Pembuatan Aplikasi Terintegrasi, Pendataan Barang di Gudang Berbasis Android

Dodi Triwibowo, Rinta Kridalukmana, Kurniawan Teguh Martono
Program Studi Sistem Komputer Fakultas Teknik Universitas Diponegoro
Jalan Prof. Sudharto, Tembalang, Semarang, Indonesia
silveriuz16@gmail.com

Abstrak - Dalam Industri yang semakin pesat perkembangannya, proses keluar masuknya barang perlu dicatat, ini diperlukan untuk mempermudah suatu perusahaan dalam mengontrol stok barang baik keluar ataupun masuk. Data fisik merupakan sesuatu yang mudah hilang, pengandaan data juga akan memberikan suatu redundant bagi seorang pekerja, dimana dimungkinkan adanya *Human Error*, salah satu akibatnya adalah dapat menyebabkan suatu perbedaan data, dimana dari beberapa data yang ada dapat berbeda nilai, ketika terdapat masalah seperti ini dapat merugikan perusahaan ataupun client dari perusahaan tersebut. Aplikasi ini juga dapat membantu perusahaan dalam melakukan *checking* barang secara cepat, sehingga perusahaan dapat menangani dengan cepat adanya suatu perubahan maupun kesalahan yang terjadi. Pembuatan aplikasi terintegrasi ini menggunakan bahasa pemrograman Java, PHP, dan database MYSQL serta menggunakan JQuery Mobile.

Kata Kunci : Android, Sistem, PHP, MySQL, WebView.

I. PENDAHULUAN

Seiring dengan perkembangan jaman, banyak hal yang dapat diselesaikan dengan Teknologi. Pada masa sekarang banyak dikembangkan Sistem Informasi yang bertujuan untuk memberikan kemudahan kepada penggunanya, Sistem Informasi dibangun sedemikian rupa sehingga dapat meningkatkan kinerja bagi suatu Perusahaan, Industri, Instansi, Organisasi, maupun Usaha Mandiri. Dalam perkembangannya integrasi antar sistem yang dimiliki sangat diperlukan. Dengan adanya integrasi ini dimungkinkan kita dapat melakukan suatu kerja dengan cepat, tepat dan lebih efektif serta efisien, dengan adanya pengintegrasian sistem secara menyeluruh dapat memberikan kemudahan kepada pengguna dimana salah satu keuntungannya adalah data cukup disimpan sekali, data dapat diakses darimana saja dan nilainya akan selalu sama, walaupun nantinya sistem yang terintegrasi ini akan memunculkan masalah-masalah baru seperti jika terjadi kesalahan data maka salah semua, kemudian jika data hilang maka akan hilang pada semua sistem, namun masalah tersebut dapat kita tangani nantinya.

Aplikasi terintegrasi ini dibuat berdasarkan Sistem Informasi perusahaan yang sudah ada, mengacu dari hal tersebut aplikasi ini dibuat agar di gudang dapat melakukan *crosscheck* terhadap pengiriman yang akan dilakukan sehingga pihak *checker* digudang dapat melakukan pemeriksaan secara cepat dengan mengacu pada basisdata yang ada pada perusahaan, sehingga ini akan membuat pekerja melakukan pekerjaannya dengan baik dan cepat..

Dalam pembuatan tugas akhir ini pembahasan masalah memiliki batasan pada permasalahan berikut :

a. Pembuatan Aplikasi Terintegrasi, Pendataan Barang di Gudang berbasis Android menggunakan bahasa pemrograman Java, PHP, HTML, CSS, Javascript, *Framework* JQuery, JQuery Mobile dan basisdata MySQL.
b. Hasil aplikasi akan disimulasikan di *Webhosting* dengan menggunakan beberapa *Gadget* Android.
c. Aplikasi yang dibuat hanya untuk sisi pengguna yaitu *Operator* yang ada digudang.
d. Aplikasi dijalankan pada perangkat Android dengan sistem operasi minimal versi 4.0 Android *IceCreamSandwich* hingga versi terbaru 5.0.1 Android *Lolipop*.

II. LANDASAN TEORI

A. *Sistem Informasi*

Sistem adalah sekumpulan objek-objek yang saling berelasi dan berinteraksi serta hubungan antar objek bisa dilihat sebagai satu kesatuan yang dirancang untuk mencapai satu tujuan. Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang.

Sistem Informasi dapat diartikan dengan suatu sistem di dalam organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi, dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan atau kumpulan elemen yang saling berhubungan satu sama lain yang membentuk satu kesatuan untuk integrasi data, memproses dan menyimpan serta mendistribusikan informasi. (Awalin 2012)

B. *Sistem Terintegrasi*

Sistem Terintegrasi atau sering disebut juga *Integrated Enterprise* merupakan tugas untuk membuat agar aplikasi-aplikasi yang bekerja pada berbagai *platform* di lokasi yang berbeda dapat bekerja sama guna menghasilkan suatu kesatuan fungsionalitas, sehingga dapat saling berbagi informasi, layanan dan proses bisnis baik di dalam *enterprise* maupun antar *enterprise*. (Kristanti 2012)

C. *Android*

Android merupakan Sistem operasi yang gratis dan terbuka dari google yang berjalan diberbagai perangkat seperti telepon, tablet bahkan televisi. Terdapat banyak perangkat yang dapat digunakan dengan satu *platform*. Aplikasi Android dapat dibangun pada Sistem Operasi Mac, Windows, dan Unix. Selain itu kita dapat mempublikasikan aplikasi yang kita bangun secara langsung tanpa perlu disetujui oleh siapapun.

Aplikasi Android memisahkan *Visual Definition* aplikasi(Seperti *XML Layout* dan *String Resource*) dari Tingkah Laku atau Tindakan yang didefinisikan pada Kode Java.

Aplikasi Android secara *Native* dibangun menggunakan bahasa pemrogramman Java, namun pada perkembangannya kita dapat menggunakan bahasa pemrogramman berbasis Web dimana ini dimungkinkan dengan memanfaatkan *Web View* yang ada pada Android. (Simon 2011)

Dalvik Virtual Machine (DVM) merupakan satu elemen kunci dari Android. Android berjalan di dalam DVM bukan di *Java Virtal Machine* (JVM). DVM adalah "*register based*" sementara JVM adalah "*stack based*". (ARIZA 2014)

D. Java

Bahasa pemrogramman Java memiliki *Syntax* yang bersahabat, menawarkan fitur berbasis *Object*, Manajemen memori, serta *Portability* yang baik. *Write-Once Run-Anywhere* menjadi kelebihan dari Java. Program java akan dijalankan pada *Java Virtual Machine*(JVM). JVM akan mengubah *bytecode* menjadi kode yang dapat dimengerti oleh *platform* kemudian menjalankan programnya. (Sierra & Bates 2005)

Ada 4 (empat) prinsip dasar dari pemrograman berorientasi obyek, yaitu abstraksi, enkapsulasi, modularitas dan hirarki. Berikut dijelaskan satu persatu secara singkat.

1. Abstraksi memfokuskan perhatian pada karakteristik obyek yang paling penting dan paling dominan yang bisa digunakan untuk membedakan obyek tersebut dari obyek lainnya.
2. Enkapsulasi menyembunyikan banyak hal yang terdapat dalam obyek yang tidak perlu diketahui oleh obyek lain. Dalam praktek pemrograman, enkapsulasi diwujudkan dengan membuat suatu kelas interface yang akan dipanggil oleh obyek lain, sementara didalam obyek yang dipanggil terdapat kelas lain yang mengimplementasikan apa yang terdapat dalam kelas interface.
3. Modularitas membagi sistem yang rumit menjadi bagian-bagian yang lebih kecil yang bisa mempermudah developermemahami dan mengelola obyek tersebut.
4. Hirarki berhubungan dengan abstraksi dan modularitas, yaitu pembagian berdasarkan urutan dan pengelompokkan tertentu. Misalnya untuk menentukan obyek mana yang berada pada kelompok yang sama, obyek mana yang merupakan komponen dari obyek yang memiliki hirarki lebih tinggi. Semakin rendah hirarki obyek berarti semakin jauh abstraksi dilakukan terhadap suatu obyek. (Haviluddin 2011)

E. PHP

PHP Pertama kali ditemukan pada 1995 oleh seorang *Software Developer* bernama Rasmus Lerdrof. Ide awal PHP adalah ketika itu Radmus ingin mengetahui jumlah pengunjung yang membaca resume *online*-nya. *script* yang dikembangkan baru dapat melakukan dua pekerjaan, yakni merekam informasi pengunjung, dan menampilkan jumlah pengunjung dari suatu website. Dan sampai sekarang kedua tugas tersebut masih tetap populer digunakan oleh dunia web saat ini. Kemudian, dari situ banyak orang di milis mendiskusikan script buatan Rasmus Lerdrof, hingga akhirnya rasmus mulai membuat sebuah *tool/script*, bernama Personal Home Page (PHP). (Dwiartara 2012)

PHP dibangun dari *scripts* yang ditulis secara *plaintext*. *PHP Interpreter* adalah bagian dari perangkat lunak yang ada pada *Web Server*, yang membaca file tersebut dan mengartikannya, memberikan keluaran HTML dan petunjuk mengenai bagaimana perilaku yang ada maupun menginterpretasikan masukan dari pengguna. (McLaughlin 2012)

F. HTML

HTML atau *HyperText Markup Language*, adalah suatu cara memberikan tanda yang memberikan perintah kepada *browser* bagaimana suatu teks terstruktur. HMTL memberikan perintah kepada *browser* bagaimana struktur dari dokumen kita, bagaimana *heading*-nya, bagaimana paragrafnya, bagaimana suatu teks akan ditampilkan, dan lainnya. Dengan informasi yang diberikan, *browsers* dibangun dengan perintah dasar bagaimana menampilkan setiap elemen yang ada. (Elisabeth & Eric 2012)

G. CSS

CSS atau *Cascading Style Sheet* adalah bahasa pemrogramman yang digunakan untuk mendeskripsikan bagaimana suatu konten akan ditampilkan. Kita memberikan karakteristik tampilan dari elemen yang ada pada HTML menggunakan CSS. (Elisabeth & Eric 2012)

Kode CSS dapat dituliskan dengan tiga cara yaitu *inline, internal dan external*. Ketiganya bisa anda lakukan sesuai dengan kebutuhan. (Ariona 2013)

H. Javascript

Javascript menjadikan suatu *web* lebih interaktif, ini menjadikannya bisa mengerti yang kita butuhkan, memroses masukkan pengguna, serta memberikan respon yang lebih baik.

Javascript digunakan bersama dengan HTML dan CSS sebagai salah satu dari tiga *Modern Web Page Construction.* HTML menyediakan struktur, CSS memberikan tampilan, dan *JavaScripts* menjalankannya dan membuat perintah berjalan. (Morrison 2007)

Javascript dikembangkan oleh Brendan Eich di Netspace pada tahun 1995-1996. Dulu disebut dengan *LiveScript*. Javascript bukanlah bahasa yang di-*compile*, ini membuatnya terlihat serta terasa memiliki banyak kelebihan.

Javascript pada awalnya terfokus pada sisi pengguna sebagai *Form-Validation*, dan digunakan dengan gambar untuk menyempurnakannya, lebih membantu, membuatnya lebih interaktif serta memberikan umpan balik bagi pengunjung. (Suehring n.d.)

I. JQuery

JQuery merupakan *Javascript Library* yang cepat, kecil, dan memiliki banyak feature. JQuery membuat sesuatu seperti *HTML Document Traversal* dan *Manipulation*, *Event Handling*, *Animation*, dan *AJAX* menjadi lebih sederhana dengan API yang mudah digunakan diberbagai *browser*. (jquery.com n.d.)

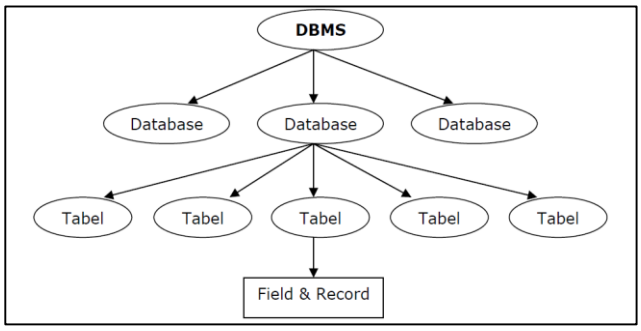
J. JQuery Mobile

JQuery adalah *HTML5-based user interface* yang merupakan *Framework* yang digunakan untuk melakukan desain *responsive website* atau aplikasi yang dijalankan pada banyak perangkat seperti *smartphone*, *tablet*, dan *desktop platform*. (Jquerymobile.com n.d.)

