PENGEMBANGAN SISTEM MANAJEMEN PERSEDIAAN BAHAN BAKU BERBASIS WEB (STUDI KASUS: RM. LALAPAN ALA DEWI)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sajarna Komputer

Disusun oleh:
Timoti Agape Siahaan
NIM: 155150201111399

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2021

PENGESAHAN

PENGEMBANGAN SISTEM MANAJEMEN PERSEDIAAN BAHAN BAKU BERBASIS WEB
(STUDI KASUS: RM. LALAPAN ALA DEWI)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh:
Timoti Agape Siahaan
NIM: 155150201111399

Skripsi ini telah diuji dan dinyatakan lulus pada
19 Maret 2021
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

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

Mengetahui
Ketua Jurusan Teknik Informatika

Achmad Basuki, S.T., M.MG., Ph.D.
NIP: 19741118 200312 1 002

PERNYATAAN ORISINALITAS

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

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

Malang, 19 Maret 2021

Timoti Agape Siahaan

NIM: 155150201111399

PRAKATA

Puji syukur kehadirat Tuhan YME yang telah melimpahkan rahmat-Nya sehingga laporan skripsi yang berjudul "Pengembangan Sistem Manajemen Persediaan Bahan Baku Berbasis Web (Studi Kasus: RM. Lalapan Ala Dewi)" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku Pembimbing skripsi yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini,
2. Kedua orang tua dan adik-adik penulis yang telah memberikan semangat dan kasih sayang yang tulus kepada penulis,
3. Deborah Florencia Gunawan yang selalu menyemangati penulis,
4. Teman-teman seperjuangan bimbingan skripsi,
5. Seluruh civitas akademika Jurusan Teknik Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Jurusan Teknik Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 19 Maret 2021

Timoti Agape Siahaan

timoti@student.ub.ac.id

ABSTRAK

Timoti Agape Siahaan, Pengembangan Sistem Manajemen Persediaan Bahan Baku Berbasis Web (Studi Kasus: RM. Lalapan Ala Dewi)

Pembimbing: Tri Astoto Kurniawan, S.T., M.T., Ph.D.

Manajemen persediaan Rumah Makan Lalapan Ala Dewi masih menggunakan catatan dan laporan yang konvensional. Jumlah cabang dan kebutuhan bahan baku setiap cabang menjadi kelemahan untuk cara konvensional pada saat mengelola persediaan bahan baku. Kelemahan tersebut mempengaruhi kinerja setiap cabang karena sering kehabisan persediaan bahan baku. Sistem manajemen persediaan bahan baku berbasis *web* diperlukan untuk membantu masalah tersebut. Sistem manajemen persediaan bahan baku akan mencatat persediaan bahan baku dan laporan setiap cabang. Sistem ini dirancangan dengan metode *waterfall model* dan menggunakan pendekatan berorientasi objek. Setelah sistem selesai dikembangkan maka sistem diuji dengan teknik pengujian *white-box testing* dan *black-box testing*. Hasil dari penelitian ini adalah sistem manajemen persediaan bahan baku

Kata kunci: rumah makan, manajemen persediaan, *waterfall model*, pendekatan berorientasi objek

ABSTRACT

Timoti Agape Siahaan, *Development of Web-Based Raw Material Inventory Management System (Case Study: RM. Lalapan Ala Dewi)*

***Supervisor*: Tri Astoto Kurniawan, S.T., M.T., Ph.D.**

Lalapan Ala Dewi Restaurant inventory management still uses conventional notes and reports. The number of branches and the need for raw ingredients of each branch becomes a weakness for the conventional way when managing the supply of raw ingredients. Such weaknesses affect the performance of each branch because it often runs out of raw ingredient supplies. A web-based raw ingredient inventory management system is needed to help with the problem. The raw ingredient inventory management system will record the raw ingredient inventory and report of each branch. The system is designed with a waterfall model method and uses an object-oriented approach. After the system was developed, the system was tested with white-box testing techniques and black-box testing techniques. The result of this research was raw ingredient inventory management system.

Keywords: restaurant, inventory management, waterfall model, object-oriented approach

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Pada era globalisasi saat ini kegiatan komersial mulai dilakukan oleh setiap elemen masyarakat untuk mendapatkan suatu keuntungan. Komersial adalah sesuatu yang memungkinkan seseorang untuk menarik keuntungan dari produk si pencipta (Hamilton, 2003). Semua yang bisa dikomersialkan akan digunakan untuk menggali pemasukan, salah satu misalnya rumah yang dikomersialkan menjadi rumah makan. Rumah makan adalah setiap tempat usaha komersial yang ruang lingkup kegiatannya menyediakan hidangan dan minuman untuk umum (SK Menteri Pariwisata, Pos dan Telekomunikasi).

Rumah makan membutuhkan sebuah manajemen persediaan dalam mempersiapkan makanan dan minuman yang akan disajikan. Setiap pengusaha pasti membutuhkan persediaan, tidak adanya persediaan para pengusaha bisa berhadapan dengan risiko tidak bisa memberikan kemauan para pelanggannya (Handoko, 1999). Jika rumah makan tidak memiliki manajemen persediaan yang tepat maka keuntungan yang diperoleh tidak maksimal karena muncul biaya-biaya lain. Setiap pengusaha yang menanamkan terlalu banyak dana dalam persediaan akan menyebabkan biaya penyimpanan yang berlebihan. Akan tetapi biaya tak terduga dari kekurangan bahan juga dapat terjadi jika pengusaha tidak mempunyai persediaan yang mencukupi (Handoko, 2000).

Objek studi kasus manajemen persediaan bahan baku pada penelitian ini adalah Rumah Makan Lalapan ala Dewi. Berdasarkan hasil observasi dan pengamatan di lapangan, Rumah Makan Lalapan ala Dewi memiliki beberapa cabang di sekitar Kota Malang dengan jarak kurang dari satu kilometer. Proses transaksi penjualan, pembelian, dan pengelolaan persediaan masih dilakukan secara konvensional atau ditulis pada buku catatan. Permasalahannya terjadi pada penyimpanan dan cara transaksi, belum adanya pencatatan langsung mengenai jumlah persediaan makanan yang ada dan yang keluar di setiap cabang. Persediaan makanan dikelola dan diatur oleh setiap cabang itu sendiri sehingga seringkali terjadi tumpang tindih persediaan makanan, bahan baku yang lebih dan tidak digunakan akhirnya rusak dan menyebabkan kerugian pada rumah makan. Perbedaan jumlah persediaan makanan antar cabang juga menyebabkan kesulitan dalam pengadaan bahan baku karena jumlah makanan yang dibutuhkan setiap cabang berbeda. Variabel yang menyebabkan terjadinya perbedaan jumlah persediaan bahan baku pada setiap cabang yaitu lokasi cabang. Akan tetapi variabel yang menyebabkan terjadinya perbedaan jumlah pengadaan bahan baku pada setiap cabang yaitu hari dan bulan. Selain itu untuk mengontrol dan mengecek setiap cabang pemilik harus datang ke cabang tersebut, sehingga pengecekan dan kontrol rutin susah dilakukan karena kendala waktu dan jarak.

Dari permasalahan yang ada yaitu pencatatan secara konvensional untuk laporan bahan baku dan perbedaan kebutuhan bahan baku setiap cabang, maka pemilik membutuhkan sistem pencatatan secara digital serta informasi jumlah

kebutuhan bahan baku setiap cabang. Penelitian ini akan mengembangkan sistem agar pemilik dapat mengelola persediaan rumah makannya dan menerima saran manajemen persediaan bahan baku secara online menggunakan aplikasi web.

1.2 Rumusan Masalah

Berdasarkan latar belakang, maka dapat disimpulkan rumusan masalah sebagai berikut:

1. Apa saja kebutuhan perangkat lunak pada Sistem Manajamen Persediaan Bahan Baku Pada Sebuah Rumah Makan Yang Lokasinya Tersebar (Studi Kasus: RM. Lalapan ala Dewi)?
2. Bagaimana rancangan perangkat lunak pada Sistem Manajamen Persediaan Bahan Baku Pada Sebuah Rumah Makan Yang Lokasinya Tersebar (Studi Kasus: RM. Lalapan ala Dewi)?
3. Bagaimana hasil implementasi perangkat lunak pada Sistem Manajamen Persediaan Bahan Baku Pada Sebuah Rumah Makan Yang Lokasinya Tersebar (Studi Kasus: RM. Lalapan ala Dewi)?
4. Bagaimana hasil pengujian perangkat lunak pada Sistem Manajamen Persediaan Bahan Baku Pada Sebuah Rumah Makan Yang Lokasinya Tersebar (Studi Kasus: RM. Lalapan ala Dewi)?

1.3 Tujuan

Tujuan dilaksanakannya penelitian ini adalah sebagai berikut:

1. Mengidentifikasi kebutuhan perangkat lunak untuk Sistem Manajamen Persediaan Bahan Baku Pada Sebuah Rumah Makan Yang Lokasinya Tersebar (Studi Kasus: RM. Lalapan ala Dewi)
2. Merancang perangkat lunak sesuai dengan analisis kebutuhan untuk Sistem Manajamen Persediaan Bahan Baku Pada Sebuah Rumah Makan Yang Lokasinya Tersebar (Studi Kasus: RM. Lalapan ala Dewi)
3. Mengimplementasikan rancangan perangkat lunak untuk Sistem Manajamen Persediaan Bahan Baku Pada Sebuah Rumah Makan Yang Lokasinya Tersebar (Studi Kasus: RM. Lalapan ala Dewi)
4. Menguji Sistem Manajamen Persediaan Bahan Baku Pada Sebuah Rumah Makan Yang Lokasinya Tersebar (Studi Kasus: RM. Lalapan ala Dewi) secara fungsional

1.4 Manfaat

Adapun manfaat dilaksanakannya penelitian ini bagi Mahasiswa yaitu menerapkan ilmu yang ditekuni saat di bangku perkuliahan dalam penelitian yang sedang dilakukan. Meningkatkan keterampilan dan kreativitas sesuai ilmu yang ditekuni. Menambah wawasan dan pengetahuan saat sedang melakukan penelitian. Bagi Rumah Makan Lalapan ala Dewi yaitu membantu pemilik cabang untuk mengetahui setiap transaksi dan jumlah persediaan makanan secara digital.

