Aplikasi Permintaan Barang Berbasis Java dan Mysql di PT. Kino Indonesia, TBK Sukabumi

Wawang Adi Darma, M. Sofwan Romli, Nunung Hasanah
Jurnal@cbi.ac.id

ABSTRACT

A Cosmetic Industry "PT. Kino Indonesia, Tbk "located on Jl. Raya Babakan Km. 12 Cikembar Sukabumi Regency is a company engaged in the Cosmetic Industry.
Product data processing system for factory purposes at PT. KINO INDONESIA, TBK has used Excel Application systems as its data processing system.

So the company PT. Kino Indonesia, Tbk requires a form of computer application that can be run quickly, efficiently and precisely in its use such as a more practical Goods Request Application. Applications that are specifically designed to be able to melt requests for goods by the User and can be accessed by the Admin without having to re-login to update the data to be pulled, there is no need to re-record the different reports to make a stock inventory report, helping to avoid the difference in the final report. The display form that will be used has several main menus such as Data menu, Transaction menu and Report menu. Java and MySql programs are used by authors to build this system

Keywords: Request, Java, MySql

I.PENDAHULUAN

1.1 Latar Belakang Masalah

Teknologi Informasi merupakan salah satu teknologi yang sedang berkembang pesat pada saat ini, dengan kemajuan teknologi informasi pengaksesan terhadap data atau Informasi yang tersedia dapat berlangsung dengan cepat, efisien dan akurat.

Perkembangan dalam bidang komputer saat ini telah membuka peluang kepeda para pakar dan para pengambil keputusan baik yang bergerak di bidang Pemerintahan dan Swasta. Untuk mempermudah dan menyelesaikan pekerjaannya dengan komputer. Sebelum datang era komputerisasi ini seluruh unit kerja menyelesaikan pekerjaannya secara manual, berbeda dengan saat ini unit kerja dapat menggunakan komputer dalam mengerjakan berbagai tugasnya dengan cepat dan tepat. Hal ini di karenakan di dalam komputer tersebut dapat di pasang bermacam-macam aplikasi yang dapat di gunakan sehingga unit kerja mendapatkan kemudahan dalam menyelesaikan pekerjaannya.

Sebuah Industri Kosmetik “PT. Kino Indonesia, Tbk” yang berlokasi di Jl. Raya Babakan Km. 12 Cikembar Kabupaten Sukabumi adalah sebuah perusahaan yang bergerak di bidang Industri Kosmetik. Dalam strukturnya perusahaan ini memiliki beberapa Departemen di antaranya Departemen Produksi, *Quality Control, Personalia, GA (General Affair)*, *Procuremen*t, *Finance, SAP (System Analysis and Program Development), PPIC (Production Planning Inventory Control), Warehouse* dan *Logistic Distribusi*. Yang akan saya angkat kali ini berhubungan dengan Departemen GA mengenai sistem Permintaan Barang pada Gudang barang keperluan Plant di PT. Kino Indonesia, Tbk.

Sistem pengolahan data barang keperluan pabrik di PT. KINO INDONESIA, TBK sudah menggunakan Ms. Excel sebagai sistem pengolahan datanya. Penggunaan aplikasi standar ini masih terkendala kurang baiknya dalam segi penginputan dan penyimpanan data karena jumlah data tabel yang diperlukan cukup banyak. Maka pihak perusahaan PT. Kino Indonesia Tbk membutuhkan suatu bentuk Aplikasi komputer yang dapat di jalankan secara cepat, *efisien* dan tepat dalam pengguunaanya seperti Aplikasi Permintaan Barang yang lebih praktis untuk dapat memudahkan proses input dan simpan data untuk item data yang banyak.

1.2 Identifikasi Masalah

Adapun masalah-masalah yang di temukan dari pengamatan penyusun tentang

sistem permintaan barang di PT. Kino Indonesia, Tbk yaitu :

1. Penginputan data yang sering mengalami kesalahan karena belum terkomputerisasi.
2. Penyimpanan data yang kurang aman, karena berbentuk arsip manual
3. Pelaporan yang kurang baik dan tidak tepat waktu

1.3 Batasan Masalah

Berdasarkan identifikasi masalah di atas penulis membatasi permasalahan yang akan di bahas di tantaranya :

1. Membatasi pembuatan Aplikasi dengan hanya mencakup permintaan Barang.
2. Pembuatan Aplikasi hanya bertujuan untuk pembutan laporan harian.

1.4 Rumusan Masalah

Beberapa rumusan masalah yang akan di bahas dalam penelitian, yaitu :

1. Bagaimana membuat suatu Aplikasi yang dapat melakukan pengolahan Data Stok Barang melingkupi Data Barang, Transaksi Permintaan, *Report* Permintan dan *Report* Stok Barang .
2. Bagaiman merancangan basisdata sebagai penyimpanan data yang aman
3. Bagaimana membuat suatu laporan yang akurat sesuai data yang ada di masukan.