K. MySQL

Basisdata adalah suatu *tools* yang digunakan untuk menyimpan informasi, mengambil informasi kapanpun dibutuhkan, dan mengatur informasi yang tersimpan. Jika digambarkan Lemari *File* merupakan suatu basisdata. (McLaughlin 2012)

Database Management System atau DBMS merupakan suatu perangkat lunak yang digunakan untuk membuat, memelihara, mengontrol, dan mengakses basisdata secara praktis dan efisien. Sedangkan RDBMS atau *Relationship DBMS* merupakan salah satu DBMS yang mendukung adanya Relasi atau hubungan antar tabel.



Gambar 1 Hierarki Basisdata.

Terdapat 3 (tiga) jenis perintah SQL, yaitu :

1. DDL atau *Data Definition Language*

DDL merupakan perintah SQL yang berhubungan dengan pendefinisian suatu struktur database, dalam hal ini *database* dan *table*. Beberapa perintah dasar yang termasuk DDL ini antara lain *CREATE, ALTER, RENAME, DROP.*

2. DML atau *Data Manipulation Language*

DML merupakan perintah SQL yang berhubungan dengan manipulasi atau pengolahan data atau *record* dalam tabel. Perintah SQL yang termasuk dalam DML antara lain *SELECT, INSERT, UPDATE, DELETE.*

3. DCL atau *Data Control Language*

DCL merupakan perintah SQL yang berhubungan dengan manipulasi *user* dan hak akses (*priviledges*). Perintah SQL yang termasuk dalam DCL antara lain *GRANT, REVOKE*. (Solichin 2010)

L. *Rapid Application Development*

Rapid Application Development (RAD) adalah salah satu metode pengembangan suatu sistem informasi dengan waktu yang relatif singkat. Untuk pengembangan suatu sistem informasi yang normal membutuhkan waktu minimal 180 hari, akan tetapi dengan menggunakan metode RAD suatu sistem dapat diselesaikan hanya dalam waktu 30-90 hari.

Adapun kelemahan-kelemahan yang terdapat pada metode konvensional adalah sebagai berikut:

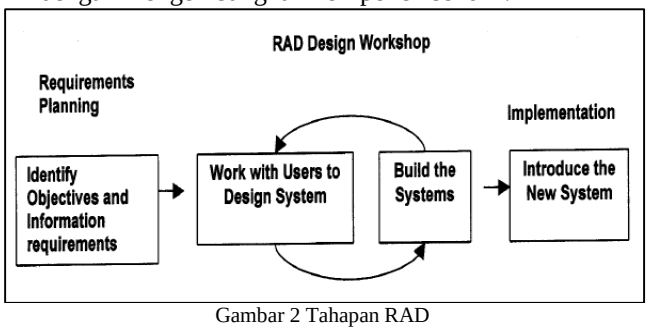
- Dengan metode konvensional, maka terdapat batas waktu yang cukup lama mulai dari pembuatan sistem sampai dengan konsumen dapat menggunakan sistem tersebut.
- Dengan metode konvensional, apabila proses pengembangan suatu sistem membutuhkan waktu yang lama maka kebutuhan konsumen pada sistem akan mengalami perubahan seiring dengan perubahan proses bisnis yang dilakukan oleh konsumen.
- Dengan metode konvensional, sistem yang dikembangkan tidak akan mempunyai manfaat apabila belum diselesaikan seluruhnya.

Pada saat mengembangkan suatu sistem pasti dihadapkan dengan 3 pilihan model yaitu:

- *Efficient Development* (model pengembangan yang mengutamakan *schedule*, ekonomi dan kualitas produk secara seimbang). *Schedule* lebih cepat dari rata-rata. Ekonomi, biaya lebih murah dari rata-rata. Produk lebih baik daripada kualitas rata-rata.
- *Sensible* RAD (model pengembangan yang mengutamakan *schedule* dibandingkan dengan ekonomi dan kualitas produk). *Schedule* lebih cepat dari rata-rata. Ekonomi, biaya lebih murah sedikit dari rata-rata. Produk lebih baik sedikit dari kualitas rata-rata.
- *All-out* RAD (model pengembangan yang mengutamakan schedule dengan mengorbankan ekonomi dan kualitas produk). *Schedule* paling cepat. Ekonomi biaya lebih mahal dari rata-rata. Produk lebih buruk dari kualitas rata-rata.

Dengan menggunakan RAD maka ada satu atau beberapa tujuan berikut ini yang tidak akan dapat dicapai secara bersamasama yaitu:

- Kemungkinan terjadi kesalahan yang kecil, karena pihak pengembang tidak mempunyai hak untuk mengubah komponen- komponen yang digunakan dalam mengembangkan suatu sistem.
- Tingkat kepuasan konsumen yang tertinggi, karena kebutuhan-kebutuhan sekunder dari konsumen harus dikorbankan supaya suatu sistem dapat diselesaikan sesuai jadwal.
- Biaya pengembangan yang termurah, karena dengan menggunakan komponen yang sudah ada dapat menyebabkan biaya yang lebih besar apabila dibandingkan dengan mengembangkan komponen sendiri.



Gambar 2 Tahapan RAD

Metode RAD mempunyai 3 tahapan utama seperti yang terlihat pada gambar 2. Berikut merupakan tahapannya :

1. Rencana Kebutuhan (*Requirement Planning*)

Pada tahap ini, *user* dan *analyst* melakukan semacam pertemuan untuk melakukan identifikasi tujuan dari aplikasi atau sistem dan melakukan identifikasi kebutuhan informasi untuk mencapai tujuan. Pada tahap ini hal terpenting adalah adanya keterlibatan dari kedua belah pihak, bukan hanya sekedar persetujuan akan proposal yang sudah dibuat. Untuk lebih jauh lagi, keterlibatan user bukan hanya dari satu tingkatan pada suatu organisasi, melainkan beberapa tingkatan organisasi sehingga informasi yang dibutuhkan untuk masingmasing *user* dapat terpenuhi dengan baik.

2. Proses Desain (*Design Workshop)*

Pada tahap ini adalah melakukan proses desain dan melakukan perbaikan perbaikan apabila masih terdapat ketidaksesuaian desain antara user dan *analyst*. Untuk tahap ini maka keaktifan *user* yang terlibat sangat menentukan untuk mencapai tujuan, karena user bisa langsung memberikan komentar apabila terdapat ketidaksesuaian pada desain. Biasanya, *user* dan *analyst* berkumpul menjadi satu dan duduk di meja melingkar dimana masing-masing orang bisa melihat satu dengan yang lain tanpa ada halangan.

3. Implementasi (*Implementation*)

Setelah desain dari sistem yang akan dibuat sudah disetujui baik itu oleh user dan *analyst*, maka pada tahap ini *programmer* mengembangkan desain menjadi suatu program. Setelah program selesai baik itu sebagian maupun secara keseluruhan, maka dilakukan proses pengujian terhadap program tersebut apakah terdapat kesalahan atau tidak sebelum diaplikasikan pada suatu organisasi. Pada saat ini maka user bias memberikan tanggapan akan sistem yang sudah dibuat serta persetujuan mengenai sistem tersebut.

4. Tahapan Keseluruhan

Dengan berdasarkan pada tahapantahapan tersebut di atas maka proses utama pengembangan suatu sistem dengan menggunakan metode RAD adalah sebagai berikut :

- Pengembang membuat *prototype* berdasarkan kebutuhan-kebutuhan yang sudah didefinisikan sebelumnya
- Desainer melakukan penilaian terhadap *prototype*

- User melakukan uji coba pada *prototype* dan memberikan masukan mengenai kebutuhan-kebutuhan yang kurang.
- *User* dan *developer* melakukan pertemuan untuk memberikan penilaian terhadap produk secara bersama-sama, menyesuaikan kebutuhan serta memberikan komentar apabila diperlukan perubahan.
- Semua kebutuhan akan sistem dan perubahan-perubahan yang terjadi dilakukan proses "*timeboxed*" dengan mempunyai 2 kemungkinan yaitu perubahan yang tidak dapat ditampung seperti yang sudah direncanakan harus dihilangkan, dan Jika diperlukan, kebutuhan-kebutuhan yang bersifat sekunder ditiadakan.

Beberapa keuntungan dalam menggunakan metode RAD adalah sebagai berikut:

- Membeli sistem yang baru memungkinkan untuk lebih menghemat biaya ketimbang mengembangkan sendiri.
- Proses pengiriman menjadi lebih mudah, hal ini dikarenakan proses pembuatan lebih banyak menggunakan potonganpotongan *script*.
- Mudah untuk diamati karena menggunakan model *prototype*, sehingga user lebih mengerti akan sistem yang dikembangkan.
- Lebih fleksibel karena pengembang dapat melakukan proses desain ulang pada saat yang bersamaan.
- Bisa mengurangi penulisan kode yang kompleks karena menggunakan *wizard*.
- Keterlibatan *user* semakin meningkat karena merupakan bagian dari tim secara keseluruhan.
- Mampu meminimalkan kesalahan-kesalahan dengan menggunakan alat-alat bantuan (*CASE tools*).
- Mempercepat waktu pengembangan sistem secara keseluruhan karena cenderung mengabaikan kualitas.
- Tampilan yang lebih standar dan nyaman dengan bantuan *software-software* pendukung.

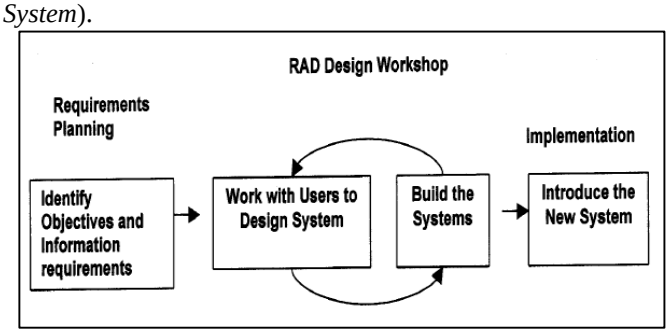
Selain itu dalam penggunaannya terdapat kerugian dalam menggunakan metode RAD, seperti berikut :

- Dengan melakukan pembelian belum tentu bisa menghemat biaya dibandingkan dengan mengembangkan sendiri.
- Membutuhkan biaya tersendiri untuk membeli peralatan-peralatan penunjang seperti misalnya *software* dan *hardware*.
- Kesulitan melakukan pengukuran mengenai kemajuan proses.
- Kurang efisien karena apabila melakukan pengkodean dengan menggunakan tangan bisa lebih efisien.
- Ketelitian menjadi berkurang karena tidak menggunakan metode yang formal dalam melakukan pengkodean.
- Lebih banyak terjadi kesalahan apabila hanya mengutamakan kecepatan dibandingkan dengan biaya dan kualitas.
- Fasilitas-fasilitas banyak yang dikurangi karena terbatasnya waktu yang tersedia.
- Sistem sulit diaplikasikan di tempat yang lain.
- Fasilitas yang tidak perlu terkadang harus disertakan, karena menggunakan komponen yang sudah jadi, sehingga hal ini membuat biaya semakin meningkat karena harga komponen yang lebih lengkap semakin mahal. (Noertjahyana 2002)

III. PERANCANGAN SISTEM

A. Tahap Pengembangan Sistem

Tahapan yang digunakan pada pembuatan sistem adalah *Rapid Application Development* (RAD). Pada Gambar 3 ditunjukkan bahwa RAD memiliki tiga tahapan utama, yaitu Rencana Kebutuhan (*Requirement Planning*), Proses Desain (*Desain Workshop*), dan Implementasi (*Implementation*). Sedangkan pada tahap Proses Desain dibagi lagi menjadi dua hal yaitu Desain Sistem (*Design System*) dan Membangun Sistem (*Build System*).



Gambar 3 Tahapan Desain RAD

- Rencana Kebutuhan.

Pada tahap ini akan dilakukan perencanaan terhadap kebutuhan sistem secara menyeluruh baik proses kerja maupun kebutuhan-kebutuhan yang diperlukan oleh pengguna. Pada tahap ini analisis terhadap sistem diperlukan untuk memilah antara kebutuhan pokok dan kebutuhan sekunder, ini diperlukan karena pada metodologi desain RAD yang ingin dicapai adalah kecepatan pembuatan produk, atau dengan kata lain waktu menjadi indikator utama dalam metodologi ini, selain waktu ada juga indikator lain seperti biaya serta kualitas dari produk itu sendiri.