1.5 Batasan Masalah

Sistem ini dikembangkan untuk membantu penyajian dan pengolahan data secara digital, termasuk setiap transaksi yang dilakukan setiap cabang. Sistem juga mencatat jumlah setiap persediaan makanan yang masuk ataupun yang keluar dan sitem akan memberikan informasi setiap menu yang tersedia di setiap cabang kepada pembeli. Sistem yang akan dikembangkan menggunakan bahasa pemrograman PHP, HTML dan Javascript. Sementara untuk penyimpanan data menggunakan MySQL.

1.6 Sistematika Pembahasan

Adapun sistematika penyusunan laporan penelitian ini meliputi:

BAB 1 PENDAHULUAN

Menguraikan mengenai latar belakang penelitian, rumusan masalah, tujuan, manfaat penelitian, serta sistematika pembahasan laporan penelitian.

BAB 2 LANDASAN KEPUSTAKAAN

Menguraikan teori-teori yang menjadi referensi dalam melakukan penelitian. Teori-teori yang diambil akan digunakan sebagai landasan penelitian.

BAB 3 METODOLOGI PENELITIAN

Menguraikan tentang metodologi yang digunakan dalam penelitian serta alur kerja yang akan digunakan selama melakukan penelitian. Alur kerja akan dimulai dari Studi Literatur, Analisis Kebutuhan, Perancangan, Implementasi, Pengujian, Kesimpulan dan Saran.

BAB 4 ANALISIS KEBUTUHAN

Bab ini berisikan teknik analisis kebutuhan yang dilakukan oleh peneliti. Analisis yang dimaksud mulai dari deskripsi sistem, karakteristik pengguna dan kebutuhan fungsionalitas sistem dan *use case*.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Bab ini berisikan perancangan sistem yang terdiri dari perancangan *sequence diagram,* perancangan *class diagram*, perancangan basis data dan perancangan antarmuka. Implementasi sistem berisikan tampilan sistem setelah diubah ke bahasa pemrograman.

BAB 6 PENGUJIAN

Bab ini berisikan pengujian yang dilakukan oleh peneliti. Pengujian terdiri dari 2 bagian yaitu *white box testing* dan *black box testing*.

BAB 7 PENUTUP

Bab ini berisikan kesimpulan yang diperoleh dari implementasi dan pengujian sistem yang dikembangkan dalam penelitian ini serta saran-saran untuk penelitian berikutnya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Penelitian mengenai Sistem Manajemen Persediaan Bahan Baku juga pernah dilakukan oleh Davin (2016) yaitu "Sistem Informasi Penjualan Menu Makanan dan Persediaan Bahan Baku Berbasis Web pada Koffie Tijd Kafe Resto". Tujuan penelitian tersebut yaitu untuk membuat perancangan sistem informasi penjualan dan persediaan bahan baku dan pembuatan laporan menggunakan sistem yang berbasis komputer untuk memudahkan pihak Koffie Tijd Kafe Resto. Tujuan penelitian tersebut yaitu, sistem dapat membantu *user* dalam pembuatan laporan, sistem juga dapat mengurangi pemesanan yang sama kepada *supplier*, dan sistem dapat menyimpan faktur pemesanan yang sudah terkomputerisasi. Kelemahan penelitian tersebut yaitu, sistem tidak dapat melakukan pembatalan pemesanan, sistem juga tidak dapat membuat laporan *service*, dan sistem tidak dapat mengelola pembayaran secara debit atau kredit.

Penelitian serupa juga pernah dilakukan oleh Kurniawan (2015) yaitu "Sistem Aplikasi Persediaan Bahan Baku Penyusun Makanan di Rumah Makan Podo Joyo". Penelitian yang bertujuan untuk membuat sebuah sistem aplikasi pendapatan persediaan bahan baku untuk melakukan proses pendataan persediaan bahan baku. Proses pendataan tidak lagi dilakukan secara manual dan menerapkan sistem aplikasi pendataan persediaan bahan baku yang dapat digunakan oleh pemilik untuk melakukan proses *monitoring* terhadap usahanya. Kelebihan dari penelitian tersebut yaitu, sistem memberikan kemudahan kepada pemilik untuk mengontrol keluar masuknya bahan baku makanan dan sistem dapat memberikan informasi stok bahan baku yang ada dalam persediaan. Kelemahan dari penelitian tersebut yaitu, fasilitas yang disediakan sistem masih sederhana karena hanya menampilkan nama, menu dan rumah makan Podo Joyo.

Dari kedua penelitian yang telah diuraikan maka persamaan dengan penelitian ini yaitu, sistem melakukan pencatatan dan penyimpanan data persediaan bahan baku serta menampilkan jumlah persediaan bahan baku terkini. Perbedaan yang ada pada penelitian ini yaitu, sistem menggunakan pendekatan berorientasi objek dan metode pengembangan sistem menggunakan *waterfall* yang berbeda dari kedua penelitian tersebut. Sistem dalam penelitian ini menggunakan web sebagai *platform* pengembangan aplikasi sedangkan sistem yang lain menggunakan desktop sebagai *platform* pengembangan aplikasinya. Web dipilih sebagai platform sistem yang akan dikembangkan karena sistem dapat diakses dimana saja dan kapan saja sehingga mempercepat waktu pelayanan kepada pasien (Simarmata, 2010).

2.2 Manajemen Rumah Makan Lalapan Ala Dewi

Rumah makan adalah suatu tempat atau bangunan yang diorganisasikan secara komersial, yang menyelenggarakan pelayanan dengan baik kepada semua tamu, baik berupa kegiatan makan maupun minum (Marsum, 2005). Keberhasilan sebuah industri rumah makan sangat tergantung pada kepuasan pembeli yang memebeli produk makanan, disamping itu mutu produk yang tinggi dan mutu pelayanan juga penting untuk meningkatkan kepuasan pembeli.

2.2.1 Menu rumah makan lalapan ala dewi

Menu adalah daftar makanan yang telah dipersiapkan yang tersedia di dalam restoran tersebut (Sugiarto, 2001). Menu pada rumah makan memiliki dua sifat yaitu, statik menu dan *cycle* menu. Statik menu adalah menu yang diganti setiap enam bulan sekali atau lebih sedangkan *cycle* menu adalah menu yang diganti satu minggu sekali atau setiap hari. Menu pada Rumah Makan Lalapan ala Dewi merupakan menu statik dan daftar menu pada setiap cabang yang ada sama dengan cabang utama.

Daftar menu yang tersedia di Rumah Makan Lalapan ala Dewi adalah sebagai berikut:

1. Ayam goreng dada
2. Ayam goreng paha
3. Ayam *crispy* dada
4. Ayam *crispy* paha
5. Ayam bakar dada
6. Ayam bakar paha
7. Jamur goreng
8. Telur
9. Tahu/tempe
10. Nasi + Ayam goreng dada
11. Nasi + Ayam goreng paha
12. Nasi + Ayam *crispy* dada
13. Nasi + Ayam *crispy* paha
14. Nasi + Ayam bakar dada
15. Nasi + Ayam bakar paha
16. Nasi + Jamur goreng
17. Nasi + Telur
18. Nasi + Tahu/tempe

2.2.2 Bahan baku rumah makan lalapan ala dewi

Bahan baku merupakan bahan yang membentuk bagian menyeluruh produk jadi (Mulyadi, 1999). Bahan baku adalah bahan yang penting pada sebuah rumah makan karena disini terletak langkah awal dalam membuat proses produksi. Bahan baku dibedakan menjadi bahan baku langsung yang artinya bahan digunakan secara langsung pada proses produksi makanan dan bahan baku tidak langsung yang artinya bahan yang penting untuk memfasilitasi proses produksi. Bahan baku pada Rumah Makan Lalapan ala Dewi merupakan bahan baku langsung dan satu menu bisa terdiri lebih dari satu bahan baku.

Daftar bahan baku yang digunakan untuk membuat menu pada Rumah Makan Lalapan ala Dewi adalah sebagai berikut:

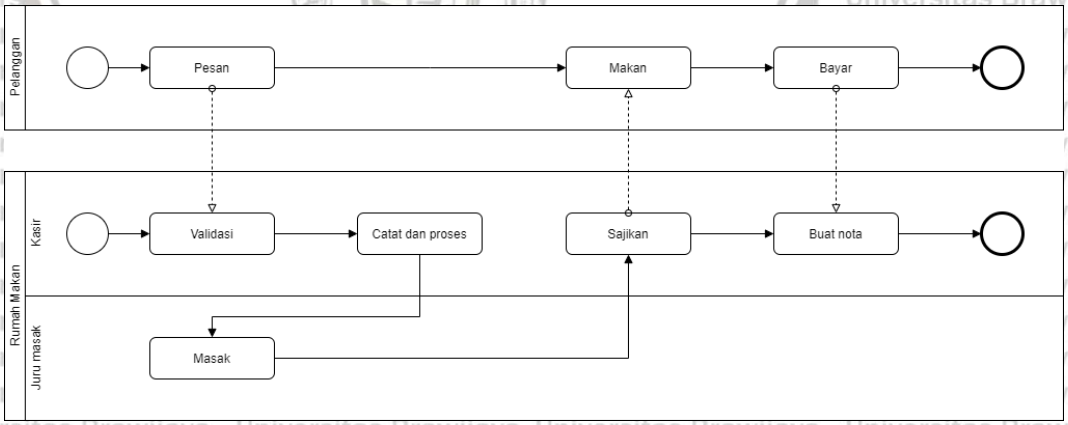
1. Ayam bagian dada
2. Ayam bagian paha
3. Jamur
4. Tahu dan tempe
5. Telur
6. Nasi

Setiap bahan baku akan disiapkan oleh pemilik rumah makan sesuai dengan kebutuhan setiap cabang. Pemilik rumah makan kemudian melakukan pembelian bahan baku kepada penyedia bahan baku, lalu membagikan bahan baku tersebut. Setiap kasir dari cabang rumah makan perlu datang ke cabang utama untuk mendapatkan bahan baku sehingga pemilik rumah makan mengetahui jumlah bahan baku setiap cabang.