1.5 Tujuan Penelitian

Adapun tujuan penelitian ini adalah sebagai berikut :

1. Untuk mengetahui sistem yang sedang berjalan.
2. Untuk mencari solusi dalam perancangan sistem Aplikasi.
3. Membantu proses pemintaan barang menjadi lebih cepat dan akurat.

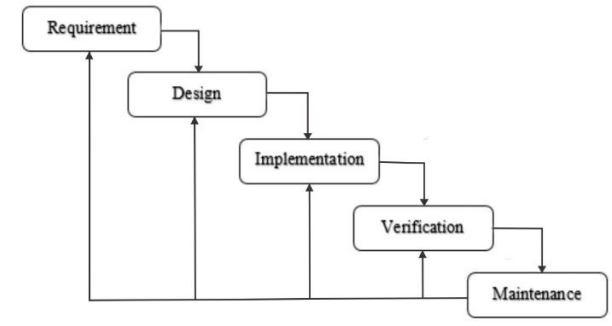
1.6 Teknik Pengumpulan Data

Untuk mendapatkan data yang di perlukan dalam penyusunan laporan ini penulis menggunakan metode deskriptif yang akan menggambarkan permasalahan yang di bahas dengan cara mengumpulkan data, menyusun, mengklasifikasi, menganalisa dan menginterprestasikan. Untuk memperoleh data yang di perlukan dalam laporan ini, pemulis menggunakan teknik pengumpulan data sebagai berikut :

1. Observasi
   Penulis melakukan pengamatan secara langsung terhadap proses pelaksanaan di tempat yang sebenarnya.
2. Wawancara
   Penulis mengumpulkan data dengan cara wawancara langsung dan tanya jawab dengan staf dan PIC pada masing-masing departemen di perusahaan tersebut mengenai sistem transaksi permintaan barang.
3. Studi kepustakaan
   Penulis mengumpulkan data sekunder dengan membaca serta menelaah buku-buku ilmiah, laporan-laporan lainyang ada hubungannya dengan masalah yang akan di bahas.

1.7 Desain Pengembangan Sistem

Desain sistem yang diguankan adalah desain sistem waterfall, dengan tahapan-tahapan sebagai berikut:



Ganbar 1.1. Waterfall
(Pressman, Roger S. 2012)

Tahapan tahapan dari metode *waterfall* adalah sebagai berikut :

1. *Requirement Analisis*
   Tahap ini penulis mencoba memahami perangkat lunak yang diharapkan oleh pengguna dan batasan perangkat lunak tersebut. Informasi ini diperoleh penulis melalui wawancara, diskusi

atau survei langsung. Informasi dianalisis untuk mendapatkan data yang dibutuhkan oleh pengguna.

2. *System Design*
   Dalam tahap ini penulis membuat Spesifikasi kebutuhan dari tahap sebelumnya akan dipelajari dalam fase ini dan desain sistem disiapkan. Desain Sistem membantu dalam menentukan perangkat keras*(hardware)* dan sistem persyaratan dan juga membantu dalam mendefinisikan arsitektur sistem secara keseluruhan.
3. *Implementation*
   Pada tahap ini, penulis mengembangkan program kecil yang disebut *unit* dan mengimplementasikan seluruh rancangan yang dibuat dan terintegrasi dalam tahap selanjutnya. Setiap *unit* dikembangkan dan diuji untuk fungsionalitas yang disebut sebagai *unit testing*.
4. *Integration & Testing*
   Seluruh *unit* yang dikembangkan dalam tahap implementasi diintegrasikan ke dalam sistem setelah pengujian yang dilakukan masing-masing *unit*. Setelah integrasi seluruh sistem diuji untuk mengecek setiap kegagalan maupun kesalahan.
5. *Operation & Maintenance*
   Tahap akhir dalam model *waterfall*. Perangkat lunak yang sudah jadi, dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi *unit* sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

1.8. Tempat Penelitian

Penulis melakukan penelitian di PT. Kino Indonesia, Tbk yang beralamat di Jl. Raya Babakan Km.12, Desa Kertaraharja, Cikembar, Sukabumi, Jawa Barat 43157.

II. LANDASAN TEORI

2.1 Perancangan Sistem Beroerientasi Objek dengan UML

UML (*Unified Modeling Language*) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax.* Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan.

Berikut ini adalah penjelasan dari beberapa diagram UML yang sering digunakan :

1. *Diagram Use Case*

Use case adalah rangkaian atau uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah aktor. *use case* digunakan untuk membentuk tingkah laku benda dalam sebuah model serta direalisasikan oleh sebuah kolaborasi.

Diagram use case menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Hal yang ditekankan pada diagram ini adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* menyatakan sebuah aktivitas atas pekerjaan tertentu, misalnya *login* ke sistem, meng-*create* sebuah daftar belanja, dan lain sebagainya. adal sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

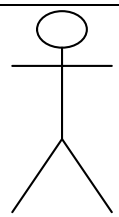
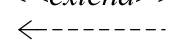
Diagram use case dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan *klien*, dan merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum, diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali

use case yang meng-*include* dieksekusi secara normal.

Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain. Oleh karena itu, duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain. (Hamim Tohari, 2014 : 47-48)

Berikut ini adalah simbol-simbol yang ada pada *diagram use case* :

Tabel 2.1 Simbol *Diagram Use Case*

No.	Simbol	Deskripsi
1		Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. Biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case.*
2		Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang. Biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
3	*Association*	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4		Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri, walaupun tanpa *use case*

		tambahan itu. Mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan. Arah panah menunjukpad *use case* yang dituju.
5	<<include>>	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini, untuk menjalankan fungsinya atau sebagai syarat yang cukup dijalankan *use case* ini. Ada dua sudut pandang yang cukup besar mengenai *include* di *use case*.

Sumber: (Hamim Tohari, 2014 : 49)

2. *Class Diagram*

Kelas (*Class*) adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan perancangan berorientasi objek. Kelas menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metode/fungsi). (Hamim Tohari, 2014 : 83-84)

Berikut ini adalah simbol-simbol yang ada pada *Class Diagram* :

Tabel 2.2 Simbol *Class Diagram*

No.	Simbol	Deskripsi
1	Package	*Package* merupakan sebuah bungkusan dari satu atau lebih kelas.
2	Nama kelas +atribut1 +atribut2 +operation1 ()	Kelas pada struktur sistem.
3	*Interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.

4	Asosiasi 1 1..*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
5	Asosiasi berarah	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
6	Generalisasi	Relasi antar kelas dengan makna generalisasi-spesialisai (umum khusus).
7	*Defendency*	Relasi antar kelas dengan makna kebergantungan antar kelas.
8	*Agregasi*	Relasi antar kelas dengan makna semua-bagian (*whole-part*).

Sumber: (Hamim Tohari, 2014 : 84)

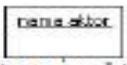
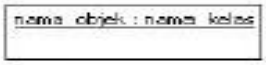
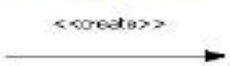
dikirim antara objek, juga interaksi antar objek yang terjadi pada titik tertentu dalam eksekusi sistem. dalam UML, objek pada *diagram sequence* digambarkan dengan segi empat, yeng berisi nama dari objek yang digarisbawahi. Terdapat tiga (3) cara untuk menamai objek yaitu, nama objek, nama objek dan *class* serta nama *class*.

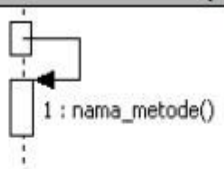
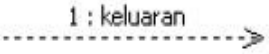
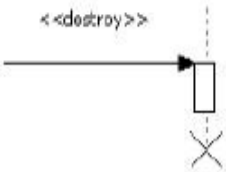
3. *Sequence Diagram*

Sequence Diagram menggambarkan interaksi antara sejumlah objek dalam urutan waktu. Kegunaannya untuk menunjukkan rangkaian pesan yang

Pada *diagram sequence*, setiap objek hanya memiliki garis yang digambarkan garis putus-putus ke bawah. Pesan antar objek digambarkan dengan anak panah dari objek yang mengirimkan pesan ke objek yang menerima pesan. (Hamim Tohari, 2014 : 101)

Table 2.3 Simbol-simbol *Sequence Diagram*

Simbol	Deskripsi
Aktor 	orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan
atau  tanpa waktu aktif	orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor
Garis hidup / *lifeline*	menyatakan kehidupan suatu objek
Objek 	menyatakan objek yang berinteraksi pesan
Waktu aktif 	menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe create 	menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe call 	menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.

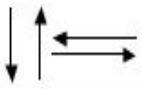
Simbol	Deskripsi
	 arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
Pesan tipe send 	menyatakan bahwa suatu objek mengirimkan data/masukan/ informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
Pesan tipe return 	menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe destroy 	menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

Sumber : http://www.sistem-informasi.xyz/2016/08/pengertian-sequence-diagram.html

4. *Activity Diagram*

Activity Diagram memodelkan *workflow* proses bisnis dan urutan aktivitas dalam sebuah proses. Digaram ini sangat mirip dengan *flowchart* karena memodelkan *workflow* dari satu aktivitas ke aktivitas lainnya atau dari aktivitas ke status. Membuat *activity diagram* pada awal pemodelan proses cukup menguntungkan untuk membantu memahami keseluruhan proses. *Activity diagram* juga bermanfaat untuk menggambarkan *paralel behaviour* atau menggambarkan interaksi antara beberap *use case*. (Hamim Tohari, 2014 : 114)

Tabel 2.3 Simbol Diagram Aktivitas/ *Activity Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Activity Final Node*	Bagaimana objek dibentuk dan diakhiri
5		*Decision*	Diguanakan untuk menggambarkan suatu keputusan / tindakan yang harus diambil pada kondisi tertentu
6		*Line Connector*	Digunakan untuk menghubungkan satu simbol dengan simbol lainnya

Sumber: Fahrul ahaddin:2015

Java sebagai salah satu bahasa pemrograman baru menjanjikan banyak kemudahan bagi programer. Java dikembangkan dengan model yang mirip dengan bahasa C++ dan Smalltalk, namun dirancang agar lebih mudah dipakai dan - platform independent, yaitu dapat dijalankan di berbagai jenis sistem operasi dan arsitektur komputer. Java Development Kit (JDK) merupakan perlengkapan tempur yang mendasar dalam pengembangan aplikasi dengan Java. (Eko:2017:2)

Class ditulis dalam kode sumber yang disimpan sebagai file teks biasa berekstension .java.