- Proses Desain.

Proses desain dilakukan oleh pengembang bersama dengan pengguna, salah satu kelebihan dari metodologi ini adalah pengguna dilibatkan langsung dalam proses pembuatan sistem, ini membuat sistem yang dibuat lebih memenuhi proses kerja (*bussiness process*) yang dilakukan oleh pengguna, sehingga dapat memberikan kepuasan kepada pengguna.

Pada tahap desain sistem ini akan dilakukan beberapa desain, seperti basisdata menggunakan *Entity Relationship Diagram*, kemudian untuk melakukan desain proses kerja akan digunakan *Sequence Diagram* dan *Use Case Diagram*, serta pembuatan tampilan dasar dari aplikasi.

- Implementasi.

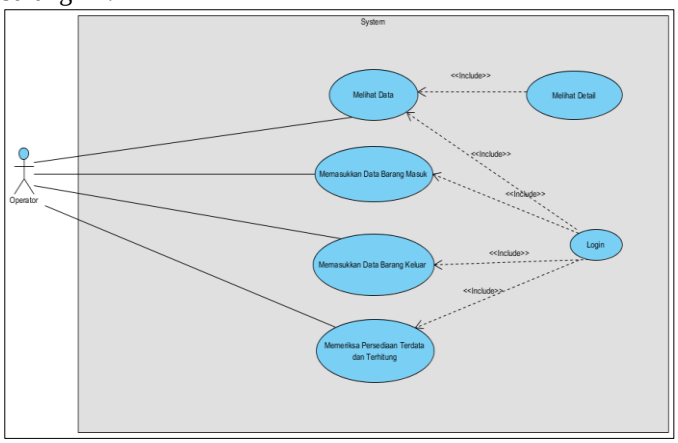
Tahap ini dilakukan setelah desain sistem disepakati oleh pengguna dan pengembang. Pada tahap ini pengembang akan menindak lanjuti desain sistem yang telah disetujui kedalam suatu sistem yang siap digunakan secara nyata. Setelah sistem selesai dibuat, akan dilakukan pengujian terhadap sistem. Keterlibatan pengguna dalam pengujian merupakan bagian penting yang memberikan kepuasan terhadap pengguna. Pada sistem ini akan dilakukan pengujian secara *White Box* atau Kotak Putih, yang artinya sistem akan diuji pada fungsi-fungsi yang ada apakah sudah sesuai dengan fungsinya, apakah sistem sudah bekerja seperti seharusnya.

B. Rencana Kebutuhan

Rencana kebutuhan merupakan tahapan yang sangat penting karena pada tahap ini dilakukan perencanaan terhadap sistem, baik perilakunya, fungsi-fungsi yang diinginkan dan pengelompokkan terhadap fitur-fitur yang harus ada dan fitur-fitur tambahan.

Dari perencanaan kebutuhan tersebut dapat digambarkan secara umum sistem yang akan dibuat dengan menggunakan UML, UML yang digunakan adalah UML versi 1.1 dimana terdapat delapan diagram (*Use Case, Activity, Sequence, Collaboration, Class, Statechart, Component, Deployment*) yang memiliki fungsi-fungsi yang berbeda dalam menggambarkan suatu sistem. Dalam menggambarkan

kebutuhan sistem digunakan *Use Case Diagram*, untuk kebutuhan dari aplikasi ini dapat dilihat pada Gambar 4 yang menggambarkan *Use Case Diagram* dari aplikasi pendataan barang ini.



Gambar 4 *Use Case Diagram* Sistem

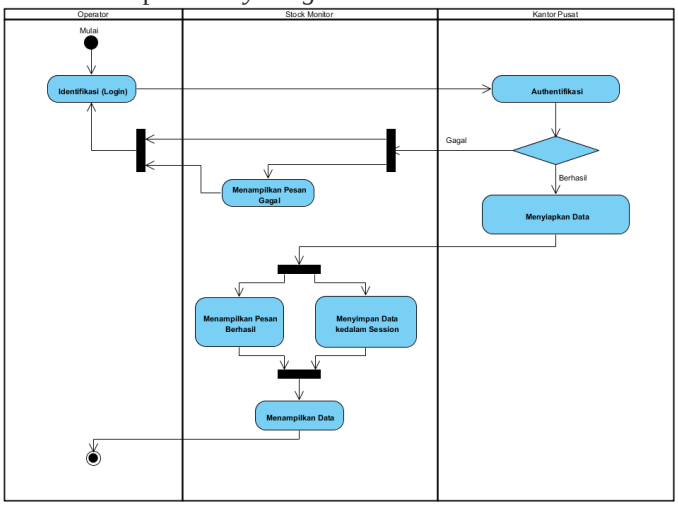
Pada Gambar 4 dapat dilihat terdapat satu *actor* yaitu *Operator*, *Operator* dalam konteks aplikasi ini merupakan orang yang bertanggung jawab terhadap pendataan barang, dengan kata lain *Operator* merupakan pengguna utama dari sistem ini. Kemudian, terdapat enam *Use Case* yaitu melihat data, memasukkan data barang masuk, memasukkan data barang keluar, memeriksa persediaan terdata dan terhitung, melihat *detail* dan *login*.

C. Proses Desain

Proses desain merupakan proses yang dilakukan setelah perencanaan kebutuhan dilakukan, ini dikarenakan dalam melakukan desain terhadap suatu sistem pasti seorang analis harus mengetahui spesifikasi atau kebutuhan dari sistem itu sendiri. Dalam melakukan desain sistem terdapat beberapa hal yang harus dibuat antara lain desain proses kerja (*bussiness process*), desain penyebaran sistem (*Deployment Design*), desain basisdata (*Database Design*), serta desain tampilan (*Layout Design*).

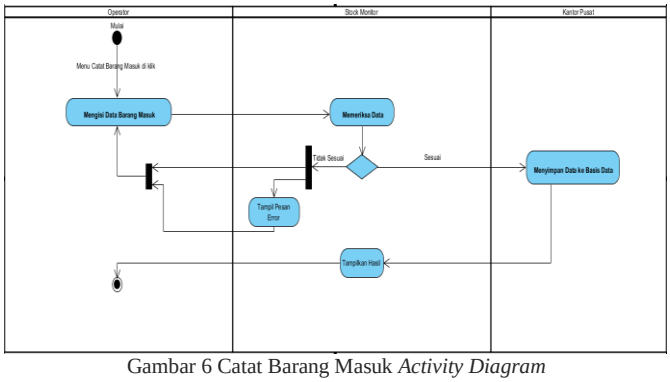
- Desain Proses Kerja

Desain proses kerja merupakan desain mendasar mengenai perilaku sistem dan aktivitas yang terjadi ketika aplikasi dijalankan. Aktivitas serta perilaku aplikasi ini digambarkan dalam beberapa *Activity Diagram*.



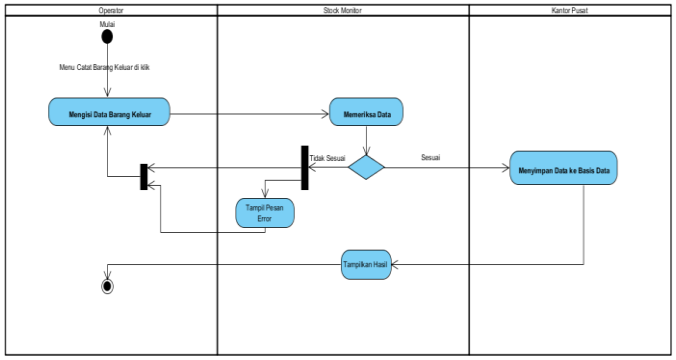
Gambar 5 *Login Activity Diagram*

Login Activity Diagram yang terdapat pada Gambar 5 menggambarkan tentang bagaimana proses yang akan terjadi ketika seorang *Operator* masuk ke dalam Aplikasi ini.



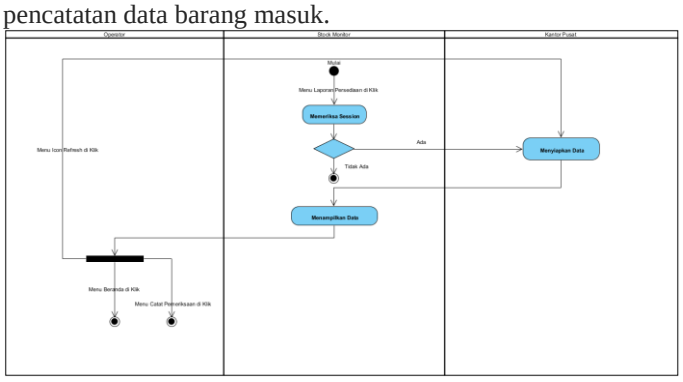
Gambar 6 Catat Barang Masuk *Activity Diagram*

Proses pendataan barang masuk dapat dilihat pada Gambar 6 dapat terlihat setelah *Operator* menekan menu Catat Barang Masuk.



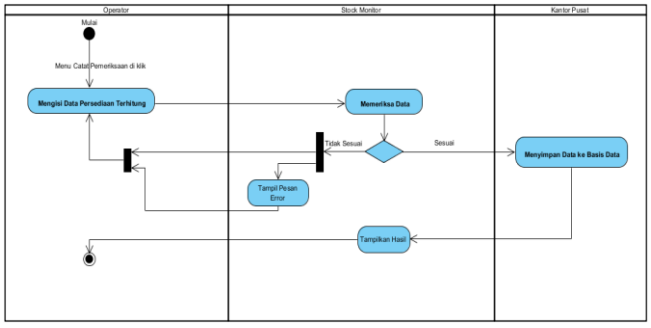
Gambar 7 Catat Barang Keluar *Activity Diagram*

Pada Gambar 7 digambarkan bagaimana proses yang terjadi ketika *Operator* melakukan pencatatan data karang keluar. Setelah pengguna menekan menu "Catat Barang Keluar. Terlihat bahwa aktivitas ini mirip dengan aktivitas pada pencatatan data barang masuk.



Gambar 8 Laporan Persediaan *Activity Diagram*

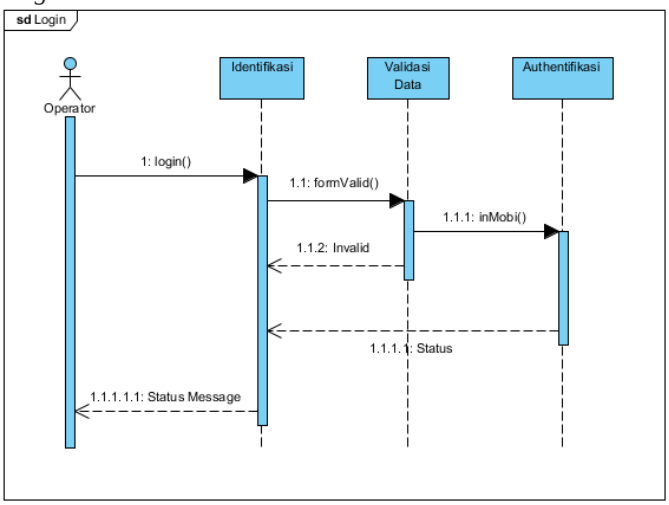
Aktivitas selanjutnya digambarkan menjadi dua buah *Activity Diagram*, yang pertama menjelaskan mengenai aktivitas perpindahan ke halaman "Laporan Persediaan", yang kedua menjelaskan mengenai aktivitas pencatatan persediaan terhitung. Pada Gambar 8 dapat dilihat ketika pengguna menekan menu "Laporan Persediaan" maka pengguna dialihkan ke halaman "Laporan Persediaan".



Gambar 9 Catat Pemeriksaan *Activity Diagram*

Aktivitas selanjutnya adalah melakukan pencatatan data persediaan barang terhitung yang dapat dilihat pada Gambar 9. Proses pada aktivitas ini mirip dengan proses yang ada pada pencatatan barang keluar dan masuk.