2.2.3 Manajemen persediaan bahan baku rumah makan lalapan ala dewi

Persediaan merupakan bahan atau barang yang disimpan untuk tujuan tertentu, antara lain untuk proses produksi (Siagian, 2005). Jadi persediaan adalah stok dari sebuah sumber daya atau bahan material yang disediakan dan disimpan oleh pengusaha guna membantu proses bisnisnya dan menyediakan permintaan dari pelanggan atau konsumen. Mengendalikan persediaan yang cukup adalah hal yang sulit, jika jumlah persediaan terlalu banyak menyebabkan adanya dana menganggur yang banyak, bertambahnya biaya penyimpanan dan resiko barang rusak menjadi lebih besar.

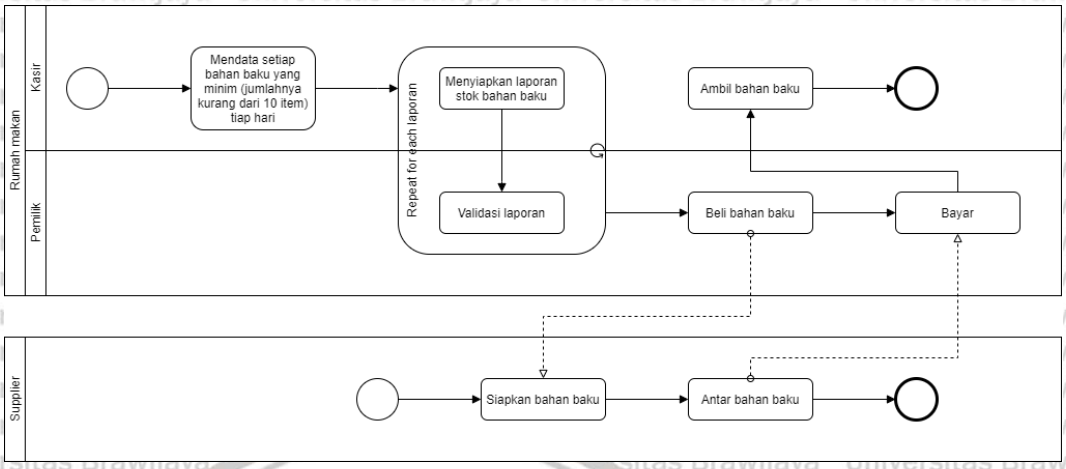
Manejemen persediaan bahan baku pada Rumah Makan Lalapan ala Dewi masih menggunakan cara konvensional karena pemilik harus menerima laporan dari cabang rumah makan. Pemilik kemudian menyiapkan bahan baku sesuai dengan kebutuhan cabang tersebut, jenis dan jumlah bahan baku yang diperlukan sesuai dengan laporan. Setiap laporan dari setiap cabang yang ada bisa saja berbeda sehingga pemilik harus menyediakan jenis dan jumlah bahan baku yang berbeda untuk setiap cabang.



Gambar 2.1 Proses bisnis pemesanan

Proses bisnis pada Rumah Makan Lalapan Ala Dewi dimulai dari pelanggan memesan menu yang tersedia pada salah satu cabang. Pesanan kemudian dicatat oleh kasir sebagai laporan transkasi, jika menu tidak tersedia maka kasir akan memberitahu pelanggan untuk mengganti pesanannya. Setelah pesanan siap

maka pesanan diberikan oleh kasir beserta nota jika diperlukan. Setiap catatan pesanan akan dikumpulkan menjadi laporan kepada pemilik.



Gambar 2.2 Proses bisnis penyediaan bahan baku

Proses penyediaan bahan baku pada Rumah Makan Lalapan ala Dewi dimulai dengan kasir mendata setiap bahan baku yang jumlahnya dibawah dari jumlah minimal yaitu sepuluh item, misalnya untuk bahan baku telur jumlah minimal adalah sepuluh biji telur. Bahan baku ayam terbagi dua bagian yaitu ayam bagian paha dan ayam bagian dada, sehingga untuk setiap bagian jumlah minimalnya juga sepuluh potong. Kemudian kasir membuat laporan stok bahan baku lalu divalidasi oleh pemilik rumah makan jika laporan sudah sesuai maka pemilik akan membeli bahan baku yang diperlukan, namun jika belum sesuai maka kasir perlu memperbaiki laporan stok bahan baku terlebih dahulu. Pemilik sudah memiliki penyedia bahan baku yang akan menyiapkan dan mengantar keperluan bahan baku, kemudian bahan baku tersebut akan diambil oleh kasir di cabang pusat.

2.3 Rekayasa Perangkat Lunak

Definisi rekayasa perangkat lunak adalah pembuatan dan penggunaan prinsip-prinsip keahlian teknik untuk mendapatkan perangkat lunak yang ekonomis, handal dan bekerja secara efisien pada mesin yang sesungguhnya (Pressman, 2001). Hal-hal yang menjadi dasar pembelajaran pada rekayasa perangkat lunak yaitu spesifikasi perangkat lunak, pengembangan perangkat lunak, validasi perangkat lunak dan evolusi perangkat lunak.

Menurut Sommerville (2011) pendekatan sistematis digunakan dalam rekayasa perangkat lunak yaitu urutan kegiatan yang mengarah pada pembuatan perangkat lunak. Urutan kegiatan tersebut adalah:

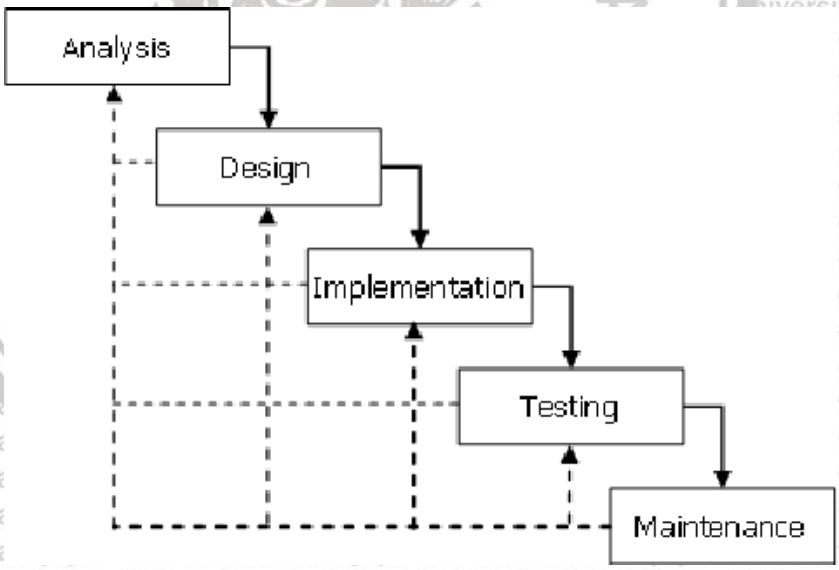
1. *Software specification*, tahapan dimana pengguna dan pengembang menentukan perangkat lunak yang akan dibangun dan batasannya
2. *Software development*, tahapan dimana perangkat lunak dirancang dan diimplementasikan.

3. *Software validation*, tahapan dimana perangkat lunak diperiksa untuk memastikan sesuai dengan kebutuhan pengguna.
4. *Software evolution*, tahapan dimana perangkat lunak dimodifikasi berdasarkan perubahan kebutuhan pengguna dan kebutuhan pasar.

Rekayasa perangkat lunak memiliki beberapa model pengembangan atau yang dikenal dengan *software development life cycle*. *Software development life cycle* adalah serangkaian kegiatan yang membentuk kerangka kerja untuk pengembangan perangkat lunak (Pressman, 2010). *Software development life cycle* terdiri dari *waterfall model*, *v-model*, *prototyping model*, *rapid application model*, *agile methodology* dan *extreme programming*. Penelitian ini menggunakan model pengembangan *Waterfall model* karena kebutuhan-kebutuhan perangkat lunak telah bisa didefinisikan sejak awal pengembangan. Kemudian pada perancangan sistem penelitian ini menggunakan pemodelan berbasis objek dan setelah sistem diimplementasi maka dilakukan pengujian terhadap sistem dengan *whitebox testing* dan *blackbox testing*.

2.3.1 *Waterfall model*

Waterfall model adalah model pengembangan perangkat lunak yang menggunakan pendekatan sistematis dan berurutan (Bassil, 2012). Hasil dari setiap tahapan dalam *waterfall model* merupakan input bagi tahap berikutnya dan setiap tahapan yang telah dilalui dapat diulang hingga sempurna.



Gambar 2.3 *Waterfall model*
Sumber: Bassil (2012)

Menurut Bassil (2012) tahapan *waterfall model* seperti yang diilustrasikan pada Gambar 2.2 terdiri dari:

1. *Analysis*

Tahap *analysis* merupakan tahap untuk mengumpulkan kebutuhan perangkat lunak (Bassil, 2012). Tahap *analysis* dilakukan agar kebutuhan yang

harus dipenuhi dapat disepakati oleh pemangku kepentingan dan menjadi dasar dari perancangan yang akan dilakukan. Proses yang dilakukan pada tahap *analysis* yaitu elisitasi kebutuhan, spesifikasi kebutuhan, dan validasi kebutuhan (Pressman 2010). Elisitasi kebutuhan adalah aktivitas untuk menggali kebutuhan melalui komunikasi dengan pelanggan, pengguna, dan pihak yang berkepentingan (Pressman, 2010). Elisitasi kebutuhan dapat dilakukan dengan teknik wawancara, kuesioner, observasi, pengamatan dokumen, *sampling*, dan *prototyping*. Spesifikasi kebutuhan adalah proses untuk memberi detail kebutuhan yang telah didefinisikan sebelumnya yang akan dijadikan dasar dari perancangan dan implementasi (Pressman, 2010). Spesifikasi kebutuhan dapat dibuat dalam sebuah dokumen *software requirements specification* (SRS). Menurut Pressman (2010), SRS adalah sebuah dokumen yang berisi deskripsi segala aspek dari sebuah perangkat lunak yang akan dibangun. Validasi kebutuhan adalah proses untuk memastikan bahwa spesifikasi kebutuhan telah tepat dan akurat (Pressman, 2010). Pada tahap *analysis* terdapat pemodelan kebutuhan perangkat lunak yang dilakukan untuk mempermudah memahami kebutuhan. Menurut Pressman (2010), pemodelan kebutuhan harus dapat menggambarkan apa yang dibutuhkan oleh pengguna, menjadi dasar dari perancangan, dan menjadi referensi dalam melakukan validasi.

