBAR: 4. 27 Tampilan Menu *About*

16. Pada menu *Logout*, jika *user* dengan *level Admin* atau *User*, sistem akan menampilkan halaman awal yaitu halaman *form login*.



GAMBAR: 4. 28 Tampilan Halaman *Login* (Setelah klik *Logout*)

4.3. Uji Coba

Metode uji coba yang digunakan oleh penulis yaitu *Black Box Testing*. *Metode Black Box* ini menguji fungsi-fungsi yang ada pada aplikasi secara keseluruhan. Berikut adalah hasil dari pengujian yang dilakukan terhadap aplikasi ini :

1. Hasil pengujian pada *Form Login*.

TABEL: 4. 1 Pengujian *Login*

No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form Username* tidak di isi. *Test Case :* 	Muncul pesan "*Username and Password Required!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 2 Pengujian *Login* (Lanjutan)

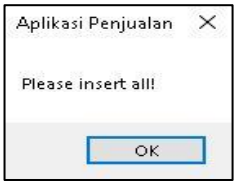
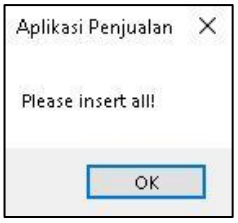
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form Username* tidak di isi. *Test Case :* 	Muncul pesan "*Username and Password Required!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
2.	*Form Password* tidak di isi. *Test Case :* 	Muncul pesan "*Username or Password is Wrong!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	*Form Username* di isi sembarang dan *password* di isi adminx. *Test Case :* 	Muncul pesan "*Username and Password Required!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	*Form Username* di isi admin dan *password* di isi sembarang. *Test Case :* 	Muncul pesan "*Username and Password Required!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 1 Pengujian *Login* (Lanjutan)

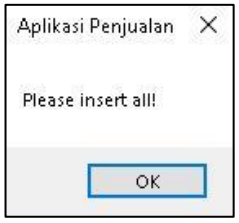
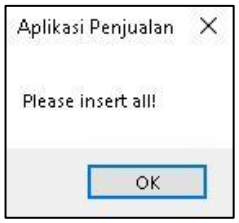
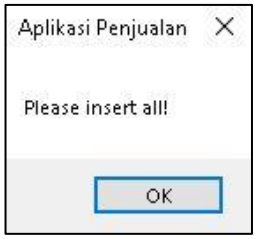
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
5.	*Form Username* di isi admin dan *Password* di isi adminx. *Test Case :* 	Muncul halaman utama aplikasi.	Sesuai yang di harapkan. *Test Result :* 	*Valid*

2. Hasil pengujian pada *Form Input* Data Barang.

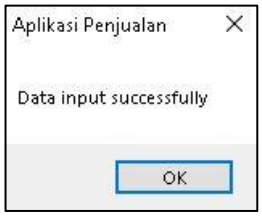
TABEL: 4. 3 Pengujian *Input* Data Barang

No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "*Please insert all*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
2.	*Form* nama barang di isi dan yang lain nya di kosongkan. *Test Case :* 	Muncul pesan "*Please insert all*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 2 Pengujian *Input* Data Barang (Lanjutan)

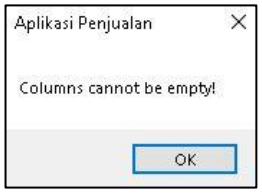
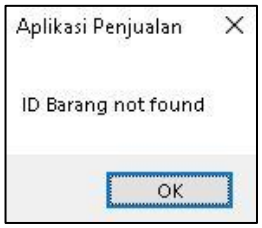
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
3.	*Form* nama barang dan brand di isi yang lain di kosongkan. *Test Case :* 	Muncul pesan "*Please insert all*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	*Form* nama barang, brand, dan *barcode* di isi yang lain di kosongkan. *Test Case :* 	Muncul pesan "*Please insert all*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
5.	*Form* nama barang, brand, *barcode*, jumlah dan satuan di isi yang lain di kosongkan. *Test Case :* 	Muncul pesan "*Please insert all*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 2 Pengujian *Input* Data Barang (Lanjutan)

