SISTEM PENGELOLAAN TRANSAKSI TOKO PERHIASAN PERAK BERBASIS DESKTOP

(Studi Kasus: Toko Perak Beben Banjarmasin)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Nadya Ramadana
NIM: 145150207111139

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

SISTEM PENGELOLAAN TRANSAKSI PERHIASAN PERAK BERBASIS *DESKTOP*
(Studi Kasus : Toko Perak Beben Banjarmasin)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Nadya Ramadana
NIM: 145150207111139

Skripsi ini telah diuji dan dinyatakan lulus pada
7 Juni 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Denny Sagita R., S.Kom, M.Kom	Bayu Priyambadha, S.Kom, M.Kom
NIP: 19851124 201504 1 001	NIP: 19820909 200812 1 004

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 27 Juni 2018

Nadya Ramadana

NIM: 145150207111139

KATA PENGANTAR

Assalamu'alaikum Wr. Wb. Puji dan syukur penulis panjatkan atas rahmat dan karunia Allah SWT, penulis dapat menyelesaikan skripsi ini yang berjudul Sistem Pengelolaan Transaksi Perhiasan Perak Berbasis Desktop. Skripsi ini diajukan sebagai syarat untuk mendapatkan gelar Sarjana Komputer di Fakultas Ilmu Komputer Universitas Brawijaya.

Selama proses penyusunan skripsi, penulis banyak belajar dan mengaplikasikan ilmu yang telah diperoleh dari perkuliahan. Semoga penulis dapat terus mengembangkan ilmu yang diperoleh selama perkuliaahan dengan baik.

Dalam penyusunan skripsi ini penulis mendapat banyak dukungan dan bantuan dari berbagai pihak. Oleh sebab itu, penulis ingin mengucapkan terimakasih kepada:

1. Bapak Denny Sagita R., S.Kom, M.Kom selaku Pembimbing I, yang telah membimbing dan mengarahkan penulis dalam proses penyelesaian skripsi
2. Bapak Bayu Priyambadha, S.Kom, M.Kom selaku Pembimbing II, yang telah membimbing dan mengarahkan penulis sehingga skripsi ini telah selesai.
3. Kedua orang tua saya, kakak dan adik saya yang selalu mendukung saya.
4. Bapak dan Ibu dosen serta staff Fakultas Ilmu Komputer Universitas Brawijaya yang telah bersedia membagi ilmu dan arahan kepada penulis.
5. Seluruh keluarga besar penulis yang telah memberikan dukungan kepada penulis.
6. Meylisa D. Marali, Khairinnisa R., Nirmala Faiza S., Tuti Wardani H., Shindy Maria Ulfa, Hilman Nihri, Uis Yudha, Landika H. Suganda, Binariyanto Aji serta teman-teman *Private Class* L yang selalu memotivasi, mendukung dan membantu penulis dalam menyelesaikan skripsi.
7. Mahardhika Hendra Bagaskara yang selalu memberikan semangat, arahan dan motivasi kepada penulis.
8. Aditya Wisnu, Devara Fikry A, Ahmad Kamil A., Ahmad Hadori, M. Syaifudin Abdullah dan teman-teman mahasiswa keminatan Rekayasa Perangkat Lunak yang lain yang telah memberikan ilmu, masukkan, saran dan motivasi.
9. Teman-teman PKL penulis yang selalu mendukung dan memberikan arahan kepada penulis.
10. Nurainie Putri, Lia Novi A., Bella Ayu Islamiyah, Dita Endah, Darin, Noerhanani dan seluruh teman-teman kos saya yang selalu memberikan dukungan dan bantuan kepada penulis.
11. Keluarga Pengpro dan PIP yang selalu menjadi penyemangat bagi penulis.
12. Rekan-rekan EMIF (Eksekutif Mahasiswa Informatika) Fakultas Ilmu Komputer yang telah mendukung dan memotivasi.
13. Rekan-rekan Forum Komunkasi Mahasiswa Banjarmasin (FKMB) yang memberikan saran dan motivasi kepada penulis.

14. Teman-teman Fakultas Ilmu Komputer yang telah membantu selama masa studi penulis.
15. Semua pihak yang tidak dapat disebutkan satu – persatu.

Penulis menyadari dalam proses penyusunan skripsi masih banyak terdapat kekurangan, oleh sebab itu kritik dan saran yang membangun sangat diperlukan dalam penulisan selanjutnya. Penulis berharap semoga skripsi ini memberikan manfaat kepada pihak yang membutuhkan.

Malang, 17 Mei 2018

Penulis

nadyaramadana@gmail.com

ABSTRAK

Toko Perhiasan Beben adalah toko yang bergerak di bidang jual beli perhiasan perak baik perak mentah, maupun perak yang sudah diolah menjadi perhiasan. Perhiasan yang dijual beraneka macam, baik berupa gelang tangan, kalung, anting, cincin, gelang kaki, dan bros. Terdapat 2 transaksi yang terjadi di Toko Perhiasan Beben yaitu transaksi penjualan dan pembelian. Proses transaksi penjualan dan pembelian yang ada pada toko perhiasan ini masih dilakukan secara konvensional, belum memiliki laporan keuangan, belum memiliki laporan stok barang, dan pemilik Toko Beben menginginkan adanya *reminder* jika sales dari toko beben belum membayar lunas transaksi pembelian barang ke toko beben dalam jangka waktu 30 hari. Untuk menangani masalah tersebut dibuatlah solusi yang dapat diterapkan yaitu membuat sistem pengelolaan transaksi perhiasan perak berbasis desktop dengan fitur utama yaitu mengelola transaksi penjualan dan pembelian, pengelolaan stok barang, laporan keuangan, pembayaran sales. Penelitian ini dimulai dengan melakukan analisis kebutuhan dimana terdapat 28 kebutuhan fungsional sistem. Kemudian melakukan tahapan perancangan dan implementasi. Dalam penelitian ini sistem dikembangkan dengan menggunakan bahasa java, pola perancangan yang digunakan adalah MVC (*Model, View, Controller*) dengan menggunakan *database* MYSQL. Setelah melakukan implementasi kemudian dilakukan tahap pengujian dengan metode pengujian yang dilakukan dalam sistem ini adalah *whitebox* dan *blackbox.* Pengujian menggunakan metode *white box* menggunakan pengujian unit, sedangkan untuk pengujian *blackbox* menggunakan dan pengujian validasi. Hasil dari pengujian sistem yang dibangun menghasilkan 100% valid dimana sistem yang dibuat telah sesuai dengan hasil analisis dan perancangan sebelumnya.

Kata kunci: *java*, perhiasan, perak, *netbeans,* desktop

ABSTRACT

Toko Perhiasan Beben is a store that sells and buys silver either as silver raw materials or jewelry goods. Silver jewelry goods that Toko Perhiasan Beben sells such as bracelets, necklaces, ear rings, rings, ankle band and brooches. In this store, there is two transactions available, which is selling goods and buying goods. transactions in this store still are handled in conventional ways, no report about financial, and no report on quantity of goods. Besides selling goods on its own store, Toko Perhiasan Beben also has third party seller (sales). The owner wants to have reminder for the sales to pay goods that he/she buys when he/she can't pay up in 30 days. To solve these problems, I make suggestion to implement a management system on silver jewelry goods based on desktop. With primal function in selling and buying management transactions, stocks management, financial statements and sales payment. This research is started by requirements gathering which analyzed to 28 functional needs from the system. The next step is to make a design and implementation. System is implemented using java language and Model, View, Controller (MVC) as design pattern with database MySQL. After implementation is finished, the system will be tested using whitebox testing and blackbox testing. The whitebox testing will be done using unit testing and the blackbox testing will be done using validation testing. Testing result of this implemented system is 100 percent valid as the system is compatible with analysis and design result.

Keywords: Java, jewelry, silver, netbeans, desktop

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Di Kalimantan, perhiasan merupakan hal yang banyak di minati. Salah satu toko yang menjual perhiasan adalah Toko Perhiasan Beben yang ada di Banjarmasin. Toko Perhiasan Beben adalah toko yang bergerak di bidang jual beli perhiasan perak baik perak mentah, maupun perak yang sudah diolah menjadi perhiasan seperti gelang tangan, kalung, anting, cincin, gelang kaki, dan bros. Terdapat 2 jenis transaksi yang terjadi di Toko Perhiasan Beben yaitu transaksi penjualan dan pembelian.

Dalam proses pembuatan nota transaksi penjualan pada Toko Perhiasan Beben masih dilakukan secara konvensional yaitu dengan mencatat barang yang dijual pada nota kemudian melakukan *cross check* untuk meminimalisir kesalahan. Jika terjadi kesalahan saat mencatat barang penjualan maka akan dilakukan *cross check* untuk memperbaiki kesalahan. Hal ini dapat mengakibatkan pembatalan transaksi karena proses transaksi yang lama.

Selain itu, pembayaran transaksi penjualan pada Toko Perhiasan Beben harus dibayarkan lunas kecuali terhadap sales, dimana sales dapat mencicil pembayaran selama 30 hari. Pemilik Toko Perhiasan Beben menginginkan adanya *reminder* untuk mengingatkan pemilik atau pegawai jika sales belum membayar lunas sampai 30 hari setelah pembelian.

Pemilik Toko Perhiasan Beben tidak memiliki catatan laporan keuangan. Hal tersebut menyebabkan pemilik Toko Perhiasan Beben tidak dapat mengetahui keuntungan, kerugian, jumlah uang yang telah masuk dan yang telah dikeluarkan oleh Toko Perhiasan Beben.

Selain itu, pemilik Toko Perhiasan Beben tidak memiliki catatan jumlah stok barang. Hal ini dapat mengakibatkan pemilik toko membeli perhiasan lagi padahal perhiasan yang dibeli masih banyak tersedia di Toko Perhiasan Beben. Pemilik atau pegawai mengira suatu barang telah habis padahal masih tersedia atau mengira barang tersedia padahal telah habis yang mengakibatkan kurang efektifnya waktu pencarian barang dan lambatnya barang terjual.

Dari uraian permasalahan, dapat diketahui bahwa seluruh proses transaksi yang terjadi di Toko Perhiasan Beben sampai saat ini masih belum terorganisir dengan baik. Untuk mengurangi permasalahan tersebut maka dibuatlah dibuatlah sistem pengelolaan transaksi pada Toko Perhiasan Beben yang dapat mengelola transaksi jual beli perhiasan, *reminder* pembayaran sales belum lunas, informasi stok barang, dan laporan keuangan. Dengan adanya sistem ini diharapkan dapat meningkatkan efektivitas, efisiensi dan kemudahan transaksi dan pelayanan di toko tersebut. Sistem yang akan dikembangkan menggunakan pemrograman berbasis *desktop*.

1.2 Rumusan masalah

Dari uraian latar belakang di atas, maka didapatkan rumusan masalah untuk Sistem Pengelolaan Transaksi Toko Perhiasan Perak Beben adalah sebagai berikut.

1. Bagaimana hasil analisis kebutuhan Sistem Pengelolaan Transaksi Toko Perhiasan Perak?
2. Bagaimana hasil rancangan Sistem Pengelolaan Transaksi Toko Perhiasan Perak yang sesuai dengan kebutuhan pengguna?
3. Bagaimana hasil implementasi Sistem Pengelolaan Transaksi Toko Perhiasan Perak?
4. Bagaimana hasil pengujian Sistem Pengelolaan Transaksi Toko Perhiasan Perak?

1.3 Tujuan

Tujuan dari penelitian ini adalah sebagai berikut:

1. Menganalisis kebutuhan Sistem Pengelolaan Transaksi Toko Perak sesuai kebutuhan pengguna.
2. Merancang Sistem Pengelolaan Transaksi Toko Perhiasan Perak yang sesuai dengan hasil analisis kebutuhan.
3. Membuat Sistem Pengelolaan Transaksi Toko Perhiasan Perak.
4. Menguji Sistem Pengelolaan Transaksi Toko Perhiasan Perhiasan Perak.

1.4 Manfaat

Dapat mengelola transaksi penjualan dan pembelian secara lebih terorganisir dengan menggunakan aplikasi berbasis *desktop*.

1.5 Batasan masalah

Lingkungan Sistem Pengelolaan Transaksi Toko Perhiasan Perak yang dibatasi hanya digunakan pada lingkup Toko Perhiasan Beben dan dibatasi pada pengelolaan transaksi penjualan, transaksi pembelian, informasi stok barang, dan *reminder* pembayaran sales. Sistem ini hanya dapat diakses oleh pemilik dan pegawai.

1.6 Sistematika pembahasan

Sistematika pembahasan deskripsi singkat bagian-bagian yang terdapat dalam skripsi ini adalah sebagai berikut :

BAB I **PENDAHULUAN**

Bab ini menjelaskan mengenai latar belakang skripsi, rumusan masalah, tujuan, manfaat, batasan masalah dan sistematika pembahasan.

BAB II **TINJAUAN PUSTAKA**

Bab ini menjelaskan dan menguraikan teori-teori dari penelitian terdahulu yang dikaji sebagai bahan pendukung untuk mendasari penelitian.

BAB III **METODE PENELITIAN**

Bab ini menjelaskan metode dan langkah kerja yang dilakukan dalam proses perancangan, analisis kebutuhan, dan implementasi yang menjadi objek studi kasus skripsi.

BAB IV **ANALISIS KEBUTUHAN**

Bab ini menguraikan tentang analisis kebutuhan dan perancangan sistem yang akan dibangun.

BAB V **PERANCANGAN DAN IMPLEMENTASI**

Bab ini menguraikan proses perancangan sesuai analisis kebutuhan dan implementasi sesuai dari perancangan sistem.

BAB VI : **PENGUJIAN DAN ANALISIS**

Bab ini menguraikan proses pengujian dan analisis terhadap sistem yang telah dibangun.

BAB VII **: PENUTUP**

Bab ini berisi kesimpulan dan saran-saran untuk pengembangan lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian pustaka

Kajian pustaka yang digunakan berisi tentang penjelasan penelitian-penelitian sebelumnya yang telah dilakukan dan membahas tema sejenis. Terdapat tiga penelitian yang digunakan sebagai bahan untuk menganalisis kebutuhan sistem. Penelitian pertama adalah penelitian yang dilakukan oleh Masriadi. Penelitian ini berjudul "Aplikasi Pengelolaan Transaksi Penjualan Perhiasan pada Toko Emas Pasamaan Indah Kabupaten Pasamaan Barat". Penelitian ini menjelaskan mengenai pengembangan Sistem Informasi Pengolahan Data Penjualan Perhiasan yang dapat membantu Toko Emas Pasamaan Indah. Pada Toko Pesaman Indah masih menggunakan cara manual yaitu masih menggunakan pencatatan kertas untuk mengitung transaksi penjualan, pesanan mapun dalam pembuatan laporan penjualan. Sistem Transaksi Penjualan Perhiasan yang dibangun, berfungsi untuk menangani proses pengolahan data penjualan dan laporan penjualan perhiasan emas. Pada kesimpulan penelitian ini menjelaskan bahwa pengelolaan data yang dilakukan semula secara manual menjadi terkomputerisasi dapat mempermudah pengelolaan data sehingga dapat dilakukan secara cepat dan akurat, serta informasi yang dihasilkan dalam laporan lebih lengkap (Masriadi, 2017).

Penelitian kedua berjudul Sistem Informasi Penjualan, Pemesanan dan Pembelian pada Toko Cahaya Murni Silver Pacitan. Penelitian ini dilakukan oleh Rizky Fandora, Bambang Eka Purnama dan Sukadi. Dalam penelitian ini menjelaskan Toko Cahaya Murni Silver dalam proses penjualan, pemesanan dan pembeliannya masih dilakukan secara konvensional dengan menggunakan alat tulis, kalkulator, dan nota kertas. Sedangkan untuk laporannya menggunakan salinan nota-nota kertas yang digunakan tiap harinya. Oleh karena itu dibuatlah sistem untuk menyelesaikan permasalahan transaksi penjualan, pembelian dan pemesanan pada toko Cahaya Murni Silver. Sistem yang dikembangkan berbasis web. Kesimpulan dari penelitian ini adalah sistem informasi pada Toko Cahaya Murni Silver memberikan kemudahan dalam mengolah data penjualan, mengolah data pembelian, mengolah data pemesanan, pembuatan laporan, dan keamanan data lebih terjamin dan meningkatkan efisiensi kerja (Fandora, et al., 2013).

Penelitian ketiga adalah penelitian yang dilakukan oleh Iqbal H. Sarker dan K. Apu yang berjudul "*MVC Architecture Driven Design and Implementation of Java Framework for Developing Desktop Application"*. Dalam penelitian ini menjelaskan mengenai mengurangi permasalahan tingginya kompleksitas dan rendahnya fleksibilitas dalam aplikasi berbasis java dengan menggunakan framework *Model View Controller (MVC)*. Dalam konsep MVC, memisahkan antara modul model, view dan controller. Kelebihan dari MVC adalah untuk aplikasi interaktif dapat menampilkan banyak representasi dari informasi yang sama, dapat dilakukan penggunaan kode kembali, dan membantu *developer* untuk fokus pada fitur tertentu. Dalam kesimpulan penelitian ini menjelaskan bahwa *framework MVC*

stabil, efisien, dan mampu mengembangkan aplikasi berkualitas tinggi (Sarker & K., 2014) .

2.2 Toko Beben

Toko Beben adalah toko perhiasan perak yang ada di Banjarmasin. Toko ini menjual perak mentah maupun perak yang telah berbentuk perhiasan. Toko ini melayani jual-beli perhiasan perak baik grosir maupun eceran. Toko ini memiliki 4 orang pegawai dan 6 orang sales yang menjualkan barang perhiasannya ke daerah-daerah di Kalimantan. Dalam menghitung harga dari perhiasan ataupun perak mentah yang dijual dengan harga pergram.

Toko Beben menjual perhiasan kepada 3 jenis pembeli yaitu konsumen, *reseller,* dan sales dimana harga pergram barang yang jual berbeda-beda sesuai dengan status pembeli tersebut. Harga untuk konsumen sesuai dengan harga ecer, harga untuk *reseller* sesuai dengan harga grosir, dan harga untuk sales sesuai dengan harga grosir dan dipotong 0,07 persen. Transaksi penjualan ke konsumen dan *reseller* harus dibayar lunas sedangkan ke sales dapat dicicil hingga 30 hari setelah pembelian.

Transaksi pembelian Toko Beben menerima barang dari konsumen, *reseller*, distributor, dan sales. Konsumen mendapat potongan Rp.6000,00/g dari harga/g pembelian. Jika *reseller* menjual maka barang tersebut dihargai dengan harga 5000/g. Jika Toko Beben membeli barang pada distributor dan sales maka harga sesuai kesepakatan dengan Toko Beben.

Toko Beben selama ini tidak memiliki laporan keuangan dan informasi stok barang. Tidak adanya laporan keuangan menyebabkan pemilik Toko Beben tidak dapat mengetahui keuntungan, kerugian, uang yang dikeluarkan (debit), dan uang yang diterima (kredit). Sedangkan tidak adanya informasi stok barang menyebabkan dapat terjadinya pembelian jenis barang yang sama padahal stok barang tersebut masih ada dan dapat menyebabkan pemilik atau pegawai mencari barang di toko tersebut padahal tidak ada.

2.3 Rekayasa perangkat lunak

Rekayasa perangkat lunak adalah pembentukan dan penggunaan prinsip-prinsip teknis untuk memperoleh perangkat lunak yang ekonomis, dapat dipercaya, dan bekerja secara efisien pada mesin (Pressman, 2001). Rekayasa perangkat lunak merupakan disiplin teknik yang berkaitan dengan semua aspek produksi perangkat lunak dari tahap awal spesifikasi sistem hingga *maintaining* setelah sistem digunakan (Sommerville, 2011).

Menurut (Sommerville, 2011), pendekatan sistematis yang digunakan dalam rekayasa perangkat lunak disebut juga *software process*. *Software* process adalah rangkaian aktivitas untuk membuat *software product. Software process* terdiri atas 4 kegiatan utama yang umum, yaitu:

1. *Software specification*, tahapan menentukan perangkat lunak yang akan dikembangkan beserta batasan-batasan operasinya.

2. *Software development*, tahapan merancang dan membangun perangkat lunak.
3. *Software validation*, tahapan pengujian perangkat lunak untuk memastikan kesesuaiannya dengan apa yang diinginkan pembeli.
4. *Software evolution,* tahapan perubahan terhadap perangkat lunak untuk menyesuaikan dengan perubahan permintaan pembeli dan pasar.

2.4 Software development life cycle

SDLC (*Software Development Life Cycle)* adalah proses yang digunakan industri perangkat lunak untuk mendesain, mengembangkan dan menguji kualitas perangkat lunak. SDLC bertujuan untuk untuk menghasilkan kualitas perangkat lunak yang tinggi yang sesuai atau melebihi harapan *customer,* menyelesaikannya dalam waktu dan biaya yang telah diperkirakan (Sharma & Singh, 2016).

SDLC (*Software Development Life Cycle)* adalah proses untuk mengembangkan perangkat lunak. Proses ini dibagi menjadi beberapa fase sebagai berikut:

1. *Requirement analysis* (Analisis kebutuhan)

   Requirement Analysis adalah fase awal dari SDLC. Tujuan dari fase ini adalah untuk memahami kebutuhan *client* dan mendokumentasikannya dengan benar. Dalam fase ini menekankan pada identifikasi apa yang dibutuhkan dari sistem. Ini adalah fase yang krusial dalam SDLC. Hasil dari fase analisis kebutuhan adalah spesifikasi kebutuhan perangkat lunak.

2. *Design* (Desain)

   Design adalah fase paling kreatif dalam SDLC. Tujuan dari fase ini adalah mentransformasikan spesifikasi kebutuhan ke dalam struktur. Hasil dari fase ini adalah *software design document* (SDD).

3. *Coding*

   Coding adalah fase SDD dirubah kedalam kode dengan menggunakan beberapa Bahasa pemrograman. Ini adalah fase logis dari SDLC. Hasil dari fase ini adalah kode program.

4. *Testing* (Pengujian)

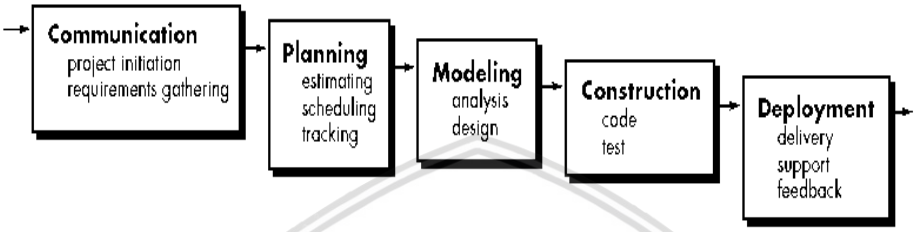
   Testing adalah fase yang sangat penting. Pengujian yang efektif akan menghasilkan produk perangkat lunak yang memliki hasil dengan kualitas tinggi, menyenangkan pengguna, biaya *maintenance* rendah, lebih akurat, dan dapat diandalkan.

5. *Maintenance* (Pemeliharaan)

   Maintenance adalah fase yang terjadi setelah penyamapaian produk ke *client*. Jika terdapat *error* yang terjadi atau perlu adanya modifkasi maka akan diimplementasikan pada fase ini (Kumar, et al., 2013).

2.4.1 Waterfall model

Menurut Pressman (2010), *waterfall model* atau terkadang disebut *classic life cycle* bersifat sistematis dengan menggunakan pendekatan sekuensial. Pengembangan perangkat lunak dimulai dengan spesifikasi kebutuhan pembeli dan kemudian dilanjutkan dengan *planning* (perencanaan), *modeling* (pemodelan)*, construction* (konstruksi) *dan deployment* (penyebaran) memuncak dalam dukungan berkelanjutan dari perangkat lunak yang telah selesai. Berikut ini adalah proses *waterfall model* dapat dilihat pada Gambar 2.1.



Gambar 2.1 *Waterfall Model*

Sumber: (Pressman, 2010)

1. *Communication*

Langkah ini merupakan analisis terhadap kebutuhan *software*, dan tahap untuk mengadakan pengumpulan data dengan melakukan pertemuan dengan *customer*, maupun mengumpulkan data-data tambahan baik yang di jurnal, artikel, maupun dari internet.

2. *Planning*

Proses *planning* merupakan lanjutan dari proses *communication (analysis requirement).* Tahapan ini akan menghasilkan dokumen *user requirement* atau bisa dikatakan sebagai data yang berhubungan dengan keinginan *user* dalam pembuatan *software,* termasuk rencana yang akan dilakukan.

3. *Modeling*

Proses *modeling* ini akan menerjemahkan syarat kebutuhan ke sebuah perancangan *software* yang dapat diperkirakan sebelum dibuat *coding.* Proses ini berfokus pada rancangan struktur data, arsitektur *software,* representasi *interface*, dan detail (algoritme) *procedural*. Tahapan ini akan menghasilkan dokumen yang disebut *software requirement.*

4. *Construction*

Construction merupakan proses membuat kode. *Coding* atau pengkodean merupakan penerjemah desain dalam bahasa yang bisa dikenali oleh komputer. *Programmer* akan menerjemahkan transaksi yang diminta oleh *user.* Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu *software,* artinya penggunaan komputer 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 sistem tersebut untuk kemudian bisa diperbaiki.

5. *Deployment*

Tahapan ini bisa dikatakan final dalam pembuatan sebuah *software* atau sistem. Setelah melakukan analisis, desain dan pengkodean maka sistem yang sudah jadi akan digunakan oleh *user.* Kemudian *software* yang telah dibuat harus dilakukan pemerliharaan secara berkala.

2.5 Pengujian perangkat lunak

Pengujian perangkat lunak bertujuan untuk mengetahui sejauh mana perangkat lunak yang diuji telah memenuhi kebutuhan pengguna dan stakeholder yang menjadi pengguna sistem (Sommerville, 2011). Tujuan utama pengujian perangkat lunak adalah untuk memastikan tidak ada kesalahan dan sesuai dengan ekspektasi yang diharapkan yang tertuang dalam rencana dan perancangan yang telah dibuat dan disetujui sebelumnya (Jambak, 2016). Teknik atau metode yang digunakan dalam kasus penelitian ini adalah *white-box-testing* dan *black-box testing.*

2.5.1 White-box testing

White-box testing merupakan salah satu pendekatan dalam pengujian program, dimana pengujian dilakukan berdasarkan pengetahuan mengenai struktur program dan komponen-komponennya. Akses pada *source code* sangat penting pada *white-box testing* (Sommerville, 2011).