Dalam file .java, dapat dideklarasikan :

1. Package
2. Import
3. satu atau lebih class

1. Deklarasi Package

Deklarasi package digunakan untuk mengelompokkan class-class. Sebuah package dapat mempunyai satu atau lebih sub-package ini sehingga dapat menyusun sebuah hirarki. Tata cara penulisan :

package <namapackage>.<namasubpackage>.<namasubsubpackage>;

Contoh deklarasi package adalah :

package java.awt;

atau

package org.apache.tomcat.core;

atau

package net.developerforce.relieve.dataaccess;

Deklarasi package bersifat opsional, tidak harus ada. Tanpa deklarasi package maka sebuah class dikelompokkan ke dalam default package.

2. Deklarasi Import

Deklarasi import, bersifat opsional, digunakan untuk menunjukkan package atau class yang digunakan dalam sebuah program Java.

Contohnya adalah :

import java.io.RandomAccessFile;

import java.net.*;

Deklarasi import tidak akan menjadikan class atau package yang Anda cantumkan digabungkan dengan program Anda saat kompilasi atau saat diluncurkan. Deklarasi import digunakan untuk menunjukkan java atau javac dalam menemukan class yang Anda gunakan.

3. Deklarasi Class

Deklarasi class merupakan kandungan utama sebuah file berekstension .java. Deklarasi class terutama memuat :

a. nama class, bisa dilengkapi dengan kendali akses, deklarasi extends maupun deklarasi implements. Pola yang lumrah adalah :
public class ClassName
extends SuperClassName
implements Interface1Name, Interface2Name, Interface3Name
constructor, yang dipanggil pada saat dibuat instans dari class.
deklarasi variabel-variabel

III. Gambaran Umum Objek Penelitian dan Perancangan Sistem

3.1 Gambaran Umum Objek Penelitian

3.1.1 Sejarah Singkat Perusahaan / Instansi

Kino Corporation berawal dari sebuah perusahaan distribusi bernama PT Duta Lestari Sentratama di tahun 1991. Saat ini, PT Duta Lestari Sentratama telah berkembang secara signifikan dengan 18 titik distribusi yang menangani distribusi di seluruh Indonesia.

Mengembangkan diri ke dalam industri manufaktur, PT Kino Sentra Industrindo didirikan sebagai sebuah perusahaan produksi makanan ringan pada tahun 1997 dengan pabrik yang berlokasi di Semarang, Jawa Tengah. PT Kino Sentra Industrindo terus aktif menciptakan berbagai macam produk makanan ringan seperti permen, *snacks*, dan cokelat.

Langkah besar lainnya dilakukan pada tahun 1999, PT Kinocare Era Kosmetindo, produsen dengan aneka produk perawatan tubuh untuk semua gender dan usia, mulai didirikan. Tahun 2003 PT Kinocare Era Kosmetindo melebarkan bisnisnya ke produk perawatan rumah di bawah naungan merek Sleek. Kemudian Kino juga melangkah lebih lanjut ke dalam industri minuman dengan memproduksi minuman berenergi Panther.

Mengembangkan bisnis hingga merambah kawasan Asia, di tahun 2004 Kino membuka kantor cabang pertamanya di Malaysia dengan Kinocare (M) Sdn.
Bhd dan Filipina dengan Kino Consumer Philippines. Kino juga menjalin kerjasama distribusi dengan beberapa jaringan distributor besar di Singapura, Brunei, Vietnam, Myanmar, Jepang, Australia, Timur Tengah, dan Afrika. Di tahun 2012, Kino Consumer Vietnam berdiri yang menambah jaringan distribusi di Asia Tenggara. Kino Corporation terus bergerak dengan mendirikan PT Prime Restaurant Indonesia pada tahun 2005 dengan dibukanya *Chicken Story*, restoran citarasa Indonesia. Selain itu, dibuka juga gerai *bubble tea* asal Taiwan di tahun 2012, *Share Tea*. Seiring dengan berjalannya waktu, Kino *Corporation* memiliki lisensi dari WenKen Group Singapura untuk memproduksi, mendistribusikan, dan memasarkan secara resmi merek Cap Kaki Tiga di Indonesia sejak tahun 2011.