2. *Design*

   Tahap *design* merupakan tahap membuat rancangan perangkat lunak (Bassil, 2012). Perancangan yang dilakukan meliputi perancangan algoritme, perancangan arsitektur, perancangan antarmuka, dan perancangan data. Perancangan algoritme dilakukan untuk merancang detail dari algoritme yang digunakan pada suatu klas (Pressman, 2010). Perancangan arsitektur dilakukan untuk merepresentasikan klas dan hubungan antar klas dalam struktur perangkat lunak (Pressman, 2010). Perancangan antarmuka dilakukan untuk merancang bagaimana informasi masuk dan keluar pada perangkat lunak dan bagaimana arus informasi tersebut berkomunikasi (Pressman, 2010). Perancangan data dilakukan untuk merancang struktur data yang diperlukan berdasarkan informasi dari hasil analisis kebutuhan untuk implementasi perangkat lunak (Pressman, 2010). Perancangan data dengan menggunakan basis data bisa dilakukan dengan pemodelan data yaitu, Conceptual *Data Model* (CDM) dan *Physical Data Model* (PDM). CDM merupakan model data yang berisi daftar entitas, atribut dan hubungan antar entitas (Guru99, 2020). PDM merupakan model data yang berisi daftar tabel, tipe data, panjang data, *primary key*, *foreign key* dan hubungan antar tabel (Guru99, 2020).

3. *Implementation*

   Tahap *implementation* merupakan tahap implementasi kebutuhan perangkat lunak menggunakan bahasa pemrograman (Bassil, 2012). Hasil dari tahap *implementation* ini berupa sebuah perangkat lunak yang siap untuk diuji dan digunakan.

4. *Testing*

Tahap *testing* atau yang diketahui sebagai tahap verifikasi dan validasi merupakan tahap pengujian perangkat lunak apakah sudah memenuhi tujuan yang dimaksudkan (Bassil, 2012). Tahap ini bertujuan untuk memastikan bahwa perangkat lunak yang telah diimplementasikan sesuai dengan spesifikasi kebutuhan dan tujuan yang diharapkan. Verifikasi merupakan proses evaluasi perangkat lunak untuk menentukan apakah perangkat lunak telah dikembangkan dengan tepat dan sesuai dengan perancangannya. Validasi merupakan proses evaluasi perangkat lunak yang dilakukan selama atau pada tahap akhir untuk menentukan apakah perangkat lunak telah sesuai dengan kebutuhan. Tahap pengujian akan dilakukan dengan menggunakan strategi pengujian. Menurut Pressman (2010), strategi pengujian adalah aktivitas untuk menjalankan pengujian yang terencana berdasarkan kasus uji yang dihasilkan dari teknik pengujian. Strategi pengujian yang dilakukan meliputi pengujian unit, pengujian integrasi, dan pengujian validasi. Teknik pengujian yang dilakukan yaitu *white-box testing* dan *black-box testing*. Pengujian unit dilakukan untuk menguji bagian terkecil dari perangkat lunak, yang meliputi komponen atau modul (Pressman, 2010). Pengujian integrasi bertujuan untuk menguji relasi antar unit (Pressman, 2010). Pengujian validasi berfokus pada input dan output untuk memastikan bahwa perangkat lunak telah sesuai dengan kebutuhan yang telah ditetapkan (Pressman, 2010).

a. *White-box tesing*

White-box testing adalah pengujian yang berfokus untuk menguji struktur internal dari perangkat lunak (Pressman, 2010). Salah satu metode dalam teknik ini yaitu *basis path testing*. *Basis path testing* dilakukan untuk mendapat nilai kompleksitas dari algoritme yang dirancang (Pressman, 2010). *Basis path testing* dilakukan dengan mengubah kode program menjadi *pseudocode* dan digambarkan dengan *flow graph*. Setelah *flow graph* dibuat, akan dihitung nilai dari *cyclomatic complexity* untuk mendapat jumlah jalur independennya. Menurut Pressman (2010) nilai *cyclomatic complexity* dapat dihitung dengan tiga cara:

1. Menghitung total region pada *flow graph*
2. Jumlah *edge* – jumlah *node* + 2
3. *Predicate node* + 2

Nilai kompleksitas yang semakin tinggi akan meningkatkan probabilitas *error* yang akan ditemukan (Pressman, 2010). Nilai kompleksitas akan menentukan berapa kasus aja yang akan digunakan. Kasus uji akan didapatkan dengan melakukan identifikasi terhadap *pseudocode* yang telah dirancang sebelumnya.

b. *Black-box testing*

Black box testing adalah pengujian yang berfokus untuk menguji fungsionalitas perangkat lunak (Pressman, 2010). *Black box testing*

digunakan saat pengujian unit, integrasi dan validasi. *Equivalence partitioning, boundary value analysis dan scenario-based testing* merupakan jenis-jenis *black box testing*. *Equivalence partitioning* adalah salah satu metode pengujian *black box* yang membagi domain masukan program menjadi klas-klas data dimana kasus uji nantinya akan diturunkan (Pressman, 2010). *Boundary value analysis* adalah salah satu metode pengujian *black box* yang menguji nilai-nilai yang terdapat di perbatasan (Pressman, 2010). *Scenario-based testing* adalah salah satu metode pengujian *black box* untuk mengetahui cara berpikir pengguna bukan cara kerja perangkat lunak (Pressman, 2010). Skenario dirancang untuk menemukan kemungkinan-kemungkinan interaksi pengguna dengan perangkat lunak.

5. *Maintenance*

Tahap *maintenance* merupakan tahap pemeliharaan perangkat lunak yang telah digunakan untuk memperbaiki kesalahan perangkat lunak. *Maintenance* merupakan tahap terakhir setelah dilakukan analisis, perancangan, implementasi serta pengujian. *Maintenance* tambahan dapat dilakukan pada tahap ini untuk mengakomodasi kebutuhan baru dari pemangku kepentingan.

2.3.2 Pendekatan berorientasi objek

Pendekatan beorientasi objek merupakan salah satu pendekatan dalam mengembangkan perangkat lunak dengan memandang informasi yang ada sebagai sekumpulan objek yang memiliki atribut dan perilaku tertentu (Pressman, 2010). Menurut Pressman (2010), pendekatan berorientasi objek dapat diterapkan pada tahap analisis, perancangan, implementasi, dan pengujian perangkat lunak. Tahap analisis atau disebut dengan tahap *Object Oriented Analysis* (OOA) merupakan sebuah proses untuk melakukan analisis terhadap kebutuhan-kebutuhan perangkat lunak dan mendefinisikan objek-objek yang ada berdasarkan domain masalah. OOA dilakukan dengan mendaftar kebutuhan perangkat lunak yang akan dikembangkan serta aktor yang terlibat. Tahap perancangan atau disebut dengan *Object Oriented Design* (OOD) merupakan tahap merancang perangkat lunak berdasarkan objek-objek yang terdapat pada model analisis. Perancangan meliputi perancangan arsitektur, perancangan komponen, perancangan data dan perancangan antarmuka. Tahap implementasi atau disebut dengan tahap *Object Oriented Programming* (OOP) merupakan tahap implementasi kode dengan menggunakan bahasa pemograman berorientasi objek. Tahap pengujian atau disebut dengan tahap *Object Oriented Testing* (OOT) merupakan tahap untuk menguji kualitas perangkat lunak berdasarkan hasil analisis dan perancangan.

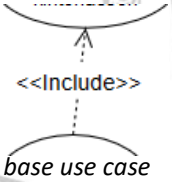
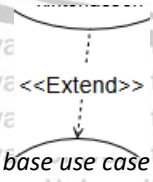
OOA dan OOD dimodelkan menggunakan *unified modeling language* (UML) meliputi *use case diagram*, *use case scenario*, *sequence diagram* serta *class diagram* (Pressman, 2010). Menurut Rumbaugh, et al. (2005) UML adalah sebuah standar untuk memvisualisasikan, mengidentifikasi, membangun dan

mendokumentasikan artefak perangkat lunak. Pengembangan perangkat lunak ini akan menggunakan tiga diagram UML yaitu:

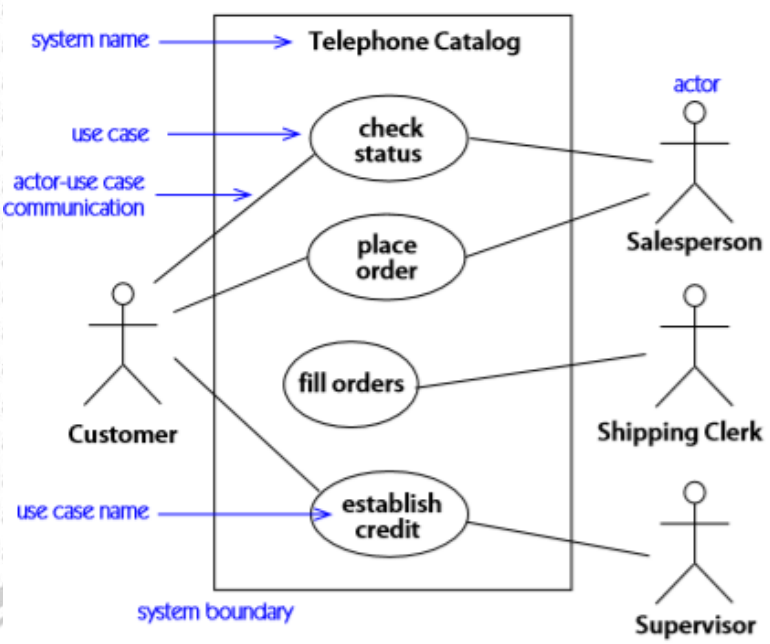
1. *Use case diagram*

Use case diagram adalah diagram yang merepresentasikan interaksi aktor dengan sistem berdasarkan kebutuhan fungsional yang telah didefinisikan (Rumbaugh, Jacobson, & Booch, 2005). *Use case diagram* menyediakan fungsi-fungsi yang harus dipenuhi sistem dengan aktornya. Notasi *use case diagram* dapat dilihat pada Tabel 2.1 dan gambar *use case diagram* dapat dilihat pada Gambar2.3.