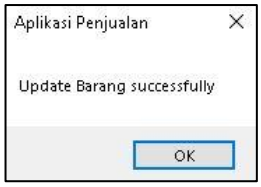
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
6.	Semua *form* di isi dengan data valid. *Test Case :* 	Muncul pesan "Data *input successfully* ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

3. Hasil pengujian pada *Form Edit* Data Barang.

TABEL: 4. 4 Pengujian *Edit* Data Barang

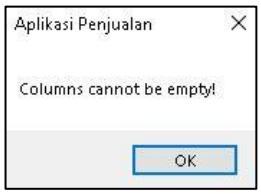
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
2.	*Form ID* Barang dimasukan data yang tidak ada. *Test Case :* 	Muncul pesan "*ID* Barang *not found*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 3 Pengujian *Edit* Data Barang (Lanjutan)

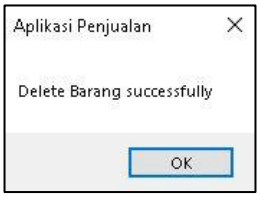
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
3.	*Form ID* Barang di isi dengan data *valid*. *Test Case :* 	Muncul semua data barang sesuai *ID* Barang yang di *input*-kan.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	*Form* jumlah barand di rubah menjadi 10. *Test Case :* 	Muncul pesan "*Update* Barang *Successfully* ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

4. Hasil pengujian pada *Form Delete* Data Barang.

TABEL: 4. 5 Pengujian *Delete* Data Barang

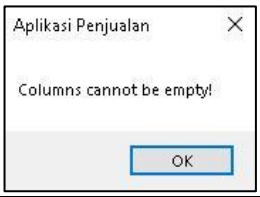
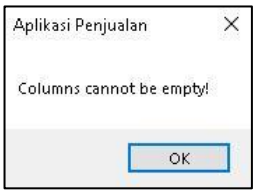
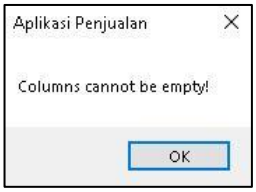
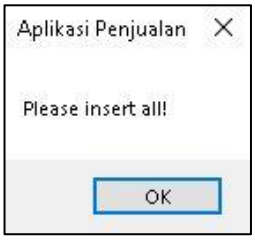
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "*Columns cannot be empty*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 6 Pengujian *Delete* Data Barang (Lanjutan)

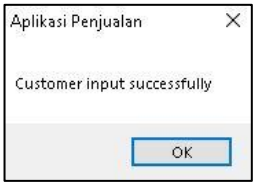
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
2.	*Form ID* Barang dimasukan data yang tidak ada. *Test Case :* 	Muncul pesan "*ID* Barang *not found*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	*Form ID* Barang di isi dengan data *valid*. *Test Case :* 	Muncul semua data barang sesuai *ID* Barang yang di *input*-kan.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	Klik Button delete *Test Case :* 	Muncul pesan "*Delete* Barang *Successfully* ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

5. Hasil pengujian pada *Form Input* Data Pelanggan.

TABEL: 4. 7 Pengujian *Input* Data Pelanggan

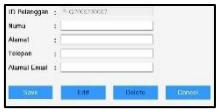
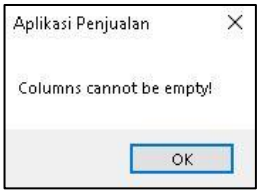
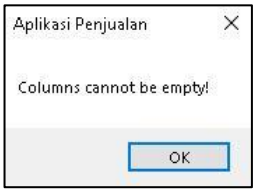
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
2.	*Form* nama di isi dan yang lain nya di kosongkan. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	*Form* nama dan alamat di isi yang lain di kosongkan. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	*Form* nama, alamat, dan telepon di isi yang lain di kosongkan. *Test Case :* 	Muncul pesan "*Please insert all!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 5 Pengujian *Input* Data Pelanggan (Lanjutan)

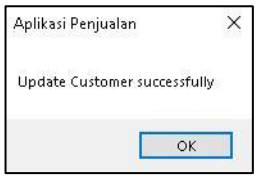
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
5.	*Form* nama, alamat, telepon dan alamat email di isi dengan data *valid*. *Test Case :* 	Muncul pesan "*Customer input successfully* ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