Kino Corporation juga telah memiliki hubungan kerjasama dengan Morinaga & Company Limited dari Jepang untuk memperluas bisnis nya di Indonesia. PT Morinaga Kino Indonesia, yang secara resmi dibentuk di tahun 2013. Setelah lebih dari dua dekade, produk-produk Kino Corporation Group kini tersedia di distribustor besar, *hypermarket, supermarket, minimarkets*, ribuan toko kosmetik, serta jutaan pasar tradisional yang ada di seluruh kepulauan Indonesia. Kualitas produk Kino telah memenuhi standar kualifikasi internasional, terbukti dengan meningkatnya permintaan pasar dari seluruh dunia. Namun Kino menyadari bahwa kepuasan tidak hanya sampai disitu, kesuksesan hari ini harus menjadi motivasi untuk semakin berprestasi di masa depan. "*The Innovator*" berarti harus terus kreatif dan inovatif untuk menjadi

pemimpin dalam industri kebutuhan konsumen.

3.1.2 Visi, Misi & Budaya

1. Visi

Menjadi perusahaan ternama di Indonesia yang berlandaskan ide & inovasi dan terus bergerak untuk menjadi

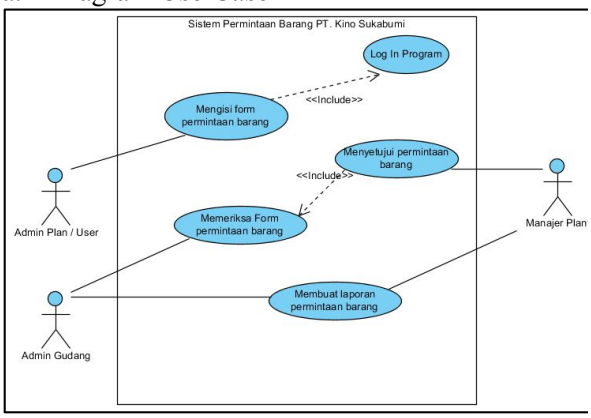
Analisa sistem terhadap sistem baru yang diusulkan yaitu penulis menganalisa kebutuhan user terhadap sistem yang akan dibangun nanti. Adapun kebutuhan user diantaranya yaitu sistem mampu menginput data permintaan barang, menyimpan data permintaan barang dan membuat laporan permintaan barang tersebut.

1. Gambaran umum Sistem yang dusulkan

Secara umum sistem yang diusulkan dapat dijelaskan melalui diagram pemodelan dengan menggunakan UML. Adapun diagram-diagram yang akan dibuat adalah Diagram *Use Case*, Diagram Aktivitas, Diagam Sequence dan Diagram ER (*Entity Relationship*).

2. Rancangan prosedur

a. Diagram *Use Case*



Gambar 3.5 Diagram *Use Case* yang diusulkan

b. Realisasi Use Case

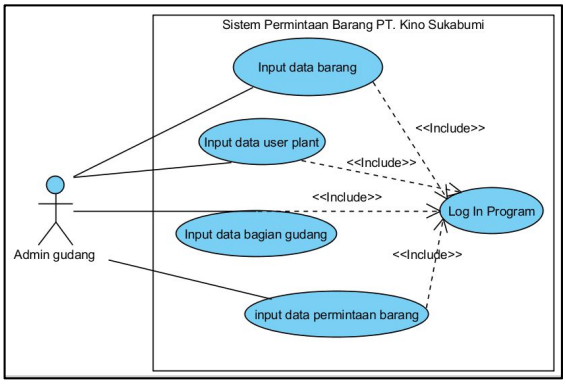
1) Realisasi Use Case permintaan barang

perusahaan yang mendunia tanpa meninggalkan nilai-nilai lokal.

2. Misi

Memperluas pasar melalui pengembangan produk yang didorong oleh semangat untuk berinovasi.

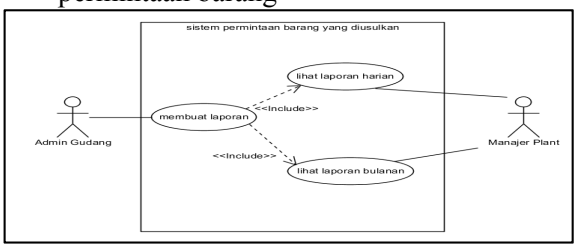
3.2 Perancangan Sistem



Gambar 3.6 Realisasi *Use Case* permintaan barang

Gambar 3.6 diatas memperlihatkan *Use Case* permintaan barang. Proses permintaan barang dimulai dari admin gudang yang menginput data permintaan yang berasal dari user plant. Sebelumnya admin gudang juga menginput data barang, data user plant dan data bagian gudang.

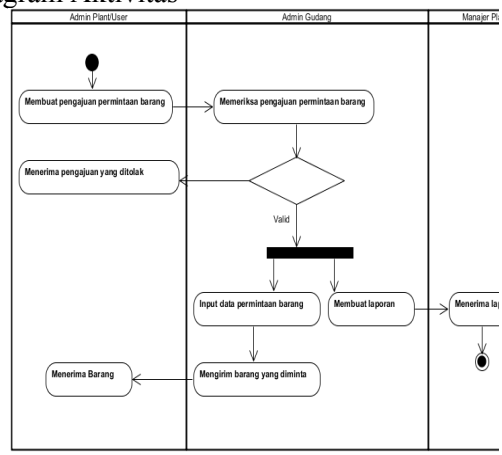
a. Realisasi *Use Case* membuat laporan permintaan barang



Gambar 3.7 Realisasi *Use case* membuat laporan permintaan barang

Gambar 3.7 memperlihatkan realisasi *Use Case* untuk membuat laporan permintaan barang. Admin gudang membuat laporan periode bulanan dan periode harian kemudian diserahkan ke manajer plant.