Tabel 2.1 Notasi *use case diagram*

Notasi	Keterangan
	*Actor* adalah representasi pengguna atau sistem lain yang berinteraksi dengan sistem yang dikembangkan.
	*Use case* adalah fungsional yang tersedia pada perangkat lunak.
	Association adalah relasi yang memperlihatkan interaksi *actor* dengan *use case* secara umum.
	*Include* adalah hubungan yang menunjukkan *base use case* selalu memerlukan *include use case* agar *base use case* dapat dipenuhi.
	*Extend* adalah hubungan yang menunjukkan *base use case* mampu diperluas oleh *extend use case*. *Extend* bersifat *optional*.
	Generalization merupakan relasi yang menunjukkan *actor parent* lebih umum dan *actor child* lebih spesifik.

Sumber: Rumbaugh, Jacobson, & Booch (2005)



Gambar 2.4 *Use case diagram*

Sumber: Rumbaugh, Jacobson, & Booch (2005)

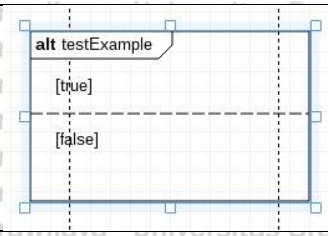
Suatu *use case* akan dijelaskan lebih detail dalam *use case scenario*. Menurut Kurniawan (2018*), use case scenario* adalah deskripsi urutan aksi atau langkah yang dilakukan aktor ketika berinteraksi dengan sistem, baik yang berhasil maupun gagal. Menurut Kurniawan (2018), *use case scenario* memiliki beberapa bagian penting, yaitu:

a. Aktor primer, yaitu aktor yang menginisiasi layanan sistem untuk mencapai tujuan dari aktor tersebut.
b. Prakondisi, yaitu kondisi spesifik yang harus terpenuhi sebelum use case dapat dieksekusi oleh aktor primer.
c. Alur utama, yaitu jalur yang mengarahkan pada skenario berhasil sehingga tujuan aktor tercapai.
d. Alur alternatif, yaitu jalur alternatif dari interaksi yang terjadi antara aktor dengan sistem yang mencakup pencabangan maupun skenario yang gagal sehingga tujuan aktor tidak terpenuhi.
e. Kondisi akhir, yaitu kondisi spesifik yang harus terjadi ketika use case berhasil dijalankan secara lengkap.

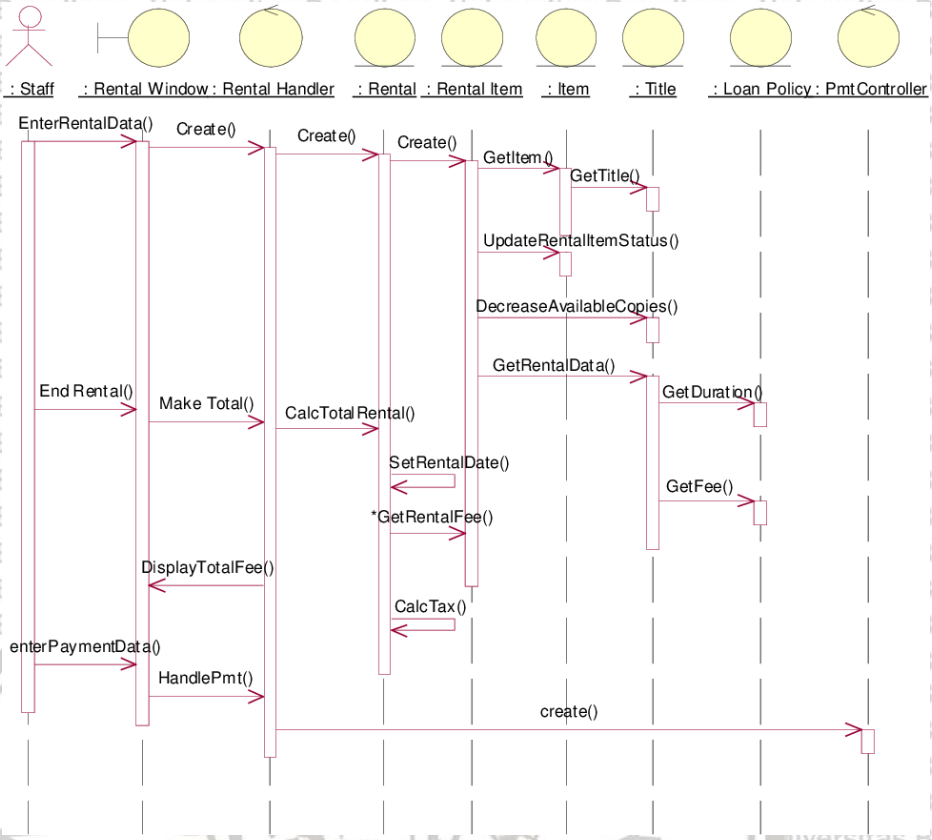
2. *Sequence diagram*

Sequence diagram adalah diagram yang merepresentasikan interaksi antar objek dengan memperlihatkan aliran pesan antar objek tersebut (Rumbaugh, Jacobson, & Booch, 2005). *Sequence diagram* menunjukkan urutan interaksi saat pengguna mengeksekusi fungsi perangkat lunak (Sommerville, 2011). Notasi *sequence diagram* dapat dilihat pada Tabel 2.2 dan gambar *sequence diagram* dapat dilihat pada Gambar 2.4.

Tabel 2.2 Notasi *sequence diagram*

Notasi	Keterangan
	*Actor* adalah representasi pengguna atau sistem lain yang berinteraksi dengan sistem yang dikembangkan.
	*Boundary Object* merupakan bagian dari perangkat lunak yang berinteraksi dengan aktor secara langsung.
	*Control Object* akan menggambarkan alur logika dari perangkat lunak dan menjadi penghubung antara *boundary object* dan *entity object*.
	*Entity Object* akan menggambarkan bagian dari perangkat lunak yang berfungsi untuk mengelola informasi.
	Lifeline akan merepresentasikan eksistensi dari objek dalam berinteraksi dengan objek lainnya.
	Message akan merepresentasikan komunikasi berupa operasi pada suatu objek.
	Return akan mendefinisikan pesan kembalian suatu objek berdasarkan pesan sebelumnya.
	*Alternative Combined Fragment* merupakan sebuah fragmen yang menggambarkan mengenai kondisi opsional atau *alternative* dari suatu proses yang ada.

Sumber: Song (2001)



Gambar 2.5 *Sequence diagram*

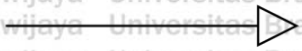
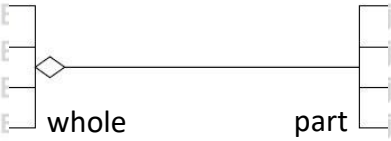
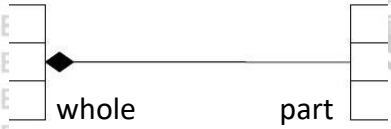
Sumber: Song (2001)

3. *Class diagram*

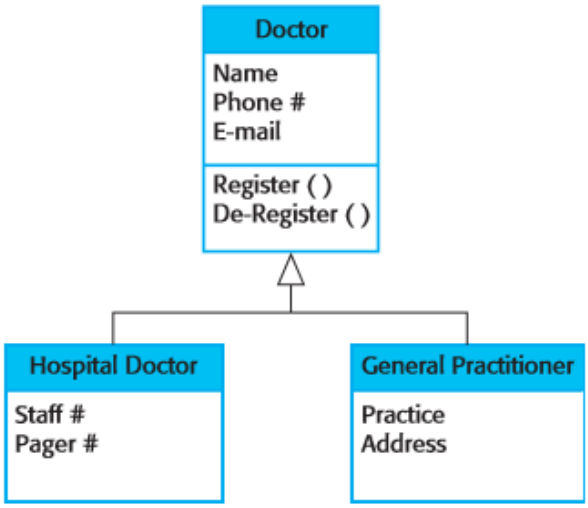
Class diagram adalah diagram yang merepresentasikan hubungan dan ketergantungan antar klas dalam perangkat lunak (Rumbaugh, Jacobson, & Booch, 2005). *Class diagram* terdiri dari klas dan relasi. Notasi *class diagram* dapat dilihat pada Tabel 2.3 dan gambar *class diagram* dapat dilihat pada Gambar 2.5.

Tabel 2.3 Notasi *class diagram*

Notasi	Keterangan
Class Attribute Operation	*Class* merupakan *blue print* dari objek yang berisi atribut dan operasi.
	Association merupakan hubungan antar klas yang menggambarkan suatu klas memiliki instance dari klas lain.

	*Generalization* merupakan hubungan antar klas yang menunjukkan klas induk merupakan klas yang lebih umum dan klas anak merupakan klas yang lebih spesifik.
	*Aggregation* merupakan hubungan yang menggambarkan *part class* bagian dari *whole class* dan setiap objek dari *part class* tidak memiliki ketergantungan siklus hidup.
	*Composition* merupakan hubungan yang menggambarkan *part class* bagian dari *whole class* dan setiap objek dari *whole class* memiliki ketergantungan siklus hidup.