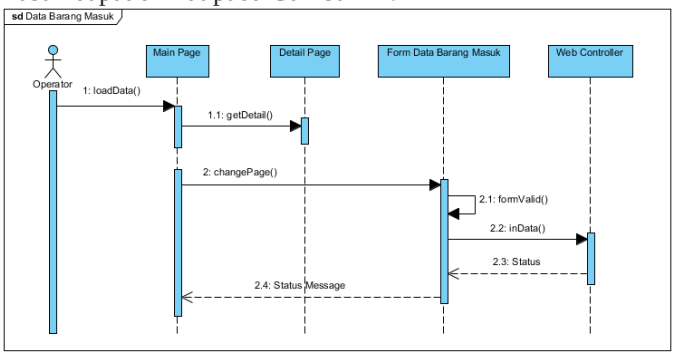
Selanjutnya untuk memberikan gambaran yang lebih jauh mengenai proses yang dilakukan oleh aplikasi akan digambarkan proses-proses yang terjadi dengan *Sequence Diagram*.



Gambar 10 *Login Sequence Diagram*

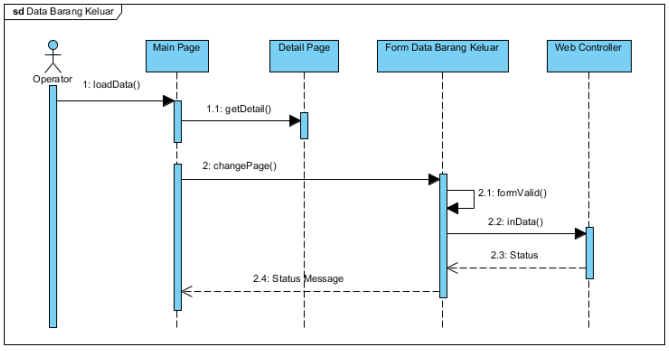
Pada Gambar 10 menggambarkan mengenai proses yang terjadi pada saat *Login*. Pada halaman utama nanti terdapat tiga *sequence*, yaitu catat barang masuk, catat barang keluar, dan catat laporan persediaan.

Proses yang terjadi ketika melakukan pendataan barang masuk dapat dilihat pada Gambar 11.



Gambar 11 Catat Barang Masuk *Sequence Diagram*

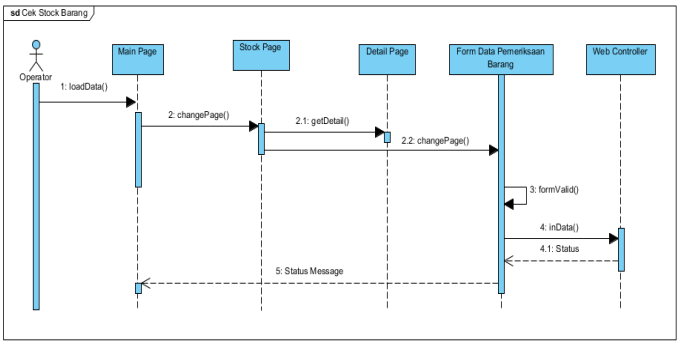
Dalam proses pendataan barang keluar dapat dilihat pada Gambar 12.



Gambar 12 Catat Barang Keluar *Sequence Diagram*

Proses yang ada pada pengelolaan laporan persediaan dapat dilihat pada Gambar 13.

Selanjutnya jika pengguna menekan laporan pemeriksaan pada halaman utama laporan persediaan, pengguna akan dialihkan ke halaman rincian. Halaman rincian memberikan informasi mengenai data pemeriksaan secara keseluruhan per pemeriksaan.

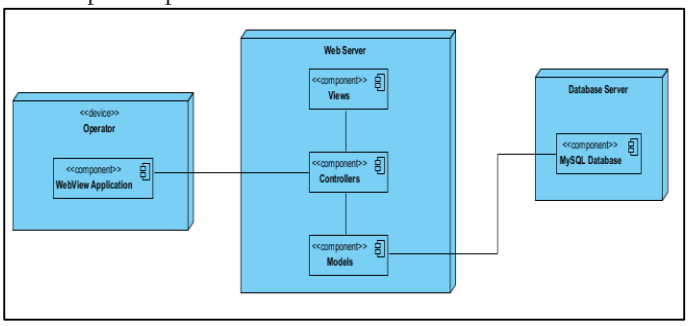


Gambar 13 Laporan Persediaan *Sequence Diagram*

Laporan persediaan memiliki keterkaitan dengan halaman utama maka pada *Sequence Diagram* Gambar 3.11 digambarkan saling terkait.

- Desain Penyebaran Sistem

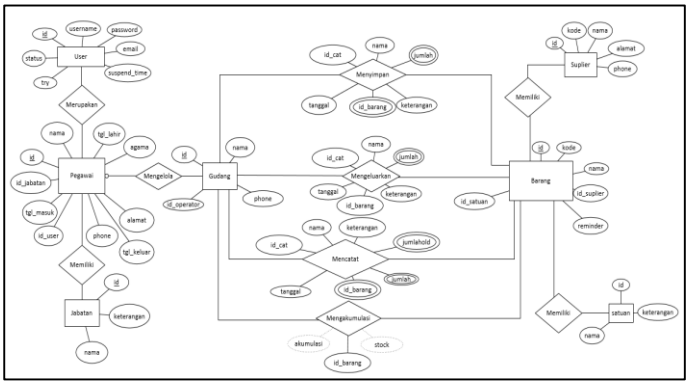
Deployment design merupakan penggambaran bagaimana suatu sistem itu akan dibangun, apa saja yang dibutuhkan untuk membangun sistem tersebut, serta bagaimana antar perangkat sistem tersebut saling terkait. Ditunjukkan desain penyebaran sistem aplikasi pada Gambar 14.



Gambar 14 *Deployment Diagram* Sistem

- Desain Basis Data

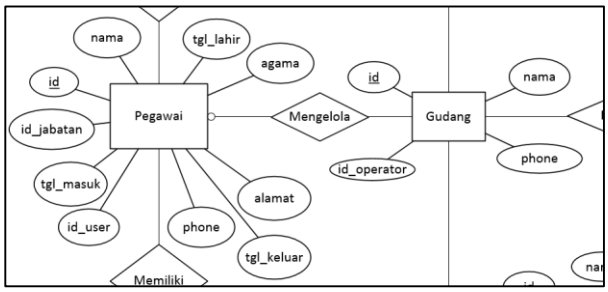
Pada perancangan perangkat lunak, desain basisdata merupakan suatu tahap yang sangat penting, dimana dalam tahap ini akan dilakukan pembuatan desain dari skema penyimpanan data dari perangkat lunak. Biasanya pembuatan desain basisdata dilakukan dengan membuat gambaran mengenai kebutuhan, serta keterkaitan antar data, gambaran yang digunakan biasanya adalah *Entity Relationship Diagram* (ERD). Untuk menggambarkan bagaimana basisdata aplikasi pendataan barang ini dibangun, terdapat ERD yang menunjukkan struktur data dari aplikasi yang terdapat pada Gambar 15.



Gambar 15 *Entity Relationship Diagram* (ERD) Aplikasi

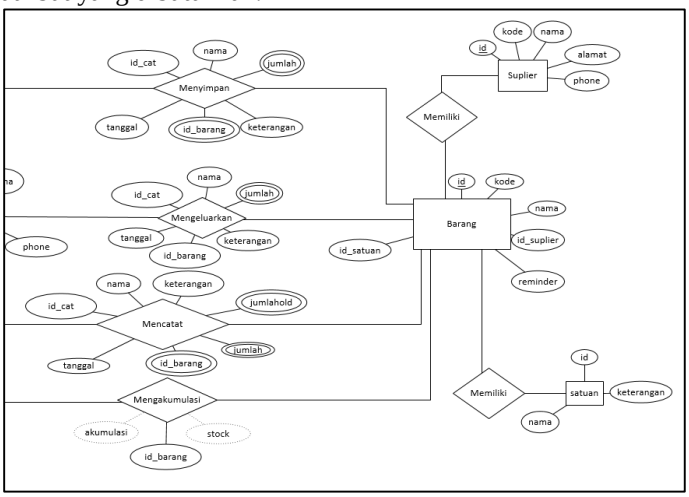
Pada Gambar 15 dapat dilihat keterkaitan antara tabel "Pegawai" dengan tabel "User" dan tabel "Jabatan".

Keterkaitan antara tabel "Pegawai" dengan tabel "Gudang" disini bertujuan agar setiap gudang yang ada memiliki *Operator* yang bertanggung jawab terhadap gudang itu sendiri pada Gambar 16 ditunjukkan keterkaitan antara keduanya.



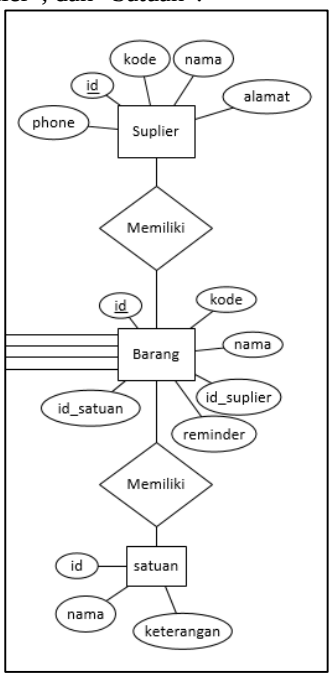
Gambar 16 ERD Keterkaitan Pegawai dan Gudang

Aplikasi ini memiliki tiga fitur utama, yaitu catat barang masuk, catat barang keluar, dan catat pemeriksaan. Dari fitur ini dikelompokkan lagi untuk setiap gudang yang dimiliki, dimana setiap gudang akan memiliki data per barang yang ada. Sehingga dapat disimpulkan bahwa gudang menyimpan data nota masuk, nota keluar, dan laporan persediaan. Pada Gambar 17 dapat dilihat bahwa keterkaitan masing-masing entitas menghasilkan atribut yang dibutuhkan.



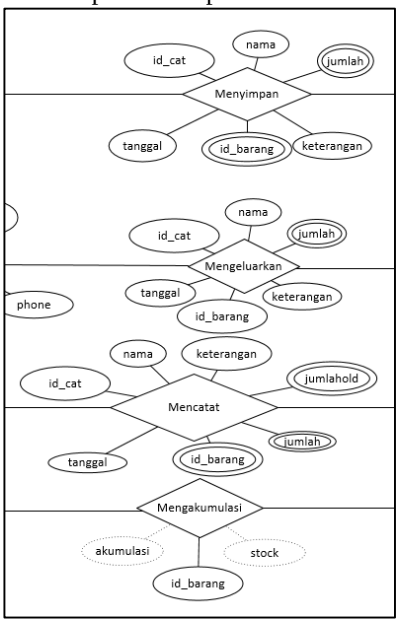
Gambar 17 ERD Keterkaitan Gudang dan Barang

Pada Gambar 18 ditunjukkan keterkaitan antara entitas "Barang", "Suplier", dan "Satuan".

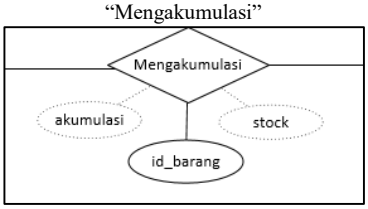


Gambar 18 ERD Keterkaitan Barang, *Suplier*, dan Satuan

Keterkaitan antara relasi "Menyimpan", relasi "Mengeluarkan", serta relasi "Mencatat" menghasilkan atribut turunan yang dihasilkan dari ketiga relasi tersebut. Keterkaitan antara ketiga relasi ini terlihat pada Gambar 19. Akumulasi barang dihasilkan dari persediaan terdata yaitu jumlah barang yang didata oleh sistem, dikurangi dengan persediaan terhitung. Atribut turunan ini dapat dilihat pada Gambar 20.



Gambar 19 Relasi "Menyimpan", "Mengeluarkan", "Mencatat", dan "Mengakumulasi"

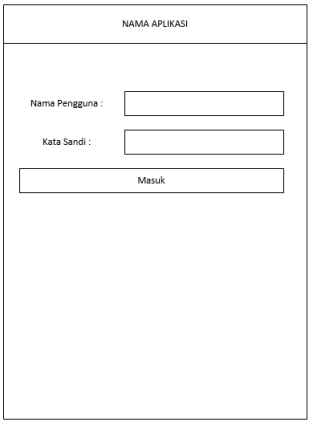


Gambar 20 *Atribut* Turunan Akumulasi dan Stock

- Desain Tampilan

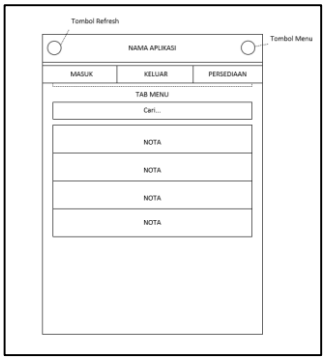
Desain tampilan merupakan tahapan yang dilakukan berikutnya, desain tampilan bertujuan memberikan gambaran secara umum dari aplikasi yang akan dibangun.