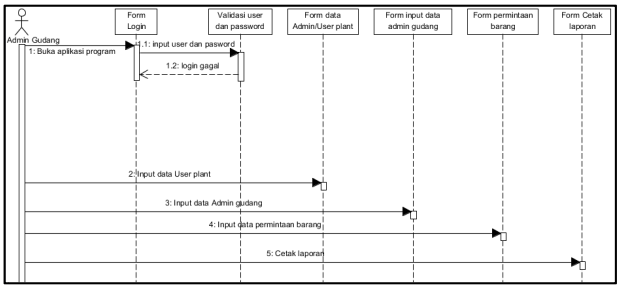
c. Diagram Aktivitas



Gambar 3.8 Diagram aktivitas sistem permintaan barang

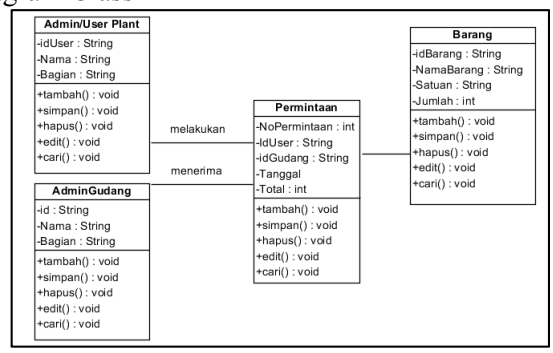
Gambar 3.8 diatas memperlihatkan alur aktivitas dari sistem permintaan barang yang diusulkan oleh penulis. Proses permintaan barang dimulai dari admin atau user plant mengisi formulir permintaan barang atau membuat surat permintaan barang. Selanjutnya diserahkan ke bagian gudang dan diterima oleh admin gudang. Admin gudang kemudian memeriksa permintaan tersebut, jika valid maka selanjutnya akan diinputkan melalui aplikasi permintaan barang. Sebelumnya admin dang juga menginput data barang, data user plant dan data bagian gudang. Selanjutnya admin gudang membuta laporan dan diserahkan ke manajer plant.

d. Diagram Sekuen sistem permintaan barang yang diusulkan

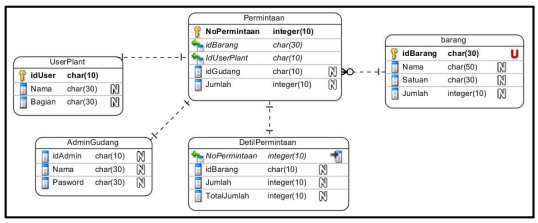


Gambar 3.9 Diagram Sequence

e. Diagram Class



Gambar 3.10 Diagram ClassDiagram ER



Gambar 3.11 Diagram ER

IV. Implementasi Sistem

Implementasi sistem ini merupakan penjelasan mengenai hasil sistem dan aplikasi yang dibuat penulis. Implementasi sistem terdiri dari dari implementasi basis data yaitu implementasi dari rancangan dan pemodelan sistem khususnya diagram class dan ERD yang penulis rancang di Bab III.

Selain itu juga implementasi sistem tediri dari implementasi program yaitu implementasi aplikasi program yang penulis bangun menggunakan Java dan MySQL.

4.1 Implementasi Basis Data

1. Nama Basis data : PermintaanBarang
   Nama Tabel : Barang
   Primary Key : idBarang
   Struktur Tabel :

#	Nama	Jenis	Penyortiran
1	idBarang	varchar(10)	latin1_swedish_ci
2	Nama	varchar(30)	latin1_swedish_ci
3	Satuan	varchar(20)	latin1_swedish_ci
4	Jenis	varchar(20)	latin1_swedish_ci

Gambar 4.1 Struktur tabel Barang

2. Nama Basis Data : Permintaanbarang
   Nama Tabel : Plant
   Primary Key : idPlant
   Struktur Tabel :

#	Nama	Jenis	Penyortiran	Atribut
1	idPlant	varchar(10)	latin1_swedish_ci	
2	Nama	varchar(30)	latin1_swedish_ci	
3	Bagian	varchar(50)	latin1_swedish_ci	
4	NoHP	varchar(12)	latin1_swedish_ci	

Gambar 4.2 Struktur tabel Plant

3. Nama Basis Data : PermintaanBarang
   Nama Tabel : Gudang
   Primary Key : idGudang
   Struktur Tabel :

#	Nama	Jenis	Penyortiran	Atribut
1	idGudang	varchar(10)	latin1_swedish_ci	
2	Nama	varchar(30)	latin1_swedish_ci	
3	Bagian	varchar(20)	latin1_swedish_ci	
4	NoHP	varchar(12)	latin1_swedish_ci	