6. Hasil pengujian pada *Form Edit* Data Pelanggan.

TABEL: 4. 8 Pengujian *Edit* Data Pelanggan

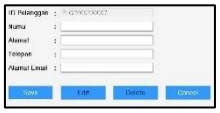
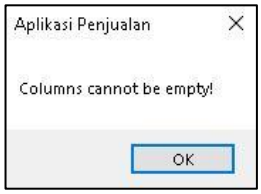
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
2.	*Form ID* Pelanggan di isi *random* dan yang lain nya di kosongkan. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 6 Pengujian *Edit* Data Pelanggan (Lanjutan)

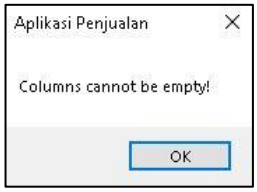
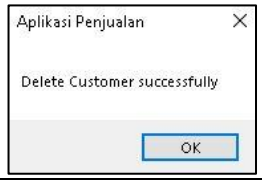
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
3.	*Form ID* Pelangan di isi dengan data valid. *Test Case :* 	Muncul data pelanggan.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	Melakukan perubahan pada *form* alamat email. *Test Case :* 	Muncul pesan "*Update customer successfully* ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

7. Hasil pengujian pada *Form Delete* Data Pelanggan.

TABEL: 4. 9 Pengujian *Delete* Data Pelanggan

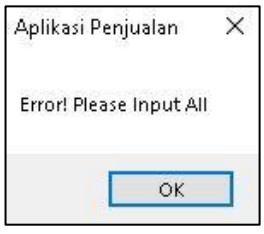
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 7 Pengujian Delete Data Pelanggan (Lanjutan)

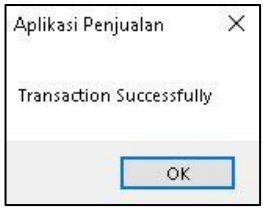
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
2.	*Form ID* Pelanggan di isi *random* dan yang lain nya di kosongkan. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	*Form ID* Pelangan di isi dengan data *valid*. *Test Case :* 	Muncul data pelanggan.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	Klik *Button* Delete. *Test Case :* 	Muncul pesan "*Delete customer successfully* ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

8. Hasil pengujian pada *Form* Transaksi.

TABEL: 4. 10 Pengujian Transaksi

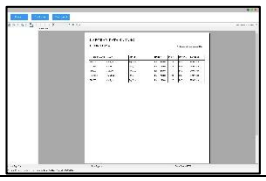
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "*Columns cannot be empty!*".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 8 Pengujian Transaksi (Lanjutan)

No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
2.	*Form ID* Pelanggan di isi dengan data valid. *Test Case :* 	Muncul data pelanggan.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	*Form* Kode Barang di isi dengan data valid. *Test Case :* 	Muncul data barang.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	*Form* jumlah barang di isi dan klik *Button Insert*. *Test Case :* 	Data tersimpan di *List*.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
5.	Memasukan jumlah uang yang di bayarkan dan klik *Button Submit.* *Test Case :* 	Muncul pesan "*Transaction successfully* ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
6.	Klik *Button Print*. *Test Case :* 	Muncul tampilan preview transaksi.	Sesuai yang di harapkan. *Test Result :* 	*Valid*

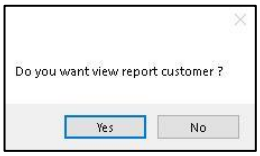
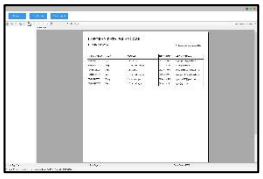
9. Hasil pengujian pada *Report* Barang.

TABEL: 4. 11 Pengujian *Report* Barang

No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	Klik button Barang di halaman *Reports*. *Test Case :* 	Muncul pesan "Do you want view *report* barang ?".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
2.	Klik yes pada pesan untuk preview data barang. *Test Case :* 	Muncul Print Preview Data Barang.	Sesuai yang di harapkan. *Test Result :* 	*Valid*

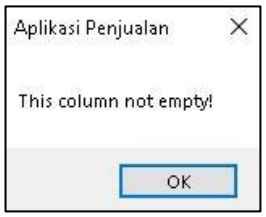
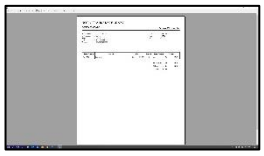
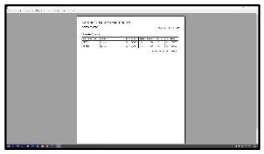
10. Hasil pengujian pada *Report* Pelanggan (*Customer*).