Pada Gambar 21 diperlihatkan desain tampilan halaman *login*.



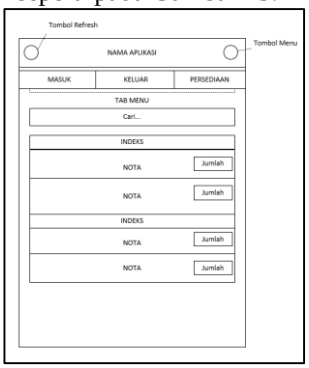
Gambar 21 Tampilan Halaman *Login*

Pada Gambar 22 merupakan tampilan halaman utama dari aplikasi ini.



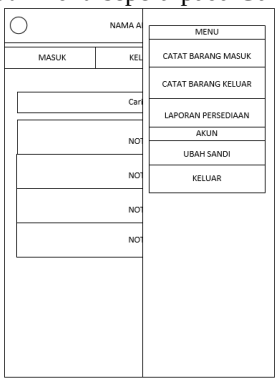
Gambar 22 Tampilan Halaman Utama

Terdapat tiga *tab menu* yang terdapat pada halaman utama, *tab menu* "Masuk" dan "Keluar" memiliki tampilan yang sama seperti pada Gambar 22, sedangkan *tab menu* "Persediaan" memiliki tampilan seperti pada Gambar 23.



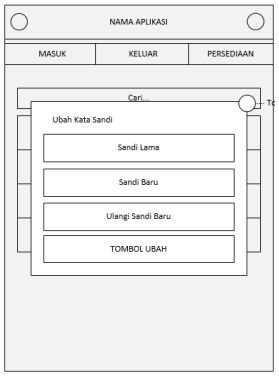
Gambar 23 Tampilan Halaman *Tab Menu* Persediaan

Pada halaman utama terdapat dua tombol pada bagian *header*, yaitu tombol "*refresh*" yang berfungsi untuk mengambil data ulang serta tombol "*menu*" yang berfungsi untuk menampilkan *sidebar menu* seperti pada Gambar 24.



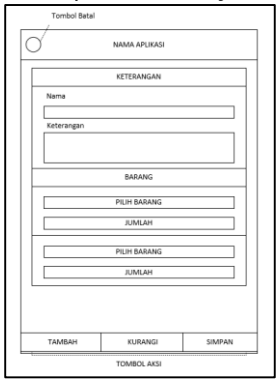
Gambar 24 Tampilan *Sidebar Menu*

Pada Gambar 25 merupakan *pop-up* yang muncul ketika "Ubah Sandi" ditekan.

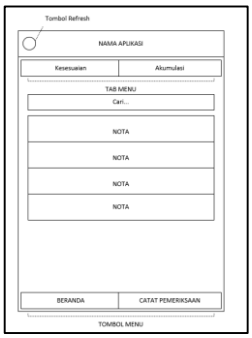


Gambar 25 Tampilan *Pop-up* Ubah Sandi

Sedangkan *menu* "Catat Barang Masuk" dan "Catat Barang Keluar" menuju kehalaman yang sama namun aksi yang dibentuk akan mengikuti *variable* action yang ada pada Javascript. Tampilan dari *form* ini ditunjukkan pada Gambar 26.



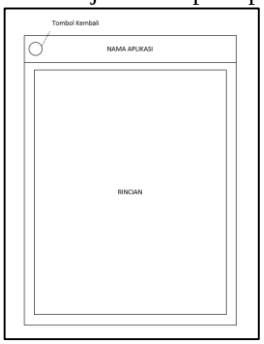
Gambar 26 Tampilan *Form* Masukkan Data



Gambar 27 Tampilan Halaman Laporan Persediaan

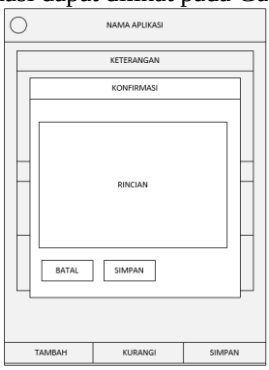
Ditunjukkan pada Gambar 27 merupakan desain tampilan dari halaman "Laporan Persediaan", serta "Catat Pemeriksaan" yang digunakan untuk masuk ke *form* masukkan data seperti pada Gambar 26 namun dengan aksi "cekstock".

Halaman "*Detail*" ditunjukkan seperti pada gambar 28.



Gambar 28 Tampilan Halaman Rincian Data

Pop-up konfirmasi dapat dilihat pada Gambar 29.



Gambar 29 Tampilan *Pop-up* Konfirmasi

IV. IMPLEMENTASI DAN PENGUJIAN

A. Tahap Pembuatan

Pembuatan Aplikasi dilakukan secara terurut berdasarkan kebutuhan dari sistem itu sendiri, oleh karena itu pada pembuatan aplikasi ini yang pertama dibuat adalah Basisdata. Tahap selanjutnya adalah membuat *Resource Sharing* yang akan digunakan pada sistem informasi yang terintegrasi dengan aplikasi ini.

- Tahap Pembuatan Basisdata

Pembuatan Basis Data dilakukan menggunakan aplikasi pihak ketiga phpmyadmin. Phpmyadmin menyediakan pembuatan basisdata secara GUI, hanya dengan mengetikkan beberapa hal dan memilih beberapa pilihan saja basisdata dapat dibuat. Namun walaupun begitu phpmyadmin juga menyediakan *Menu* SQL yang dapat digunakan untuk melakukan eksekusi *query* langsung ke basisdata.

```
#TABEL USER
CREATE TABLE tb_user(
id int( 5 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
username varchar( 20 ) UNIQUE KEY NOT NULL ,
PASSWORD varchar( 40 ) NOT NULL ,
email varchar( 50 ) NOT NULL UNIQUE KEY ,
STATUS set( 'Active', 'Disable', 'Suspend' ) NOT NULL DEFAULT
'Disable',
try int( 1 ) UNSIGNED NOT NULL ,
suspend_time timestamp NOT NULL DEFAULT NOW( )
);
```

Gambar 30 Membuat tabel tb_user

Pada kode Gambar 30 di atas menunjukkan bagaimana tabel tb_user dibuat.

```
#TABEL SATUAN
CREATE TABLE tb_satuan(
id int( 2 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
nama varchar( 20 ) NOT NULL ,
keterangan text NOT NULL DEFAULT ''
);
```

Gambar 31 Membuat tabel tb_satuan

Kode *query* pada Gambar 31 digunakan untuk membuat tabel yang berisi satuan-satuan dari barang-barang yang ada di gudang.

```
#TABEL SUPLIER
CREATE TABLE tb_suplier(
id int( 3 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
kode varchar( 5 ) NOT NULL UNIQUE KEY ,
nama varchar( 50 ) NOT NULL ,
alamat varchar( 200 ) NOT NULL ,
phone varchar( 20 ) NOT NULL
);
```

Gambar 32 Membuat tabel tb_suplier

Tabel tb_suplier dibuat dengan menjalankan kode pada Gambar 32.

```
#TABEL JABATAN
CREATE TABLE tb_jabatan(
id int( 2 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
nama varchar( 50 ) NOT NULL ,
keterangan text NOT NULL
);
```

Gambar 33 Membuat tabel tb_jabatan

Tabel tb_jabatan digunakan sebagai acuan untuk melihat *level user*. Kode *query* yang digunakan ditunjukkan pada Gambar 33.

```
#TABEL PEGAWAI
CREATE TABLE tb_pegawai(
id int( 4 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
nama varchar( 100 ) NOT NULL ,
tgl_lahir date NOT NULL ,
agama  set(  'Islam',  'Katholik',  'Kristen',  'Hindu',  'Budha',
'Lainnya' ) NOT NULL DEFAULT 'Lainnya',
id_jabatan int( 2 ) UNSIGNED NOT NULL ,
alamat varchar( 200 ) NOT NULL ,
phone varchar( 20 ) NOT NULL ,
tgl_masuk date NOT NULL ,
tgl_keluar date,
id_user int( 5 ) UNSIGNED,
FOREIGN KEY ( id_jabatan ) REFERENCES tb_jabatan( id ) ON DELETE
CASCADE ON UPDATE CASCADE ,
FOREIGN KEY ( id_user ) REFERENCES tb_user( id ) ON DELETE SET
NULL ON UPDATE CASCADE
);
```

Gambar 34 Membuat tabel tb_pegawai

Pada Gambar 34 ditunjukkan kode yang digunakan untuk membuat tabel tb_pegawai.

```
#TABEL GUDANG
CREATE TABLE tb_gudang(
id int( 2 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
nama varchar( 30 ) NOT NULL UNIQUE KEY ,
id_operator int( 4 ) UNSIGNED,
phone varchar( 20 ) NOT NULL DEFAULT '',
FOREIGN KEY ( id_operator ) REFERENCES tb_pegawai( id ) ON DELETE
SET NULL ON UPDATE CASCADE
);
```

Gambar 35 Membuat tabel tb_gudang

Tabel gudang dibuat menggunakan kode pada Gambar 35.

```
#TABEL BARANG
CREATE TABLE tb_barang(
id int( 5 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
kode varchar( 5 ) NOT NULL UNIQUE KEY ,
nama varchar( 100 ) NOT NULL ,
id_suplier int( 3 ) UNSIGNED,
id_satuan int( 2 ) UNSIGNED NOT NULL ,
reminder int( 10 ) UNSIGNED NOT NULL ,
FOREIGN KEY ( id_suplier ) REFERENCES tb_suplier( id ) ON DELETE
SET NULL ON UPDATE CASCADE ,
FOREIGN KEY ( id_satuan ) REFERENCES tb_satuan( id ) ON UPDATE
CASCADE
);
```

Gambar 36 Membuat tabel tb_barang

Tabel tb_barang digunakan sebagai acuan dari masukkan barang masuk, barang keluar, serta laporan persediaan. Dengan kata lain daftar dari barang-barang yang ada di gudang disimpan pada tabel ini. Untuk membuat tabel tb_barang digunakan kode seperti pada Gambar 36.

```
#TABEL CAT MASUK
CREATE TABLE tb_catmasuk(
id int( 16 ) UNSIGNED PRIMARY KEY AUTO_INCREMENT ,
tanggal timestamp NOT NULL ,
nama varchar( 30 ) NOT NULL ,
id_gudang int( 2 ) unsigned,
id_operator int(4) unsigned,
keterangan text NOT NULL DEFAULT '',
FOREIGN KEY ( id_gudang ) REFERENCES tb_gudang( id ) ON DELETE SET
NULL ON UPDATE CASCADE ,
FOREIGN KEY ( id_operator ) REFERENCES tb_pegawai( id ) ON DELETE
SET NULL ON UPDATE CASCADE
);
#TABEL MASUK
CREATE TABLE tb_masuk(
id int( 16 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
id_cat int(16) UNSIGNED NOT NULL,
id_barang int( 16 ) UNSIGNED NOT NULL ,
jumlah int( 16 ) UNSIGNED NOT NULL DEFAULT 0,
FOREIGN KEY ( id_cat ) REFERENCES tb_catmasuk( id ) ON DELETE
CASCADE ON UPDATE CASCADE ,
FOREIGN KEY ( id_barang ) REFERENCES tb_barang( id ) ON DELETE
CASCADE ON UPDATE CASCADE
);
```

Gambar 37 Membuat tabel tb_catmasuk dan tb_masuk

Pada Gambar 37 ditunjukkan kode untuk membuat tabel tb_catmasuk dan tabel tb_masuk.

```
#TABEL CAT KELUAR
CREATE TABLE tb_catkeluar(
id int( 16 ) UNSIGNED PRIMARY KEY AUTO_INCREMENT ,
tanggal timestamp NOT NULL ,
nama varchar( 30 ) NOT NULL ,
id_gudang int( 2 ) unsigned,
id_operator int(4) unsigned,
keterangan text NOT NULL DEFAULT '',
FOREIGN KEY ( id_gudang ) REFERENCES tb_gudang( id ) ON DELETE SET
NULL ON UPDATE CASCADE ,
FOREIGN KEY ( id_operator ) REFERENCES tb_pegawai( id ) ON DELETE
SET NULL ON UPDATE CASCADE
);
#TABEL KELUAR
CREATE TABLE tb_keluar(
id int( 16 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
id_cat int(16) UNSIGNED NOT NULL,
id_barang int( 16 ) UNSIGNED NOT NULL ,
jumlah int( 16 ) UNSIGNED NOT NULL DEFAULT 0,
FOREIGN KEY ( id_cat ) REFERENCES tb_catkeluar( id ) ON DELETE
CASCADE ON UPDATE CASCADE,
FOREIGN KEY ( id_barang ) REFERENCES tb_barang( id ) ON DELETE
CASCADE ON UPDATE CASCADE
);
```

Gambar 38 Membuat tabel tb_catkeluar dan tb_keluar

Pada Gambar 39 ditunjukkan kode untuk membuat tabel yang digunakan pada pengolahan data keluar.