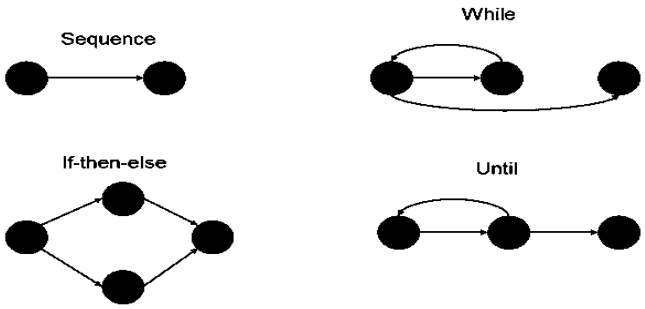
White-box testing atau bisa disebut *glass-box testing*, menggunakan struktur kontrol, yang merupakan bagian dari *component level design*, untuk menghasilkan *test cases*. Dengan kata lain, pengujian *whitebox* hanya dapat didesain apabila *component-level design* (atau *source code*) sudah dibuat. *Test cases* yang dihasilkan dengan menggunakan metode *white-box testing*:

1. Menjamin bahwa seluruh *independent path* yang ada di dalam suatu *modul* telah dieksekusi paling tidak satu kali.
2. Mencoba seluruh *logical decision* dan memberikan kesesuaian hasil nilai *true* atau *false* berdasarkan kondisi yang diberikan.
3. Melakukan percobaan pada semua *loops* sesuai dengan batasan yang diberikan dan dalam batas operasionalnya.
4. Melakukan percobaan pada struktur data internal untuk memastikan bahwa struktur tersebut valid (Pressman, 2010).`

Menurut (Guru99, 2018), *basis path testing* adalah salah satu teknik *white-box* testing yang dapat menghasilkan kompleksitas logis (*logical complexity*) dari rancangan secara prosedural dan menggunakan pengukuran ini sebagai dasar kumpulan jalur yang akan dieksekusi. *Test case* yang dihasilkan dapat menjamin eksekusi setiap program paling tidak dilakukan sekali selama pengujian.

Cyclomatic Complexity adalah *software metric* yang digunakan untuk mengukur kompleksitas suatu program dengan mengukur jalur independen melalui kode program. Jalur independen adalah jalur yang paling tidak memiliki 1 *edge* yang belum pernah dilewati sebelum nya oleh jalur yang lain. Notasi

flowgraph untuk program telah ditetapkan. Beberapa *node* dihubungkan melalui *edges.* Dibawah ini adalah *flow diagram* untuk pernyataan seperti *if-else, While, until,* dan *normal sequence.* Berikut gambar notasi *flowgraph* pada Gambar 2.2.



Gambar 2.2 Notasi *Flowgraph*

Sumber: (Guru99, 2018)

Kompleksitas suatu program dapat didefinisikan dengan:

$$V(G) = E - N + 2$$

Dimana,

E = Jumlah *edge*

N = Jumlah *node*

$$V(G) = P + 1$$

Dimana,

P = Jumlah *predicate node (node* yang berisi kondisi)

Cyclomatic Complexity dapat dihitung secara manual pada program sederhana. *Automated tools* diperlukan jika program sangat kompleks yang melibatkan banyak *flow graph*. Pada Tabel 2.1 dibawah ini merupakan tabel yang memberikan gambaran kompleksitas dan arti dari nilai V(G).

Tabel 2.1 Makna *Complexity Number*

Complexity Number	Meaning
1-10	Struktur dan Penulisan Baik *High Testability* Biaya dan Usaha Rendah
10-20	Kode Kompleks *Medium Testability* Biaya dan Usaha Medium
20-40	Kode sangat kompleks *Low testability*

Complexity Number	Meaning
20-40	Biaya dan Usaha Tinggi
>40	Tidak semuanya bisa diuji Biaya dan Usaha sangat tinggi

2.5.2 *Black-box testing*

Black-box testing merupakan sebuah pendekatan yang ditujukan untuk menguji sistem berdasarkan spesifikasi sistem tanpa harus mengakses atau melakukan uji terhadap kode program (Sommerville, 2011)

Black-box testing, disebut juga sebagai *behavioural testing*, dikhususkan pada kebutuhan fungsional perangkat lunak. Teknik *black-box testing* digunakan untuk mengetahui kesesuaian fungsi tertentu dari suatu produk dengan tujuan awal didesainnya produk tersebut. Pengujian *black-box* dilaksanakan untuk menemukan *error* ketika skrip dieksekusi. *Black-box testing* melakukan pencarian *error* dalam kategori (Pressman, 2010, pp.495,539):

1. Fungsi yang salah atau fungsi yang tidak lengkap,
2. Antar muka yang *error*,
3. *Error* dalam melakukan akses *database* eksternal atau akses data struktur,
4. *Error* dalam *behavior* atau kinerja, dan
5. *Error* dalam inisialisasi dan *termination*.

Pressman (2010) menjelaskan bahwa *black-box testing* bukan merupakan pengganti dari teknik *white-box*. Akan tetapi, merupakan pendekatan yang melengkapi teknik *white-box* dengan menemukan *error* yang pada kelas yang berbeda dari metode *white-box*. *Black-box testing* cenderung dilaksanakan pada langkah terakhir karena pengujian ini tidak mempedulikan struktur kontrol.

Black-box testing dilakukan dengan membuat sekumpulan *test case* yang telah didesain untuk dapat menemukan *error* pada level validasi perangkat lunak (*software validation level*). Pada setiap *case* yang diberikan, diharapkan agar dapat menemukan *error* sebanyak mungkin dengan waktu dan usaha seminimal mungkin. Setiap *test case* dapat dibuat untuk dapat melakukan uji coba terhadap setiap fungsi yang dibuat. Kemudian akan di validasi apakah telah beroperasi sepenuhnya dan disaat yang bersamaan melakukan pencarian *error* yang mungkin terjadi. Teknik pengujian dilakukan pada bagian antarmuka dengan memberikan sekumpulan kondisi *input* yang akan dijalankan secara keseluruhan dalam kebutuhan fungsi program.

2.7 Pola-pola Perancangan

Pola-pola perancangan berasal dari ide yang dikemukakan oleh Christopher Alexander bahwa dalam melakukan pembuatan rancangan terdapat pola-pola yang dapat digunakan secara efektif dan dengan pewarisan yang ada. Pola

merupakan gambaran masalah dan solusi untuk menyelesaikan permasalahan tersebut agar solusi yang ada dapat digunakan kembali dalam pengaturan yang berbeda (Sommerville, 2011).

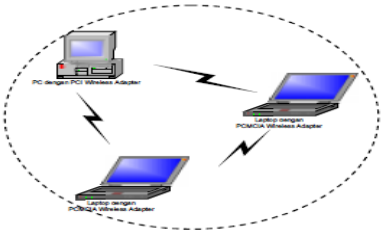
Pola-pola perancangan adalah pola-pola antar kelas yang dapat digunakan lagi sebagai solusi dari masalah perancangan dalam organisasi kelas-kelas. Pola-pola perancangan adalah "taktik" yang menghasilkan struktur dan perilaku kelas-kelas dan hubungan-hubungan antar kelas. Pola-pola perancangan sangat penting dalam pemrograman berorientasi objek karena menawarkan motif desain, solusi yang elegan untuk masalah desain yang dapat digunakan kembali, meningkatkan kualitas dari perangkat lunak (Gue´he´neuc & Antoniol, 2008).

2.8 Model-View-Controller

MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, user interface, dan bagian yang menjadi kontrol aplikasi. Terdapat 3 jenis komponen yang membangun suatu MVC pattern dalam suatu aplikasi yaitu *model, view dan controller. Model*, biasanya berhubungan langsung dengan database untuk memanipulasi data (*insert, update, delete* dan *search*), menangani validasi dari bagian controller, namun tidak dapat berhubungan langsung dengan bagian view. *View* merupakan bagian yang menangani presentation logic. bagian ini biasanya diatur oleh controller. View berfungsi untuk menerima dan merepresentasikan data kepada user. Bagian ini tidak memiliki akses langsung terhadap bagian model. Sedangkan, *Controller* merupakan bagian yang mengatur hubungan antara bagian model dan bagian view, controller berfungsi untuk menerima request dan data dari user kemudian menentukan apa yang akan diproses oleh aplikasi (Riyandwyana & Mukhlason, 2012).

2.9 Topologi *wireless LAN* mode *Ad-Hoc*

Mode *Ad-Hoc* adalah suatu kondisi jaringan Mode Ad-Hoc adalah suatu kondisi jaringan wireless yang tidak menggunakan access point. Artinya, antar client langsung terkoneksi satu dengan yang lainnya. Jika merasa asing dengan istilah *Ad-Hoc*, mungkin istilah Peer-to-peer dapat lebih mempermudah mengenali koneksi *Ad-Hoc*. Prinsip kerjanya sama saja dengan *Peer-to-peer*. Disini setiap client akan saling terkoneksi secara langsung (Arianto, 2009).

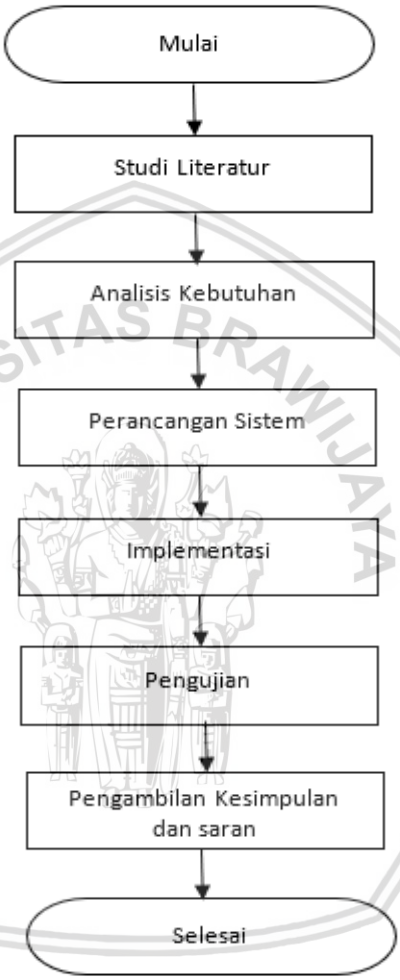


Gambar 2.3 Topologi Mode *Ad-Hoc*

Sumber: (Arianto, 2009)

BAB 3 METODOLOGI

Pada bab ini akan menjelaskan tentang metodologi yang akan dikerjakan untuk menyelesaikan skripsi ini. Untuk menyelesaikan penelitian pada skripsi ini akan dilakukan beberapa tahapan yaitu studi literatur, analisis kebutuhan, perancangan sistem, implementasi sistem, pengujian, pengambilan kesimpulan dan saran. Sebagaimana yang digambarkan pada gambar dibawah ini:



Gambar 3.1 Diagram Alur Penelitian

3.1 Studi literatur

Studi literatur adalah tahapan-tahapan yang dilakukan untuk penyusunan dasar teori sebagai bahan pendukung dan penunjang yang digunakan dalam penelitian skripsi ini. Literatur yang digunakan adalah bersumber dari berbagai jurnal ilmiah, internet, buku, dan laporan ilmiah.

Teori dan pustaka yang berkaitan mengenai skripsi ini adalah:

1. Toko Beben
2. Rekayasa perangkat lunak

3. *Software development life cycle*
4. *Waterfall model*
5. Pengujian perangkat lunak
6. *White-box testing*
7. *Black-box testing*
8. Pola-pola perancangan
9. *Model-view-controller*

3.2 Analisis kebutuhan

Analisis kebutuhan dilakukan untuk menganalisis permasalahan yang terjadi dan mendefinisikan apa saja yang yang harus ada pada sistem yang akan dibangun. Tahap ini sangat penting, karena jika terdapat kesalahan pada pendefinisian maka akan mempengaruhi semua tahapan setelah analisis kebutuhan karena akan terjadi perbedaan antara apa yang dibutuhkan oleh klien dengan software yang dihasilkan.

Pada pembuatan Sistem Pengelolaan Transaksi Toko Perak Beben pada daerah Banjarmasin, harus dilakukan studi literatur dan studi lapangan berupa observasi dan wawancara yang bertujuan untuk menganalisis kebutuhan sistem dari pengguna, seperti mengetahui proses bisnis, pihak-pihak yang terlibat dan fitur dan yang dibutuhkan dalam pembangunan sistem. Hal ini bertujuan untuk menghindari kesalahpahaman dalam mendefiniskan sistem. Studi literatur yang digunakan sebagai bahan acuan adalah dengan mencari referensi baik dari buku, jurnal ilmiah, internet dan laporan ilmiah. Hal tersebut dilakukan sebagai pendukung pada sistem yang akan dibangun.

Hasil dari studi lapangan dan studi literatur digunakan untuk mengidentifikasi semua kebutuhan yang diperlukan oleh Pemilik dan pegawai, kemudian hasilnya diajukan kepada klien untuk di validasi dan di verifikasi sesuai kebutuhan Pemilik dan Pegawai Toko Perhiasan Beben. Hasil dari tahap ini adalah identifikasi aktor-aktor yang telibat didalam sistem, *usecase diagram* dan *usecase scenario*.

3.3 Perancangan sistem

Perancangan sistem adalah tahapan yang dilakukan untuk menerjemahkan kebutuhan menjadi suatu model perangkat lunak. Tahapan ini dilakukan setelah melakukan analisis kebutuhan, yang mengacu pada hasil analisis kebutuhan. Hasil perancangan sistem adalah pemodelan arsitektur sistem, *sequence diagram, class diagram,* perancangan algoritme, kemudian dilanjutkan dengan pembuatan ERD (*Entity Relationship Diagram)* dan perancangan antarmuka*.*

Sequence diagram dibuat berdasarkan *use case diagram* dan *use case scenario*. Setelah selesai membuat *sequence diagram* maka akan dibuat *class diagram* yang dibuat berdasarkan *sequence diagram*. kemudian dibuat *entitiy relationship diagram.* Terakhir, dilakukan pemodelan antarmuka sistem sesuai dengan kebutuhan pengguna sistem.

3.4 Implementasi

Implementasi yang akan dibuat yaitu sistem pengelolaan transaksi Toko Perhiasan Beben berbasis *desktop*. Proses implementasi meliputi pemilihan bahasa pemrograman yang akan digunakan dan melakukan proses *coding. Coding* adalah proses untuk menerjemahkan rancangan sistem menjadi bentuk kode program. Pola arsitektur yang digunakan adalah *Model-View-Controller* (MVC) menggunakan database MySQL. Sistem ini dibangun menggunakan bahasa java dengan menggunakan aplikasi NetBeans.

Dalam mengimplementasikan sistem dengan menggunakan pola arsitektur *Model-View-Controller*(MVC) ini terdiri dari 3 modul. Bagian pertama adalah *View* yang merupakan antarmuka dari sistem yang dapat dilihat oleh pengguna sistem. Bagian kedua adalah *controller* yang berfungsi untuk menghubungkan *model* dengan *view*. Bagian ketiga adalah *model* yang berfungsi untuk mengelola *database* yang berisi masukkan, membaca, mengedit, dan menghapus data pada *database*.

3.5 Pengujian

Pengujian perangkat lunak dilakukan untuk menganalisis perbedaan antara spesifikasi kebutuhan yang telah didefinisikan dengan perangkat lunak yang telah dibuat. Apabila ditemukan kekurangan atau kesalahan maka akan dilakukan perbaikan pada sistem. Pengujian yang dilakukan yaitu pengujian unit dan pengujian validasi. Pengujian unit adalah pengujian yang dilakukan dengan menguji bagian terkecil dari perangkat lunak. Pengujian unit dilakukan dengan metode *white-box* testing yang diuji dengan menggunakan jenis *basis path testing*. Pengujian validasi dilakukan untuk menguji kesesuaian antara kebutuhan yang telah dispesifikasikan dengan perangkat lunak yang dibuat telah sesuai atau belum.

3.6 Kesimpulan dan saran

Kesimpulan dan saran dilakukan setelah tahapan pengujian selesai dilaksanakan berdasarkan kesesuaian antara teori dan praktik. Kesimpulan yang diambil didasarkan pada pengujian dan analisis pada sistem yang telah dibuat. Kesimpulan dibuat dengan tujuan untuk menjawab rumusan masalah. Tahap saran dilakukan sebagai masukan untuk memperbaiki kesalahan-kesalahan yang terjadi pada pengembangan perangkat lunak selanjutnya dan untuk menyempurnakan penulisan.

BAB 4 ANALISIS KEBUTUHAN

Analisis kebutuhan adalah tahapan pertama yang dilakukan dalam mengembangkan perangkat lunak. Pada bagian ini akan menjelaskan tahapan dalam menentukan kebutuhan pada sistem pengelolaan transaksi perangkat lunak dan menentukan aktor yang akan berinteraksi dengan sistem jika sistem telah selesai dikembangkan. Analisis kebutuhan didapat dari memahami domain masalah dan hasil dari elisitasi.

4.1 Gambaran umum sistem

Sistem Pengelolaan Transaksi Toko Perhiasan Perak Berbasis *Desktop* dibuat dengan tujuan untuk membantu proses operasional yang ada pada Toko Perhiasan Beben. Sistem ini terdiri dari 5 bagian utama, yaitu:

1. Transaksi Penjualan

Dalam transaksi penjualan, pegawai dan pemilik dapat melakukan proses menambah transaksi penjualan, mengedit barang pada *form* tambah transaksi penjualan, menghapus barang pada *form* tambah transaksi penjualan, menampilkan transaksi penjualan, dan mencetak transaksi penjualan, sedangkan penjual dapat melakukan proses menambah, menampilkan. Selain itu terdapat proses yang hanya bisa dilakukan pemilik Toko Beben yaitu menghapus transaksi penjualan.

2. Transaksi Pembelian

Dalam transaksi pembelian, pegawai dan pemilik dapat melakukan proses menambah transaksi pembelian, menampilkan transaksi pembelian, dan mecetak transaksi pembelian. Terdapat proses yang hanya dapat dilakukan pemilik yaitu menghapus transaksi pembelian.

3. Laporan Keuangan

Pemilik dapat melihat laporan keuangan Toko Perhiasan Beben berdasarkan rentang waktu tertentu.

4. Informasi Stok Barang

Informasi stok barang dapat diakses oleh pegawai dan pemilik. Pegawai dan pemilik dapat melakukan proses menampilkan stok barang. Terdapat proses yang hanya dapat dilakukan pemilik yaitu pemilik dapat mengedit stok barang.

5. Pembayaran Sales

Pembayaran sales hanya dapat diakses oleh pemilik. Dalam pembayaran sales pemilik dapat menambahkan nama sales, mengedit nama sales, melihat data pembayaran sales yang belum lunas, menambahkan data pembayaran sales, mengedit data pembayaran sales, menghapus data pembayaran sales, menampilkan notifikasi pembayaran sales yang belum lunas.

4.2 Analisis kebutuhan

Analisis kebutuhan merupakan aktivitas awal dalam proses pengembangan perangkat lunak. Proses analisis kebutuhan yaitu dimulai degan melakukan elisitasi kebutuhan, identifikasi aktor, menentukan spesifikasi kebutuhan yang akan digambarkan dengan *use case diagram* kemudian akan dilakukan tahapan validasi dan verifikasi.

4.2.1 Elisitasi kebutuhan

Elisitasi kebutuhan adalah proses pertama dalam analisis kebutuhan. Elisitasi kebutuhan dilakukan untuk memahami masalah yang terjadi pada Toko Perhiasan Beben. Kemudian, dari permasalahan yang didapatkan akan diajukan solusi-solusi yang dapat membantu menyelesaikan permasalahan yang terjadi pada Toko Perhiasan Beben.

Teknik elisitasi yang digunakan dalam penelitian ini adalah wawancara dan observasi. Proses Wawancara dilakukan untuk mengetahui permasalahan yang terjadi, orang-orang yang terlibat, dan proses bisnis yang terjadi pada Toko Perhiasan Beben yang hasilnya akan didapatkan fitur-fitur dan aktor-aktor yang terlibat dalam sistem. Sedangkan, proses observasi dilakukan dengan mengamati proses yang terjadi di Toko Perhiasan Beben yang nantinya digunakan untuk mendapatkan fitur-fitur yang digunakan dalam sistem.

Berdasarkan hasil observasi dan wawancara didapatkan kebutuhan dalam sistem pengelolaan transaksi pada Toko Perhiasan Beben, yaitu:

1. Kegiatan Transaksi

   1.1 Bagian ini harus memiliki fungsi untuk mengelola transaksi pembelian

   1.2 Bagian ini harus memiliki fungsi untuk mengelola transaksi penjualan

   1.3 Bagian ini harus memiliki fungsi untuk mecetak transaksi

2. Kegiatan Pengelolaan Stok Barang

   2.1 Bagian ini harus memiliki fungsi untuk menampilkan stok barang

   2.2 Bagian ini harus memiliki fungsi untuk mengedit stok barang

3. Laporan Keuangan

   Bagian ini harus memiliki fungsi untuk menampilkan laporan keuangan Toko Perhiasan Beben.

4. Pembayaran Sales

   4.1 Bagian ini harus memiliki fungsi untuk menampilkan riwayat pembayaran sales

   4.2 Bagian ini harus memiliki fungsi untuk menambah pembayaran sales

   4.3 Bagian ini harus memiliki fungsi untuk menghapus pembayaran sales

   4.4 Bagian ini harus memiliki fungsi untuk mengedit pembayaran sales

4.5 Bagian ini harus memiliki fungsi untuk menambahkan nama sales

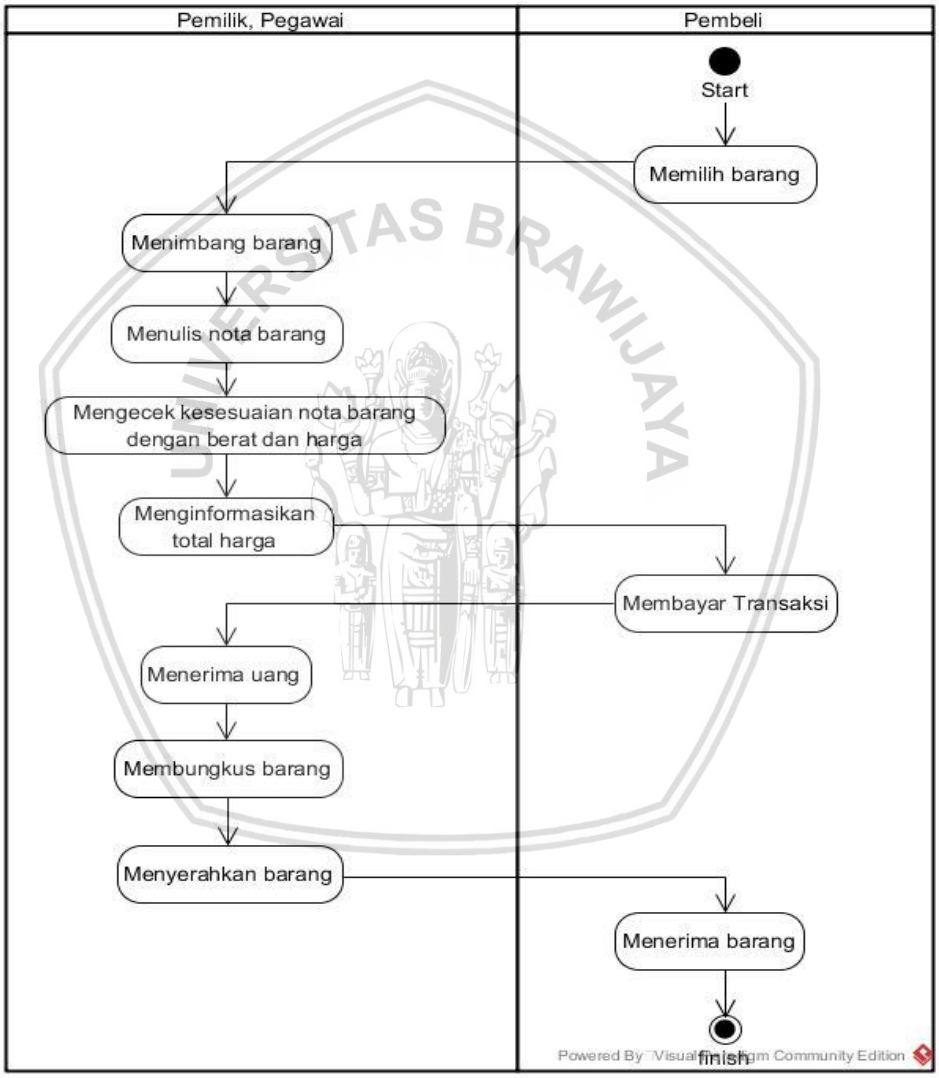
4.6 Bagian ini harus memiliki fungsi untuk mengedit nama sales

4.7 Bagian ini harus memiliki fungsi untuk menampilkan pembayaran sales yang belum lunas

5. Umum

5.1 Bagian ini harus menyediakan fungsi *login* untuk dapat mengidentifikasi pengguna

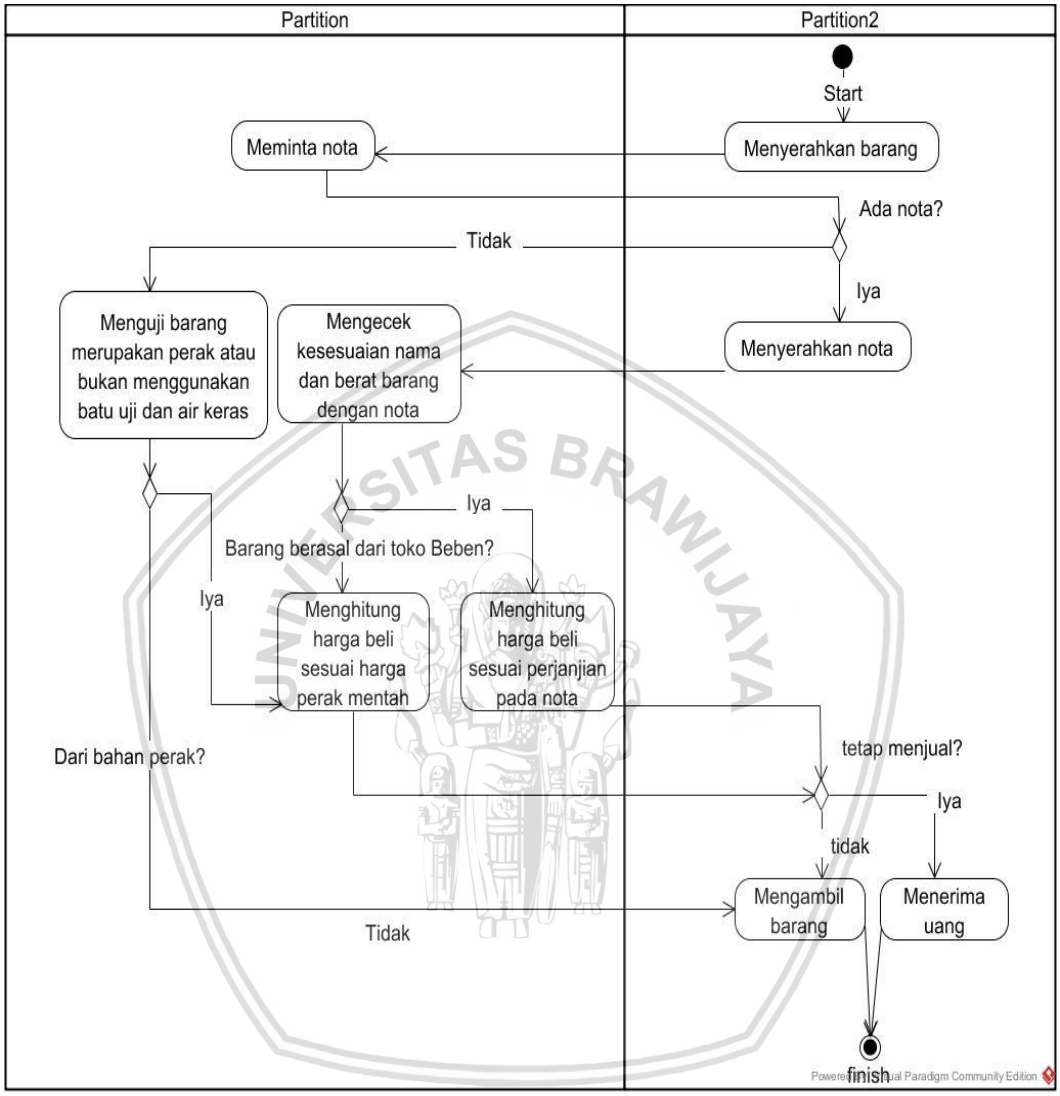
5.2 Bagian ini harus memiliki fungsi untuk *logout*



Gambar 4.1 Proses Bisnis Transaksi Penjualan

Proses bisnis transaksi penjualan bisa dilihat pada Gambar 4.1. Transaksi penjualan ini adalah transaksi yang dilakukan oleh pembeli (konsumen atau *reseller)* dengan pemilik atau pegawai. Proses ini diawali pembeli memilih barang yang ingin dibeli, kemudian pemilik atau pegawai menimbang barang, menulis nota barang, mengecek kesesuaian nota dengan berat dan harga, dan menginformasikan kepada pembeli biaya total. Setelah itu pembeli membayar

transaksi, kemudian pemilik atau pegawai menerima uang, membungkus perhiasan yang dibeli, dan menyerahkan barang kepada pembeli, dan terakhir pembeli menerima perhiasan yang dibelinya.