TABEL: 4. 12 Pengujian *Report* Pelanggan (*Customer*)

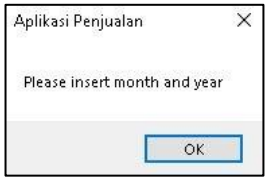
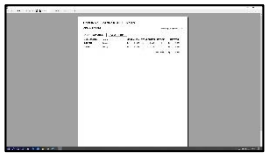
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	Klik *button Customer* di halaman *Reports*. *Test Case :* 	Muncul pesan "Do you want view *report* barang ?".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
2.	Klik yes pada pesan untuk preview data barang. *Test Case :* 	Muncul Print Preview Data Barang.	Sesuai yang di harapkan. *Test Result :* 	*Valid*

11. Hasil pengujian pada *Report* Transaksi.

TABEL: 4. 13 Pengujian *Report* Transaksi

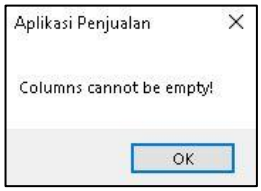
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* no transaksi di kosongkan. *Test Case :* 	Muncul pesan "This column not empty!".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
2.	*Form* no transaksi di isi dengan data valid. *Test Case :* 	Muncul Print Preview Data Transaksi.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	*Form* tanggal transaksi di isi dengan data valid. *Test Case :* 	Muncul Print Preview Data Transaksi.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	*Form* tanggal awal dan tanggal akhir di isi dengan data valid. *Test Case :* 	Muncul Print Preview Data Transaksi.	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 11 Pengujian Report Transaksi (Lanjutan)

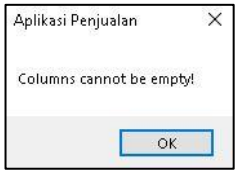
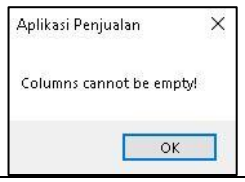
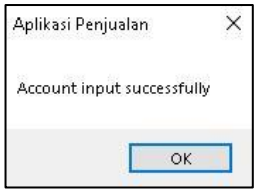
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
5.	*Form* Bulan dan Tahun dikosongkan. *Test Case :* 	Muncul pesan "Please insert month and year".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
6.	*Form* Bulan dan Tahun di isi dengan data valid. *Test Case :* 	Muncul Print Preview Data Transaksi.	Sesuai yang di harapkan. *Test Result :* 	*Valid*

12. Hasil pengujian pada *Form Input* Data *Account*.

TABEL: 4. 14 Pengujian *Input* Data *Account*

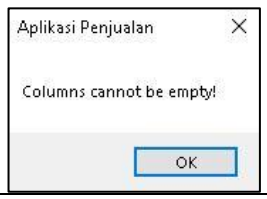
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	Form tidak di isi. *Test Case :* 	Muncul pesan "Columns cannot be empty!".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 12 Pengujian *Input* Data *Account* (Lanjutan)

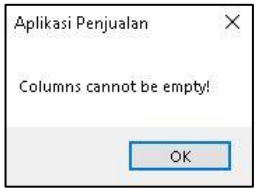
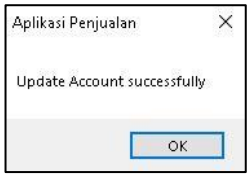
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
2.	Form username di isi dan yang lain nya di kosongkan. *Test Case :* 	Muncul pesan "Columns cannot be empty!".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	Form nama dan password di isi yang lain di kosongkan. *Test Case :* 	Muncul pesan "Columns cannot be empty!".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	Form nama, alamat, dan telepon di isi yang lain di kosongkan. *Test Case :* 	Muncul pesan "Account input successfully ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