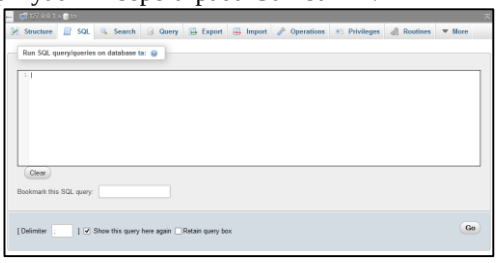
```
#TABEL CAT STOCK
CREATE TABLE tb_catcekstock(
id int( 16 ) UNSIGNED PRIMARY KEY AUTO_INCREMENT ,
tanggal timestamp NOT NULL ,
nama varchar( 30 ) NOT NULL ,
id_gudang int( 2 ) unsigned,
id_operator int(4) unsigned,
keterangan text NOT NULL DEFAULT '',
FOREIGN KEY ( id_gudang ) REFERENCES tb_gudang( id ) ON DELETE SET
NULL ON UPDATE CASCADE ,
FOREIGN KEY ( id_operator ) REFERENCES tb_pegawai( id ) ON DELETE
SET NULL ON UPDATE CASCADE
);

#TABEL CEK STOCK
CREATE TABLE tb_cekstock(
id int( 16 ) UNSIGNED AUTO_INCREMENT PRIMARY KEY ,
id_cat int(16) UNSIGNED NOT NULL,
id_barang int( 16 ) UNSIGNED NOT NULL ,
jumlahold int( 16 ) NOT NULL DEFAULT 0,
jumlah int( 16 ) NOT NULL DEFAULT 0,
FOREIGN KEY ( id_cat ) REFERENCES tb_catcekstock( id ) ON DELETE
CASCADE ON UPDATE CASCADE,
FOREIGN KEY ( id_barang ) REFERENCES tb_barang( id ) ON DELETE
CASCADE ON UPDATE CASCADE
);
```

Gambar 40 Membuat tabel tb_catcekstock dan tb_cekstock

Pada Gambar 40 ditunjukkan kode untuk membuat tabel yang dibutuhkan untuk mengolah data laporan persediaan.

Query yang digunakan dieksekusi pada menu *SQL* yang ada pada phpMyadmin seperti pada Gambar 41.



Gambar 41 Menu *SQL* pada phpMyAdmin

View adalah tabel yang dibuat secara *virtual* yang sebenarnya merupakan *stored query* yang nantinya akan dieksekusi ketika sebuah *View* dipanggil, untuk melihat suatu.

```
#VIEW PEGAWAI
CREATE OR REPLACE VIEW vw_pegawai AS
SELECT a. * , b.nama AS jabatan, IFNULL( c.username, 'Tidak Ada' )
AS username, IFNULL( c.email, 'Tidak Ada' ) AS email, IFNULL(
c.status, 'Tidak Ada' ) AS
status
FROM tb_pegawai AS a
LEFT JOIN tb_jabatan AS b ON a.id_jabatan = b.id
LEFT JOIN tb_user AS c ON a.id_user = c.id
```

Gambar 42 Membuat view vw_pegawai

View vw_pegawai digunakan untuk menggabungkan data sehingga informasi yang terbentuk dapat digunakan oleh pengguna sistem, *view* ini dibuat menggunakan kode seperti pada Gambar 42.

```
#VIEW GUDANG
CREATE OR REPLACE VIEW vw_gudang AS
SELECT a.*, b.nama AS operator, b.phone AS phone_op
FROM tb_gudang AS a
LEFT JOIN tb_pegawai AS b ON a.id_operator = b.id
```

Gambar 43 Membuat view vw_gudang

Pada vw_gudang dibuat menggunakan kode seperti pada Gambar 43.

```
#VIEW HELP MASUK
CREATE OR REPLACE VIEW vw_hmasuk AS
SELECT a.*, b.id AS id_masuk, b.id_barang, b.jumlah
FROM tb_catmasuk AS a
LEFT JOIN tb_masuk AS b ON a.id = b.id_cat;
#VIEW HELP KELUAR
CREATE OR REPLACE VIEW vw_hkeluar AS
SELECT a.*, b.id AS id_keluar, b.id_barang, b.jumlah
FROM tb_catkeluar AS a
LEFT JOIN tb_keluar AS b ON a.id = b.id_cat;
#VIEW HELP STOCK
CREATE OR REPLACE VIEW vw_hstock AS
SELECT    a.nama,a.id    as    id_gudang,b.id,b.kode,b.nama    as
barang,b.reminder,
IFNULL( (select sum(c.jumlah) from vw_hmasuk as c where (a.id =
c.id_gudang and b.id=c.id_barang)) , 0 ) -
IFNULL( (select sum(d.jumlah) from vw_hkeluar as d where (a.id =
d.id_gudang and b.id=d.id_barang)) , 0 ) AS jumlah
FROM tb_gudang AS a
LEFT JOIN tb_barang AS b on true;
#VIEW HELP BARANG
CREATE OR REPLACE VIEW vw_hbarang AS
SELECT a.*, b.kode AS kd_suplier, b.nama AS suplier, b.alamat,
b.phone, c.nama AS satuan, IFNULL( sum( d.jumlah ) , 0 ) AS stock
FROM tb_barang AS a
LEFT JOIN tb_suplier AS b ON a.id_suplier = b.id
LEFT JOIN tb_satuan AS c ON a.id_satuan = c.id
LEFT JOIN vw_hstock as d ON a.id = d.id
GROUP BY a.id;
```

Gambar 44 Membuat view vw_hmasuk, vw_hkeluar, vw_hstock, dan vw_hbarang

Kode *views* pada Gambar 44 digunakan untuk membantu dalam membuat sebuah *view* yang lebih kompleks lagi. Contoh pada *view* vw_hstock dibuat agar dapat menghitung nilai persediaan hanya dari data barang keluar-masuk, namun pada tingkat yang lebih jauh, *Operator* membutuhkan nilai jumlah persediaan yang sudah dikurangi dengan data terhitung (tb_cekstock) sehingga persediaan yang terhitung lebih sesuai atau *valid*.

```
#VIEW MASUK
CREATE OR REPLACE VIEW vw_masuk AS
SELECT a.id, a.id_cat, e.tanggal,e.nama as subject, e.keterangan,
b.id AS id_barang, b.kode, b.nama, b.suplier, b.satuan, a.jumlah,
b.reminder, b.stock AS stocktotal, c.id AS id_gudang, c.nama AS
gudang, d.id AS id_operator, d.nama AS operator
FROM tb_masuk AS a
LEFT JOIN tb_catmasuk as e on a.id_cat = e.id
LEFT JOIN vw_hbarang AS b ON a.id_barang = b.id
LEFT JOIN tb_gudang AS c ON e.id_gudang = c.id
LEFT JOIN tb_pegawai AS d ON e.id_operator = d.id;
#VIEW KELUAR
CREATE OR REPLACE VIEW vw_keluar AS
SELECT a.id,a.id_cat, e.tanggal,e.nama as subject, e.keterangan,
b.id AS id_barang, b.kode, b.nama, b.suplier, b.satuan,
a.jumlah,b.reminder,b.stock as stocktotal, c.id AS id_gudang,
c.nama AS gudang, d.id AS id_operator, d.nama AS operator
FROM tb_keluar AS a
LEFT JOIN tb_catkeluar as e on a.id_cat = e.id
LEFT JOIN vw_hbarang AS b ON a.id_barang = b.id
LEFT JOIN tb_gudang AS c ON e.id_gudang = c.id
LEFT JOIN tb_pegawai AS d ON e.id_operator = d.id;
#VIEW CEKSTOCK
CREATE OR REPLACE VIEW vw_cekstock AS
SELECT a.id,a.id_cat, e.tanggal,e.nama as subject, e.keterangan,
b.id AS id_barang, b.kode, b.nama, b.suplier,
b.satuan,a.jumlahold, a.jumlah,b.reminder,b.stock as stocktotal,
c.id AS id_gudang, c.nama AS gudang, d.id AS id_operator, d.nama
AS operator
FROM tb_cekstock AS a
LEFT JOIN tb_catcekstock as e on a.id_cat = e.id
LEFT JOIN vw_hbarang AS b ON a.id_barang = b.id
LEFT JOIN tb_gudang AS c ON e.id_gudang = c.id
LEFT JOIN tb_pegawai AS d ON e.id_operator = d.id;
```

Gambar 45 Membuat view vw_masuk, vw_keluar, vw_cekstock

Pada *Query* yang terdapat pada Gambar 45 digunakan untuk membuat *views* yang menggabungkan data masuk, keluar, dan cekstock yang sudah ternormalisasi.

```
#VIEW AKUMULASI
CREATE OR REPLACE VIEW vw_akumulasi AS
SELECT a.nama, a.id AS id_gudang, b.id, b.kode, b.nama AS barang,
IFNULL( sum( c.jumlah - c.jumlahold ) , 0 ) AS akumulasi
FROM tb_gudang AS a
LEFT JOIN tb_barang AS b ON
TRUE LEFT JOIN vw_cekstock AS c ON a.id = c.id_gudang
AND b.id = c.id_barang
GROUP BY a.id, b.id;
```

Gambar 46 Membuat view vw_akumulasi

View vw_akumulasi dibuat dengan menggunakan kode seperti pada Gambar 46.

.

```
#VIEW STOCK
CREATE OR REPLACE VIEW vw_stock AS
SELECT a.nama, a.id_gudang, a.id, a.kode, a.barang, a.reminder, (
a.jumlah + b.akumulasi
) AS jumlah
FROM vw_hstock AS a
INNER JOIN vw_akumulasi AS b ON ( a.id = b.id
AND a.id_gudang = b.id_gudang );
```

Gambar 47 Membuat view vw_stock

Pada Gambar 47 ditunjukkan kode yang digunakan untuk membuat *view* vw_stock.

```
#VIEW BARANG
CREATE OR REPLACE VIEW vw_barang AS
SELECT a.*, b.kode AS kd_suplier, b.nama AS suplier, b.alamat,
b.phone, c.nama AS satuan, IFNULL( sum( d.jumlah ) , 0 ) AS stock
FROM tb_barang AS a
LEFT JOIN tb_suplier AS b ON a.id_suplier = b.id
LEFT JOIN tb_satuan AS c ON a.id_satuan = c.id
LEFT JOIN vw_stock as d ON a.id = d.id
GROUP BY a.id;
```

Gambar 48 Membuat view vw_barang

View vw_barang dibuat menggunakan kode seperti pada Gambar 48.

```
#VIEW USER
CREATE OR REPLACE VIEW vw_user AS
SELECT a.*, IFNULL( b.id, 'all' ) AS id_pegawai, b.nama,
IFNULL( c.id, 'all' ) AS id_gudang, c.nama AS gudang, b.id_jabatan
FROM tb_user AS a
LEFT JOIN tb_pegawai AS b ON a.id = b.id_user
LEFT JOIN tb_gudang AS c ON b.id = c.id_operator
```

Gambar 49 Membuat view vw_user

View vw_user yang dibuat dengan menggunakan kode seperti pada Gambar 49.

Dari pembuatan *views* tersebut maka diharapkan aplikasi ini dapat mempermudah *programmer* dengan menyediakan informasi yang sudah diolah, sehingga pada sisi pemrogramman hanya cukup menampilkannya saja.