Gambar 4.2 Proses Bisnis Transaksi Pembelian

Proses bisnis pembelian barang dilakukan seperti pada Gambar 4.2. prosesnya dimulai dari konsumen menyerahkan barang yang ingin dijualnya, kemudian pemilik atau pegawai meminta nota kepada konsumen

1. jika konsumen tidak memiliki nota maka akan dilakukan proses pengujian perak.

   1.1 Jika terbukti perak maka akan dikenakan biaya seharga perak mentah. Apabila konsumen setuju maka penjual memberi uang pembelian ke konsumen.

   1.2 Jika tidak terbukti perak maka barang dikembalikan kepada konsumen.

2 Jika konsumen memiliki nota, maka konsumen menyerahkan nota ke pemilik atau pegawai.

2.1 Jika barang yang diserahkan dan nota yang diberikan dibandingkan dan barang tersebut memang benar berasal dari Toko Beben maka penjual membeli barang sesuai dengan perjanjian pada nota.

2.2 Jika barang yang diserahkan dan nota yang diberikan tidak sesuai maka pemilik atau pegawai membeli barang tersebut seharga perak mentah.

4.2.2 Identifikasi aktor

Berdasarkan hasil dari elisitasi kebutuhan, aktor yang terdapat pada sistem yaitu pengguna, pemilik dan pegawai yang didapat dari hasil observasi dan wawancara dan observasi dengan pegawai dan pemilik toko. Pada tabel 4.1 terdapat 2 kolom yaitu kolom aktor yang berisi informasi aktor yang terlibat didalam sistem dan deskripsi dari peran aktor yang terdapat didalam sistem. Berikut adalah hasil identifikasi aktor pada Tabel 4.1:

Tabel 4.1 Tabel Identifikasi Aktor

Aktor	Deskripsi
Pengguna	Pengguna merupakan orang yang belum terautentikasi oleh sistem.
Pemilik	Pemilik merupakan pemilik toko perhiasan yang memiliki hak akses pada semua fungsi yang ada di dalam sistem.
Pegawai	Pegawai merupakan orang yang bekerja kepada pemilik yang didalam sistem berperan untuk menambah transaksi pembelian, mengedit barang pada *form* tambah transaksi pembelian, menghapus barang pada *form* tambah transaksi pembelian, menambah transaksi penjualan, mengedit barang pada *form* tambah transaksi penjualan, menghapus barang pada *form* tambah transaksi penjualan, mencetak transaksi pembelian, mencetak transaksi penjualan, menampilkan transaksi penjualan, menampilkan transaksi pembelian dan melihat stok barang

4.3 Daftar kebutuhan fungsional

Kebutuhan fungsional merupakan suatu layanan yang harus ada didalam sistem. Kebutuhan fungsional didapat dari hasil wawancara dengan pemilik sistem dan pegawai. Validasi menunjukkan kebutuhan fungsional dalam sistem telah benar dan lengkap sesuai dengan kebutuhan aktor yang terlibat didalam sistem. Verifikasi menunjukkan bahwa kebutuhan yang telah didefinisikan mudah diidentifikasi, lengkap, dan tidak ambigu. Setiap kebutuhan fungsional diberikan kode SPTTB_F_XX. SPTTB adalah singkatan dari Sistem Pengelolaan Transaksi Toko Beben, F adalah singkatan dari Fungsional dan XX adalah nomor kebutuhan fungsional yang didefinisikan. Berikut terdapat 28 kebutuhan fungsional di gambarkan dalam tabel spesifikasi kebutuhan fungsional dalam sistem pada tabel 4.2.

Tabel 4.2 Tabel Spesifikasi Kebutuhan Fungsional Sistem

No	*Use Case*	Aktor	Spesifikasi	Kode
1.	*Login*	Pengguna	Sistem harus dapat melakukan autentikasi terhadap pengguna sebagai pegawai atau pemilik	SPTTB_F_01
2.	*Logout*	Pemilik, Pegawai	Sistem harus dapat mengeluarkan pegawai atau pemilik dari sistem sebagai pengguna	SPTTB_F_02
3.	Menambah Akun	Pemilik	Sistem harus dapat menambahkan akun	SPTTB_F_03
			Sistem harus menyediakan pilihan akun yang akan ditambahkan berstatus sebagai pemilik atau pegawai	
4.	Menghapus Akun	Pemilik	Sistem harus dapat menghapus akun	SPTTB_F_04
5.	Mengedit Akun	Pemilik	Sistem harus dapat mengedit akun	SPTTB_F_05
6.	Melihat Akun	Pemilik	Sistem harus dapat melihat data akun yang telah terdaftar	SPTTB_F_06
			Data akun yang ditampilkan adalah no, *username,* nama lengkap, status user dan status aktif	

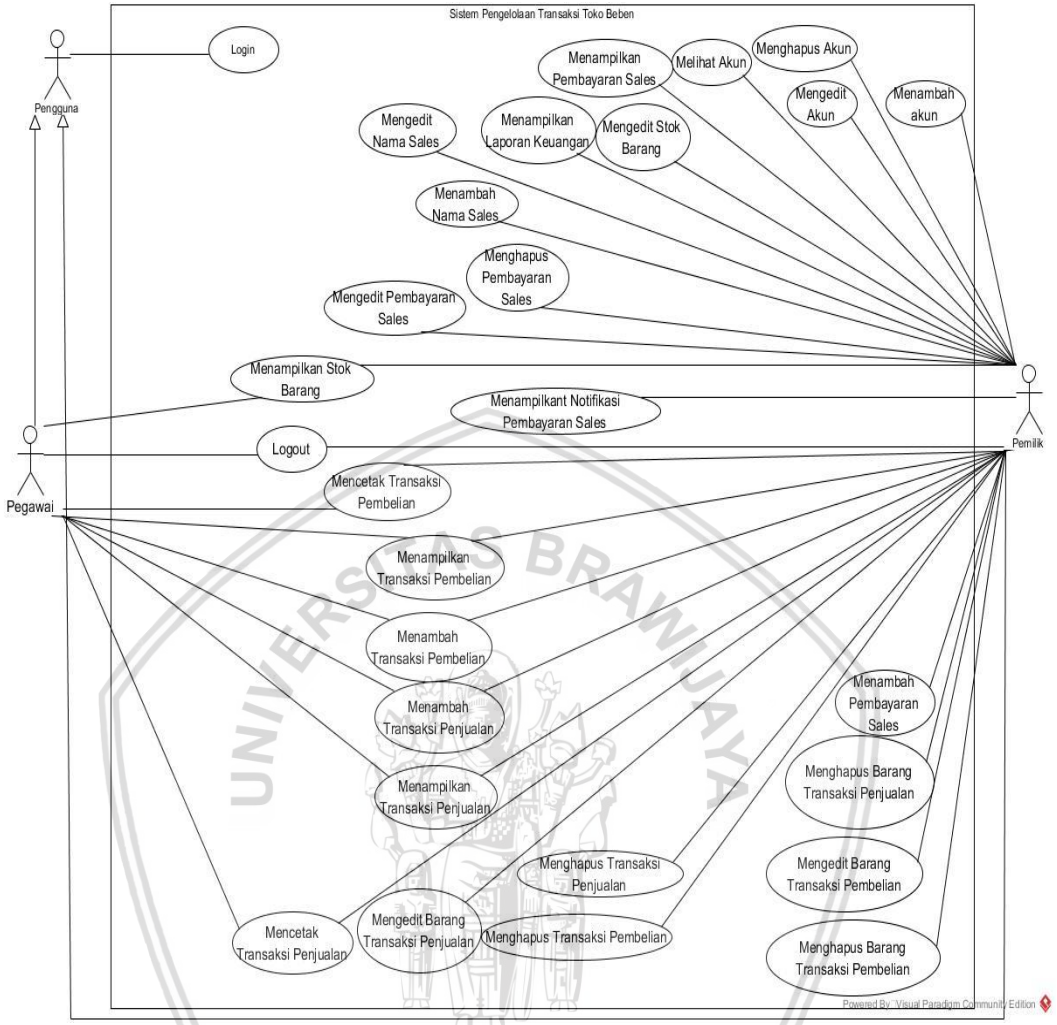
7.	Menambah Transaksi Pembelian	Pemilik, Pegawai	Sistem harus dapat menambahkan transaksi pembelian	SPTTB_F_07
			Sistem harus dapat menambahkan jenis barang baru dan jenis barang lama pada transaksi pembelian	
8.	Mengedit Barang Transaksi Pembelian	Pemilik, Pegawai	Sistem harus dapat mengedit barang pada *form* tambah transaksi pembelian	SPTTB_F_08
9.	Menghapus Barang Transaksi Pembelian	Pemilik, Pegawai	Sistem harus dapat menghapus barang pada *form* tambah transaksi pembelian	SPTTB_F_09
10.	Menghapus Transaksi Pembelian	Pemilik	Sistem harus dapat menghapus transaksi pembelian	SPTTB_F_10
11.	Menampilkan Transaksi Pembelian	Pemilik, Pegawai	Sistem harus dapat menampilkan transaksi pembelian	SPTTB_F_11
			Halaman transaksi pembelian berisi data no transaksi, tanggal, status transaksi, nama penjual, status penjual, total harga, dan total harga akhir	
12.	Mencetak Transaksi Pembelian	Pemilik, Pegawai	Sistem harus dapat mencetak transaksi pembelian	SPTTB_F_12
13.	Menambah Transaksi Penjualan	Pemilik, Pegawai	Sistem harus dapat menambah transaksi penjualan	SPTTB_F_13
14.	Mengedit Barang Transaksi Penjualan	Pemilik, Pegawai	Sistem harus dapat mengedit barang pada *form* tambah transaksi penjualan	SPTTB_F_14
15.	Menghapus Barang	Pemilik, Pegawai	Sistem harus dapat menghapus barang pada	SPTTB_F_15

	Transaksi Penjualan		*form* tambah transaksi penjualan	
16.	Menghapus Transaksi Penjualan	Pemilik	Sistem harus dapat menghapus transaksi penjualan	SPTTB_F_16
17.	Menampilkan Transaksi Penjualan	Pemilik, Pegawai	Sistem harus dapat menampilkan transaksi penjualan	SPTTB_F_17
			Halaman transaksi penjualan terdiri dari no transaksi, tanggal, status transaksi, nama pembeli, status pembeli, total harga, potongan harga, total harga akhir, no, id barang, nama barang, banyak, berat, harga, dan total harga	
18.	Mencetak Transaksi Penjualan	Pemilik, Pegawai	Sistem harus dapat mencetak transaksi penjualan	SPTTB_F_18
19.	Menampilkan Stok Barang	Pemilik, Pegawai	Sistem harus dapat menampilkan informasi stok barang	SPTTB_F_19
			Informasi stok barang terdiri dari no, id barang, nama barang, banyak, berat, harga modal, harga jual eceran, harga jual grosir	
20.	Mengedit Stok Barang	Pemilik	Sistem harus dapat mengedit stok barang	SPTTB_F_20
21.	Menampilkan Laporan Keuangan	Pemilik	Sistem harus dapat menampilkan laporan keuangan berdasarkan rentang waktu tertentu	SPTTB_F_21
			Laporan keuangan terdiri dari rentang tanggal transaksi yang ingin ditampilkan, no, no transaksi, laba bersih, rugi, kredit, debit , nama user,	

			total debit, total kredit, total untung bersih dan total rugi	
22.	Menampilkan Pembayaran Sales	Pemilik	Sistem harus dapat menampilkan halaman pembayaran sales	SPTTB_F_22
			Halaman pembayaran sales terdiri dari no, no pembayaran, no transaksi, tanggal transaksi, tanggal pembayaran, bayar, sisa belum lunas, dan status pembayaran	
23.	Menambah Pembayaran Sales	Pemilik	Sistem harus dapat menambahkan data pembayaran sales	SPTTB_F_23
24.	Menghapus Pembayaran Sales	Pemilik	Sistem harus dapat menghapus data pembayaran sales	SPTTB_F_24
25.	Mengedit Pembayaran Sales	Pemilik	Sistem harus dapat mengedit data pembayaran sales	SPTTB_F_25
26.	Menampilkan Notifikasi Pembayaran Sales	Pemilik	Sistem harus dapat menampilkan notifikasi pembayaran sales yang belum melunasi pembayaran selama sebulan	SPTTB_F_26
27.	Mengedit Nama Sales	Pemilik	Sistem harus dapat mengedit nama sales	SPTTB_F_27
28.	Menambah Nama Sales	Pemilik	Sistem harus dapat menambah nama sales	SPTTB_F_28

4.3.1 *Use case diagram*

Use case diagram adalah diagram yang dibuat untuk memodelkan perilaku sistem yang akan dibangun dan aktor yang terlibat pada sistem. *Use case diagram* dibuat berdasarkan kebutuhan fungsional yang telah dibuat sebelumnya. *Use Case* yang terdapat dalam sistem ini sebanyak 28 *use case.* Berikut *Use case diagram* sistem transaksi Toko Perhiasan Beben dapat dilihat pada Gambar 4.3.



Gambar 4.3 *Use Case Diagram* Sistem Transaksi Toko Perhiasan Beben

4.3.2 Use case scenario

4.3.2.1 Login

Use case scenario login dapat dilihat pada Tabel 4.3. Aktor yang terlibat dalam *use case* ini adalah pengguna. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case login*. Terdapat *alternative flow* apabila *username* atau *password* yang dimasukkan salah.

Tabel 4.3 *Use Case Scenario Login*

Login	
Actor	Pengguna
Objective	Mengizinkan *user* mengakses sistem sebagai pemilik atau pegawai
Pre-Condition	Sistem menampilkan halaman *login*

Main Flow	1. Pengguna memasukkan *username* dan *password* dan memilih tombol *login* 2. Sistem mengecek status validitas input dari pengguna kemudian menampilkan halaman daftar transaksi penjualan
Alternative Flow: Jika *username* atau *password* salah	1. Sistem menampilkan pesan "U*sername* atau *password* anda salah".
Post-Condition	Sistem menampilkan halaman daftar transaksi penjualan

4.3.2.2 Logout

Use case scenario logout dapat dilihat pada Tabel 4.4. Aktor yang terlibat dalam *use case* ini adalah pemilik atau pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case logout*.

Tabel 4.4 *Use Case Scenario Logout*

Logout	
Actor	Pemilik, pegawai
Objective	Untuk keluar dari sistem sebagai pengguna
Pre-Condition	Aktor telah masuk ke dalam sistem sebagai pemilik atau pengguna dan berada pada halaman daftar transaksi penjualan
Main Flow	1. Aktor memilih tombol logout 2. Sistem mengeluarkan aktor dari sistem sebagai pengguna dan menampilkan halaman *login*
Alternative Flow	-
Post-Condition	Sistem menampilkan halaman *login*

4.3.2.3 Menambah akun

Use case scenario menambah akun dapat dilihat pada Tabel 4.5. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* Menambah Akun. Terdapat *alternative flow* jika *username* telah digunakan sebelumnya.

Tabel 4.5 *Use Case Scenario* Menambah Akun

Menambah Akun	
Actor	Pemilik
Objective	Untuk menambahkan akun
Pre-Condition	Sistem menampilkan *form* tambah akun
Main Flow	1. Aktor mengisi *form* tambah akun yang berisi *username*, nama lengkap, password, dan status kemudian memilih tombol tambah 2. Sistem mengecek *username* telah digunakan sebelumnya atau belum, jika belum maka sistem menyimpan masukkan aktor dan menampilkan pesan "Pendaftaran berhasil"
Alternative Flow: Jika *username* telah digunakan sebelumnya	Sistem menampilkan pesan "*username* telah digunakan"
Post-Condition	Sistem menyimpan masukkan aktor, menampilkan pesan "pendaftaran berhasil" dan menampilkan halaman data akun yang telah di perbarui

4.3.2.4 Menghapus akun

Use case scenario menghapus akun dapat dilihat pada Tabel 4.6. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menghapus akun.

Tabel 4.6 *Use Case Scenario* Menghapus Akun

Menghapus Akun	
Actor	Pemilik
Objective	Untuk menghapus akun
Pre-Condition	Sistem menampilkan *form* data akun
Main Flow	1. Aktor memilih tombol hapus 2. Sistem menghapus akun dari *database* dan menampilkan halaman data akun yang telah diperbarui

Alternative Flow	-
Post-Condition	Sistem menampilkan halaman data akun yang telah diperbarui

4.3.2.5 Mengedit akun

Use case scenario mengedit akun dapat dilihat pada Tabel 4.7. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* mengedit akun. Terdapat *alternative flow* apabila *username* yang dimasukkan telah digunakan sebelumnya.

Tabel 4.7 *Use Case Scenario* Mengedit Akun

Mengedit Akun	
Actor	Pemilik
Objective	Untuk mengedit akun
Pre-Condition	Sistem menampilkan *form* edit akun
Main Flow	1. Aktor mengisi *form* edit berupa nama lengkap, password, dan status kemudian memilih tombol edit 2. sistem menyimpan masukkan aktor dan menampilkan pesan "Akun berhasil diedit", kemudian menampilkan halaman data akun yang telah diperbarui
Post-Condition	Sistem menyimpan masukkan aktor dan menampilkan pesan "Akun berhasil diedit", kemudian menampilkan halaman data akun yang telah diperbarui

4.3.2.6 Melihat akun

Use case scenario melihat akun dapat dilihat pada Tabel 4.8. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* melihat akun.

Tabel 4.8 *Use Case Scenario* Melihat Akun

Melihat akun	
Actor	Pemilik
Objective	Untuk melihat data akun yang telah terdaftar

Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman daftar transaksi penjualan
Main Flow	1. Aktor memilih menu akun 2. Sistem menampilkan halaman data akun yang terdapat data no, *username*, nama lengkap, status user dan status aktif
Alternative Flow	-
Post-Condition	Sistem menampilkan halaman data akun yang terdapat data no, *username*, nama lengkap, status user, dan status aktif.

4.3.2.7 Menambah transaksi pembelian

Use case scenario menambah transaksi pembelian dapat dilihat pada Tabel 4.9. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menambah transaksi pembelian. Terdapat *alternative flow* dalam *use case scenario* ini jika aktor melakukan pembayaran namun uang yang dibayarkan kurang dari total akhir transaksi pembelian.

Tabel 4.9 *Use Case Scenario* Menambah Transaksi Pembelian

Menambah Transaksi Pembelian	
Actor	Pemilik, Pegawai
Objective	Untuk menambahkan transaksi pembelian pada sistem
Pre-Condition	Aktor telah berhasil *login* dan sistem menampilkan halaman *form* tambah transaksi pembelian
Main Flow	1. Aktor mengisi nama pada bagian tuan/nyonya dan memilih status penjual kemudian memilih tombol simpan data 2. Sistem menyimpan masukkan dan menampilkan halaman transaksi pembelian kemudian menampilkan pesan "Anda berhasil menambahkan data penjual" 3. Aktor memilih tombol tambah

	4. Sistem menampilkan *form* tambah barang 5. Aktor memilih barang yang ingin ditambahkan pada tabel kemudian mengisi *form* tambah jenis barang lama setelah itu menekan tombol tambah 6. Sistem menyimpan masukkan aktor kemudian menampilkan halaman *form* transaksi pembelian 7. Aktor memilih tambah jenis barang baru pada *form* tambah barang 8. Sistem menampilkan *form* tambah jenis barang baru 9. Aktor mengisi *form* tambah jenis baru berupa data nama barang, banyak, berat, harga beli, harga tambahan, harga untung ecer, dan harga untung grosir kemudian memilih tombol simpan 10. Sistem menyimpan masukkan aktor dan menampilkan halaman transaksi pembelian yang telah diperbarui 11. Aktor memilih tombol bayar 12. Sistem menampilkan *form* pembayaran 13. Aktor mengisi bayar dan uang di terima pada *form* pembayaran dan menekan tombol bayar pada form pembayaran 14. Sistem menyimpan data pembayaran dan menampilkan uang kembalian pembayaran serta menampilkan notifikasi pembayaran selesai dilakukan
Alternative Flow: Jika aktor melakukan pembayaran namun uang yang dibayarkan kurang dari total akhir transaksi pembelian	1. Sistem menampilkan pesan “Anda harus membayar lunas”

Post-Condition	Sistem menyimpan pembayaran transaksi, kemudian sistem menampilkan pesan "Pembayaran telah selesai dilakukan" dan menampilkan *form* transaksi pembayaran yang berstatus lunas

4.3.2.8 Mengedit barang transaksi pembelian

Use case scenario mengedit barang transaksi pembelian dapat dilihat pada Tabel 4.10. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* mengedit transaksi pembelian.

Tabel 4.10 *Use Case Scenario* Mengedit Barang Transaksi Pembelian

Mengedit Transaksi Pembelian	
Actor	Pemilik, Pegawai
Objective	Untuk mengedit barang pada *form* tambah transaksi pembelian
Pre-Condition	Aktor telah berhasil *login* dan sistem menampilkan halaman *form* tambah transaksi pembelian yang tabelnya telah terisi barang pembelian
Main Flow	1. Aktor memilih tombol edit 2. Sistem menampilkan *form* edit barang 3. Aktor memilih barang yang ingin diedit pada tabel kemudian mengisi *form* edit barang setelah itu memilih tombol edit 4. Sistem memperbarui data sesuai masukkan aktor kemudian menampilkan halaman *form* tambah transaksi pembelian
Post-Condition	Sistem memperbarui data sesuai masukkan aktor kemudian menampilkan halaman *form* tambah transaksi pembelian

9. Menghapus barang transaksi pembelian

Use case scenario menghapus barang transaksi pembelian dapat dilihat pada Tabel 4.11. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario*

ini menggambarkan alur yang terjadi pada *use case* menghapus barang transaksi pembelian.

Tabel 4.11 *Use Case Scenario* Menghapus Barang Transaksi Pembelian

Menghapus Transaksi Pembelian	
Actor	Pemilik, Pegawai
Objective	Untuk menghapus barang transaksi pada *form* tambah transaksi pembelian
Pre-Condition	Aktor telah *login* dan sistem menampilkan *form* transaksi pembelian
Main Flow	1. Aktor memilih barang pada tabel kemudian memilih tombol hapus transaksi 2. Sistem menghapus barang yang dipilih aktor dan menampilkan halaman tambah transaksi pembelian yang telah diperbarui
Alternative Flow	-
Post-Condition	Sistem berhasil menghapus barang yang dipilih aktor dan menampilkan halaman tambah transaksi pembelian yang telah diperbarui

10. Menghapus transaksi pembelian

Use case scenario menghapus transaksi pembelian dapat dilihat pada Tabel 4.12. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menghapus transaksi pembelian.

Tabel 4.12 *Use Case Scenario* Menghapus Transaksi Pembelian

Menghapus Transaksi Pembelian	
Actor	Pemilik
Objective	Untuk menghapus transaksi pembelian yang telah tersimpan
Pre-Condition	Aktor telah *login* dan sistem menampilkan *form* transaksi pembelian
Main Flow	1. Aktor memilih tombol hapus transaksi

	2. Sistem menampilkan pesan "Apakah anda yakin ingin menghapus transaksi ini?". 3. Aktor memilih tombol iya 4. Sistem menghapus transaksi yang dipilih aktor dan menampilkan halaman daftar transaksi yang telah diperbarui
Alternative Flow	-
Post-Condition	Sistem menghapus transaksi yang dipilih aktor dan menampilkan halaman daftar transaksi pembelian yang telah diperbarui

4.3.2.11 Menampilkan transaksi pembelian

Use case scenario menampilkan transaksi pembelian dapat dilihat pada Tabel 4.13. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menampilkan transaksi pembelian.