Sumber: Rumbaugh, Jacobson, & Booch (2005)



Gambar 2.6 *Class diagram*

Sumber: Sommerville (2011)

Class diagram yang ditunjukkan pada Gambar 2.5 memiliki tiga klas meliputi *Doctor*, *Hospital Doctor* dan *General Practitioner*. Klas *Doctor* memiliki atribut *Name*, *Phone*, *E-mail* serta *method Register( )* dan *De-Register( )*. Klas *Hospital Doctor* serta *General Practitioner* mewarisi atribut dan method klas *Doctor*.

2.4 Sistem Basis Data

Menurut Date (2010), basis data adalah kumpulan data yang saling berhubungan dan digunakan pada suatu sistem aplikasi. Menurut Date (2010), tempat untuk menampung kumpulan data yang saling terkait dan terkomputerisasi disebut DBMS (*Database Management System*). MySQL merupakan sebuah DBMS yang bersifat relasional, yaitu mendukung adanya hubungan antar tabel (Damodaran, Shirin & Surekha, 2016). CDM dan PDM pada perancangan data digunakan sebagai acuan untuk merancang basis data pada MySQL.

2.5 *Framework* CodeIgniter

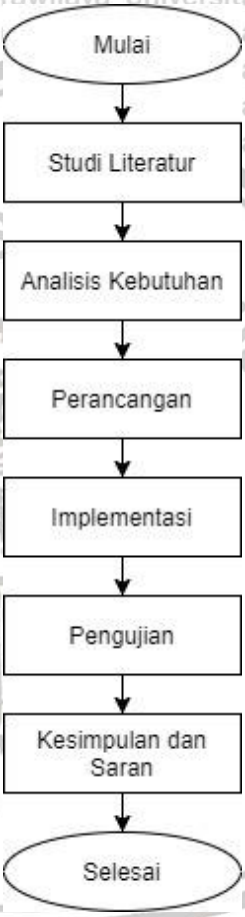
Penelitian ini menggunakan *framework* CodeIgniter untuk mempercepat proses pengerjaan perangkat lunak. *Framework* dapat diartikan sebagai kerangka kerja yang berisi sekumpulan *script* terstruktur untuk mempermudah pengembangan *website* dan menangani masalah pemrograman dengan mudah (Pratama, 2010). Manfaat dari penggunaan *framework* yaitu penghematan waktu dalam pengerjaan kode program dan pengaturan berkas-berkas kode program. Menurut Blanco & Upton (2009), Codeigniter merupakan *framework* yang mudah dipahami dalam mengembangkan aplikasi berbasis web. CodeIgniter memiliki konsep MVC yaitu memisahkan antara *model*, *view* dan *controller*. Penelitian ini menggunakan *framework* CodeIgniter karena membutuhkan manajemen konfigurasi *file* dari setiap komponen-komponen yang ada, sehingga konsep MVC pada *framework* CodeIgniter tidak digunakan dan diganti dengan konsep ECB (*entity-control-boundary*). *Model* pada MVC dengan *entity* pada ECB berbeda karena *model* hanya berkaitan dengan database sedangkan *entity* berkaitan dengan objek sehingga bisa mengolah informasi. *Controller* pada MVC dengan *control* pada ECB juga berbeda karena *controller* mengolah interaksi antar *view* sedangkan *control* mengolah interaksi antar objek yaitu *boundary* dan *entity*. Penerapan konsep ECB ke dalam *framerwork* CI dengan menempatkan *entity* ke dalam *model*, *control* ke dalam *controller* dan *boundary* ke dalam *view*.

Konsep pemrograman berorinetasi objek pada CodeIgniter menggunakan dua klas induk yaitu, klas CI_Controller dan klas CI_Model. Klas CI_Controller memiliki *method construct* dan attribut *load* sedangkan klas CI_Model memliki *method construct* dan *getKey* serta attribut *load* dan *key*. Setiap *method* dan attribut dari klas CI_Controller harus dimiliki oleh semua klas *controller* sehingga semua klas *controller* perlu melakukan *extend* klas CI_Contrroller. Semua klas *model* juga melakukan hal yang sama yaitu melakukan *extend* klas CI_Model. Kedua klas induk harus membuat hak akses *method* dan attribut menjadi *protected* sehingga dapat digunakan oleh setiap klas turunannya yang berarti *framework* CodeIgniter menggunakan kosep *inheritance*. Konsep *inheritance* digunakan untuk menghindari duplikasi kode program.

BAB 3 METODOLOGI PENELITIAN

3.1 Penjelasan Umum

Metodologi penelitian berisi tahapan dalam mengembangkan sistem manajemen persediaan bahan baku berbasis web di Rumah Makan Lalapan ala Dewi. Tahap pertama dimulai dengan studi literatur, analisis kebutuhan, perancangan, implementasi, pengujian serta penarikan kesimpulan dan saran. Tahap analisis kebutuhan sampai ke tahap pengujian mengikuti tahapan pada *waterfall* model. Tahap-tahap tersebut terdapat pada Gambar 3.1.



Gambar 3.1 Tahap penelitian

3.2 Tahap Penelitian

3.2.1 Studi literatur

Studi literatur adalah langkah yang dilakukan untuk memperdalam konsep yang digunakan dalam penelitian ini. Studi literatur berfokus dalam memperlajari teori yang terdapat pada jurnal, buku, artikel ilmiah, dan informasi yang tersedia dengan landasan yang kuat dan berkaitan dengan penelitian ini. Informasi yang diperoleh akan membantu proses penelitian ini. Studi literatur yang digunakan meliputi:

1. Kajian pustaka
2. Rumah makan
3. Rekayasa perangkat lunak
   3.1 *Waterfall model*
   3.2 Pendekatan berorientasi objek
4. Sistem basis data
5. *Framework* CodeIgniter

3.2.2 Analisis

Analisis merupakan langkah-langkah dalam menggali, mengumpulkan, dan memodelkan kebutuhan-kebutuhan yang diperlukan. Hasil dari proses analisis kebutuhan merupakan kebutuhan fungsional dan nonfungsional, aktor yang terlibat, *use case diagram*, dan *use case scenario*. Mekanisme analisis kebutuhan dilakukan dengan beberapa tahap, yaitu:

1. Melakukan elisitasi kebutuhan. Elisitasi kebutuhan dilakukan dengan melakukan wawancara dengan pemilik Rumah Makan Lalapan ala Dewi.
2. Melakukan identifikasi aktor untuk menentukan daftar aktor yang terlibat, kemudian membuat daftar kebutuhan fungsional.
3. Memodelkan kebutuhan fungsional dalam bentuk *use case diagram*.
4. Membuat spesifikasi langkah-langkah untuk menjalankan setiap *use case* dalam bentuk *use case scenario*.

3.2.3 Perancangan

Perancangan merupakan cara atau langkah – langkah dalam merancang dan memodelkan suatu perangkat lunak. Perancangan bertujuan sebagai dasar membangun sistem manajemen persediaan bahan baku. Hasil dari perancangan berupa *sequence diagram*, *class diagram*, *pseudocode*, *conceptual data model*, *physical data model* dan tampilan antarmuka. Mekanisme perancangan dilakukan dengan beberapa tahap, yaitu:

1. Pemodelan *sequence diagram*. Pemodelan *sequence diagram* dilakukan dengan cara mendefinisikan hubungan antar objek dari tiga sampel yang telah ditentukan.
2. Pemodelan *class diagram.* Pemodelan *class diagram* dilakukan dengan mendefinisikan daftar klas beserta hubungan antar klas berdasarkan objek yang ada pada *sequence diagram*.
3. Perancangan komponen. Perancangan komponen dilakukan dengan merancang algoritme dari *method* yang ada pada klas. Hasil dari perancangan algoritme yaitu sebuah *pseudocode* untuk setiap algoritme. Perancangan komponen mengacu pada tiga sampel yang telah ditentukan sebelumnya.

4. Perancangan data. Perancangan data dilakukan dengan pemodelan *conceptual data model* (CDM) dan *physical data model* (PDM). CDM mendefinisikan daftar entitas, relasi antar entitas, dan atribut dari setiap entitas. PDM mendefinisikan daftar tabel, relasi antar tabel, daftar kolom di setiap tabel, *primary key*, *foreign key*, dan tipe data dari setiap kolom.
5. Perancangan antarmuka. Perancangan antarmuka dibuat dengan menggambarkan rancangan antarmuka pengguna dari perangkat lunak yang akan dibangun.

3.2.4 Implementasi

Implementasi merupakan proses perubahan rancangan perangkat lunak menjadi bentuk kode. Implementasi bertujuan untuk merealisasikan sistem manajemen persediaan bahan baku. Implementasi pada penelitian ini menggunakan pendekatan berorientasi objek. Mekanisme implementasi dilakukan dengan beberapa tahap, yaitu:

1. Implementasi klas dan algoritme. Implementasi klas berdasarkan pemodelan *class diagram* dan implementasi algoritme berdasarkan perancangan komponen.
2. Implementasi basis data. Implementasi basis data dilakukan berdasarkan pada pemodelan basis data bagian *physical data model*.
3. Implementasi antarmuka pengguna. Implementasi antarmuka dilakukan berdasarkan pada rancangan antarmuka pengguna yang telah dibuat sebelumnya.

3.2.5 Pengujian

Pengujian merupakan proses untuk menguji hasil dari implementasi. Pengujian pada penelitian ini menggunakan pendekatan berorientasi objek. Pengujian dilakukan dengan menggunakan dua teknik, yaitu pengujian *black-box* dan pengujian *white-box*. Strategi pengujian dilakukan dengan beberapa tahapan, yaitu:

1. Pengujian Unit

   Pengujian unit dilakukan pada setiap unit yaitu klas. Pengujian unit dilakukan dengan metode *white-box testing* dengan menggunakan *basis path testing*. Mekanisme proses pengujian unit dilakukan dengan menjalankan klas *driver* pada setiap kasus uji yang ada. Penelitian ini menggunakan tiga sampel uji untuk melakukan pengujian unit.
2. Pengujian integrasi

   Pengujian integrasi merupakan jenis pengujian yang berfokus untuk menguji hubungan atau integrasi antar unit. Pengujian integrasi dilakukan dengan metode *white-box testing* dengan menggunakan *basis path testing*. Mekanisme proses pengujian unit dilakukan dengan menjalankan klas *driver* pada setiap

kasus uji yang ada. Penelitian ini menggunakan tiga sampel uji untuk melakukan pengujian integrasi.