- Pembuatan *Resource Sharing*

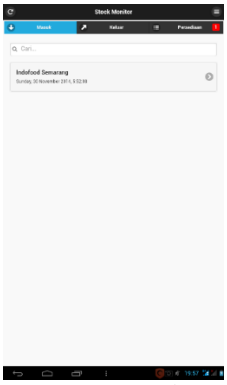
Pembuatan *Resource sharing* ini bertujuan agar aplikasi yang dibuat dapat menggunakan sumberdaya yang ada pada Sistem Informasi sehingga basisdatanya akan terpusat atau terintegrasi. Langkah pertama dalam membuat *Resource Sharing* ini adalah mengijinkan *Webserver* diakses oleh aplikasi dengan cara mengaktifkan *Cross Origin Resource Sharing*(CORS).

- Pembuatan Aplikasi

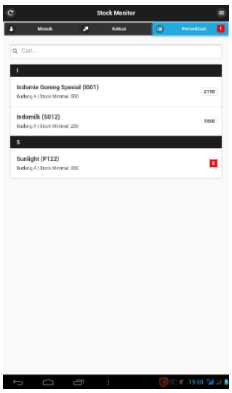
Pada Gambar 60 merupakan tampilan dari Halaman *Login*, selain itu nantinya ketika tombol *menu native android* ditekan maka akan muncul *menu* yang berfungsi untuk menutup aplikasi.

Gambar 60 Tampilan setelah loadUrl.

Pada *menu* Masuk menampilkan daftar Nota Masuk seperti pada Gambar 61, *menu* Keluar menampilkan daftar Nota Keluar seperti pada Gambar 62, sedangkan *menu* Persediaan menampilkan jumlah persediaan barang seperti yang ada pada Gambar 63.

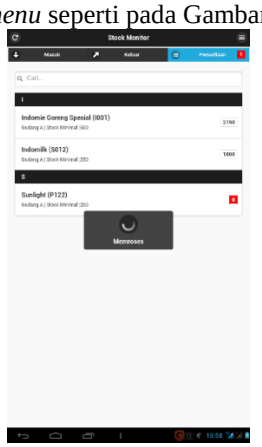


Gambar 61 Tampilan Daftar Barang.

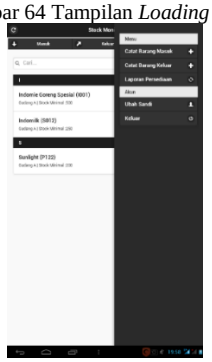


Gambar 63 Tampilan Data Persediaan Barang.

Proses pengambilan data ulang ditunjukkan pada Gambar 64. Tampilan *Sidemenu* seperti pada Gambar 65.



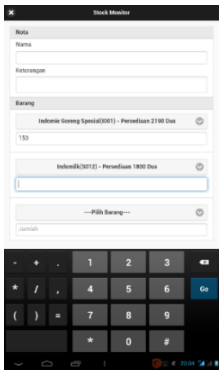
Gambar 64 Tampilan *Loading* Data.



Gambar 65 Tampilan *Sidemenu*.

Tampilan dari halaman masukkan dapat dilihat pada Gambar 66 hingga Gambar 68. Gambar 66 merupakan Masukkan Barang, pada Gambar 67 dan 68 menunjukkan ketika pengguna mengisi kolom masukkan.

Gambar 66 Tampilan Masukkan Barang Masuk.



Gambar 67 Tampilan Masukkan Barang.

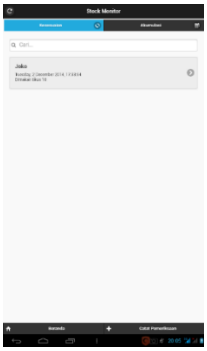
Gambar 68 Tampilan Pilihan Barang.

Pada Gambar 69 merupakan tampilan dari *Pop-up* yang masukkan datanya sudah benar, sedangkan pada Gambar 70 merupakan tampilan dari pesan *error* karena data yang dimasukkan belum sesuai.

Gambar 69 Tampilan *Pop-up* konfirmasi simpan data.

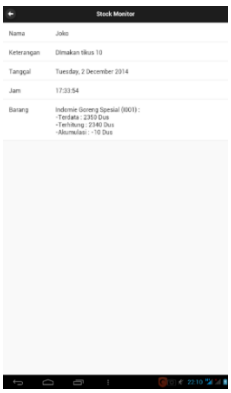
Gambar 70 Tampilan Pesan ketika *Field* Nama dikosongkan.

Pada Gambar 71 ditunjukkan Tampilan Halaman Laporan Persediaan.



Gambar 71 Tampilan Halaman Laporan Persediaan.

Tampilan dari halaman “Rincian” dapat dilihat pada Gambar 72.



Gambar 72 Tampilan Halaman Rincian.

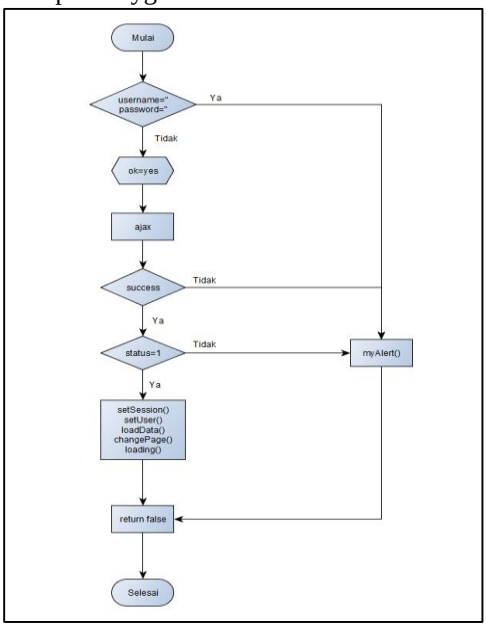
Pada Gambar 73 merupakan Tampilan *Pop-up* ketika *Menu* Ubah Kata Sandi ditekan.

Gambar 73 Tampilan *Pop-up* Ubah Kata Sandi.

B. Pengujian

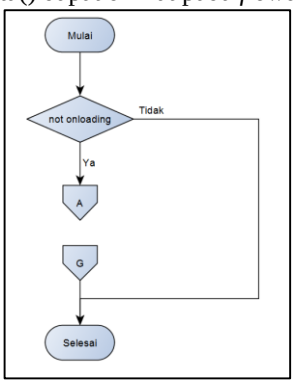
Pengujian pada pembuatan aplikasi ini akan menggunakan pengujian *white box*, dimana setiap fungsi yang ada akan diuji secara rinci jalur-jalur logika yang ada. Metode pengujiannya menggunakan Struktur Kontrol Program untuk memperoleh kasus uji. Pengujian akan dilakukan untuk masing-masing jalur, cabang dan perintah.

Fungsi *login*, terdapat beberapa fungsi yang digunakan pada saat pengguna melakukan *login*, yaitu fungsi pemeriksaan yang ada pada sisi Aplikasi, fungsi proses yang ada sisi *webserver* serta fungsi pemeriksaan hasil dari proses *login* yang telah dilakukan pada sisi Aplikasi, *flowchart* seperti pada Gambar 74 menunjukkan proses yg dilakukan.



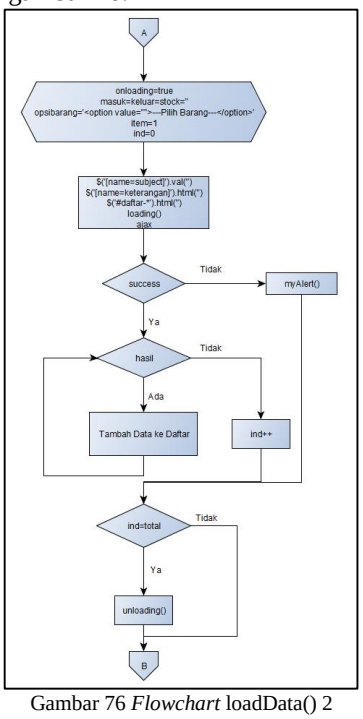
Gambar 74 *Flowchart* proses *login* sisi Pengguna

Proses loadData() dapat dilihat pada *flowchart* gambar 75.



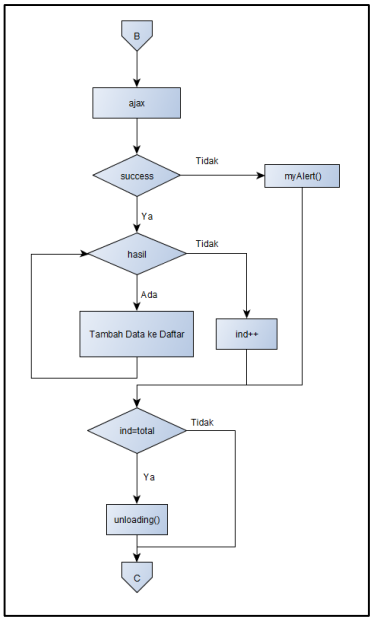
Gambar 75 *Flowchart* loadData() 1

Pada *flowchart* gambar 75 terlihat fungsi akan mengecek terlebih dahulu apakah aplikasi sedang melakukan *loading* atau tidak, setelah itu dilanjutkan ke proses pertama yang terlihat pada *flowchart* gambar 76.

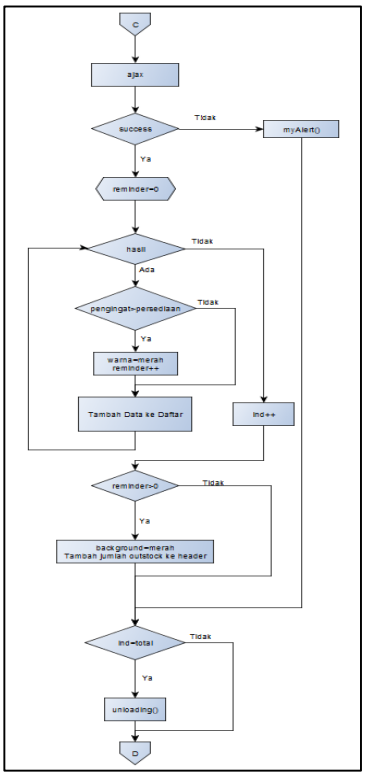


Gambar 76 *Flowchart* loadData() 2

Pada Gambar 77 terdapat tiga fungsi ajax, yaitu barang keluar, persediaan barang dan pilihan barang yang digunakan untuk menghasilkan pilihan barang yang dinamis mengikuti basisdata pada saat memasukkan data.

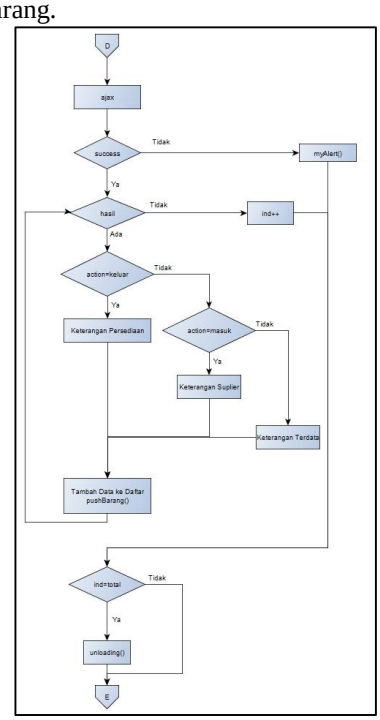


Gambar 77 *Flowchart* loadData() 3



Gambar 78 *Flowchart* loadData() 4

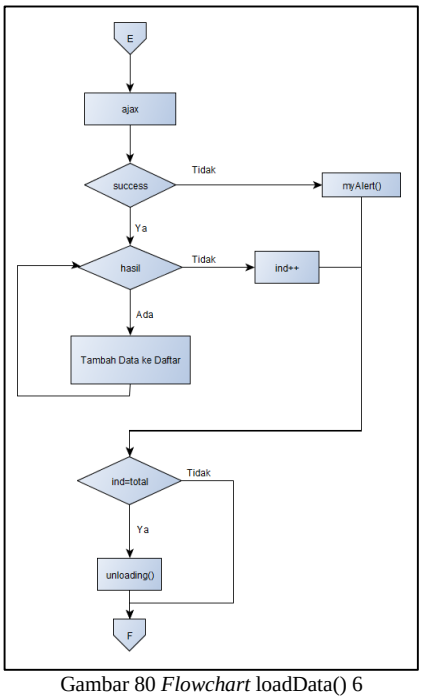
Pada *flowchart* berikutnya seperti pada Gambar 78 merupakan ajax yang digunakan untuk menambahkan ke daftar persediaan barang.