Tabel 4.13 *Use Case Scenario* Menampilkan Transaksi Pembelian

Menampilkan Transaksi Pembelian	
Actor	Pemilik, Pegawai
Objective	Untuk dapat menampilkan transaksi pembelian yang telah dilakukan
Pre-Condition	Aktor telah *login* dan sistem menampilkan daftar transaksi penjualan
Main Flow	1. Aktor memilih menu transaksi pembelian 2. Sistem menampilkan halaman daftar transaksi pembelian 3. Aktor memilih transaksi dalam tabel yang ingin ditampilkan 4. Sistem menampilkan halaman transaksi pembelian yang telah dipilih
Alternative Flow	-

Post-Condition	Sistem menampilkan halaman transaksi pembelian

4.3.2.12 Mencetak transaksi pembelian

Use case scenario mencetak transaksi pembelian dapat dilihat pada Tabel 4.14. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* mencetak transaksi pembelian.

Tabel 4.14 *Use Case Scenario* Mencetak Transaksi Pembelian

Mencetak Transaksi Pembelian	
Actor	Pemilik, Pegawai
Objective	Untuk mencetak transaksi pembelian.
Pre-Condition	Aktor telah login dan sistem menampilkan halaman transaksi pembelian
Main Flow	1. Aktor memilih tombol Cetak Transaksi 2. Sistem menyimpan detail transaksi pembelian pada *file* dengan format pdf kemudian menampilkan *file* detail transaksi pembelian dengan format pdf tersebut kepada aktor
Alternative Flow:	-
Post-Condition	Sistem menyimpan detail transaksi pembelian pada *file* dengan format pdf kemudian menampilkan *file* detail transaksi pembelian dengan format pdf tersebut kepada aktor

4.3.2.13 Menambah transaksi penjualan

Use case scenario menambah transaksi penjualan dapat dilihat pada Tabel 4.15. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menambah transaksi penjualan. Terdapat 3 *alternative flow* dalam *use case scenario* ini yaitu apabila status pembeli yang dimasukkan aktor adalah sales namun nama yang dimasukkan tidak terdaftar sebagai sales, pembayaran yang dilakukan lunas dan uang yang dibayarkan kurang dari yang seharusnya dibayarkan.

Tabel 4.15 *Use Case Scenario* Menambah Transaksi Penjualan

Menambah Transaksi Penjualan	
Actor	Pemilik, Pegawai
Objective	Aktor dapat menambah transaksi penjualan
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman transaksi penjualan
Main Flow	1. Aktor mengisi nama pembeli dan status pembeli, kemudian memilih tombol simpan data 2. Sistem menyimpan masukan aktor dan menampilkan pesan Anda berhasil menambahkan data pembeli 3. Aktor memilih tombol tambah 4. Sistem menampilkan *form* tambah barang penjualan 5. Aktor memilih barang yang ingin ditambahkan pada tabel kemudian mengisi *form* tambah barang penjualan. Setelah itu, aktor memilih tombol tambah 6. Sistem menyimpan masukkan dan menampilkan transaksi penjualan yang telah diperbarui 7. Aktor memilih tombol bayar 8. Sistem menampilkan *form* pembayaran 9. Aktor mengisi bayar dan uang diterima pada *form* pembayaran kemudian memilih tombol bayar 10. Sistem menyimpan data pembayaran kemudian menampilkan uang kembalian, dan pesan “Pembayaran telah selesai dilakukan”

Alternative Flow 1: Jika status pembeli yang dimasukkan aktor adalah sales namun nama yang dimasukkan tidak terdaftar sebagai sales	1. Sistem menampilkan pesan "Nama yang diinputkan belum terdaftar sebagai sales"
Alternative Flow 2: Jika pembayaran yang dilakukan lunas	1. Sistem merubah status transaksi menjadi lunas
Alternative Flow 3: Jika uang yang dibayarkan kurang dari yang seharusnya dibayarkan	1. Sistem menampilkan pesan "Pembeli harus membayar lunas"
Post-Condition	Sistem memperbarui status transaksi menjadi lunas, menampilkan uang kembalian, dan menampilkan pesan "Pembayaran telah selesai dilakukan"

4.3.2.14 Mengedit barang transaksi penjualan

Use case scenario mengedit barang transaksi penjualan dapat dilihat pada Tabel 4.16. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* mengedit barang transaksi penjualan.

Tabel 4.16 *Use Case Scenario* Mengedit Barang Transaksi Penjualan

Mengedit Barang Transaksi Penjualan	
Actor	Pemilik, Pegawai
Objective	Aktor dapat mengedit barang pada halaman tambah transaksi penjualan
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman tambah transaksi penjualan dimana tabelnya telah terisi barang yang akan dijual ke pembeli
Main Flow	1. Aktor memilih barang yang ingin diedit pada tabel kemudian memilih tombol edit 2. Sistem menampilkan *form* edit barang 3. Aktor memilih barang pada tabel kemudian mengisi *form* edit barang. Setelah itu aktor memilih tombol edit pada *form* edit 4. Sistem memperbarui data barang sesuai masukkan aktor kemudian

	menampilkan halaman tambah transaksi penjualan yang telah diperbarui
Alternative Flow:	-
Post-Condition	Sistem memperbarui data barang sesuai masukkan aktor kemudian menampilkan halaman tambah transaksi penjualan yang telah diperbarui

4.3.2.15 Menghapus barang transaksi penjualan

Use case scenario menghapus barang transaksi penjualan dapat dilihat pada Tabel 4.17. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menghapus transaksi penjualan.

Tabel 4.17 *Use Case Scenario* Menghapus Barang Transaksi Penjualan

Menghapus Barang Transaksi Penjualan	
Actor	Pemilik, pegawai
Objective	Aktor dapat menghapus barang pada tabel di halaman tambah transaksi penjualan
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman tambah transaksi penjualan dimana tabel barang yang diakan dibeli pembeli telah terisi
Main Flow	1. Aktor memilih barang pada tabel kemudian memilih tombol hapus 2. Sistem menghapus data barang dan menampilkan halaman transaksi penjualan
Alternative Flow:	-
Post-Condition	Sistem menghapus data barang dan menampilkan halaman transaksi penjualan

4.3.2.16 Menghapus transaksi penjualan

Use case scenario menghapus transaksi penjualan dapat dilihat pada Tabel 4.18. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menghapus transaksi penjualan.

Tabel 4.18 *Use Case Scenario* Menghapus Transaksi Penjualan

Menghapus Transaksi Penjualan	
Actor	Pemilik
Objective	Aktor dapat menghapus transaksi penjualan
Pre-Condition	Aktor telah *login* dan sistem menampilkan *form* transaksi penjualan yang telah selesai dilakukan
Main Flow	1. Aktor memilih tombol hapus transaksi pada *form* transaksi penjualan 2. Sistem menampilkan pesan konfirmasi "Apakah anda yakin ingin menghapus transaksi ini?" 3. Aktor memilih tombol iya 4. Sistem menghapus transaksi penjualan yang dipilih aktor dan menampilkan halaman daftar transaksi penjualan yang telah diperbarui
Alternative Flow:	-
Post-Condition	Sistem berhasil menghapus transaksi penjualan dan menampilkan halaman daftar transaksi penjualan yang telah diperbarui

4.3.2.17 Menampilkan transaksi penjualan

Use case scenario menampilkan transaksi penjualan dapat dilihat pada Tabel 4.19. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menampilkan transaksi penjualan.

Tabel 4.19 *Use Case Scenario* Menampilkan Transaksi Penjualan

Menampilkan Transaksi Penjualan	
Actor	Pemilik, Pegawai
Objective	Untuk menampilkan transaksi penjualan

Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman daftar transaksi penjualan
Main Flow	1. Aktor memilih menu transaksi penjualan 2. Sistem menampilkan halaman daftar transaksi penjualan 3. Aktor memilih transaksi penjualan pada tabel 4. Sistem menampilkan halaman transaksi penjualan yang dipilih
Alternative Flow	-
Post-Condition	Sistem menampilkan halaman transaksi penjualan

4.3.2.18 Mencetak transaksi penjualan

Use case scenario mencetak transaksi penjualan dapat dilihat pada Tabel 4.20. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* mencetak transaksi penjualan.

Tabel 4.20 *Use Case Scenario* Mencetak Transaksi Penjualan

Mencetak Transaksi Penjualan	
Actor	Pemilik, Pegawai
Objective	Untuk mencetak transaksi penjualan.
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman lihat *form* transaksi penjualan
Main Flow	1. Aktor memilih tombol cetak transaksi 2. Sistem menyimpan detail transaksi pembelian pada *file* dengan format pdf kemudian menampilkan *file* detail transaksi penjualan dengan format pdf tersebut kepada aktor
Alternative Flow:	-
Post-Condition	Sistem menyimpan detail transaksi pembelian pada *file* dengan format pdf kemudian menampilkan *file* detail

	transaksi penjualan dengan format pdf tersebut kepada aktor

4.3.2.19 Menampilkan stok barang

Use case scenario menampilkan stok barang dapat dilihat pada Tabel 4.21. Aktor yang terlibat dalam *use case* ini adalah pemilik dan pegawai. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menampilkan stok barang.

Tabel 4.21 *Use Case Scenario* Menampilkan Stok Barang

Menampilkan Stok Barang	
Actor	Pemilik, Pegawai
Objective	Untuk dapat menampilkan stok barang yang tersedia
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman daftar transaksi penjualan
Main Flow	1. Aktor memilih menu stok barang 2. Sistem menampilkan halaman stok barang
Alternative Flow	-
Post-Condition	Sistem menampilkan halaman stok barang

4.3.2.20 Mengedit stok barang

Use case scenario mengedit stok barang dapat dilihat pada Tabel 4.22. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* mengedit stok barang.

Tabel 4.22 *Use Case Scenario* Mengedit Stok Barang

Mengedit Stok Barang	
Actor	Pemilik
Objective	Untuk mengedit stok barang
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman stok barang
Main Flow	1. Aktor memilih barang yang ingin di edit pada tabel kemudian memilih tombol edit

	2. Sistem menampilkan *form edit* stok barang 3. Aktor mengedit *form* dan memilih tombol edit pada *form* edit stok barang 4. Sistem menampilkan pesan konfirmasi 5. Aktor memilih tombol ya 6. Sistem menyimpan masukkan aktor dan menampilkan halaman daftar stok barang yang telah diperbarui
Alternative Flow	-
Post-Condition	Sistem menyimpan masukkan aktor dan menampilkan halaman stok barang yang telah diperbarui

4.3.2.21 Menampilkan laporan keuangan

Use case scenario menampilkan laporan keuangan dapat dilihat pada Tabel 4.23. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menampilkan laporan keuangan.

Tabel 4.23 *Use Case Scenario* Menampilkan Laporan Keuangan

Menampilkan Laporan Keuangan	
Actor	Pemilik
Objective	Aktor dapat melihat halaman laporan keuangan berdasarkan rentang waktu yang diinginkan aktor
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman daftar transaksi penjualan
Main Flow	1. Aktor memilih menu laporan keuangan 2. Sistem menampilkan halaman laporan keuangan yang berisi keterangan dari tanggal, hingga tanggal, no, tanggal pembayaran, no transaksi, kredit, debit, nama

	user, total untung bersih, total rugi, total kredit dan total debit 3. Aktor mengisi dari tanggal dan hingga tanggal kemudian menekan tombol tampilkan 4. Sistem menampilkan halaman laporan keuangan berdasarkan rentang tanggal yang dipilih aktor
Alternative Flow	-
Post-Condition	Sistem menampilkan halaman laporan keuangan berdasarkan rentang tanggal yang dipilih aktor

4.3.2.22 Menampilkan pembayaran sales

Use case scenario menampilkan pembayaran sales dapat dilihat pada Tabel 4.24. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menampilkan pembayaran sales.

Tabel 4.24 *Use Case Scenario* Menampilkan Riwayat Pembayaran Sales

Menampilkan Pembayaran Sales	
Actor	Pemilik
Objective	Untuk menampilkan pembayaran sales yang belum lunas
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman daftar transaksi penjualan
Main Flow	1. Aktor memilih menu pembayaran sales 2. Sistem menampilkan halaman pembayaran sales 3. Aktor memilih nama sales yang ingin ditampilkan 4. Sistem menampilkan riwayat pembayaran sales yang dipilih dengan status belum lunas
Alternative Flow	-
Post-Condition	Sistem menampilkan halaman pembayaran sales yang dipilih

4.3.2.23 Menambah pembayaran sales

Use case scenario menambah pembayaran sales dapat dilihat pada Tabel 4.25. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menambah pembayaran sales. Terdapat *alternative flow* yang ada pada *use case* scenario, yaitu jika pembayaran telah lunas.

Tabel 4.25 *Use Case Scenario* Menambah Pembayaran Sales

Menambah Pembayaran Sales	
Actor	Pemilik
Objective	Untuk menambahkan data pembayaran transaksi sales yang berstatus belum lunas
Pre-Condition	Aktor telah *login* dan sistem menampilkan halaman pembayaran sales yang berisi data sales yang ingin ditambahkan
Main Flow	1. Aktor memilih transaksi pada tabel kemudian memilih tombol tambah 2. Sistem menampilkan *form* pembayaran yang berisi tanggal pembayaran, no transaksi, bayar, uang diterima, tombol bayar, dan kembalian 3. Aktor mengisi *form* pembayaran dan memilih tombol bayar 4. Sistem menyimpan masukkan aktor, menampilkan pesan "Pembayaran telah ditambahkan" dan menampilkan halaman pembayaran sales yang telah diperbarui
Alternative Flow: Jika pembayaran telah lunas	Sistem melakukan *update* status transaksi menjadi lunas
Post-Condition	Sistem menampilkan halaman transaksi pembayaran sales yang telah diperbarui dan menampilkan pesan "Pembayaran telah ditambahkan"

4.3.2.24 Menghapus pembayaran sales

Use case scenario menghapus pembayaran sales dapat dilihat pada Tabel 4.26. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menghapus pembayaran sales.

Tabel 4.26 *Use Case Scenario* Menghapus Pembayaran Sales

Menghapus Pembayaran Sales	
Actor	Pemilik
Objective	Untuk menghapus pembayaran sales
Pre-Condition	Aktor telah *login*, sistem menampilkan halaman pembayaran sales dan aktor telah memilih nama sales yang data pembayaran belum lunasnya ingin dihapus
Main Flow	1. Aktor memilih pembayaran yang ingin dihapus pada tabel 2. Sistem menampilkan pesan konfirmasi 3. Aktor memilih tombol iya 4. Sistem menghapus riwayat transaksi sales kemudian menampilkan halaman pembayaran sales yang telah diperbarui
Alternative Flow	-
Post-Condition	Sistem berhasil menghapus riwayat pembayaran sales yang dipilih aktor dan menampilkan halaman pembayaran sales yang telah di perbarui

25. Mengedit pembayaran sales

Use case scenario mengedit pembayaran sales dapat dilihat pada Tabel 4.27. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* mengedit pembayaran sales. Terdapat *alternative flow* pada *use case scenario* yang terjadi jika hutang telah terlunasi.

Tabel 4.27 *Use Case Scenario* Mengedit Pembayaran Sales

Mengedit Pembayaran Sales	
Actor	Pemilik
Objective	Untuk mengedit pembayaran sales
Pre-Condition	Aktor telah *login*, sistem menampilkan halaman pembayaran sales dan aktor telah memilih nama sales yang data pembayarannya ingin diedit
Main Flow	1. Aktor memilih pembayaran yang ingin diedit kemudian memilih tombol edit 2. Sistem menampilkan *form* edit pembayaran sales 3. Aktor mengisi *form* edit pembayaran sales dan menekan tombol edit 4. Sistem menyimpan masukkan, menampilkan pesan "Anda berhasil mengedit pembayaran", kemudian menampilkan halaman pembayaran sales yang telah diperbarui
Alternative Flow: Jika hutang telah terlunasi	Sistem memperbarui status transaksi menjadi lunas
Post-Condition	Sistem menyimpan masukkan, menampilkan pesan "Anda berhasil mengedit pembayaran", kemudian menampilkan halaman pembayaran sales yang telah diperbarui

4.3.2.26 Menampilkan notifikasi pembayaran sales

Use case scenario menampilkan notifikasi pembayaran sales dapat dilihat pada Tabel 4.28. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menampilkan notifikasi pembayaran sales.

Tabel 4.28 *Use Case Scenario* Menampilkan Notifikasi Pembayaran Sales

Menampilkan noifikasi pembayaran sales	
Actor	Pemilik

Objective	Untuk mengingatkan aktor jika sales belum melunasi pembayaran selama 30 hari setelah transaksi
Pre-Condition	Sistem menampilkan halaman riwayat pembayaran sales
Main Flow	1. Aktor memilih tombol notifikasi 2. Sistem menampilkan *form* notifikasi pembayaran sales belum lunas
Alternative Flow	-
Post-Condition	Sistem menampilkan *form* notifikasi pembayaran sales yang berisi data transaksi belum lunas

4.3.2.27 Mengedit nama sales

Use case scenario mengedit nama sales dapat dilihat pada Tabel 4.29. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* mengedit nama sales. *Use case scenario* ini memiliki *alternative flow* yang terjadi jika nama yang dimasukkan aktor telah ada sebelumnya.

Tabel 4.29 *Use Case Scenario* Mengedit Nama Sales

Mengedit Nama Sales	
Actor	Pemilik
Objective	Untuk mengedit nama sales
Pre-Condition	Aktor telah login dan sistem menampilkan *form* edit nama sales
Main Flow	1. Aktor mengisi nama sales pada *form* edit nama sales kemudian memilih tombol edit 2. Sistem menampilkan pesan "Nama sales berhasil diedit" dan menutup *form* edit nama sales
Alternative Flow: Jika nama yang dimasukkan telah ada sebelumnya	Sistem menampilkan pesan "Nama sales telah digunakan sebelumnya"
Post-Condition	Sistem menyimpan masukkan sales, menampilkan halaman pembayaran sales, dan menampilkan pesan "Nama sales berhasil diedit"

4.3.2.28 Menambah nama sales

Use case scenario menambah nama sales dapat dilihat pada Tabel 4.30. Aktor yang terlibat dalam *use case* ini adalah pemilik. *Use case scenario* ini menggambarkan alur yang terjadi pada *use case* menambah nama sales. Terdapat *alternative flow* pada *use case scenario* ini yaitu jika sistem menampilkan pesan "Nama sales telah digunakan sebelumnya".

Tabel 4.30 *Use Case Scenario* Menambah Nama Sales

Menambah Nama Sales	
Actor	Pemilik
Objective	Untuk menambahkan nama sales
Pre-Condition	Aktor telah *login* dan sistem menampilkan *form* tambah nama sales
Main Flow	1. Aktor mengisi nama sales dan memilih tombol tambah 2. Sistem menyimpan masukkan, menampilkan pesan "Nama sales berhasil ditambahkan" dan menutup *form* tambah nama sales
Alternative Flow	Sistem menampilkan pesan "Nama sales telah digunakan sebelumnya"
Post-Condition	Sistem menyimpan masukkan, menampilkan pesan "Nama sales berhasil ditambahkan" dan menutup *form* tambah nama sales

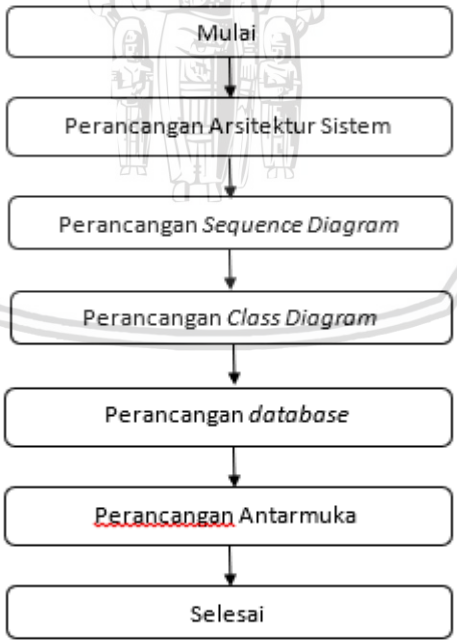
BAB 5 PERANCANGAN DAN IMPLEMENTASI

Dalam bab ini menjelaskan mengenai perancangan sistem pengelolaan transaksi perhiasan perak berbasis *desktop*. Tahapan perancangan dan implementasi dilakukan apabila tahapan analisis telah selesai. Tahapan yang dilakukan pada tahap perancangan adalah melakukan perancangan arsitektur sistem, *activity diagram, sequence diagram, class diagram, entity relationsip diagram* dan perancangan antarmuka.

Sedangkan tahapan yang dilakukan pada tahap implementasi adalah tahapan yang dilakukan saat tahapan perancangan telah selesai dilakukan. Implementasi dilakukan berdasarkan hasil dari analisis dan perancangan. Pada tahapan ini akan menjelaskan mengenai implementasi yang telah dilakukan oleh peneliti, berupa spesifikasi sistem, batasan implementasi, implementasi *database* dan implementasi antarmuka sistem.

5.1 Perancangan sistem

Dalam tahap perancangan dilakukan menggunakan pendekatan berorientasi objek. Perancangan yang dibuat yaitu perancangan arsitektur sistem, perancangan *sequence diagram,* perancangan *class diagram, perancangan entity relationship diagram (ERD),* perancangan *user interface.* Berikut perancangan diagram alir sistem pengelolaan transaksi perhiasan perak digambarkan pada Gambar 5.1.



Gambar 5.1 Perancangan Diagram Alir Sistem

5.1.1 Perancangan arsitektur jaringan sistem

Pada Gambar 5.2 merupakan rancangan arsitektur jaringan sistem pengelolaan transaksi toko perhiasan perak berbasis *desktop*. Desain arsitektur

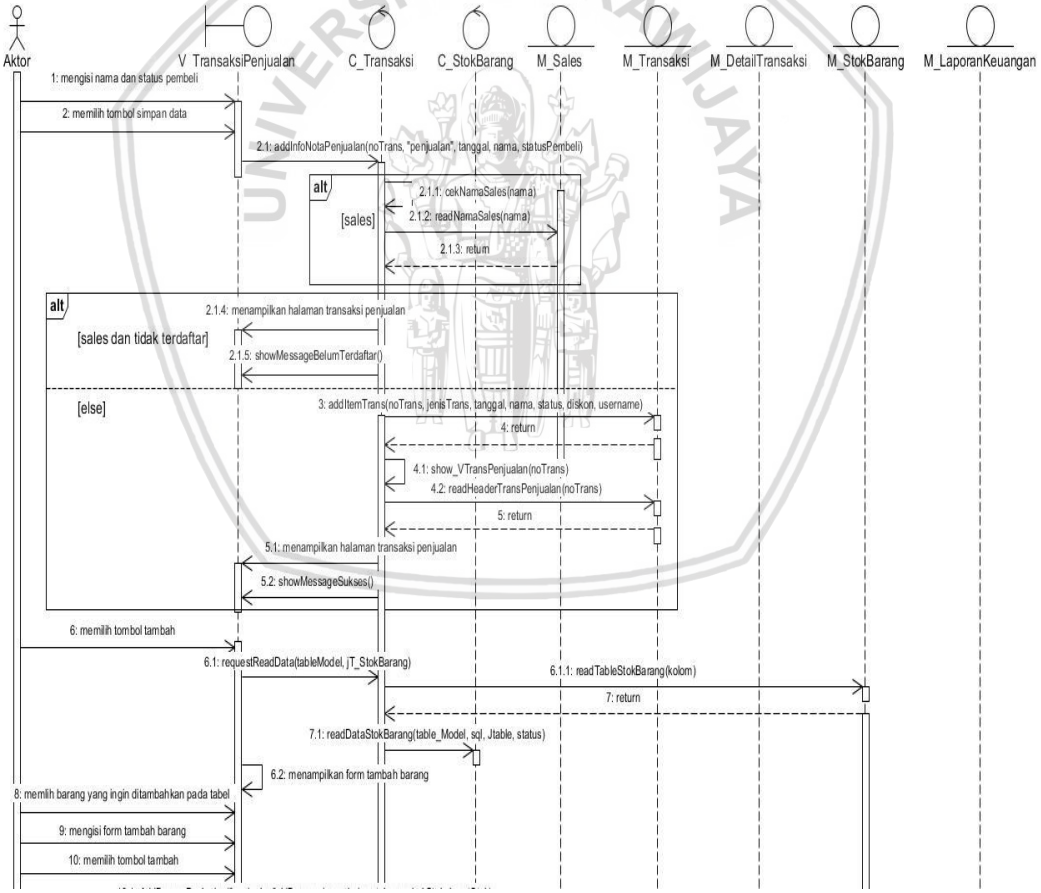
yang digunakan adalah arsitektur jaringan *peer to peer* menggunakan topologi *wireless LAN* mode *ad hoc network*.

Gambar 5.2 Desain *Arsitektur* Jaringan sistem

5.1.2 Perancangan sequence diagram

Perancangan *sequence diagram* dilakukan untuk menggambarkan perilaku antar objek dalam bentuk diagram alir. *Sequence diagram* yang digambarkan mengacu pada *usecase diagram* dan *usecase scenario. Sequence diagram* yang dibuat sebanyak jumlah *usecase* yaitu 28 *sequence diagram*, namun penulis mencantumkan 3 *sequence diagram* dalam skripsi ini.