Gambar 4.3 Struktur tabel Gudang

4. Nama Basis Data : PermintaanBarang
   Nama Tabel : Permintaan
   Primary Key : NoPermintaan
   Struktur Tabel :

#	Nama	Jenis	Penyortiran	Atribut
1	NoPermintaan	int(11)		
2	idPlant	varchar(10)	latin1_swedish_ci	
3	idGudang	varchar(10)	latin1_swedish_ci	
4	Tanggal	varchar(15)	latin1_swedish_ci	
5	Total	int(11)		

Gambar 4.3 Struktur tabel Permintaan

5. Nama Basis Data : PermintaanBarang
   Nama Tabel : DetilPermintaan
   Primary Key : -
   Struktur Tabel :

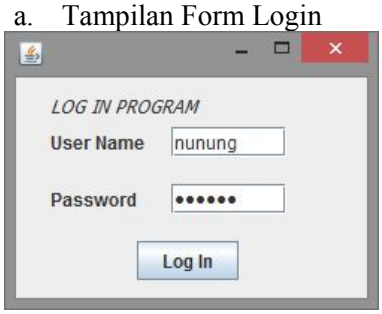
Jelajahi | Struktur | SQL | Ca

#	Nama	Jenis	Penyortiran	Atribut
1	NoPermintaan	int(11)		
2	IdBarang	int(11)		
3	Jumlah	int(11)		

Gambar 4.4 Struktur tabel DetilPermintaan

4.2 Implementasi Program

1. Form Login
   a. Tampilan Form Login



Gambar 4.4 Tampilan Form Login Program

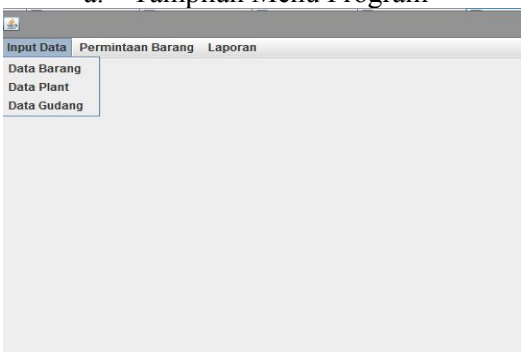
   b. Koding Program

```
private void
jButton1ActionPerformed(java.a
wt.event.ActionEvent evt) {
if ((txtNama.ge

                    new
FormLogin().setVisible(true);
                }
            });
        }

        // Variables declaration - do not
modify
        private javax.swing.JButton
jButton1;
        private javax.swing.JLabel
jLabel1;
        private javax.swing.JLabel
jLabel2;
```

```
        } catch
(javax.swing.UnsupportedLookA
ndFeelException ex) {

java.util.logging.Logger.getLogge
r(FormLogin.class.getName()).lo
g(java.util.logging.Level.SEVER
E, null, ex);
        }

java.awt.EventQueue.invokeLater
(new Runnable() {
            public void run() {
```

```
    private javax.swing.JLabel
jLabel3;
    private javax.swing.JTextField
txtNama;
    private
javax.swing.JPasswordField
txtPwd;
    // End of variables declaration
```

2. Form Menu Program utama
   a. Tampilan Menu Program



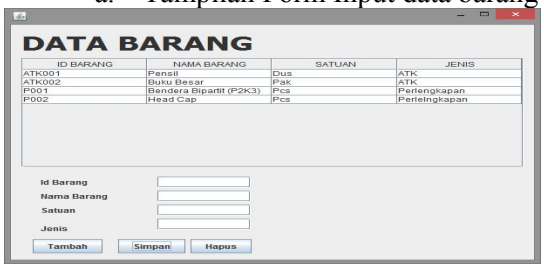
Gambar 4.5 Tampilan Form Menu

Kode Program

```
private void
jMenuItem1ActionPerformed(jav
a.awt.event.ActionEvent evt) {
      // TODO add your handling
code here:
  FormBarang fBarang=new
FormBarang();
      fBarang.setVisible(true);
   }
      JFrame form = new
JFrame();
      form.setSize(300, 300);
      new Tengah(this);
   }
   private void
jMenuItem3ActionPerformed(jav
a.awt.event.ActionEvent evt) {
      // TODO add your handling
code here:
  formGudang fGudang=new
formGudang();
      fGudang.setVisible(true);
   }
   private void
jMenuItem2ActionPerformed(jav
a.awt.event.ActionEvent evt) {
  FormPlant fPlant=new
FormPlant();
      fPlant.setVisible(true);
   }

   private void
jMenuItem4ActionPerformed(jav
a.awt.event.ActionEvent evt) {
      // TODO add your handling
         for
(javax.swing.UIManager.LookAn
dFeelInfo info :
javax.swing.UIManager.getInstall
edLookAndFeels()) {
            if
("Nimbus".equals(info.getName()
)) {

javax.swing.UIManager.setLook
AndFeel(info.getClassName());
               break;
            }
         }
      } catch
(ClassNotFoundException ex) {

java.util.logging.Logger.getLogge
r(FormMenu.class.getName()).log
(java.util.logging.Level.SEVERE,
null, ex);
      } catch
(InstantiationException ex) {
```