Gambar 79 *Flowchart* loadData() 5

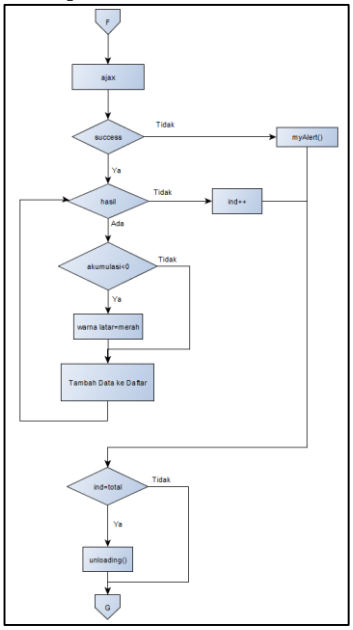
Fungsi ajax selanjutnya digunakan untuk mengambil data untuk membuat pilihan barang ketika memasukkan data barang masuk, keluar, serta laporan persediaan. Pada *flowchart* gambar 79 proses pengiriman ajax akan dilakukan ketika prosesnya sukses maka akan dijalankan perulangan untuk menambahkan data ke dalam daftar.

Pada *flowchart* gambar 80 menunjukkan proses yang akan dilakukan oleh fungsi ajax laporan persediaan.



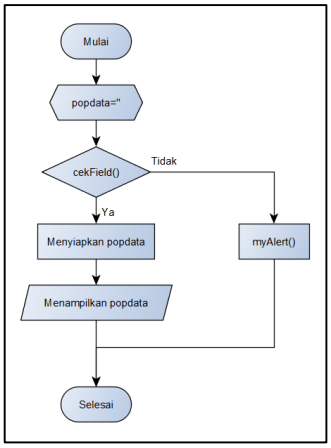
Gambar 80 *Flowchart* loadData() 6

Pada gambar 81 menunjukkan proses yang akan dilakukan, proses yang terjadi mirip dengan proses yang ada pada fungsi ajax persediaan barang, perbedaannya adalah pada fungsi ajax ini header akumulasi persediaan tidak akan diubah.



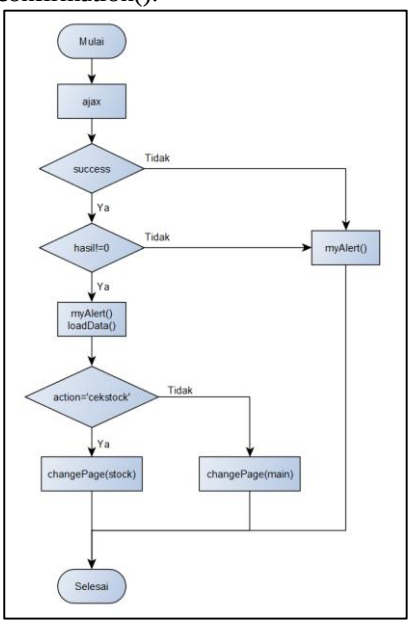
Gambar 81 *Flowchart* loadData() 7

Pada Gambar 82 menunjukkan bagaimana proses fungsi createPop() berjalan.

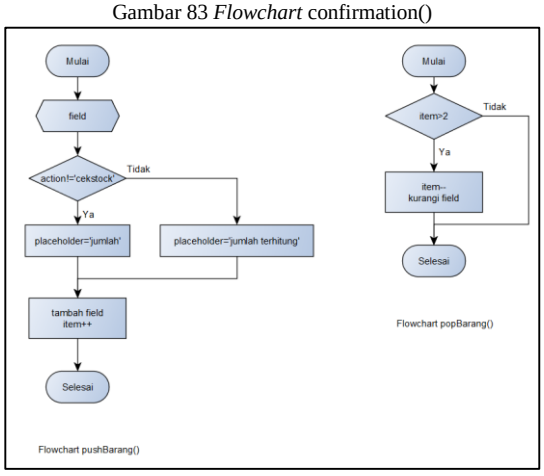


Gambar 82 *Flowchart* createPop()

Pada gambar 83 menunjukkan proses yang akan dijalankan pada fungsi confirmation().



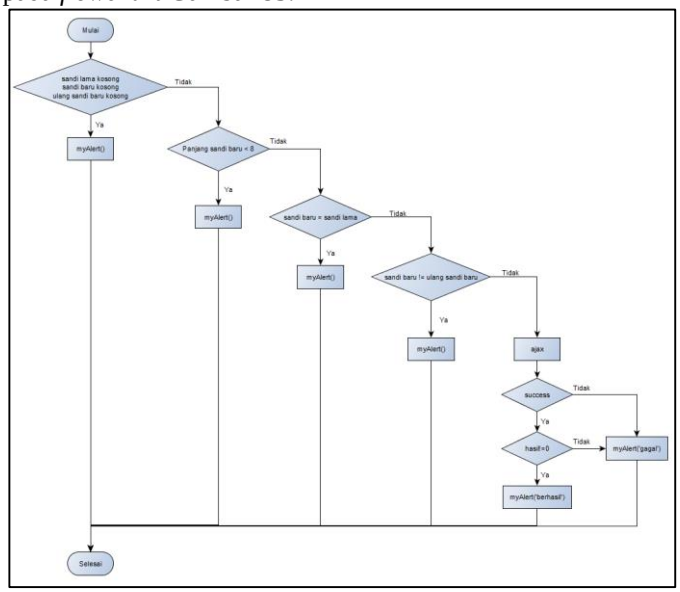
Gambar 83 *Flowchart* confirmation()



Gambar 84 *Flowchart* pushBarang() dan popBarang()

Pada gambar 84 kiri yang merupakan *flowchart* pushBarang(), sedangkan pada *flowchart* kanan gambar 84 menunjukkan proses yang terjadi ketika terjadi popBarang().

Fungsi terakhir adalah fungsi gantiPass(), ini dapat dilihat pada *flowchart* Gambar 85.



Gambar 85 *Flowchart* gantiPass()

KESIMPULAN

Dari hasil pengujian dan analisis pendataan barang di gudang berbasis Android dapat disimpulkan hal-hal sebagai berikut:

1. Pembuatan sistem terintegrasi pendataan barang di gudang dapat digunakan untuk membantu proses pendataan barang.
2. Pembuatan *view* pada basisdata memberikan kemudahan dalam melakukan perancangan suatu sistem, karena dengan *view* penyediaan informasi yang dibutuhkan oleh sistem diolah sekali tanpa perlu mengolahnya dalam aplikasi.
3. Pembuatan sistem terintegrasi yang dinamis sangat diperlukan sehingga ketika membuat suatu aplikasi terintegrasi, aplikasi tersebut dapat diimplementasikan diberbagai sistem yang berbeda tanpa harus melakukan pengubahan secara drastis terhadap sistem yang sudah ada.
4. Pembuatan aplikasi berbasis *Mobile* dengan menggunakan *WebView* memberikan kemudahan bagi pengembang dalam membuat suatu aplikasi berbasis *mobile*, karena dengan pemanfaatan *WebView* aplikasi yang dikembangkan dapat langsung diberikan karakteristik yang dimiliki setiap *platform* yang berbeda sehingga dalam mengembangkan aplikasi ini hanya cukup membuat satu aplikasi yang dapat dipindahkan langsung baik dari iOs ke Android, maupun sebaliknya dari Android ke iOs, atau bahkan *Mobile Web Application* yang ditempatkan pada *Webserver*.
5. Sistem terintegrasi pendataan barang di gudang bermanfaat dalam membantu proses pendataan barang, yang memberikan efisiensi waktu karena pengubahan data pada semua sistem secara *realtime*.
6. Sistem terintegrasi pendataan barang di gudang memberikan efektifitas kerja kepada *Operator* karena dengan data yang terintegrasi, informasi yang dibutuhkan oleh *Operator* dapat langsung dilihat.

SARAN

Saran yang diberikan dalam upaya pengembangan aplikasi yang lebih baik dikemudian hari.

1. Penambahan fitur *barcode* untuk masukkan data barang.
2. Penambahan fitur ubah data yang melibatkan Operator Sistem Informasi dan Operator di Gudang dengan tujuan memberikan keamanan data dari pengubahan data yang tidak bertanggung jawab.
3. Penambahan fitur obrolan dan pesan untuk memberikan kemudahan dalam melakukan komunikasi dari Gudang dan Pusat.
4. Pengubahan fungsi "Kurangi" pada saat memasukkan data, dengan memberikan fungsi hapus per *field* sehingga ketika data pada bagian awal ingin dihapus tidak perlu mengubah-ubah data lainnya.
5. Penambahan fitur *multi* masukkan sehingga pengguna dapat menambahkan data beberapa nota berbeda dalam satu waktu

DAFTAR PUSTAKA

Ariona, R., 2013. *Belajar HTML dan CSS - Tutorial Fundamental dalam Mempelajari HTML dan CSS*,

ARIZA, K., 2014. Pembuatan Aplikasi Informasi Tagihan Listrik Berbasis Android. *Skripsi, Fakultas Ilmu Komputer*, p.3. Available at: http://eprints.dinus.ac.id/13343/ [Accessed November 23, 2014].

Awalin, R.A.N., 2012. Analisis dan Perancangan Sistem Informasi Perpustakaan Berbasis Java di SD Negeri Jebeng. , pp.5–7.

Cahyono, H., 2014. Mengenal Java sebagai Pemrograman Berorientasi Objek dam Implementasi Thread di Lingkungan UNIX/LINUX. , 1(1), pp.2–11.

Dwiartara, L., 2012. Menyelam dan Menaklukan Samudra PHP. *Ilmuwebsite. com*, p.3. Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:menyelam+dan+menaklukan+samudra+php#0 [Accessed November 9, 2014].

Elisabeth, R. & Eric, F., 2012. *Head first HTML and CSS* 2nd ed., Available at: http://dl.acm.org/citation.cfm?id=2458651 [Accessed November 9, 2014].

Haviluddin, 2011. Memahami Penggunaan UML (Unified Modelling Language). , 6(1), p.2.

jquery.com, jQuery. Available at: http://jquery.com/ [Accessed November 15, 2014].

Jquerymobile.com, jQuery Mobile. Available at: http://jquerymobile.com/ [Accessed November 15, 2014].

Kristanti, T., 2012. Integrasi Enterprise (Studi Kasus: Yayasan Pendidikan "X"). *Jurnal Sistem Informasi*, 4, p.19. Available at: http://majour.maranatha.edu/index.php/jurnal-sistem-informasi/article/viewFile/515/pdf [Accessed November 23, 2014].

McLaughlin, B., 2012. *PHP & MySQL: The Missing Manual*, Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0 [Accessed November 15, 2014].

Morrison, M., 2007. *Head first javascript*, Available at: http://books.google.com/books?hl=en&lr=&id=xsmBjw814qwC&oi=fnd&pg=PR9&dq=Head+First+Javascript&ots=bqGRalwTcB&sig=VmwauvAc22uIrqGoHGD-IoMKS0Y [Accessed November 15, 2014].

Noertjahyana, A., 2002. Studi Analisis Rapid Application Development sebagai Perangkat Lunak. , 3(2), pp.74–79.

Sierra, K. & Bates, B., 2005. *Head First Java* 2nd ed., Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0 [Accessed November 9, 2014].

Simon, J., 2011. Head First Android Development. , p.v,vi. Available at: http://hydracortex1.torrenticity.com/torrent/9369097/Head_First_Android_Development.pdf [Accessed November 9, 2014].

Solichin, A., 2010. MySQL 5: Dari Pemula Hingga Mahir. *Jakarta: Achmatim. net*, pp.6–8,28,29,35,36. Available at: http://books.google.com/books?hl=en&lr=&id=HCNGBAAAQBAJ&oi=fnd&pg=PA3&dq=MySQL+-+Dari+Pemula+Hingga+Mahir&ots=FsFoUluhmW&sig=LCznHlnEiO_OzO1ui51BPdCCyic [Accessed November 15, 2014].

Solichin, A., 2005. Pemrograman Web dengan PHP dan MySQL. *Universitas Budi Luhur., Jakarta*, p.14,15. Available at: http://110.138.248.171/materi/Pemrograman Web dengan PHP MySQL.pdf [Accessed November 21, 2014].

Suehring, S., *JavaScript Step by Step* , *Second Edition* 2nd ed.,