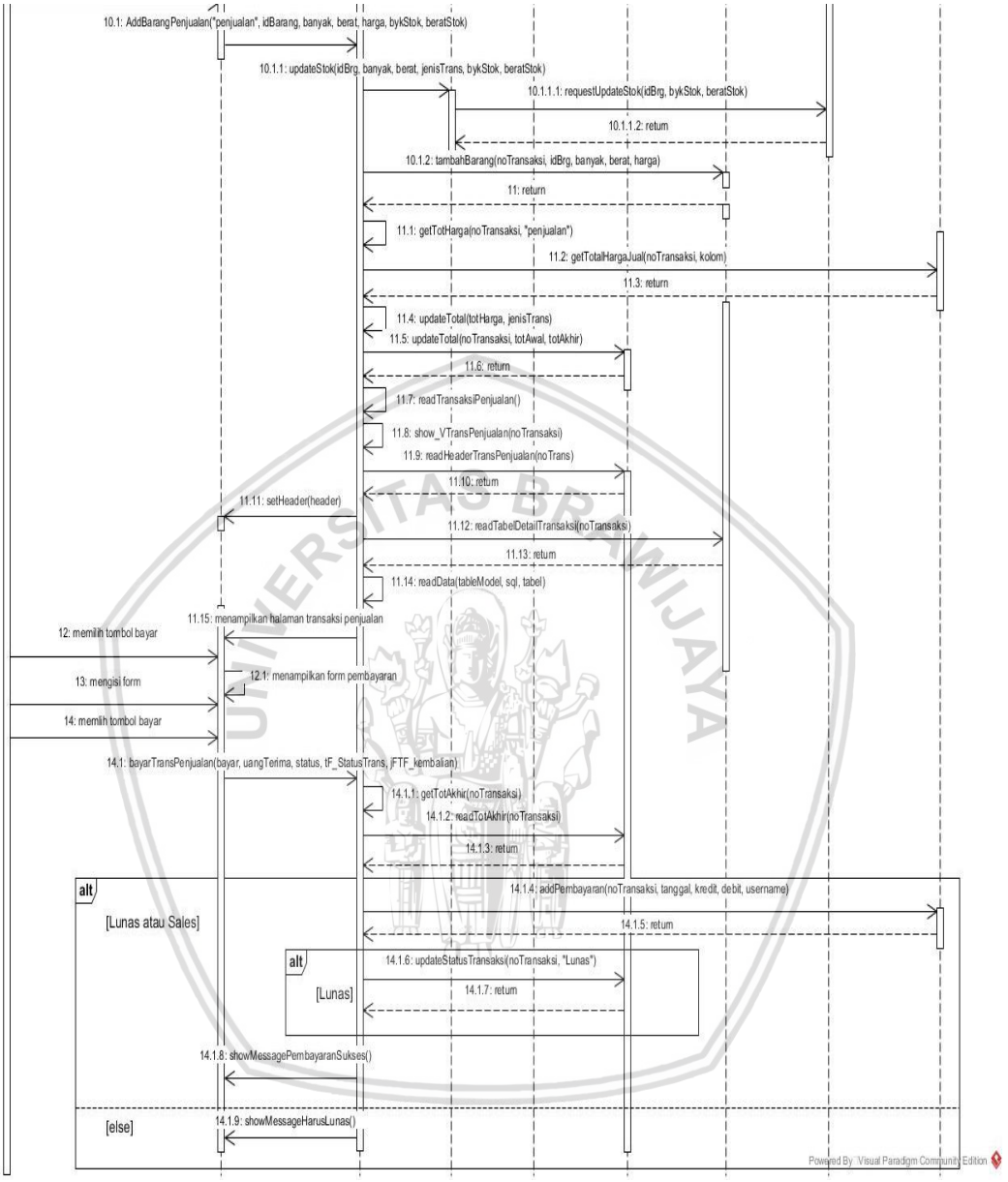
1. *Seqeuence* menambah transaksi penjualan



Gambar 5.3 *Sequence Diagram* Menambah Transaksi Penjualan (1/2)

Sequence diagram menambah transaksi penjualan dimulai dari proses aktor mengisi nama pembeli dan status pembeli kemudian memilih tombol simpan. Setelah itu, jika status pembeli yang diinputkan aktor adalah sales, maka akan dilakukan pengecekan untuk memverifikasi nama pembeli sudah terdaftar

sebagai sales atau belum. Jika belum terdaftar maka akan menampilkan pesan belum terdaftar, namun jika telah terdaftar sistem akan menyimpan masukkan aktor. Kemudian aktor memilih tombol tambah.



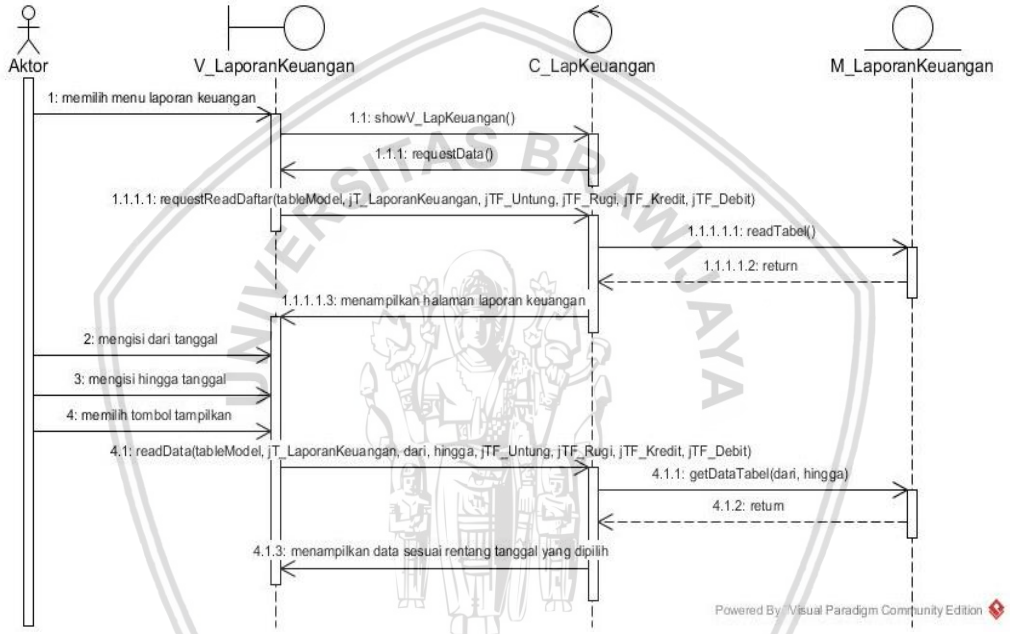
Gambar 5.4 *Sequence Diagram* Menambah Transaksi Penjualan (2/2)

Setelah aktor memilih tombol tambah kemudian sistem menampilkan *form* tambah barang penjualan. Setelah itu aktor mengisi *form* tambah barang penjualan dan memilih tombol tambah yang ada pada *form* tersebut. Sistem melakukan *update* data stok barang, menambahkan barang yang dibeli pada transaksi, melakukan *update* total harga, dan menampilkan data barang yang dibeli pada halaman transaksi penjualan. Setelah itu, aktor memilih tombol bayar kemudian sistem menampilkan *form* pembayaran, aktor mengisi *form* pembayaran dan memilih tombol bayar. Jika lunas, maka sistem akan melakukan *update* status transaksi menjadi lunas. Kemudian sistem menampilkan pesan

pembayaran sukses. Gambar dari sequence diagram menambah transaksi penjualan sales dapat dilihat pada Gambar 5.3 dan Gambar 5.4.

2. *Seqeuence diagram* menampilkan laporan keuangan

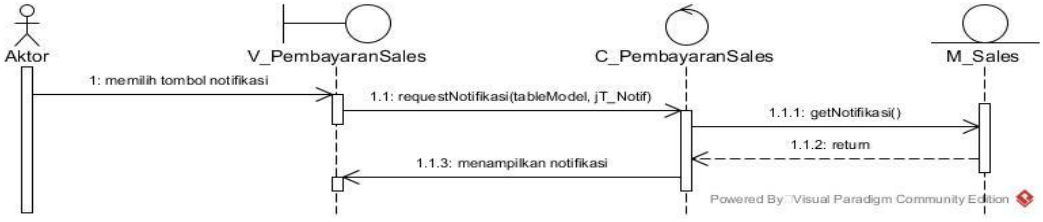
Sequence Diagram menampilkan laporan keuangan menggambarkan proses interaksi objek dalam diagram alir menampilkan laporan keuangan. Prosesnya dimulai dari aktor mengisi data rentang tanggal laporan keuangan yang ingin ditampilkan yaitu pada data dari tanggal dan hingga tanggal. Kemudian aktor memilih tombol tampilkan. Setelah itu, sistem mengambil data pada database dan menampilkan data laporan keuangan sesuai rentang waktu yang diinginkan aktor. *Sequence diagram* menampilkan laporan keuangan dapat dilihat pada Gambar 5.5.



Gambar 5.5 *Sequence Diagram* Menampilkan Laporan Keuangan

3. *Seqeuence Diagram* Menampilkan Notifikasi Pembayaran Sales

Sequence diagram menampilkan notifikasi pembayaran sales menggambarkan perilaku antar objek dalam bentuk diagram alir dari proses menampilkan notifikasi pembayaran sales. Prosesnya dimulai dari aktor memilih tombol notifikasi, kemudian dilanjutkan proses pengambilan data pada *database*. Setelah itu, data ditampilkan pada *form* notifikasi pembayaran sales belum lunas. Berikut adalah gambar *sequence diagram* menampilkan notifikasi pembayaran sales pada Gambar 5.6.



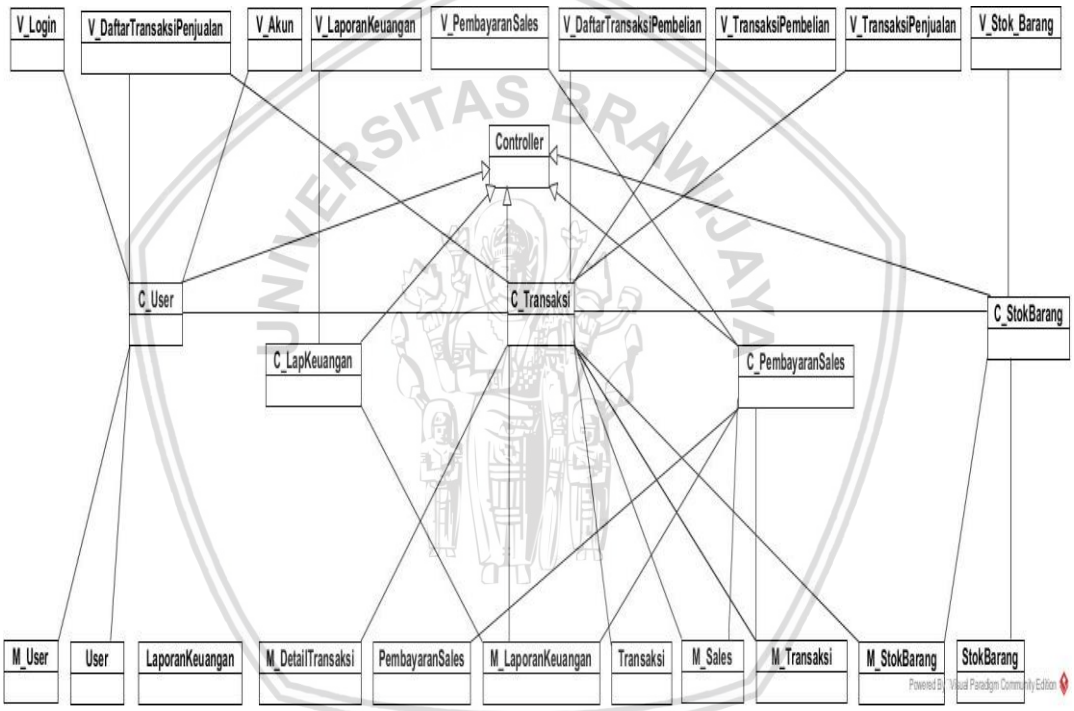
Gambar 5.6 *Sequence Diagram* Menampilkan notifikasi pembayaran sales

5.1.3 Perancangan class diagram

Dalam perancangan *class diagram* terdapat perancangan umum dan perancangan detail. Perancangan umum menggambarakan relasi antar kelas sedangkan perancangan detail menggambarkan atribut dan operasi yang ada pada *class.*

1. Perancangan umum

Berikut ini adalah perancangan umum *class diagram* dari sistem transaksi penjualan perhiasan perak. Perancangan *class diagram* ini dibuat dengan mengacu kepada arsitektur MVC (*Model-View-Controller).* Terdapat kelas-kelas *view* yang memiliki hubungan asosiasi dengan dengan kelas-kelas *controller,* kelas-kelas *controller* berhubungan asosiasi dengan kelas *view,* dan kelas *model Class diagram* perancangan umum dapat dilihat pada Gambar 5.7.



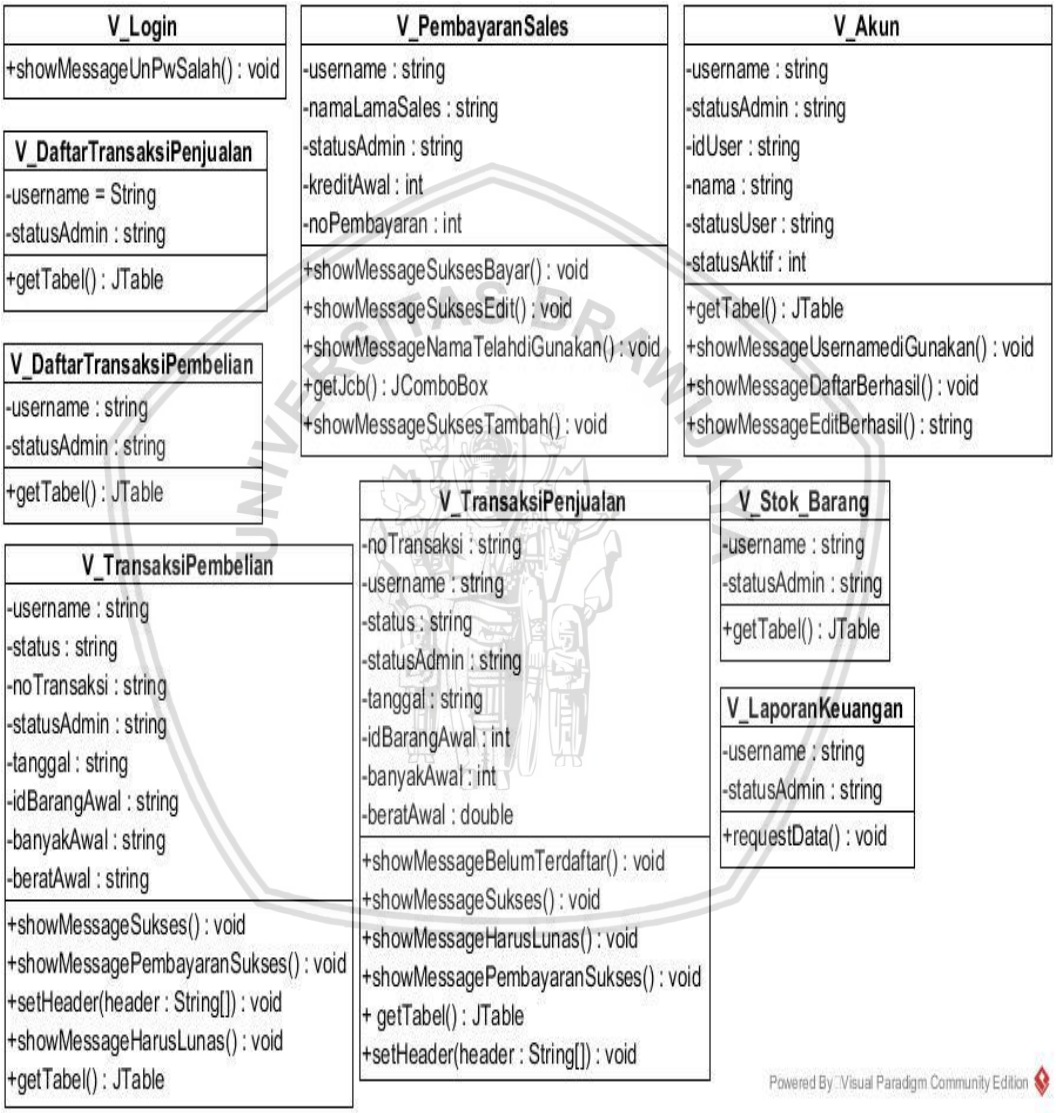
Gambar 5.7 *Class Diagram* Perancangan Umum

2. Perancangan detail

Dalam penggambaran *class diagram* perancangan detail, dibagi menjadi 3 bagian. Bagian pertama menggambarkan kelas model yang dapat dilihat pada Gambar 5.8, kemudian bagian kedua ada kelas *view* yang dapat dilihat pada Gambar 5.9. dan bagian ketiga ada kelas *controller* yang dapat dilihat pada Gambar 5.10. *Class diagram* pada *model* terdapat 13 kelas, yaitu kelas M_LaporanKeuangan, LaporanKeuangan, M_User, User, M_StokBarang, StokBarang, M_DetailTransaksi, Barang, M_Transaksi, Transaksi, M_Sales dan PembayaranSales.

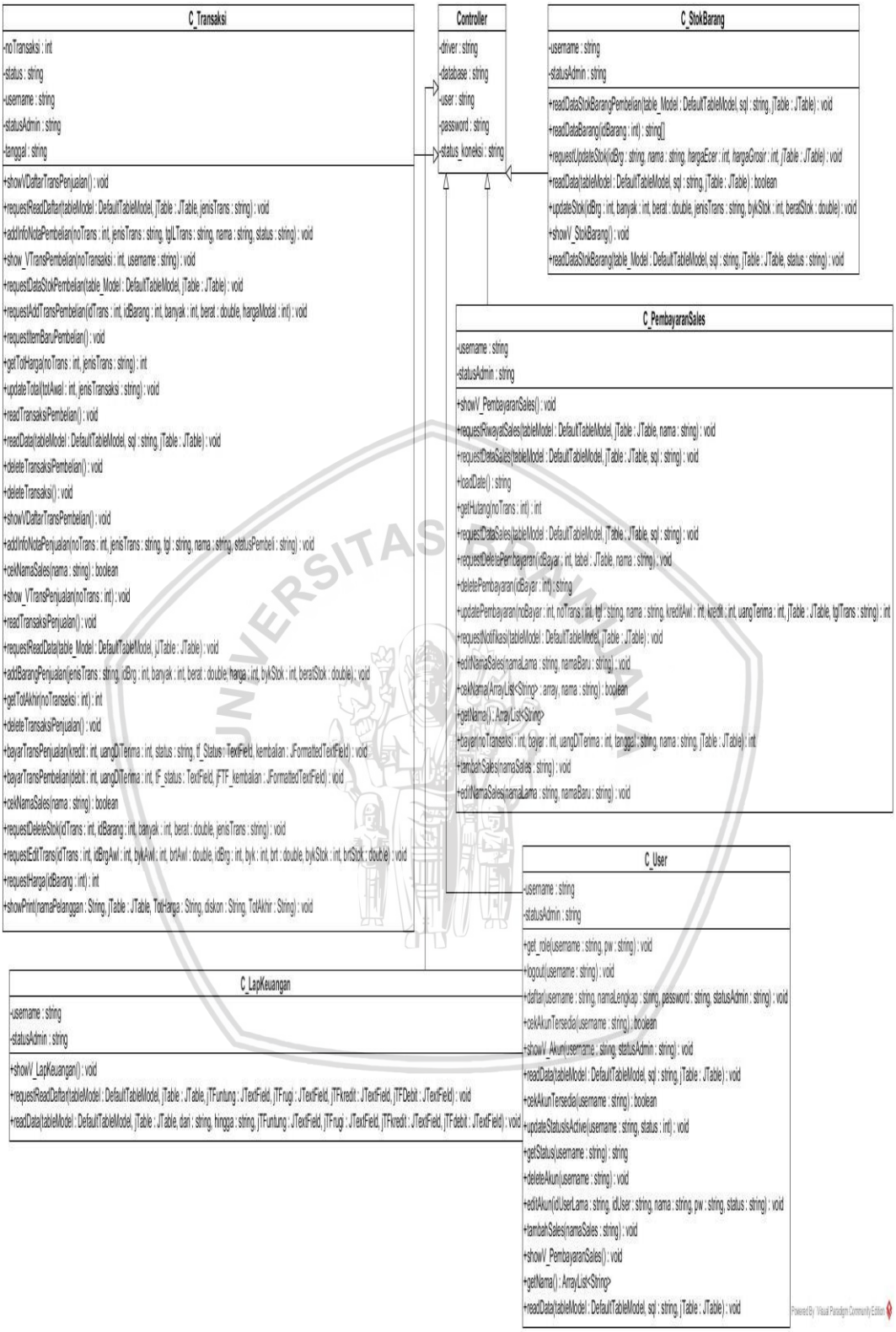
Gambar 5.8 *Class Diagram* pada *Model*

Class diagram pada view terdapat 9 kelas seperti yang terdapat pada Gambar 5.9 yaitu V_Login sebagai halaman login, V_DaftarTransaksiPenjualan sebagai halaman daftar transaksi penjualan, V_DaftarTransaksiPembelian sebagai halaman daftar transaksi pembelian, V_Akun sebagai halaman yang menampilkan data akun, V_StokBarang sebagai halaman stok barang, V_TransaksiPembelian sebagai halaman transaksi pembelian, V_PembayaranSales sebagai halaman pembayaran sales, V_TransaksiPenjualan sebagai halaman transaksi penjualan, dan V_LaporanKeuangan sebagai halaman laporan keuangan.



Gambar 5.9 *Class Diagram* pada *View*

Berikut ini pada Gambar 5.10 menggambarkan *Class diagram* pada modul *controller. Class diagram* pada *controller* terdapat kelas Controller yang memiliki 5 kelas turunan, yaitu C_Transkasi, C_User, C_LapKeuangan, C_PembayaranSales dan C_StokBarang.



Gambar 5.10 *Class Diagram* pada *Controller*

5.1.4 Perancangan algoritme

Dalam perancangan algoritme ini akan menjelaskan proses yang terjadi pada beberapa bagian dari sub-sistem untuk menjalankan fungsionalitas sistem. Perancangan algoritme yang dibuat digunakan sebagai acuan dalam membuat kode program pada tahap implementasi. Dalam perancangan algoritme ini, akan dijelaskan 3 perancangan algoritme yaitu algoritme menambah barang penjualan, menampilkan laporan keuangan berdasarkan rentang waktu dan menampilkan notifikasi pembayaran sales.

1. Perancangan algoritme menambah barang penjualan

Berikut adalah perancangan algoritme untuk menambah barang penjualan yang i dapat dilihat pada Tabel 5.1. Dalam algoritme menambah barang penjualan dilakukan proses *update* pada stok barang, menambah barang dari transaksi penjualan, melakukan update total harga dan total harga akhir, kemudian menampilkan halaman transaksi penjualan yang telah diperbarui.

Nama Kelas: C_Transaksi

Nama Operasi: AddBarangPenjualan()

Tabel 5.1 Perancangan algoritme menambah barang penjualan

Algoritme menambah barang penjualan	
1	Mulai TRY
2	Memanggil operasi updateStok untuk melakukan *update* pada stok barang Memanggil operasi tambahBarang untuk menambahkan barang yang dijual pada detail transaksi penjualan Inisialisasi totHarga=hasil kembalian dari operasi getTotHarga Memanggil operasi updateTotal untuk melakukan update total harga dan total harga akhir Memanggil operasi readTransaksiPenjualan untuk menampilkan transaksi penjualan yang telah diperbarui
3	Jika gagal mengeksekusi sql
4	Menampilkan pesan *error*
5	END TRY Selesai

2. Perancangan algoritme menampilkan laporan keuangan berdasarkan rentang waktu

Berikut adalah perancangan algoritme readData() untuk menampilkan laporan keuangan berdasarkan rentang waktu. Algoritme ini dapat dilihat pada Tabel 5.2. Dalam algoritme menampilkan laporan keuangan berdasarkan rentang waktu dilakukan proses penghapusan baris pada tabel jika tabel telah berisi data, mengambil data laporan keuangan pada *database* berdasarkan rentang waktu tertentu, kemudian menampilkan halaman laporan keuangan yang telah diperbarui.

Nama Kelas: C_LapKeuangan

Nama Operasi: readData()

Algoritme:

Tabel 5.2 Perancangan Algoritme Menampilkan Laporan Keuangan

Algoritme menampilkan laporan keuangan	
1	Mulai Deklarasi variabel noTransaksi, labaBersih, rugi, kredit, hutang, debit, rowCount, tanggalBayar, idUser, jenisTrans Deklarasi dan inisialisasi totKredit = 0, totDebit = 0, totLabaBersih = 0, totRugi = 0 Deklarasi Array data Deklarasi variabel kursIndonesia dan formatRp Mengeset kursIndonesia dan formatRp untuk mengatur format menampilkan angka dalam format rupiah rowCount = jumlah baris pada tabel
2	FOR nilai i awal bernilai 0 hingga i bernilai rowCount-1
3	Hapus baris ke i dalam tabel
4	END FOR
5	Mengosongkan kotak isian untung Mengosongkan kotak isian rugi Mengosongkan kotak isian kredit Mengosongkan kotak isian debit Inisialisasi rowCount=0 Inisialisasi sql = nilai kembalian dari operasi getDataTabel(dari, hingga)
6	TRY

7	Mengeksekusi sql
8	WHILE terdapat hasil *query* SQL{
9	Menambahkan objek pada data(Memasukkan hasil *query* dari kolom jenis_transaksi Memasukkan hasil *query* dari kolom no_transaksi Memasukkan hasil *query* dari kolom tanggal_bayar Memasukkan hasil *query* dari kolom laba bersih Memasukkan hasil *query* dari kolom kredit Memasukkan hasil *query* dari kolom debit Memasukkan hasil *query* dari kolom id_user) Inisialisasi jenisTrans = kembalian dari operasi getJenisTrans() Inisialisasi tanggalBayar = kembalian dari operasi getTanggalBayar() Inisialisasi noTransaksi= kembalian dari operasi getNoTransaksi() Inisialisasi labaBersih = kembalian dari operasi getLabaBersih() Inisialisasi rugi = kembalian dari operasi getRugi() Inisialisasi kredit = kembalian dari operasi getKredit() Inisialisasi hutang = kembalian dari operasi getHutang() Inisialisasi debit = kembalian dari operasi getDebit() Inisialisasi idUser = kembalian dari operasi getIdUser() Inisialisasi totKredit=totKredit+kredit Inisialisasi totDebit=totDebit+debit Inisialisasi totLabaBersih = totLabaBersih + labaBersih Inisialisasi totRugi = totRugi +rugi Memasukkan pada tableModel berupa objek(

	Memasukkan no tabel Memasukkan tanggalBayar Memasukkan jenisTrans Memasukkan noTransaksi Memasukkan kredit Memasukkan idUser) Inisialisasi nilai rowcount=rowcount+1
10	END WHILE Deklarasi variabel sorter Mengeset tabel agar bisa melakukan *sorting* Inisialisasi *field* untung bernilai totLabaBersih dengan format rupiah Inisialisasi *field* rugi bernilai totRugi dengan format rupiah Inisialisasi *field* kredit bernilai totKredit dengan format rupiah Inisialisasi *field* debit bernilai totDebit dengan format rupiah
11	Jika gagal mengeksekusi sql
12	Menampilkan pesan *error*
13	END TRY Selesai

3. Perancangan algoritme menampilkan notifikasi pembayaran sales

Berikut adalah perancangan algoritme untuk menampilkan *form* notifikasi pembayaran sales, yang dapat dilihat pada Tabel 5.3. Dalam algoritme menambah barang penjualan dilakukan proses menghapus baris tabel jika dalam tabel terdapat baris, mengambil data pada database dan menampilkan data hasil *query* sql ke tabel *form* notifikasi.