3. Pengujian Validasi

Pengujian validasi memiliki tujuan untuk menguji kebutuhan perangkat lunak apakah sudah sesuai dengan hasil dari analisis kebutuhan atau tidak. Pengujian dilakukan dengan cara menguji seluruh kebutuhan perangkat lunak. Pengujian validasi dilakukan dengan metode *black-box testing* dengan menggunakan *scenario-based testing*.

3.2.6 Kesimpulan dan saran

Tahap ini merupakan tahap terakhir yaitu menyimpulkan dan memberi saran. Kesimpulan akan menjawab rumusan masalah berdasarkan pengimplementasian sistem yang telah dilakukan. Saran bertujuan untuk memperbaiki kekurangan sistem pada penelitian selanjutnya agar penelitian ini dapat dikembangkan lagi. Representasi kesimpulan dan saran terdapat pada bab kesimpulan dan saran.

BAB 4 ANALISIS KEBUTUHAN

4.1 Elisitasi Kebutuhan

Elisitasi kebutuhan dilakukan dengan cara melakukan wawancara dan melakukan observasi di lapangan. Wawancara yang dilakukan dalam penelitian ini dilaksanakan pada salah satu cabang Rumah Makan Lalapan ala Dewi dengan narasumber Bapak Derwanto. Hasil wawancara tersebut menghasilkan daftar kebutuhan fungsional dan aktor yang terlibat.

4.1.1 Identifikasi aktor

Sistem manajemen persediaan bahan baku memiliki empat aktor, yaitu pengguna, karyawan, pemilik dan admin. Pengidentifikasian aktor memiliki tujuan untuk mendeskripsikan fungsi dari setiap aktor yang terlibat, serta apa saja yang bisa dilakukan oleh masing-masing aktor. Jenis aktor serta deskripsi dari masing-masing aktor dijelaskan pada Tabel 4.1.

Tabel 4.1 Identifikasi aktor

Aktor	Deskripsi
Pengguna	Aktor yang belum diverifikasi oleh sistem dan dapat membuka sistem
Kasir	Aktor yang sudah diverifikasi oleh sistem dan dapat melakukan transaksi
Pemilik	Aktor yang sudah diverifikasi oleh sistem dan dapat melakukan pengecekan bahan baku
Admin	Aktor yang sudah diverifikasi oleh sistem dan dapat melakukan pemeliharaan sistem

4.1.2 Kebutuhan fungsional

Elisitasi kebutuhan menghasilkan 12 kebutuhan fungsional. Setiap kebutuhan fungsional diuraikan dengan jelas. Kebutuhan fungsional sistem manajemen persediaan bahan baku berbasis web antara lain:

1. Sistem memiliki fungsi untuk masuk
2. Sistem memiliki fungsi untuk keluar
3. Sistem memiliki fungsi untuk pesan menu yang tersedia
4. Sistem memiliki fungsi untuk ubah jumlah persediaan bahan baku
5. Sistem memiliki fungsi untuk hitung jumlah persediaan bahan baku
6. Sistem memiliki fungsi untuk cari setiap cabang rumah makan
7. Sistem memiliki fungsi untuk tambah, hapus dan ubah cabang rumah makan
8. Sistem memiliki fungsi untuk unduh data laporan transaksi

9. Sistem memiliki fungsi untuk tambah, laporan transaksi
10. Sistem memiliki fungsi untuk cari kasir pada setiap cabang rumah makan
11. Sistem memiliki fungsi untuk tambah, hapus dan ubah kasir setiap cabang
12. Sistem memiliki fungsi untuk lihat daftar pesanan harian

4.2 Spesifikasi Kebutuhan

Spesifikasi kebutuhan digunakan untuk mendeskripsikan bagaimana spesifikasi dari setiap fungsi yang dimiliki oleh sistem. Setiap kebutuhan fungsional dan spesifikasi akan diberikan kode "MPBB_XXX_YY". Kode "MPBB" menunjukkan sistem manajemen persediaan bahan baku. Kode "XXX" menunjukkan penomoran kebutuhan fungsional. Kode "YY" menunjukkan penomoran spesifikasi kebutuhan fungsional. Daftar kebutuhan fungsional sistem bisa dilihat pada Tabel 4.2.

Tabel 4.2 Kebutuhan fungsional

No	Kebutuhan fungsional dan spesifikasi	Kode	Aktor	*Use case*
1	Sistem harus memiliki fungsi untuk masuk bagi pengguna agar dapat mengakses sistem	MPBB_001_00	Pengguna	*Login*
1.1	Halaman *login* berisi kolom *username*, kolom *password* dan tombol *login*	MPBB_001_01		
2	Sistem harus memiliki fungsi untuk keluar bagi karyawan, pemilik dan admin agar dapat keluar dari sistem	MPBB_002_00	Kasir, Pemilik dan Admin	Keluar
3	Sistem harus memiliki fungsi untuk membuat pesanan	MPBB_003_00	Kasir	Buat pesanan
3.1	Halaman pemesanan berisi kolom nama menu, kolom nama pemesanan, kolom nomor meja, kolom jumlah pesanan dan tombol *submit*	MPBB_003_01		
4	Sistem harus memiliki fungsi ubah persediaan bahan baku	MPBB_004_00	Pemilik	Ubah persediaan bahan baku

4.1	Halaman ubah persediaan menu berisi data bahan baku, jumlah dan nomor cabang	MPBB_004_01		
5	Sistem harus memiliki fungsi tambah cabang rumah makan	MPBB_005_00	Admin	Tambah cabang
51	Halaman tambah nomor cabang rumah makan berisi data cabang dan lokasi	MPBB_005_01		
6	Sistem harus memiliki fungsi hapus cabang rumah makan	MPBB_006_00		Hapus cabang
6.1	Halaman hapus nomor cabang rumah makan berisi data cabang, lokasi dan kasir	MPBB_006_01		
7	Sistem harus memiliki fungsi ubah cabang rumah makan	MPBB_007_00		Ubah cabang
7.1	Halaman ubah nomor cabang rumah makan berisi data cabang, lokasi dan kasir	MPBB_007_01		
8	Sistem harus memiliki fungsi untuk unduh laporan setiap cabang rumah makan	MPBB_008_00	Kasir dan Pemilik	Unduh laporan cabang
8.1	Laporan setiap cabang rumah makan ditampilkan dalam tabel dengan data cabang, detail pengeluaran dan keuntungan	MPBB_008_01		
9	Sistem harus memiliki fungsi tambah laporan cabang setiap rumah makan	MPBB_009_00	Kasir	Tambah laporan cabang
9.1	Halaman tambah laporan cabang setiap rumah	MPBB_009_01		

	makan berisi data cabang, detail pengeluaran dan keuntungan			
10	Sistem harus memiliki fungsi tambah kasir setiap cabang rumah makan	MPBB_010_00	Admin	Tambah kasir
10.1	Halaman tambah kasir setiap cabang rumah makan berisi data nomor cabang, nama kasir dan alamat	MPBB_010_01		
11	Sistem harus memiliki fungsi hapus kasir setiap cabang rumah makan	MPBB_011_00		Hapus kasir
11.1	Halaman hapus kasir setiap cabang rumah makan berisi id kasir, nama kasir, nomor telpon dan alamat	MPBB_011_01		
12	Sistem harus memiliki fungsi ubah kasir setiap cabang rumah makan	MPBB_012_00		Ubah kasir
12.1	Halaman ubah kasir setiap cabang rumah makan berisi data nomor cabang, nama kasir dan alamat	MPBB_012_01		
13	Sistem harus memiliki fungsi tambah menu rumah makan	MPBB_013_00	Pemilik	Tambah menu
13.1	Halaman tambah menu berisi kolom nama menu, kolom harga menu, kolom bahan baku dan tombol *submit*	MPBB_013_01		
14	Sistem harus memiliki fungsi ubah menu rumah makan	MPBB_014_00		Ubah menu
14.1	Halaman ubah menu rumah makan berisi data menu dan harga	MPBB_014_01		

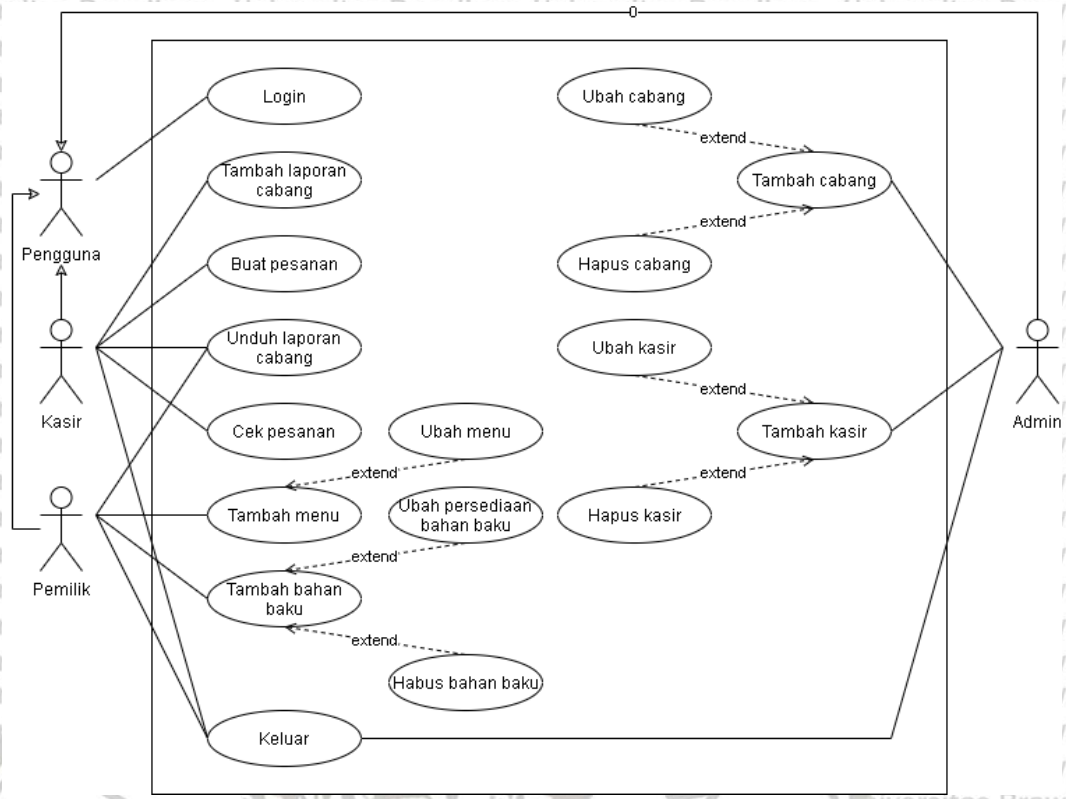
15	Sistem harus memiliki fungsi tambah bahan baku rumah makan	MPBB_015_00		Tambah bahan baku
15.1	Halaman tambah bahan baku berisi kolom nama bahan baku, kolom jumlah bahan baku, kolom jumlah persediaan dan tombol *submit*	MPBB_015_01		
16	Sistem harus memiliki fungsi hapus bahan baku rumah makan	MPBB_016_00		Hapus bahan baku
16.1	Halaman hapus bahan baku rumah makan berisi data bahan baku	MPBB_016_01		
17	Sistem harus memiliki fungsi menampilkan tabel pesanan rumah makan	MPBB_017_00	Kasir	Cek pesanan
17.1	Halaman tabel pesanan rumah makan berisi data setiap pemesanan	MPBB_017_01		