13. Hasil pengujian pada *Form Edit* Data *Account*.

TABEL: 4. 15 Pengujian *Edit* Data *Account*

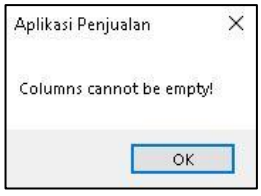
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "Columns cannot be empty!".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 13 Pengujian *Edit* Data *Account* (Lanjutan)

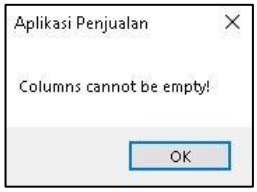
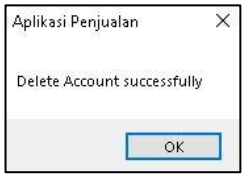
No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
2.	*Form ID* di isi random dan yang lain nya di kosongkan. *Test Case :* 	Muncul pesan "Columns cannot be empty!".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	*Form ID* di isi dengan data valid. *Test Case :* 	Muncul data *account*.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	Melakukan perubahan pada *form* nama. *Test Case :* 	Muncul pesan "Update *account* successfully ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

14. Hasil pengujian pada *Form Delete* Data *Account*.

TABEL: 4. 16 Pengujian *Delete* Data *Account*

No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
1.	*Form* tidak di isi. *Test Case :* 	Muncul pesan "Columns cannot be empty!".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

TABEL: 4. 14 Pengujian *Delete* Data *Account* (Lanjutan)

No.	Skenario Pengujian	Yang Diharapkan	Output	Kesimpulan
2.	*Form ID* di isi random dan yang lain nya di kosongkan. *Test Case :* 	Muncul pesan "Columns cannot be empty!".	Sesuai yang di harapkan. *Test Result :* 	*Valid*
3.	*Form ID* di isi dengan data valid. *Test Case :* 	Muncul data *account*.	Sesuai yang di harapkan. *Test Result :* 	*Valid*
4.	Klik *Button Remove*. *Test Case :* 	Muncul pesan "Delete *account* successfully ".	Sesuai yang di harapkan. *Test Result :* 	*Valid*

4.4. Pemeliharaan (Operation And Maintenance)

Tahap ini merupakan akhir dari metode *Waterfall*, yaitu pemeliharaan program, dimana ini adalah tugas dari pengguna aplikasi untuk mengelola dan merawat aplikasi, juga menambahkan bila ada yang ingin dirubahan atau penambahan dalam sistem baru.

BAB V

PENUTUP

5.1. Kesimpulan

Dari hasil penulisan dan penelitian yang penulis lakukan, dapat disimpulkan bahwa :

1. Aplikasi Penjualan ini atau Aplikasi POS ini dapat membantu untuk mengetahui barang apa saja yang masuk serta barang apa saja yang keluar dalam kurun waktu harian, mingguan, bulanan serta tahunan oleh pemilik optik.
2. Pemilik optic tidak perlu mengkhawatirkan proses pelayanan terhadap konsumen dikarenakan dengan Aplikasi Penjualan atau Aplikasi POS ini proses pencetakan bukti pembelian atau pembayaran bisa jalan secara otomatis dan lebih efisien tanpa harus dilakukan pencatatan secara manual.
3. Dengan adanya Aplikasi Penjualan ini atau Aplikasi POS ini pemilik optic dapat pendatapan dari hasil penjualan dengan kurun waktu tertentu dan dapat menjadikan evaluasi terhadapt barang barang yang cepat terjual dan menjadikan penghasilan kedepan lebih baik lagi.

5.2. Saran

Berdasarkan dengan apa yang telah penulis bahas, aplikasi ini masih ada beberapa kekurangan. Adapun saran dari penulis untuk pengembangan aplikasi ini adalah :

1. Sistem dapat dikembangkan lebih lanjut dengan memberikan fungsi *Retur* Barang.
2. Sistem dapat dikembangkan lebih lanjut dengan menambahkan fungsi cetak report tahunan atau berdasarkan dari tahun ke tahun.
3. Sistem dapat dikembangkan dengan adanya proses atau masukan data baru sebagai pendukung dari penjualan, yaitu adanya Data Lensa Kacamata.

DAFTAR PUSTAKA

Abidin, I.Z. and Putro, H.P., 2020. Penerapan MVC dalam Pengembangan Sistem *Point of Sale* (Studi Kasus TPOS PT. Java Signa Intermedia). *AUTOMATA*, *1*(2).