Nama Kelas: C_PembayaranSales

Nama Operasi: requestNotifikasi()

Algoritme:

Tabel 5.3 Perancangan Algoritme Menampilkan Notifikasi Pembayaran Sales

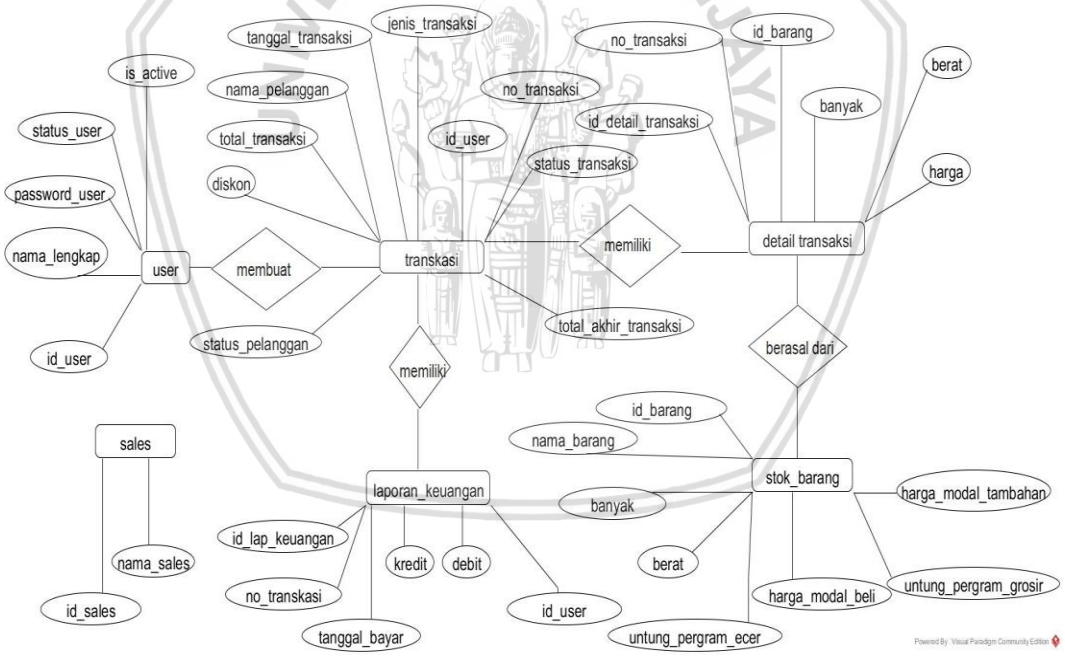
Algoritme menampilkan laporan keuangan	
1	Mulai Deklarasi tanggalTrans, namaSales, sql, rowCount, noTransaksi, hutang Deklarasi array data

	Inisialisasi sql = mengambil kembalian dari pemanggilan operasi getNotifikasi() Inisialisasi rowCount= jumlah baris pada tabel
2	FOR i bernilai dari rowCount-1 hingga 0 lakukan
3	Hapus baris ke-i dalam tabel
4	END FOR
5	Inisialisasi rowCount = 0
6	Try
7	Mengeksekusi *query* sql
8	WHILE terdapat hasil *query* sql{
9	Menambahkan objek pada data(Memasukkan hasil *query* dari kolom no_transaksi Memasukkan hasil *query* dari kolom tanggal_transaksi Memasukkan hasil *query* dari kolom nama_sales Memasukkan hasil *query* dari kolom hutang) Inisialisasi noTransaksi= kembalian dari operasi getNoTransaksi() Inisialisasi tanggalTrans = kembalian dari Operasi getTanggalTrans() Inisialisasi namaSales= kembalian dari operasi getNamaSales() Inisialisasi hutang = kembalian dari operasi getHutang() Memasukkan pada tableModel berupa objek(Memasukkan noTransaksi Memasukkan tanggalTrans Memasukkan nama_sales Memasukkan hutang) Inisialisasi nilai rowcount=rowcount+1
10	END WHILE Inisialisasi variabel sorter Mengeset tabel agar bisa melakukan *sorting*

11	Jika terjadi error
12	Menampilkan pesan
13	END TRY Selesai

5.1.5 Perancangan *database*

Perancangan *database* pada sistem transaksi toko perhiasan perak ini digambarkan pada Gambar 5.11 yang menggambarkan *Entity Relationship Diagram* (ERD) level implementasi dari Sistem Pengelolaan Transaksi Perhiasan Perak Toko Beben. Dimana pada ERD ini terdapat 6 entitas, yaitu user, transaksi, detail_transaksi, sales, laporan_keuangan, dan stok barang. Entitas user memiliki 5 atribut, entitas ini berhubungan dengan entitas transaksi. Entitas detail transaksi memiliki 6 atribut, entitas ini berhubungan dengan entitas stok barang. Entitas transaksi memiliki 10 atribut, entitas ini berhubungan dengan entitas laporan keuangan. Entitas laporan keuangan memiliki 6 atribut dimana entitas ini berhubungan dengan entitas transaksi dan entitas sales memiliki 2 atribut dimana entitas ini tidak terhubung dengan entias yang lainnya.

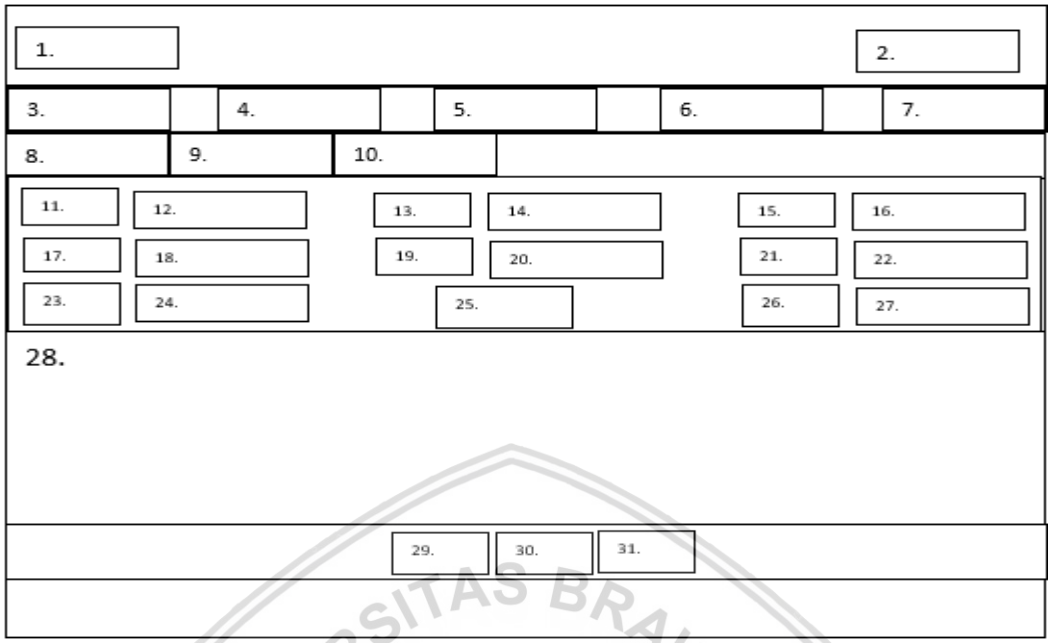


Gambar 5.11 *Entity Relationship Diagram*

5.1.6 Perancangan antarmuka

Sistem ini menggunakan antarmuka berbasis *desktop*. Berikut ini pada Gambar 5.12 adalah perancangan antarmuka halaman transaksi penjualan.

5.1.6.1. Perancangan antarmuka halaman transaksi penjualan



Gambar 5.12 Perancangan antarmuka halaman transaksi penjualan

informasi mengenai objek yang terlibat didalam perancangan antarmuka transaksi penjualan terdapat pada Tabel 5.4.

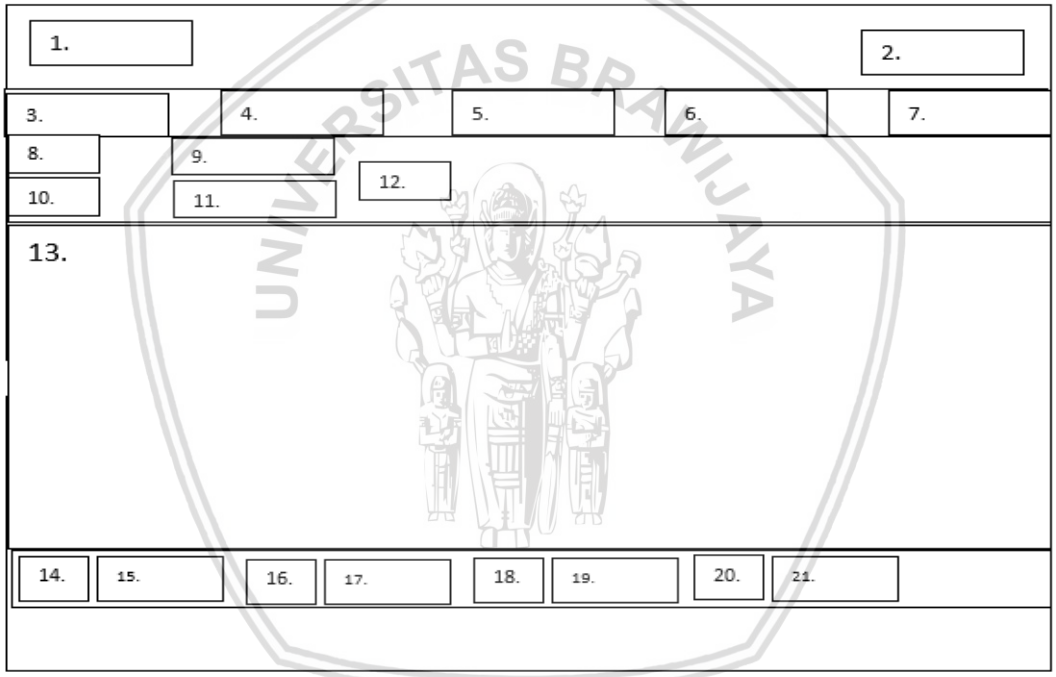
Tabel 5.4 Penjelasan Antarmuka Transaksi Penjualan

No	Nama Objek	Tipe	Keterangan
1	Nama halaman	JLabel	Menampilkan nama halaman berupa transaksi penjualan
2	Nama user	JLabel	Menampilkan nama user
3	Menu transaksi penjualan	JButton	Untuk menampilkan halaman transaksi penjualan
4	Menu transaksi pembelian	JLabel	Untuk menampilkan transaksi pembelian
5	Menu stok barang	JButton	Untuk menampilkan stok barang
6	Menu transaksi sales	JButton	Untuk menampilkan transaksi sales
7	Menu laporan keuangan	JButton	Untuk menampilkan laporan keuangan

8	Tambah	JButton	Untuk menambahkan barang yang ingin dijual
9	Edit	JButton	Untuk mengedit barang yang ingin dijual
10	Hapus	JButton	Untuk menghapus barang yang ingin dijual
11	No transaksi	JLabel	Keterangan no transaksi
12	*Field* no transaksi	JTextField	Menampilkan no transaksi
13	Nama pembeli	JLabel	Keterangan nama pembeli
14	*Field* pembeli	JTextField	Menampilkan nama pembeli
15	Total harga	JLabel	Keterangan total harga
16	*Field* total harga	JTextField	Menampilkan total harga
17	Tanggal	JLabel	Keterangan tanggal
18	*Field* tanggal	JTextField	Menampilkan tanggal transaksi
19	Status pembeli	JLabel	Keterangan status
20	*Field* status pembeli	JTextField	Menampilkan status pembeli
21	Potongan harga	JLabel	Keterangan potongan harga
22	*Field* potongan harga	JTextField	Menampilkan potongan harga
23	Status transaksi	JLabel	Keterangan status transaksi
24	*Field* status transaksi	JTextField	Menampilkan status transaksi
25	Simpan data	JButton	Menyimpan data pembeli

26	Total harga akhir	JLabel	Keterangan total harga akhir
27	*Field* total harga akhir	JTextField	Menampilkan total harga akhir
28	Tabel transaksi penjualan	JTable	Daftar item barang-barang yang dijual
29	Hapus transaksi	JButton	Menghapus transaksi
30	Bayar	JButton	Menampilkan *form* pembayaran
31	Cetak transaksi	JButton	Mencetak transaksi

5.1.6.2. Perancangan Antarmuka Halaman Laporan Keuangan



Gambar 5.13 Perancangan Antarmuka Halaman Laporan Keuangan

Perancangan antarmuka halaman lapora keuangan dapat dilihat pada Gambar 5.13. informasi mengenai objek yang terlibat didalam perancangan antarmuka halaman laporan keuangan terdapat pada Tabel 5.5.

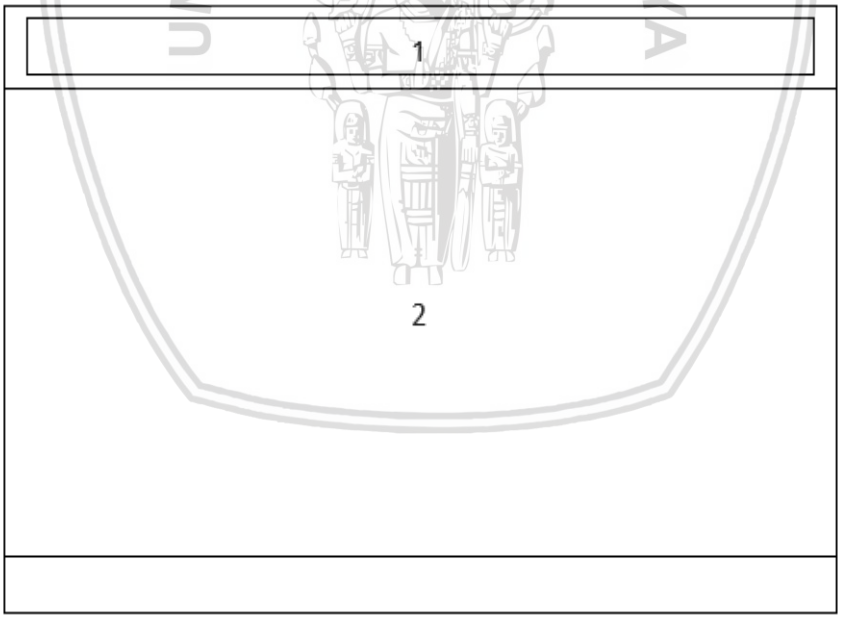
Tabel 5.5 Penjelasan Antarmuka Transaksi Penjualan

No	Nama Objek	Tipe	Keterangan
1	Nama halaman	JLabel	Menampilkan nama halaman berupa transaksi penjualan

2	Nama user	JLabel	Menampilkan nama user
3	Menu transaksi penjualan	JButton	Untuk menampilkan halaman transaksi penjualan
4	Menu transaksi pembelian	JLabel	Untuk menampilkan halaman transaksi pembelian
5	Menu stok barang	JButton	Untuk menampilkan halaman stok barang
6	Menu transaksi sales	JButton	Untuk menampilkan halaman transaksi sales
7	Menu laporan keuangan	JButton	Untuk menampilkan halaman laporan keuangan
8	Dari tanggal	JLabel	Keterangan dari tanggal
9	*Field* dari tanggal	Date chooser combo	Sebagai salah satu parameter pencarian data transaksi untuk ditampilkan tabel
10	Hingga tanggal	JLabel	Keterangan hingga tanggal
11	*Field* hingga tanggal	Date chooser combo	Sebagai salah satu parameter pencarian data transaksi untuk ditampilkan tabel
12	tampilkan	JButton	Untuk memulai pencarian data transaksi berdasarkan data dari tanggal dan hingga tanggal
13	Tabel laporan keuangan	JTable	Berisi data transaksi yang sudah lunas
14	Total untung bersih	JLabel	Keterangan total untung bersih

15	*Field* total untung bersih	JTextField	Menampilkan data untung bersih dari transaksi penjualan
16	Total rugi	JLabel	Keterangan total rugi
17	*Field* rugi	JTextField	Menampilkan data rugi dari transaksi penjualan
18	Total kredit	JLabel	Keterangan total kredit
19	*Field* Total kredit	JTextField	Menampilkan data total kredit dari transaksi penjualan
20	Total debit	JLabel	Keterangan total debit
21	*Field* Total debit	JTextField	Menampilkan data total debit dari transaksi pembelian

5.1.6.3. Perancangan antarmuka notifikasi pembayaran sales



Gambar 5.14 Perancangan *Form* Notifikasi Pembayaran Sales

Perancangan antarmuka halaman *form* notifikasi pembayaran sales dapat dilihat pada Gambar 5.14. Informasi mengenai objek yang terlibat didalam perancangan antarmuka halaman laporan keuangan terdapat pada Tabel 5.6.

Tabel 5.6 Penjelasan Antarmuka Transaksi Penjualan

No	Nama Objek	Tipe	Keterangan
1	Nama *form*	JLabel	Menampilkan judul *form* berupa daftar transaksi sales belum lunas sesuai perjanjian
2	Tabel data sisa pembayaran	jTabel	Menampilkan data sisa pembayaran

5.1.7 Spesifikasi sistem

Spesifikasi sistem yang digunakan untuk membangun sistem ini meliputi spesifikasi *hardware* dan spesifikasi *software.*

5.1.7.1. Spesifikasi hardware

Spesifikasi *hardware* yang digunakan untuk mengembangkan sistem pengelolaan transaksi perhiasan perak dapat dilihat pada Tabel 5.7.

Tabel 5.7 Spesifikasi *Hardware*

Nama Komponen	Spesifikasi
System Model	ASUS A442UA
Processor	Intel® Core™ i5-8250U CPU @ 1.600GHz 1.80 GHz
Memory	RAM 4,00 GB (3.88 GB usable)
Display	NVIDIA GT 930MX
Hard Disk	1 TB
Resolusi Layar	1366x768

5.1.7.2. Spesifikasi *software*

Spesifikasi *software* yang digunakan untuk mengembangkan sistem pengelolaan transaksi perhiasan perak dapat dilihat pada Tabel 5.8.

Tabel 5.8 Spesifikasi *Software*

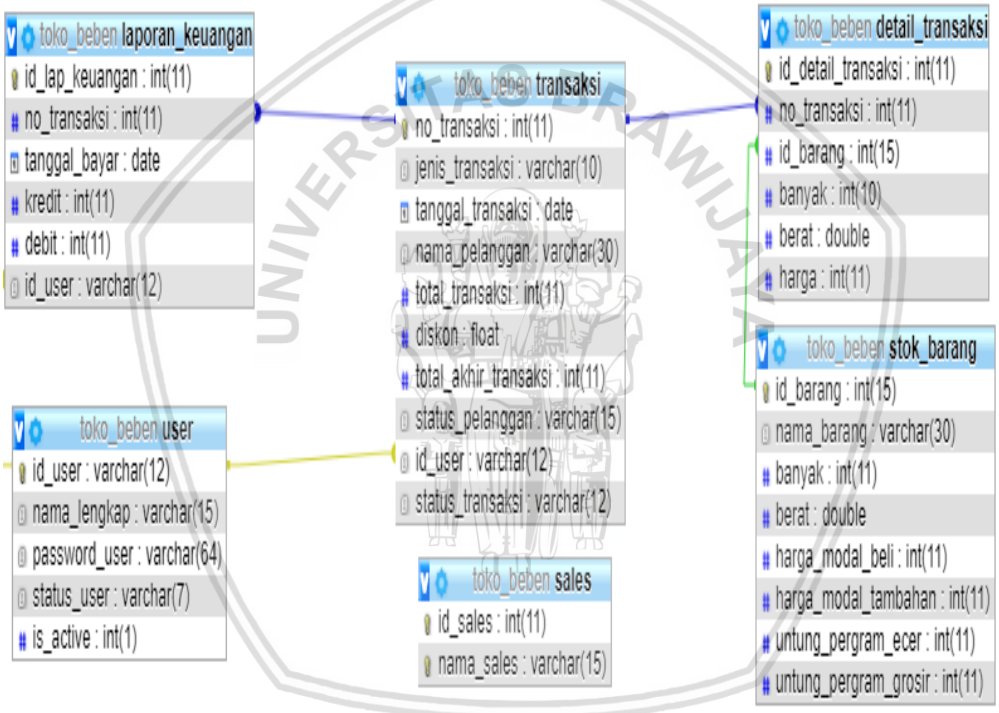
Nama Komponen	Spesifikasi
Operation System	Windows 10 Home 64-bit
Editor Perancangan	Visual Paradigm 14.2
Editor Pemrograman	Netbeans IDE 8.2
Bahasa Pemrograman	Java
DBMS	Mysql

Aplikasi Pendukung Sistem	XAMPP v3.2.2 Microsoft Word 2016
Browser	Google Chrome

5.2 Implementasi sistem

5.2.1 Implementasi *database*

Sistem Pengelolaan Transaksi Toko Perhiasan Perak Berbasis *Desktop* dibuat dengan tujuan untuk membantu proses operasional yang ada pada Toko Perhiasan Beben. Gambar 5.15 menggambarkan skema *database* sistem pengelolaan transaksi perhiasan perak. Dalam skema database tersebut terdapat 6 tabel yaitu tabel laporan_keuangan, transaksi, detail transaksi, stok barang, sales dan user.



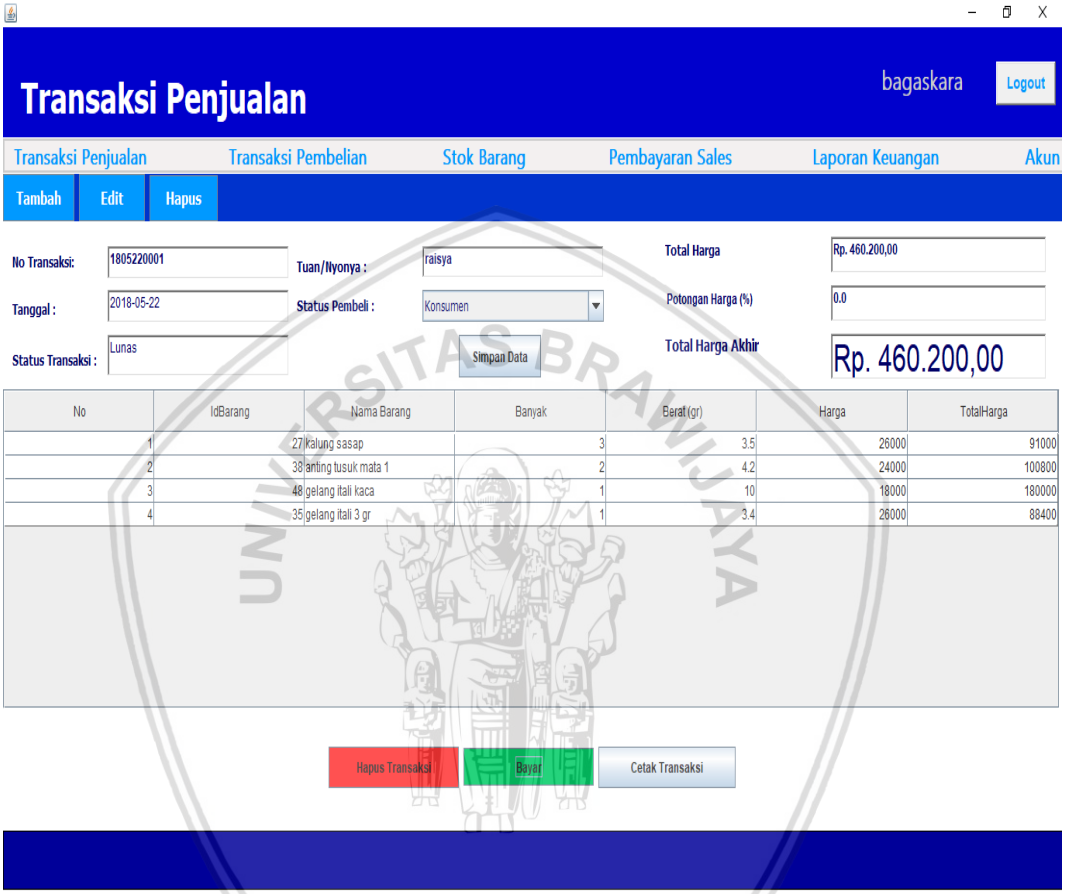
Gambar 5.15 Skema *Database* Sistem Pengelolaan Transaksi Perhiasan Perak

5.2.2 Implementasi antarmuka

Implementasi antarmuka dibuat berdasarkan perancangan antarmuka yang telah dilakukan sebelumnya. Pada Gambar 5.16 menggambarkan implementasi antarmuka halaman transaksi penjualan, Gambar 5.17 menggambarkan hasil implementasi antarmuka pada halaman laporan keuangan dan Gambar 5.18 adalah *form* notifikasi pembayaran sales.

1. Implementasi antarmuka transaksi penjualan

Pada implementasi antarmuka transaksi penjualan terdapat no transaksi, tanggal, status transaksi, tuan/nyonya, status pembeli, total harga, potongan harga, total harga akhir, dan tabel yang berisi data no, id barang, nama barang, banyak, berat, harga, dan total harga. Gambar implementasi antarmuka transaksi penjualan dapat dilihat pada Gambar 5.16.



Gambar 5.16 Implementasi Antarmuka Halaman Transaksi Penjualan

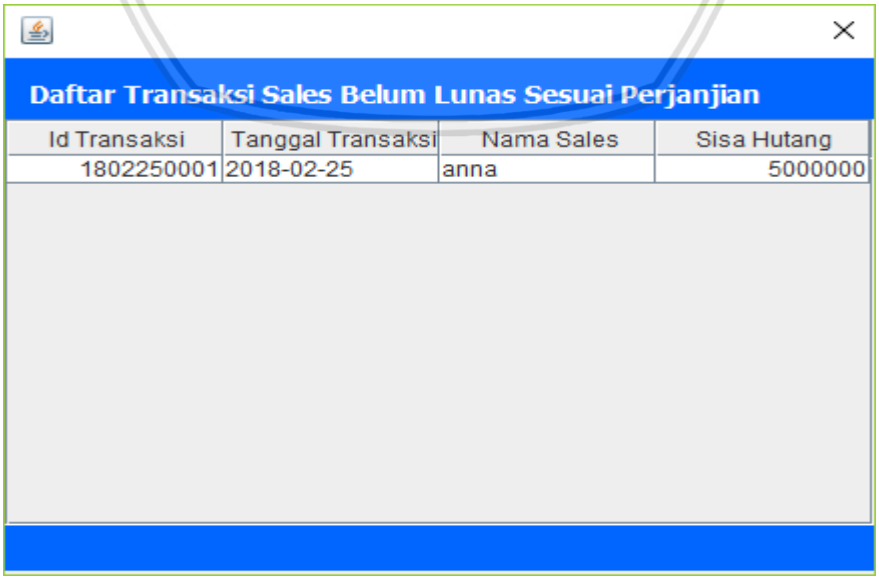
2. Implementasi antarmuka laporan keuangan

Pada halaman laporan keuangan terdapat data dari tanggal, hingga tanggal, *button* tampilkan, total untung bersih, total rugi, dan tabel yang berisi data laporan keuangan yang isinya no, tanggal pembayaran, no transaksi, kredit, debit, saldo, dan nama user. Gambar antarmuka laporan keuangan dapat dilihat pada Gambar 5.17.