3. Form Input Data Barang

   a. Tampilan Form Input data barang



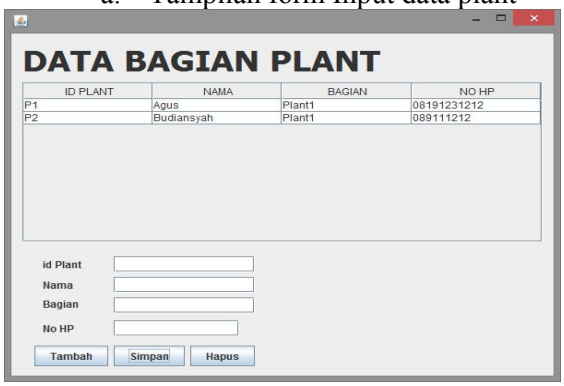
Gambar 4.6 Tampilan Form Input data barang

   b. Kode Program

```
package permintaanbarang;

import java.sql.ResultSet;
```

```
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import
javax.swing.table.DefaultTableM
odel;

public class FormBarang extends
javax.swing.JFrame {
public boolean databaru;
   public FormBarang() {
      initComponents();
      tampilkan_data();
      reset();
   }
   private void reset(){
   // mengosongkan textbox
   jTextField1.setText("");
   jTextField2.setText("");
   jTextField3.setText("");
   jTextField4.setText("");
   }
            res.getString(4),
            });
        }
          TabelData.setModel(dtm);
       }catch(SQLException e){
```

4. Form Input Data Plant
   a. Tampilan form Input data plant



Gambar 4.7 Tampilan Form input Data Plant

Kode program

```
package permintaanbarang;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import
javax.swing.table.DefaultTableM
odel;
   public FormPlant() {
      initComponents();
      tampilkan_data();
      reset();
   }
```

5. Form input Data gudang
   a. Tampilan form input data gudang

Gambar 4.7 Tampilan Form input data gudang

   b. Kode program

```
   private void tampilkan_data(){
      DefaultTableModel dtm =
new DefaultTableModel();
      dtm.addColumn("ID
GUDANG");
      dtm.addColumn("NAMA");

dtm.addColumn("BAGIAN");
      dtm.addColumn("NO HP");
      try{
   koneksi konek=new koneksi();
   konek.bukaDb();
       ResultSet res;

res=konek.stat.executeQuery("sel
ect * from tbGudang");
        while(res.next()){
         dtm.addRow(new
Object[]{
            res.getString(1),
            res.getString(2),
```

```
                    res.getString(3),
                    res.getString(4),
                   });
                }
                 TabelData.setModel(dtm);
               }catch(SQLException e){
```

6. Form Transaksi Permintaan Barang
   a. Tampilan form Transaksi permintaan barang

Gambar 4.8 Tampilan Form transaksi permintaan barang

   b. Kode program

```
private void
TabelDataMouseClicked(java.awt
.event.MouseEvent evt) {
      try{
         int row;
         row =
TabelData.getSelectedRow();

jTextField1.setText(TabelData.ge
tValueAt(row, 0).toString());

jTextField2.setText(TabelData.ge
tValueAt(row, 1).toString());

jTextField3.setText(TabelData.ge
tValueAt(row, 2).toString());

jTextField4.setText(TabelData.ge
tValueAt(row, 3).toString());
```

V. Simpulan Dan Saran

5.1 Simpulan

1. Sistem permintaan barang yang sedang berjalan di PT. Kino Indonesia Sukabumi sudah menggunakan sistem dengan menggunakan catatan-catatan yang kemudian direkap di Ms. Excel 2010, sehingga untuk lebih mempercepat dan mendapatkan data yang baik dan akurat maka dibutuhkan suatu aplikasi untuk membantu pengelolaan data permintaan barang tersebut.
2. Solusi perancangan sistem menggunakan UML mampu memberikan penyelesaian terhadap masalah yang ada dan pembuatan aplikasi menggunakan Java bisa memberikan pemrosesan data yang baik dan cepat.
3. Dengan adanya sistem baru berbasis Java ini mampu memberikan peningkatan kinerja karyawan khususnya untuk penyediaan barang-barang yang dibutuhkan oleh karyawan tersebut.

5.2 Saran

1. Penggunaan sistem yang terbatas hanya untuk versi Desktop bisa dikembangkan untuk versi web dan mobile.
2. Aplikasi sistem disarankan menggunakan tampilan-tampilan yang dinamis dan menarik untuk user.
3. Laporan yang disajikan disarankan menggunakan teknik pelaporan yang berbasis PDF.

DAFTAR PUSTAKA

Fatanshah, 2015. Sistem Basis Data. Yogyakarta : CV. Andi Ofset

Nugroho ADI, 2012. Rekayas Perangkat Lunak Metode USDP UML Yogyakarta : CV. Andi Ofset

Tata Sutabri, 2012. Analisa Sistem Informasi. Yogyakarta : CV. Andi Offset

Tohari-Hamim, ASTAH UML, 2014. Jakarta : Ikapi