Cahyodi, S.C. and Arifin, R.W., 2017. Sistem Informasi *Point Of Sales* Berbasis Web Pada Colony Amaranta Bekasi. *INFORMATION SYSTEM FOR EDUCATORS AND PROFESSIONALS: Journal of Information System*, *1*(2), pp.189-204.

Juansyah, A. (2015). Pembangunan Aplikasi *Child Tracker* Berbasis *Assisted–Global Positioning System* (A-Gps) Dengan *Platform Android*. Jurnal Ilmiah Komputer dan Informatika (KOMPUTA), 1(1), 2-3.

Koyuko, H., Sinsuw, A. A., & Najoan, X. B. (2016). Perancangan Aplikasi *Monitoring* Pemadaman Listrik Berbasis *Android* Studi kasus PT. PLN area Manado. Jurnal Teknik Informatika, 9(1).

Marisa, F. and Yuarita, T.G., 2017. Perancangan Aplikasi *Point of Sales* (POS) Berbasis Web Menggunakan Metode Siklus Hidup Pengembangan Sistem. *Jurnal Teknologi dan Manajemen Informatika*, *3*(2).

Nopriansyah, D., 2019. Pengertian *Borland Delphi* Dan Contohnya.

Permana, S.D.H., 2015. Analisa Dan Perancangan Aplikasi *Point Of Sale* (POS) Untuk Mendukung Manajemen Hubungan Pelanggan. *Jurnal Teknologi Informasi dan Ilmu Komputer*, *2*(1), pp.20-28.

Saputra, M.R. and Riyadi, S., 2019. Sistem Informasi Populasi Dan Historikal *Unit* Alat-alat Berat Pada PT. Daya Kobelco Construction Machinery Indonesia. *Jurnal Penelitian Dosen FIKOM (UNDA)*, *6*(2).

Sri, R. and Dwirgo, S., 2017. Aplikasi *Point Of Sale* Berbasis Web Menggunakan *Framework Codeigniter* Pada Martabak ABC. *Makalah Ilmiah Mahasiswa*.

Sugianto, Y. and Tjandra, S., 2016. Aplikasi *Point Of Sale* Pada Toko *Retail* Dengan Menggunakan *Dynamic Software Development Method*. *J. Ilm. Teknol. dan Rekayasa*, *8*(1), pp.1-8.

Tristianto, C., 2018, July. Penggunaan Metode *Waterfall* Untuk Pengembangan Sistem *Monitoring* Dan Evaluasi Pembangunan Pedesaan. In *ESIT* (Vol. 12, No. 1, pp. 8-22).

Wahyudi, I.K.A.B., Putra, I.A.W. and Datya, A.I., 2018. Aplikasi Penjualan *Point Of Sale* (POS) Menggunakan *Barcode* Pada Koperasi Bina Kasih Sejahtera Berbasis *Desktop* Dengan Metode *First In First Out* (FIFO). *Jurnal Teknologi Informasi dan Komputer*, *3*(2).

Wiguna, P.D.A., Swastika, I.P.A. and Satwika, I.P., 2018. Rancang Bangun Aplikasi *Point of Sales Distro Management System* dengan Menggunakan Framework React Native. *Jurnal Nasional Teknologi dan Sistem Informasi*, *4*(3), pp.149-159.

Wikipedia Indonesia, https://id.wikipedia.org/wiki/Microsoft_Visual_Studio, Diakses 10 Agustus 2020.

LAMPIRAN

Surat Keterangan Penelitian

SURAT KETERANGAN

Yang bertanda tangan dibawah ini :

Nama : Ryan Ramadhan Harahap.

NIM : 361762001.

Program Studi : Teknik Informatika – S.1

Perguruan Tinggi : STMIK – Indonesia Mandiri.

Telah melaksanakan penelitian di Optik Tazma selama 2 (dua) bulan terhitung mulai 1 Agustus 2020 s/d 23 September 2020.

Demikian Surat Keterangan ini dibuat untuk digunakan sebagaimana mestinya.

Bandung, 23 September 2020

Pemilik Optik,

OPTIK TAZMA
JL. ALHATERI NO. 45 BANDUNG
022 20505138 / 081250383914

Egi Nurjaman, Amd.RO