Gambar 5.17 Implementasi Antarmuka Halaman Laporan Keuangan

3. Implementasi antarmuka *form* notifikasi pembayaran sales

Pada Gambar 5.18 adalah *form* notifikasi pembayaran sales, dimana *form* ini menggambarkan data pembayaran sales yang belum membayar lunas dalam waktu 30 hari setelah transaksi. Didalam *form* notifikasi pembayaran sales ini terdapat id transaksi, tanggal transaksi, nama sales dan sisa hutang.



Gambar 5.18 Implementasi Antarmuka *Form* Notifikasi Pembayaran Sales

BAB 6 PENGUJIAN DAN ANALISIS

Pengujian adalah tahapan yang dilakukan setelah melakukan tahapan implementasi. Pengujian bertujuan untuk memastikan kesesuaian hasil implementasi telah sesuai dengan analisis kebutuhan dan perancangan sistem yang telah dibuat sebelumnya. Pengujian yang dilakukan ada 2 jenis yaitu pengujian *whitebox* yaitu pengujian unit, sedangkan pengujian *blackbox* berupa pengujian validasi.

6.1 Pengujian unit

Pengujian unit adalah pengujian untuk menguji komponen/unit yang ada di dalam *source code*. Pengujian ini dilakukan untuk memastikan algoritme yang telah diimplementasikan telah sesuai atau tidak terhadap analisis kebutuhan dan perancangan sistem yang dibuat.

6.1.1 Pengujian unit algoritme menambah transaksi penjualan

1. *Pseudocode Algoritme Menambah Barang Penjualan*

Nama Kelas: C_Transaksi

Nama Operasi: AddBarangPenjualan()

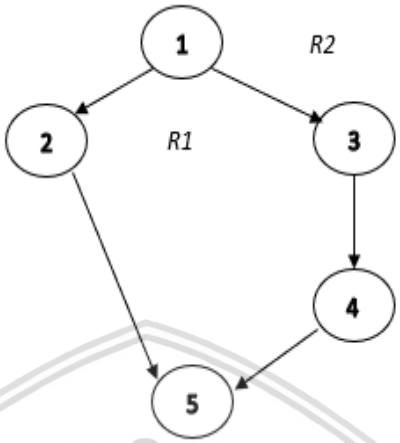
Algoritme:

Tabel 6.1 *Pseudocode* Algoritme Menambah Barang Penjualan

Algoritme menambah barang penjualan	
1	Mulai TRY
2	Memanggil operasi updateStok untuk melakukan *update* pada stok barang Memanggil operasi tambahBarang untuk menambahkan barang yang dijual pada detail transaksi penjualan Inisialisasi totHarga=hasil kembalian dari operasi getTotHarga Memanggil operasi updateTotal untuk melakukan update total harga dan total harga akhir Memanggil operasi readTransaksiPenjualan untuk menampilkan transaksi penjualan yang telah diperbarui
3	Jika gagal mengeksekusi sql
4	Menampilkan pesan *error*
5	END TRY Selesai

2. *Basis Path Testing*

2.1 Flow Graph



Gambar 6.1 *Flow Graph* Menambah Barang Penjualan

2.2 Cyclomatic Complexity

V(G) = R, dimana R adalah jumlah region pada graf

V(G) = 2, ada 2 region yaitu R1 dan R2

V(G) = *5 edges* – 5 *nodes* + 2

= 2

V(G) = *predicate node + 1*

= 1+1

= 2

2.3 Independent Path

Jalur 1 = 1 – 2 – 5

Jalur 2 = 1 – 3 – 4 – 5

Tabel 6.2 Kasus Uji Algoritme Menambah Transaksi Penjualan

No. Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil	Status
1	Mengisi *form* tambah barang dengan id barang = 9, banyak=2, berat=3.2 kemudian memilih tombol tambah maka akan dilakukan pemanggilan operasi updateStok(),	Berhasil menambahkan barang pada transaksi penjualan dan menampilkan halaman	Berhasil menambahkan barang pada transaksi penjualan dan menampilkan halaman	Valid

	tambah barang(), updateTotal(), dan readTransaksiPenjualan()	transaksi penjualan yang telah diperbarui	transaksi penjualan yang telah diperbarui	
2	Penguji mengisi *form* tambah barang dengan id= 23, banyak = 4 dan berat = 4.6. Kemudian memilih tombol tambah. Setelah itu penguji mematikan komponen Apache dan MySQL pada XAMPP yang menyebabkan tidak bisa terhubungnya program ke *database* sehingga akan melakukan proses menampilkan pesan *error.*	Menampilkan pesan error	Barang yang ingin ditambahkan tidak berhasil disimpan dalam *database* dikarenakan tidak terhubungnya aplikasi dengan *database,* sehingga sistem menampilkan pesan *error*	Valid

6.1.2 Pengujian unit algoritme menampilkan laporan keuangan

1. Berikut ini pada Tabel 6.3 adalah tabel yang berisi *pseudocode* perancangan algoritme menampilkan laporan keuangan berdasarkan rentang waktu

Nama Kelas: C_LapKeuangan
Nama Operasi: readData()

Algoritme:

Tabel 6.3 *Pseudocode* Algoritme Menampilkan Laporan Keuangan

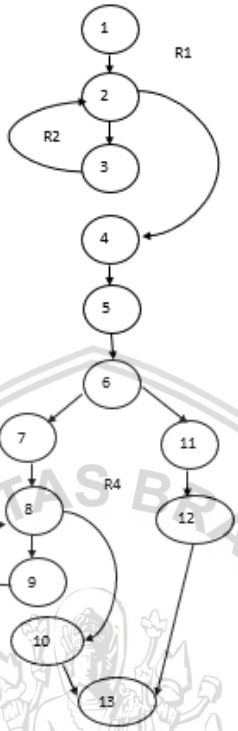
Algoritme menampilkan laporan keuangan	
1	Mulai Deklarasi variabel noTransaksi, labaBersih, rugi, kredit, hutang, debit, rowCount, tanggalBayar, idUser, jenisTrans Deklarasi dan inisialisasi totKredit = 0, totDebit = 0, totLabaBersih = 0, totRugi = 0 Deklarasi Array data Deklarasi variabel kursIndonesia dan formatRp Mengeset kursIndonesia dan formatRp untuk mengatur format menampilkan angka dalam format rupiah rowCount = jumlah baris pada tabel

2	FOR nilai i awal bernilai 0 hingga i bernilai rowCount-1
3	Hapus baris ke i dalam tabel
4	END FOR
5	Mengosongkan kotak isian untung Mengosongkan kotak isian rugi Mengosongkan kotak isian kredit Mengosongkan kotak isian debit Inisialisasi rowCount=0 Inisialisasi sql = nilai kembalian dari operasi getDataTabel(dari, hingga)
6	TRY
7	Mengeksekusi sql
8	WHILE terdapat hasil *query* SQL{
9	Menambahkan objek pada data(Memasukkan hasil *query* dari kolom jenis_transaksi Memasukkan hasil *query* dari kolom no_transaksi Memasukkan hasil *query* dari kolom tanggal_bayar Memasukkan hasil *query* dari kolom laba bersih Memasukkan hasil *query* dari kolom kredit Memasukkan hasil *query* dari kolom debit Memasukkan hasil *query* dari kolom id_user) Inisialisasi jenisTrans = kembalian dari operasi getJenisTrans() Inisialisasi tanggalBayar = kembalian dari operasi getTanggalBayar() Inisialisasi noTransaksi= kembalian dari operasi getNoTransaksi() Inisialisasi labaBersih = kembalian dari operasi getLabaBersih()

	Inisialisasi rugi = kembalian dari operasi getRugi() Inisialisasi kredit = kembalian dari operasi getKredit() Inisialisasi hutang = kembalian dari operasi getHutang() Inisialisasi debit = kembalian dari operasi getDebit() Inisialisasi idUser = kembalian dari operasi getIdUser() Inisialisasi totKredit=totKredit+kredit Inisialisasi totDebit=totDebit+debit Inisialisasi totLabaBersih = totLabaBersih + labaBersih Inisialisasi totRugi = totRugi +rugi Memasukkan pada tableModel berupa objek(Memasukkan no tabel Memasukkan tanggalBayar Memasukkan jenisTrans Memasukkan noTransaksi Memasukkan kredit Memasukkan idUser) Inisialisasi nilai rowcount=rowcount+1
10	END WHILE Deklarasi variabel sorter Mengeset tabel agar bisa melakukan *sorting* Inisialisasi *field* untung bernilai totLabaBersih dengan format rupiah Inisialisasi *field* rugi bernilai totRugi dengan format rupiah Inisialisasi *field* kredit bernilai totKredit dengan format rupiah Inisialisasi *field* debit bernilai totDebit dengan format rupiah
11	Jika gagal mengeksekusi sql
12	Menampilkan pesan *error*
13	END TRY Selesai

2. Basis Path Testing

2.1 Flow Graph



Gambar 6.2 *Flow Graph* Menampilkan Laporan Keuangan

2.2 Cyclomatic Complexity

V(G) = R, dimana R adalah jumlah region pada graf
V(G) = 6, ada 6 region yaitu R1 dan R2

V(G) = edges – nodes + 2

= 15 – 13 + 2

= 4

V(G)= prediacate node + 1

= 3+1

= 4

2.3 Independent Path

Jalur 1: 1 – 2 – 4 – 5 – 6 – 7 – 8 – 10 – 13

Jalur 2: 1 – 2 – 4 – 5 – 6 – 7 – 8 – 9 – 8 – 10 -13

Jalur 3: 1 – 2 – 3 – 2 - 4 – 5 – 6 – 7 – 8 – 9 – 8 – 10 – 13

Jalur 4: 1 – 2 – 3 – 2 – 4 – 5 – 6 – 11 – 12 – 13

Tabel 6.4 Kasus Uji Algoritme Menampilkan Laporan Keuangan

No. Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	Kondisi awal tabel berisi 0 baris, kemudian penguji memilih tanggal 2 Maret 2018 pada bagian dari tanggal dan memilih tanggal 5 Maret 2018 pada bagian hingga tanggal. Setelah itu penguji memilih tombol tampilkan. Kondisi baris pada tabel yang kosong menyebabkan tidak melewati perulangan FOR, kemudian melakukan proses pengosongan nilai pada kotak isian yang tersedia menjadi 0 dan menjalankan proses TRY, kemudian sistem berhasil mengeksekusi *query* SQL, namun tidak terdapat *query result*. Setelah itu dilakukan mengatur tabel agar bisa melakukan *sorting*	Sistem menampilkan halaman laporan keuangan berdasarkan rentang tanggal yang dipilih aktor	Sistem menampilkan halaman laporan keuangan berdasarkan rentang tanggal yang dipilih aktor. Pada halaman laporan keuangan yang ditampilkan, tabel terdiri dari 0 baris.	Valid
2	Kondisi awal tabel berisi 0 baris, kemudian penguji memilih tanggal 1 April 2018 pada	Sistem menampilkan halaman laporan keuangan	Sistem menampilkan halaman laporan keuangan	Valid

	bagian dari tanggal dan memilih tanggal 20 April 2018 pada bagian hingga tanggal. Setelah itu penguji memilih tombol tampilkan. Kondisi baris pada tabel yang kosong sehingga tidak melewati perulangan FOR, kemudian melakukan proses pemberian nilai pada kotak isian yang tersedia menjadi 0 dan menjalankan proses TRY. Sistem berhasil mengeksekusi *query* SQL dan terdapat hasil *query* sql. setelah itu melakukan proses pemasukkan hasil *query* ke objek data, selanjutnya dilakukan pemasukkan data pada tabel, kotak dan mengatur tabel agar bisa melakukan *sorting*	berdasarkan rentang tanggal yang dipilih aktor	berdasarkan rentang tanggal yang dipilih aktor.	
3	Kondisi awal tabel berisi beberapa baris. Kemudian penguji mengisi data dari tanggal dengan nilai 1 April 2018 dan nilai hingga tanggal menjadi 28 April 2018 kemudian	Sistem menampilkan halaman laporan keuangan berdasarkan rentang tanggal yang dipilih aktor	Sistem menampilkan halaman laporan keuangan berdasarkan rentang tanggal yang dipilih aktor	valid

	memilih tombol tampilkan. Sistem merespon dengan memproses masuk ke ke perulangan FOR untuk menghapus isi tabel, kemudian menjalankan proses TRY dan berhasil mengeksekusi *query* SQL. selanjutnya dilakukan pemasukkan data pada tabel, kotak dan mengatur agar tabel agar bisa melakukan *sorting.*			
4	Kondisi awal tabel berisi beberapa baris, kemudian penguji memilih tanggal 31 Maret 2018 pada bagian dari tanggal dan memilih tanggal 29 April 2018 pada bagian hingga tanggal. Setelah itu penguji mematikan komponen MYSQL dan Apache pada XAMPP kemudian penguji memilih tombol tampilkan. Kondisi tabel yang berisi data sehingga masuk pada proses perulangan FOR untuk menghapus data, kemudia memperbarui nilai semua kotak isian pada halaman	Sistem menampilkan pesan error	Berhasil menampilkan pesan *error*	Valid

	laporan keuangan menjadi 0. Setelah itu melakukan proses eksekusi SQL. Namun eksekusi SQL tidak berhasil dikarenakan komponen Apache dan MySQL dimatikan pada XAMPP dengan tujuan agar tidak terhubungnya dengan benar antara program dengan *database,* sehingga akan menjalankan CATCH			

6.1.3 Pengujian unit algoritme menampilkan notifikasi pembayaran sales

1. Berikut ini pada Tabel 6.5 berisi *pseudocode* perancangan algoritme menampilkan notifikasi pembayaran sales

Nama Kelas: C_PembayaranSales

Nama Operasi: requestNotifikasi()

Algoritme:

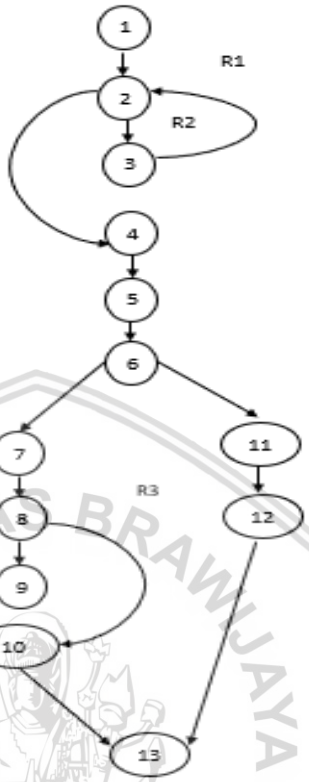
Tabel 6.5 Perancangan Algoritme Menampilkan Notifikasi Pembayaran Sales

Algoritme menampilkan laporan keuangan	
1	Mulai Deklarasi tanggalTrans, namaSales, sql, rowCount, noTransaksi, hutang Deklarasi array data Inisialisasi sql = mengambil kembalian dari pemanggilan operasi getNotifikasi() Inisialisasi rowCount= jumlah baris pada tabel
2	FOR i bernilai dari rowCount-1 hingga 0 lakukan
3	Hapus baris ke-i dalam tabel

4	END FOR
5	Inisialisasi rowCount = 0
6	Try
7	Mengeksekusi *query* sql
8	WHILE terdapat hasil *query* sql{
9	Menambahkan objek pada data(Memasukkan hasil *query* dari kolom no_transaksi Memasukkan hasil *query* dari kolom tanggal_transaksi Memasukkan hasil *query* dari kolom nama_sales Memasukkan hasil *query* dari kolom hutang) Inisialisasi noTransaksi= kembalian dari operasi getNoTransaksi() Inisialisasi tanggalTrans = kembalian dari Operasi getTanggalTrans() Inisialisasi namaSales= kembalian dari operasi getNamaSales() Inisialisasi hutang = kembalian dari operasi getHutang() Memasukkan pada tableModel berupa objek(Memasukkan noTransaksi Memasukkan tanggalTrans Memasukkan nama_sales Memasukkan hutang) Inisialisasi nilai rowcount=rowcount+1
10	END WHILE Inisialisasi variabel sorter Mengeset tabel agar bisa melakukan *sorting*
11	Jika terjadi error
12	Menampilkan pesan
13	END TRY Selesai

2. Basis Path Testing

2.1 Flow Graph



Gambar 6.3 *Flow Graph* Menampilkan Notifikasi Pembayaran

2.2 Cyclomatic Complexity

V(G) = R, dimana R adalah jumlah region pada graf

V(G) = 4, ada 4 region yaitu R1, R2, R3, R4 dan R5

V(G) = *15 edges* – 13 *nodes* + 2

= 4

V(G) = *predicate node + 1*

= 3+1

= 4

2.3 Independent Path

Jalur 1: 1 – 2 – 4 – 5 – 6 – 7 – 8 – 10 – 13

Jalur 2: 1 – 2 – 4 – 5 – 6 – 7 – 8 – 9 – 8 – 10 -13

Jalur 3: 1 – 2 – 3 – 2 - 4 – 5 – 6 – 7 – 8 – 9 – 8 – 10 – 13

Jalur 4: 1 – 2 – 3 – 2 – 4 – 5 – 6 – 11 – 12 – 13

Tabel 6.6 Kasus Uji Algoritme Menampilkan Notifikasi Pembayaran Sales

No	No. Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil	Status
1	1	Kondisi awal tabel kosong kemudian dilakukan proses eksekusi sql, namun tidak terdapat hasil dari *query* sql, setelah itu dilakukan pengaturan agar tabel bisa melakukan *sorting*	Sistem menampilkan *form* notifikasi pembayaran sales belum lunas	Sistem menampilkan *form* notifikasi pembayaran sales belum lunas. Namun, isi tabel kosong.	Valid
2	2	Kondisi awal tabel kosong kemudian dilakukan proses eksekusi sql, hasil dari *query dimasukkan* ke *array* data, kemudian memasukkan data ke tabel, dan dilakukan pengaturan agar tabel bisa melakukan *sorting*	Sistem menampilkan *form* notifikasi pembayaran sales belum lunas	Sistem menampilkan *form* notifikasi pembayaran sales belum lunas	Valid
3	3	Kondisi awal tabel berisi baris kemudian dalam perulangan FOR baris dalam sql dihapus, selanjutnya dilakukan proses eksekusi sql, hasil dari *query dimasukkan* ke *array* data, kemudian memasukkan data ke tabel,	Sistem menampilkan *form* notifikasi pembayaran sales belum lunas	Sistem menampilkan *form* notifikasi pembayaran sales belum lunas	valid

		setelah itu dilakukan pengaturan agar tabel bisa melakukan *sorting*			
4	4	Kondisi awal tabel berisi baris kemudian baris dalam sql dihapus, kemudian menjalankan FOR untuk menghapus baris tabel. Setelah itu dilakukan eksekusi sql tidak berhasil dikarenakan komponen Apache dan MySQL dimatikan pada XAMPP dengan tujuan tidak terhubungnya dengan benar antara program dengan *database,* sehingga akan menjalankan CATCH.	Menampilkan pesan error	Menampilkan pesan error	Valid

6.1.4 Hasil kasus uji

Pengujian pada kasus uji yang pertama memiliki nilai *cyclomatic complexity* bernilai 2, kasus uji yang kedua memiliki nilai *cyclomatic complexity* bernilai 4, dan kasus uji yang ketiga memiliki nilai *cyclomatic complexity* bernilai 4. Nilai *cyclomatic complexity* pada tiga algoritme yang diuji termasuk dalam rentang nilai 1-10. Menurut (Guru99, 2018), rentang nilai 1-10 berarti sistem yang dibangun memiliki kode struktur yang baik dan *high testability.*

6.2 Pengujian validasi

Pengujian validasi adalah pengujian yang bertujuan untuk memastikan sistem yang dibangun telah sesuai dengan analisis kebutuhan yang telah dibuat

sebelumnya. Pengujian yang dilakukan menggunakan blackbox *testing*. Dalam pengujian validasi ini akan dilakukan pemeriksaan kesesuaian kebutuhan fungsional yang telah didefinisikan sebelumnya.

1. Kasus Uji Login

   1.1 Kasus Uji Validasi *Login* Berhasil

Tabel 6.7 Kasus Uji Validasi Login Berhasil

Nomor kasus uji	PF_SPTF_0101
Nama kasus uji	Kasus Uji Validasi *Login*
Prosedur pengujian	1. Penguji masuk ke halaman *login* 2. Penguji memasukkan *username* dan *password* 3. Penguji menekan tombol *login*
Hasil uji yang diharapkan	Sistem menampilkan halaman daftar transaksi penjualan
Hasil uji	Sistem menampilkan halaman daftar transaksi penjualan
Status Pengujian	Valid

1.2 Kasus Uji Validasi *Login* dengan *Username* dan *Password* Salah

Tabel 6.8 Kasus Uji Validasi *Login* dengan Username dan Password Salah

Nomor kasus uji	PF_SPTF_0102
Nama kasus uji	Kasus Uji Validasi *Login* dengan *Username* dan *password* salah
Prosedur pengujian	1. Penguji masuk ke halaman *login* 2. Penguji memasukkan *username* dan *password* yang salah 3. Penguji menekan tombol *login*
Hasil uji yang diharapkan	Sistem menampilkan pesan "*Username* atau *password* anda salah".
Hasil uji	Sistem menampilkan pesan "*Username* atau *password* anda salah".
Status Pengujian	Valid

2. Kasus Uji Logout

Tabel 6.9 Kasus Uji Logout

Nomor Kasus Uji	PF_SPTF_0201
Nama Kasus Uji	Kasus Uji Validasi Logout
Prosedur	1. Penguji masuk kehalaman daftar transaksi penjualan 2. Penguji memilih tombol logout
Hasil yang diharapkan	Sistem menampilkan halaman *login*
Hasil	Sistem menampilkan halaman *login*
Status	Valid

3. Kasus uji menambah akun

3.1 menambah akun berhasil

Tabel 6.10 Kasus Uji Menambah Akun

Nomor Kasus Uji	PF_SPTF0301
Nama Kasus Uji	Kasus Uji Validasi Menambah Akun
Prosedur	1. Penguji mengisi *form* tambah akun berupa *username*, nama lengkap, password, dan status 2. Penguji memilih tombol tambah
Hasil yang diharapkan	Sistem menyimpan masukkan aktor, menampilkan pesan "pendaftaran berhasil" dan menampilkan halaman data akun yang telah di perbarui
Hasil	Sistem menyimpan masukkan aktor, menampilkan pesan "pendaftaran berhasil" dan menampilkan halaman data akun yang telah di perbarui
Status	Valid

3.2 Kasus uji menambah akun dengan *username* yang telah ada

Tabel 6.11 Kasus Uji Menambah Akun dengan *Username* yang Telah Ada

Nomor Kasus Uji	PF_SPTF0302
Nama Kasus Uji	Kasus Uji Validasi Menambah Akun dengan *Username* telah Ada

Prosedur	1. Penguji masuk ke halaman *form* tambah akun 2. Penguji mengisi *username* dengan *username* yang telah terdaftar pada sistem, kemudian mengisi nama lengkap, password, dan status 3. Penguji memilih tombol tambah
Hasil yang diharapkan	Sistem menampilkan pesan "*username* telah digunakan"
Hasil	Sistem menampilkan pesan "*username* telah digunakan"
Status	Valid

4. Kasus Uji Menghapus Akun

Tabel 6.12 Kasus Uji Menghapus Akun

Nomor Kasus Uji	PF_SPTF_0401
Nama Kasus Uji	Kasus Uji Validasi Menghapus Akun
Prosedur	1. Penguji masuk kehalaman data akun 2. Penguji memilih tombol hapus
Hasil yang diharapkan	Sistem menghapus akun dari *database* dan menampilkan halaman data akun yang telah diperbarui
Hasil	Sistem menghapus akun dari *database* dan menampilkan halaman data akun yang telah diperbarui
Status	Valid

5. Kasus uji mengedit akun

5.1 Kasus uji mengedit akun berhasil

Tabel 6.13 Kasus Uji Mengedit Akun

Nomor Kasus Uji	PF_SPTF_0501
Nama Kasus Uji	Kasus Uji Validasi Mengedit Akun
Prosedur	1. Penguji masuk kehalaman *form* edit akun 2. Penguji mengisi *form* edit dan memilih tombol edit 3. Penguji memilih tombol edit

Hasil yang diharapkan	Sistem menyimpan masukkan aktor dan menampilkan pesan "Akun berhasil diedit", kemudian menampilkan halaman data akun yang telah diperbarui
Hasil	Sistem menyimpan masukkan aktor dan menampilkan pesan "Akun berhasil diedit", kemudian menampilkan halaman data akun yang telah diperbarui
Status	Valid

5.2 Kasus uji mengedit akun dengan mengisi username yang telah terdaftar pada sistem

Tabel 6.14 Kasus Uji Mengedit Akun

Nomor Kasus Uji	PF_SPTF_0502
Nama Kasus Uji	Kasus Uji Validasi Mengedit Akun
Prosedur	1. Penguji masuk kehalaman *form* edit akun 2. Penguji mengisi *form* edit akun berupa *username* dengan mengisikan *username* yang telah terdaftar pada sistem, nama lengkap, *password* dan status 3. Penguji memilih tombol edit pada *form* edit akun
Hasil yang diharapkan	Sistem menampilkan pesan "username telah digunakan"
Hasil	Sistem menampilkan pesan "username telah digunakan"
Status	Valid

6. Kasus uji melihat akun

Tabel 6.15 Kasus Uji Melihat Akun

Nomor Kasus Uji	PF_SPTF_0601
Nama Kasus Uji	Kasus Uji Validasi Melihat Akun Pegawai
Prosedur	1. Penguji masuk kehalaman daftar transaksi penjualan 2. Penguji memilih tombol Akun
Hasil yang diharapkan	Sistem menampilkan halaman data akun

Hasil	Sistem menampilkan halaman data akun
Status	Valid

7. Kasus uji menambah transaksi pembelian

Tabel 6.16 Kasus Uji Menambah Transaksi Pembelian

Nomor Kasus Uji	PF_SPTF_0701
Nama Kasus Uji	Kasus Uji Menambah Transaksi Pembelian
Prosedur	1. Penguji masuk ke halaman transaksi pembelian 2. Penguji mengisi nama dan status penjual kemudian memilih tombol simpan 3. Penguji memilih tombol tambah pada *form* tambah transaksi pembelian 4. Penguji memilih barang yang ingin ditambahkan pada tabel, mengisi *form* dan memilih tombol tambah pada *form* tambah barang 5. Penguji memilih tombol tambah pada *form* tambah transaksi 6. Penguji memilih tombol tambah jenis barang baru pada *form* tambah barang 7. penguji mengisi *form* tambah jenis barang baru kemudian memilih tombol tambah 8. Penguji memilih tombol bayar 9. Penguji mengisi bayar dan uang di terima kemudian memilih tombol bayar pada *form* pembayaran
Hasil yang diharapkan	Sistem menyimpan data pembayaran penguji, kemudian sistem menampilkan pesan "Pembayaran telah selesai dilakukan" dan menampilkan *form* transaksi pembayaran yang berstatus lunas
Hasil	Sistem menyimpan data pembayaran penguji, kemudian sistem menampilkan pesan "Pembayaran telah selesai dilakukan" dan menampilkan *form* transaksi pembayaran yang berstatus lunas
Status	Valid

8. Kasus uji mengedit barang transaksi pembelian

Tabel 6.17 Kasus Uji Mengedit Transaksi Pembelian

Nomor Kasus Uji	PF_SPTF_0801
Nama Kasus Uji	Kasus Uji Mengedit barang Transaksi Pembelian
Prosedur	.1 Penguji masuk kehalaman tambah transaksi pembelian yang tabelnya telah terisi data barang yang akan dibeli .2 Penguji memilih barang pada tabel *form* tambah transaksi pembelian .3 Penguji memilih tombol hapus .4 Penguji memilih barang pada tabel *edit* barang kemudian mengisi *form* edit barang. Setelah itu penguji memilih tombol edit
Hasil yang diharapkan	Sistem memperbarui data sesuai masukkan aktor kemudian menampilkan halaman *form* tambah transaksi pembelian
Hasil	Sistem memperbarui data sesuai masukkan aktor kemudian menampilkan halaman *form* tambah transaksi pembelian
Status	Valid

9. Kasus uji menghapus barang transaksi pembelian

Tabel 6.18 Kasus Uji Menghapus Barang Transaksi Pembelian

Nomor Kasus Uji	PF_SPTF_0901
Nama Kasus Uji	Kasus Uji Menghapus barang Transaksi Pembelian
Prosedur	.1 Penguji masuk ke halaman tambah transaksi pembelian yang tabelnya telah terisi data barang yang akan dibeli .2 Penguji memilih barang pada tabel kemudian memilih tombol hapus
Hasil yang diharapkan	Sistem berhasil menghapus barang yang dipilih kemudian menampilkan halaman tambah transaksi yang telah diperbarui

Hasil	Sistem berhasil menghapus barang yang dipilih kemudian menampilkan halaman tambah transaksi yang telah diperbarui
Status	Valid

10. Kasus uji menghapus transaksi pembelian

Tabel 6.19 Kasus Uji Menghapus Transaksi Pembelian

Nomor Kasus Uji	PF_SPTF_1001
Nama Kasus Uji	Kasus Uji Menghapus Transaksi Pembelian
Prosedur	1.1 Penguji masuk ke halaman transaksi pembelian yang telah tersimpan 1.2 Penguji memilih tombol hapus transaksi 1.3 Penguji memilih tombol iya
Hasil yang diharapkan	Sistem menghapus transaksi yang dipilih aktor dan menampilkan halaman daftar transaksi pembelian yang telah diperbarui
Hasil	Sistem menghapus transaksi yang dipilih aktor dan menampilkan halaman daftar transaksi pembelian yang telah diperbarui
Status	valid

11. Kasus uji menampilkan transaksi pembelian

Tabel 6.20 Kasus Uji Menampilkan Transaksi Pembelian

Nomor Kasus Uji	PF_SPTF_1101
Nama Kasus Uji	Kasus Uji Validasi Menampilkan Transaksi Pembelian
Prosedur	1.1.1 Penguji masuk ke halaman daftar transaksi penjualan 1.1.2 Penguji memilih menu transaksi pembelian 1.1.3 Penguji memilih transaksi yang ingin ditampilkan dalam tabel
Hasil yang diharapkan	Sistem menampilkan halaman transaksi pembelian yang telah dipilih
Hasil	Sistem menampilkan halaman transaksi pembelian yang telah dipilih
Status	Valid

12. Kasus uji mencetak transaksi pembelian

Tabel 6.21 Kasus Uji Mencetak Transaksi Pembelian

Nomor Kasus Uji	PF_SPTF_1201
Nama Kasus Uji	Kasus Uji Mencetak Transaksi Pembelian
Prosedur	1.1 Penguji masuk kehalaman tambah transaksi pembelian yang telah terisi barang-barang pembelian 1.2 Penguji memilih tombol cetak transaksi
Hasil yang diharapkan	Sistem menampilkan *file* pdf yang berisi detail transaksi pembelian yang dipilih
Hasil	Sistem menampilkan *file* pdf yang berisi detail transaksi pembelian yang dipilih
Status	valid

13. Kasus uji menambah transaksi penjualan

13.1 Kasus uji menambah transaksi penjualan sebagai konsumen, *reseller,* atau sales dan membayar lunas transaksi

Tabel 6.22 Kasus Uji Menambah Transaksi Penjualan

Nomor Kasus Uji	PF_SPTF_1301
Nama Kasus Uji	Pengujian validasi menambah transaksi penjualan
Prosedur	1. Penguji masuk ke halaman transaksi penjualan 1.1 Penguji mengisi nama pembeli dan status pembeli, kemudian memilih tombol simpan data 1.2 Penguji memilih tombol tambah 1.3 Penguji memilih barang yang ingin ditambahkan pada tabel kemudian mengisi *form* tambah barang. Setelah itu penguji memilih tombol tambah pada *form* tambah barang 1.4 Penguji memilih tombol bayar pada halaman transaksi penjualan 1.5 Penguji mengisi *form* pembayaran berupa uang yang dibayarkan dan diterima (pembeli

	membayar lunas) kemudian memilih tombol bayar
Hasil yang diharapkan	Sistem menampilkan uang kembalian, merubah status transaksi menjadi lunas, dan menampilkan pesan "Pembayaran telah selesai dilakukan"
Hasil	Sistem menampilkan uang kembalian, merubah status transaksi menjadi lunas, dan menampilkan pesan "Pembayaran telah selesai dilakukan"
Status	valid

13.2 Kasus uji menambah transaksi penjualan sebagai konsumen atau *reseller* dan tidak membayar lunas

Tabel 6.23 Kasus Uji Menambah Transaksi Penjualan

Nomor Kasus Uji	PF_SPTF_1302
Nama Kasus Uji	Pengujian validasi menambah transaksi penjualan
Prosedur	2. Penguji masuk ke halaman transaksi penjualan 1.1 Penguji mengisi nama pembeli dan status pembeli sebagai konsumen atau *reseller*, kemudian memilih tombol simpan data 1.2 Penguji memilih tombol tambah 1.3 Penguji memilih barang yang ingin ditambahkan pada tabel kemudian mengisi *form* tambah barang. Setelah itu penguji memilih tombol tambah pada *form* tambah barang 1.4 Penguji memilih tombol bayar pada halaman transaksi penjualan 1.5 Penguji mengisi *form* pembayaran berupa uang yang dibayarkan dan diterima dimana uang yang dibayarkan belum dapat melunasi tagihan total akhir belanja, kemudian penguji memilih tombol bayar
Hasil yang diharapkan	Sistem menampilkan pesan "Pembeli harus membayar lunas"

Hasil	Sistem menampilkan pesan "Pembeli harus membayar lunas"
Status	valid

13.3 Kasus uji menambah transaksi penjualan sebagai sales dan tidak membayar lunas

Tabel 6.24 Kasus Uji Menambah Transaksi Penjualan

Nomor Kasus Uji	PF_SPTF_1303
Nama Kasus Uji	Pengujian validasi menambah transaksi penjualan
Prosedur	1. Penguji masuk ke halaman transaksi penjualan 1.1 Penguji mengisi nama pembeli dan status pembeli, kemudian memilih tombol simpan data 1.2 Penguji memilih tombol tambah 1.3 Penguji memilih barang yang ingin ditambahkan pada tabel kemudian mengisi *form* tambah barang. Setelah itu penguji memilih tombol tambah pada *form* tambah barang 1.4 Penguji memilih tombol bayar pada halaman transaksi penjualan 1.5 Penguji mengisi *form* pembayaran berupa uang yang dibayarkan dan diterima dimana uang yang dibayarkan belum dapat melunasi tagihan total akhir transaksi pembelian, kemudian penguji memilih tombol bayar
Hasil yang diharapkan	Sistem menyimpan data pembayaran kemudian menampilkan uang kembalian, dan pesan "Pembayaran telah selesai dilakukan
Hasil	Sistem menyimpan data pembayaran kemudian menampilkan uang kembalian, dan pesan "Pembayaran telah selesai dilakukan
Status	valid

14. Kasus uji mengedit barang transaksi penjualan

Tabel 6.25 Kasus Uji Mengedit Barang Transaksi Penjualan

Nomor Kasus Uji	PF_SPTF_1401
Nama Kasus Uji	Kasus Uji Mengedit Barang Transaksi Penjualan
Prosedur	1.1 Penguji masuk ke halaman tambah transaksi penjualan dimana tabel yang ada pada halaman tersebut telah terisi data barang 1.2 Penguji memilih barang yang ingin diedit pada tabel kemudian memilih tombol tambah 1.3 Penguji memilih barang pada tabel kemudian mengisi data *form* setelah itu memilih tombol edit pada *form* edit barang
Hasil yang diharapkan	Sistem memperbarui data barang sesuai masukkan aktor kemudian menampilkan halaman tambah transaksi penjualan yang telah diperbarui
Hasil	Sistem memperbarui data barang sesuai masukkan aktor kemudian menampilkan halaman tambah transaksi penjualan yang telah diperbarui
Status	valid

15. Kasus uji menghapus barang transaksi penjualan

Tabel 6.26 Kasus Uji Menghapus Transaksi Penjualan

Nomor Kasus Uji	PF_SPTF_1501
Nama Kasus Uji	Kasus Uji Menghapus Barang pada halaman Tambah Transaksi Penjualan
Prosedur	1.1 Penguji masuk kehalaman tambah transaksi penjualan dimana tabelnya telah terisi dengan data barang yang akan dibeli 1.2 Penguji memilih barang kemudian memilih tombol hapus
Hasil yang diharapkan	Sistem berhasil menghapus barang yang dipilih dan menampilkan halaman tambah transaksi penjualan yang telah diperbarui

Hasil	Sistem berhasil menghapus barang yang dipilih dan menampilkan halaman tambah transaksi penjualan yang telah diperbarui
Status	valid

16. Kasus uji menghapus transaksi penjualan

Tabel 6.27 Kasus Uji Menghapus Transaksi Penjualan

Nomor Kasus Uji	PF_SPTF_1601
Nama Kasus Uji	Kasus Uji Menghapus Transaksi Penjualan
Prosedur	1.1 Penguji masuk kehalaman transaksi penjualan yang telah tersimpan 1.2 Penguji memilih tombol hapus transaksi 1.3 Penguji memilih tombol iya
Hasil yang diharapkan	Sistem berhasil menghapus transaksi penjualan dan menampilkan halaman daftar transaksi penjualan yang telah diperbarui
Hasil	Sistem berhasil menghapus transaksi penjualan dan menampilkan halaman daftar transaksi penjualan yang telah diperbarui
Status	valid

17. Kasus uji menampilkan transaksi penjualan

Tabel 6.28 Kasus Uji Menampilkan Transaksi Penjualan

Nomor Kasus Uji	PF_SPTF_1701
Nama Kasus Uji	Kasus Uji Menampilkan Transaksi Penjualan
Prosedur	1.1 Penguji memilih menu transaksi penjualan 1.2 Penguji masuk ke halaman daftar transaksi penjualan 1.3 Penguji memilih transaksi pada tabel
Hasil yang diharapkan	Sistem menampilkan halaman transaksi penjualan
Hasil	Sistem menampilkan halaman transaksi penjualan
Status	valid

18. Kasus uji mencetak transaksi penjualan

Tabel 6.29 Kasus Uji Mencetak Transaksi Penjualan

Nomor Kasus Uji	PF_SPTF_1801
Nama Kasus Uji	Kasus Uji Mencetak Transaksi Penjualan
Prosedur	1.1 Penguji masuk kehalaman tambah transaksi penjualan yang telah terisi barang-barang pembelian 1.2 Penguji memilih tombol cetak transaksi
Hasil yang diharapkan	Sistem menampilkan *file* pdf yang berisi detail transaksi penjualan yang dipilih
Hasil	Sistem menampilkan *file* pdf yang berisi detail transaksi penjualan yang dipilih
Status	valid

19. Kasus uji menampilkan stok barang

Tabel 6.30 Kasus Uji Menampilkan Stok Barang

Nomor Kasus Uji	PF_SPTF_1901
Nama Kasus Uji	Kasus Uji Menampilkan Stok Barang
Prosedur	1.1 Penguji masuk ke halaman daftar transaksi penjualan 1.2 Penguji memilih menu stok barang
Hasil yang diharapkan	Sistem menampilkan halaman stok barang
Hasil	Sistem menampilkan halaman stok barang
Status	valid

20. Kasus uji mengedit stok barang

Tabel 6.31 Kasus Uji Mengedit Stok Barang

Nomor Kasus Uji	PF_SPTF_2001
Nama Kasus Uji	Kasus Uji Mengedit Stok Barang
Prosedur	1.1 Penguji masuk kehalaman stok barang 1.2 Penguji memilih barang yang ingin diedit kemudian memilih tombol edit 1.3 Penguji mengisi *form* edit barang kemudian memilih tombol edit

Hasil yang diharapkan	Sistem menyimpan masukkan aktor dan menampilkan halaman daftar stok barang yang telah diperbarui
Hasil	Sistem menyimpan masukkan aktor dan menampilkan halaman daftar stok barang yang telah diperbarui
Status	valid

21. Kasus uji menampilkan laporan keuangan

Tabel 6.32 Kasus Uji Menampilkan Laporan Keuangan

Nomor Kasus Uji	PF_SPTF_2101
Nama Kasus Uji	Kasus Uji Menampilkan Laporan Keuangan
Prosedur	1.1 Penguji masuk kehalaman daftar transaksi penjualan 1.2 Penguji memilih tombol laporan keuangan 1.3 Penguji mengisi dari tanggal, hingga tanggal kemudian memilih tombol tampilkan
Hasil yang diharapkan	Sistem menampilkan halaman laporan keuangan berdasarkan rentang tanggal yang dipilih aktor
Hasil	Sistem menampilkan halaman laporan keuangan berdasarkan rentang tanggal yang dipilih aktor
Status	valid

22. Kasus uji menampilkan pembayaran sales

Tabel 6.33 Kasus Uji Menampilkan Pembayaran Sales

Nomor Kasus Uji	PF_SPTF_2201
Nama Kasus Uji	Kasus Uji Menampilkan Pembayaran Sales
Prosedur	1.1 Penguji masuk ke halaman daftar transaksi penjualan 1.2 Penguji memilih menu pembayaran sales 1.3 Penguji memilih nama sales yang ingin ditampilkan pada halaman pembayaran sales

Hasil yang diharapkan	Sistem menampilkan pembayaran sales yang dipilih
Hasil	Sistem menampilkan pembayaran sales yang dipilih
Status	valid

23. Kasus uji menambah pembayaran sales

Tabel 6.34 Kasus Uji Menambah Pembayaran Sales

Nomor Kasus Uji	PF_SPTF_2301
Nama Kasus Uji	Kasus Uji Menambah Pembayaran Sales
Prosedur	1.1 Aktor masuk ke halaman pembayaran sales yang berisi data sales yang ingin ditambahkan 1.2 Aktor memilih transaksi yang ingin ditambahkan kemudian memilih tombol tambah 1.3 Aktor mengisi *form* pembayaran dan memilih tombol bayar
Hasil yang diharapkan	Sistem menampilkan halaman transaksi pembayaran sales yang telah diperbarui dan menampilkan pesan "Pembayaran telah ditambahkan"
Hasil	Sistem menampilkan halaman transaksi pembayaran sales yang telah diperbarui dan menampilkan pesan "Pembayaran telah ditambahkan"
Status	valid

24. Kasus uji menghapus pembayaran sales

Tabel 6.35 Kasus Uji Menghapus Pembayaran Sales

Nomor Kasus Uji	PF_SPTF_2401
Nama Kasus Uji	Kasus Uji Menghapus Pembayaran Sales
Prosedur	1.1.1 Penguji masuk ke halaman pemayaran sales dan penguji telah memilih nama sales yang data pembayaran belum lunasnya ingin ditampilkan 1.1.2 Penguji memilih pembayaran yang ingin dihapus

	1.1.3 Penguji memilih tombol hapus 1.1.4 Penguji memilih tombol iya pada pesan konfirmasi yang muncul
Hasil yang diharapkan	Sistem berhasil menghapus riwayat pembayaran sales yang dipilih aktor dan menampilkan halaman pembayaran sales yang telah di perbarui
Hasil	Sistem berhasil menghapus riwayat pembayaran sales yang dipilih aktor dan menampilkan halaman pembayaran sales yang telah di perbarui
Status	valid

25. Kasus uji mengedit pembayaran sales

Tabel 6.36 Kasus Uji Mengedit Pembayaran Sales

Nomor Kasus Uji	PF_SPTF_2501
Nama Kasus Uji	Kasus Uji Mengedit Pembayaran Sales
Prosedur	1.1 Aktor masuk ke halaman pembayaran sales dan telah memilih nama sales yang pembayarannya ingin diedit 1.2 Aktor memilih pembayaran yang ingin di edit pada tabel 1.3 Aktor memilih tombol edit 1.4 Aktor mengisi *form* edit pembayaran kemudian memilih tombol edit
Hasil yang diharapkan	Sistem menyimpan masukkan, menampilkan pesan “Anda berhasil mengedit pembayaran”, kemudian menampilkan halaman pembayaran sales yang telah diperbarui
Hasil	Sistem menyimpan masukkan, menampilkan pesan “Anda berhasil mengedit pembayaran”, kemudian menampilkan halaman pembayaran sales yang telah diperbarui
Status	valid

26. Kasus uji menampilkan notifikasi pembayaran sales

Tabel 6.37 Kasus Uji Menampilkan Notifikasi Pembayaran Sales

Nomor Kasus Uji	PF_SPTF_2601
Nama Kasus Uji	Kasus Uji Menampilkan Notifikasi Pembayaran Sales
Prosedur	1.1 Penguji masuk ke halaman pembayaran sales 1.2 Penguji memilih tombol notifikasi
Hasil yang diharapkan	Sistem menampilkan *form* notifikasi pembayaran sales yang berisi data transaksi belum lunas
Hasil	Sistem menampilkan *form* notifikasi pembayaran sales yang berisi data transaksi belum lunas
Status	valid

27. Kasus uji mengedit nama sales

Tabel 6.38 Kasus Uji Mengedit Nama Sales

Nomor Kasus Uji	PF_SPTF_2701
Nama Kasus Uji	Kasus Uji Mengedit Nama Sales
Prosedur	1.1 Penguji masuk ke halaman pembayaran sales dan sistem menampilkan *form* edit nama sales 1.2 Penguji mengisi nama sales dan menekan tombol edit
Hasil yang diharapkan	Sistem menyimpan masukkan sales, menampilkan halaman pembayaran sales yang telah diperbarui, dan menampilkan pesan "Nama sales berhasil diedit"
Hasil	Sistem menyimpan masukkan sales, menampilkan halaman pembayaran sales yang telah diperbarui, dan menampilkan pesan "Nama sales berhasil diedit"
Status	valid

28. Kasus uji menambah nama sales

28.1 Kasus uji menambah nama sales berhasil

Tabel 6.39 Kasus Uji Menambah Nama Sales

Nomor Kasus Uji	PF_SPTF_2801
Nama Kasus Uji	Kasus Uji Menambah Nama Sales
Prosedur	1. Penguji masuk ke halaman pembayaran sales dan sistem menampilkan *form* tambah nama sales 1.1 Penguji mengisi nama sales dan menekan tombol tambah
Hasil yang diharapkan	Sistem menyimpan masukkan, menampilkan halaman pembayaran sales yang telah diperbarui dan menampilkan pesan "Nama sales berhasil ditambahkan"
Hasil	Sistem menyimpan masukkan, menampilkan halaman pembayaran sales yang telah diperbarui dan menampilkan pesan "Nama sales berhasil ditambahkan"
Status	

28.2 Kasus uji menambah nama sales dengan nama yang telah terdaftar pada sistem.

Tabel 6.40 Kasus Uji Menambah Nama Sales

Nomor Kasus Uji	PF_SPTF_2802
Nama Kasus Uji	Kasus Uji Menambah Nama Sales
Prosedur	1. Penguji masuk ke halaman pembayaran sales dan sistem menampilkan *form* tambah nama sales 1.2 Penguji mengisi nama sales dengan nama yang telah terdaftar sebelumnya kemudian menekan tombol tambah
Hasil yang diharapkan	Sistem menampilkan pesan "Nama sales telah digunakan sebelumnya
Hasil	Sistem menampilkan pesan "Nama sales telah digunakan sebelumnya
Status	valid

BAB 7 PENUTUP

Pada bagian ini menjelaskan mengenai hasil dari penelitian yang telah dilakukan oleh penulis dan saran untuk pengembangan lebih lanjut dari Sistem Pengelolaan Transaksi Perhiasan Perak.

7.1 Kesimpulan

Hasil analisis yang dilakukan dalam membangun sistem pengelolaan transaksi perhiasan perak adalah membuat sistem yang memiliki *fitur* utama yaitu untuk mengelola kegiatan transaksi, informasi stok barang, laporan keuangan dan pembayaran sales dengan 28 kebutuhan fungsional sistem. Aktor yang terdapat pada sistem ini ada 3 yaitu pengguna, pemilik dan pegawai. Hasil analisis ini didapatkan dari studi literatur, observasi, wawancara dan validasi kesesuaian kebutuhan yang dibuat dengan keinginan *client.*

Hasil perancangan yang dilakukan menggambarkan pemodelan dari *fitur-fitur* hasil dari analisis. Dalam tahap perancangan dibuat arsitektur sistem, *sequence diagram, class diagram*, perancangan *database*, dan perancangan *system interface.*

Hasil implementasi yang dilakukan adalah menghasilkan sistem pengelolaan transaksi perhiasan perak yang memiliki memiliki *fitur* untuk mengelola kegiatan transaksi, stok barang, laporan keuangan, dan riwayat pembayaran sales sesuai dengan tahapan analisis yang telah dilakukan sebelumnya. Sistem ini dibangun dengan menggunakan arsitektural *Mode-View-Controller* (MVC) dengan menggunakan bahasa java.

Hasil pengujian yang dilakukan dengan menggunakan metode *whitebox testing* berupa pengujian unit. Hasil dari pengujian unit pada ketiga algoritme yang diuji adalah termasuk dalam *range* 1-10. Menurut (Guru99, 2018), hasil pengujian unit dalam *range* 1-10 berarti sistem yang dibangun memiliki kode struktur yang baik dan *high testability.* Pengujian *blackbox* yang dilakukan berupa validasi menghasilkan validitas sebesar 100%. Maka dapat disimpulkan aplikasi yang telah selesai dikembangkan bekerja dengan baik.

7.2 Saran

Saran untuk pengembangan lebih lanjut adalah:

1. Menambahkan kebutuhan fungsional dimana sales pun dapat mengakses sistem untuk mengetahui riwayat pembayaran sales
2. Menambahkan kebutuhan fungsional dimana pengguna dapat mengecek sendiri harga barang yang dijual di Toko Beben
3. Menambahkan kebutuhan fungsional dan non fungsional lain yang belum didefinisikan

DAFTAR PUSTAKA

Arianto, T., 2009. Implementasi Wireless Local Area Network dalam RT/RW Net. *Jurnal Teknologi Informasi Dinamik,* pp. 152-157.

Fandora, R., Bambang, E. P. & Sukadi, 2013. Sistem Informasi Penjualan, Pemesanan dan Pembelian pada Toko Cahaya Murni Silver Pacitan. *Indonesian Journal on Networking and Security.*

Gue´he´neuc, Y.-G. & Antoniol, G., 2008. DeMIMA: A Multilayered Approach. *IEEE Transactions On Software Engineering,* p. 667.

Guru99, 2018. *Learn Mccabe's Cyclomatic Complexity with Example.* [Online] Tersedia di: <https://www.guru99.com/cyclomatic-complexity.html> [Di akses 23 Maret 2018]

Jambak, M. I., 2016. *Pengujian Perangkat Lunak.* [Online] Tersedia di: <https://mti.binus.ac.id/2016/04/08/pengujian-perangkat-lunak/>

Kumar, N., Zadgaonkar, A. & Shukla, A., 2013. Evolving a New Software Development Life Cycle Model SDLC-2013 with Client Satisfaction. *International Journal of Soft Computing and Engineering (IJSCE),* p. 216.

Larasati, S. S. A., 2018. *Pembangunan Sistem Ujian Harian Siswa Berbasis Web dengan Mengacu pada Standar Kualitas ISO 25010.* Malang: Fakultas Ilmu Komputer. Universitas Brawijaya..

Masriadi, 2017. Aplikasi Pengelolaan Transaksi Penjualan Perhiasan pada Toko Emas Pasamaan Indah Kabupaten Pasamaan Barat. *UPI YPTK Jurnal KomTekInfo,* pp. 40-51.

Pressman, R. S., 2001. *Software Engineering.* 5th ed. New York: McGraw-Hill Companies, Inc.

Pressman, R. S., 2010. *Software Engineering.* 7th ed. New York: McGraw-Hill Companies, Inc.

Riyandwyana, A. & Mukhlason, E. S. A., 2012. Pengembangan Sistem Rekomendasi Peminjaman Buku Berbasis Web Menggunakan Metode Self Organizing Map Clustering Pada Badan Perpustakaan Dan Kearsipan (BAPERSIP) Provinsi Jawa Timur. *Jurnal Teknik ITS,* pp. A-376.

Sarker, H. I. & K., A., 2014. MVC Architecture Driven Design and Implementation of Java Framework for Developing Desktop Application. *International Journal of Hybrid Information Technology,* pp. 317-322.

Sharma, K. S. D. & Singh, A., 2016. Software Development Life Cycle. *International Journal of Modern Engineering and Research Technology,* p. 5.

Sommerville, I., 2011. *Software Engineering.* 9th ed. USA: Pearson Education, Inc.