4.3 Pemodelan Kebutuhan

Pemodelan kebutuhan merupakan proses agar pembacaan suatu kebutuhan yang telah didefinisikan lebih mudah. Pemodelan kebutuhan dilakukan dengan menggunakan diagram UML yaitu *use case diagram*. Setiap *use case* pada *use case diagram* akan dijelaskan lebih detail dalam *use case scenario*.

4.3.1 *Use case diagram*

Use case diagram berisi kebutuhan fungsional yang telah didefinisikan kemudian dimodelkan dalam bentuk diagram. *Use case diagram* dapat dilihat pada Gambar 4.1.



Gambar 4.1 *Use case diagram*

4.3.2 *Use case scenario*

Use case scenario berisi detail kebutuhan fungsional yang telah didefinisikan. Jumlah *use case scenario* sama dengan jumlah *use case* pada *use case diagram*. *Use case scenario* dapat dilihat pada Tabel 4.3 hingga Tabel 4.14.

Tabel 4.3 *Use case scenario login*

Login (MPBB_001_00)	
Aktor	Pengguna
Objective	Fungsi untuk masuk ke sistem
Precondition	Halaman *login* yang terdiri dari kolom *username*, kolom *password* dan tombol *login* sudah tersedia
Main Flow	1. Aktor mengisi kolom *username* dan kolom *password* kemudian memerintahkan sistem untuk memproses data 2. Sistem menampilkan halaman *index* dengan hak akses yang berbeda untuk setiap aktor
Alternative Flows	2. a Jika salah satu kolom salah maka sistem menampilkan pesan "*Username* atau *password* salah" 2. b Jika salah satu kolom kosong maka sistem menampilkan pesan "*Please fill out this field*"
Postcondition	Aktor sudah teridentifikasi oleh sistem sebagai kasir, pemilik atau *admin*. Untuk kasir sistem menampilkan halaman *index* yang terdiri dari pesan selamat datang. Halaman *index* untuk pemilik terdiri dari nama pengguna dan *dashboard*, sedangkan halaman *index* untuk *admin* terdiri dari nama pengguna dan *dashboard*.

Tabel 4.4 *Use case scenario* keluar

Keluar (MPBB_002_00)	
Aktor	Kasir, Pemilik dan Admin
Objective	Fungsi untuk keluar dari sistem
Precondition	Aktor sudah teridentifikasi oleh sistem
Main Flow	1. Aktor memerintahkan sistem untuk mengeluarkan aktor 2. Sistem mengeluarkan aktor 3. Sistem menampilkan halaman *login*
Alternative Flows	-
Postcondition	Aktor berhasil keluar dari sistem

Tabel 4.5 *Use case scenario* buat pesanan

Buat pesanan (MPBB_003_00)	
Aktor	Kasir
Objective	Fungsi untuk membuat pesanan
Precondition	Aktor sudah berada di halaman *index* kasir
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar menu 2. Sistem menampilkan halaman daftar menu yang berisi nama menu, jumlah menu, keterangan dan tombol tambah pesanan 3. Aktor memilih menu yang akan dipesan 4. Aktor mengisi kolom nama pemesan, kolom nomor meja, kolom jumlah pesanan 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar menu
Alternative Flows	3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar menu
Postcondition	Aktor berhasil menambah pesanan

Tabel 4.6 *Use case scenario* ubah persediaan bahan baku

Ubah persediaan bahan baku (MPBB_004_00)	
Aktor	Pemilik
Objective	Fungsi untuk mengubah jumlah persediaan bahan baku
Precondition	Aktor sudah berada di halaman *index* pemilik
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar bahan baku 2. Sistem menampilkan halaman daftar bahan baku yang berisi id bahan baku, nama bahan baku, jumlah bahan baku dan jumlah persediaan 3. Aktor memilih bahan baku yang akan diubah 4. Aktor mengisi kolom jumlah persediaan 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar bahan baku
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan tambah bahan baku (MPBB_015_00) 2. b Aktor memerintahkan sistem untuk melakukan hapus bahan baku (MPBB_016_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar bahan baku
Postcondition	Aktor berhasil mengubah jumlah persediaan bahan baku

Tabel 4.7 *Use case scenario* tambah cabang

Tambah cabang (MPBB_005_00)	
Aktor	Admin
Objective	Fungsi untuk menambah cabang rumah makan
Precondition	Aktor sudah berada di halaman *index admin*
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar cabang 2. Sistem menampilkan halaman daftar cabang yang berisi id cabang, nama cabang, nomor telpon dan alamat 3. Aktor memerintahkan sistem untuk menambahkan cabang 4. Aktor mengisi kolom nama cabang, kolom nomor telpon dan kolom alamat 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar cabang
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *update* cabang (MPBB_007_00) 2. b Aktor memerintahkan sistem untuk melakukan *delete* cabang (MPBB_006_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar cabang 4. a Jika ada kolom yang kosong maka sistem menampilkan pesan *required* untuk kolom yang kosong
Postcondition	Aktor berhasil menambah cabang rumah makan

Tabel 4.8 *Use case scenario* hapus cabang

Hapus cabang (MPBB_006_00)	
Aktor	Admin
Objective	Fungsi untuk menghapus cabang rumah makan
Precondition	Aktor sudah berada di halaman *index admin*
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar cabang 2. Sistem menampilkan halaman daftar cabang yang berisi id cabang, nama cabang, nomor telpon dan alamat 3. Aktor memilih cabang yang akan dihapus 4. Sistem menampilkan halaman konfirmasi hapus cabang 5. Aktor memerintahkan sistem untuk melakukan hapus cabang 6. Sistem menampilkan halaman daftar cabang
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *update* cabang (MPBB_007_00) 2. b Aktor memerintahkan sistem untuk melakukan tambah cabang (MPBB_005_00) 5. a Aktor memerintahkan sistem untuk melakukan batal hapus cabang
Postcondition	Aktor berhasil menghapus cabang rumah makan

Tabel 4.9 *Use case scenario* ubah cabang

Ubah cabang (MPBB_007_00)	
Aktor	Admin
Objective	Fungsi untuk mengubah cabang rumah makan
Precondition	Aktor sudah berada di halaman *index admin*
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar cabang 2. Sistem menampilkan halaman daftar cabang yang berisi id cabang, nama cabang, nomor telpon dan alamat 3. Aktor memilih cabang yang akan diubah 4. Aktor mengisi kolom nama cabang, kolom nomor telpon dan kolom alamat 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar cabang
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *delete* cabang (MPBB_006_00) 2. b Aktor memerintahkan sistem untuk melakukan tambah cabang (MPBB_005_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar cabang
Postcondition	Aktor berhasil mengubah cabang rumah makan

Tabel 4.10 *Use case scenario* unduh laporan cabang

Unduh laporan cabang (MPBB_008_00)	
Aktor	Kasir dan Pemilik
Objective	Fungsi untuk mengunduh laporan transaksi
Precondition	Aktor sudah berada di halaman *index* pemilik atau *index* kasir
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar laporan transaksi 2. Sistem menampilkan halaman daftar laporan transaksi yang berisi id laporan, nama laporan, tanggal laporan dan *link* isi laporan 3. Aktor memilih laporan transaksi yang akan diunduh
Alternative Flows	2. a Jika daftar laporan transaksi kosong maka sistem menampilkan pesan "Data laporan transaksi kosong" 2. b Aktor memerintahkan sistem untuk melakukan tambah laporan (MPBB_009_00)
Postcondition	Aktor berhasil mengunduh laporan transaksi

Tabel 4.11 *Use case scenario* tambah laporan cabang

Tambah laporan cabang (MPBB_009_00)	
Aktor	Kasir
Objective	Fungsi untuk menambah laporan transaksi
Precondition	Aktor sudah berada di halaman *index* kasir
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar laporan transaksi 2. Sistem menampilkan halaman daftar laporan transaksi yang berisi id laporan, nama laporan, tanggal laporan dan *link* isi laporan 3. Aktor memerintahkan sistem untuk menambahkan laporan 4. Aktor mengisi kolom nama laporan, kolom tanggal laporan dan *file* isi laporan 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar laporan transaksi
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan unduh laporan (MPBB_008_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar laporan transaksi
Postcondition	Aktor berhasil menambah laporan transaksi

Tabel 4.12 *Use case scenario* tambah kasir

Tambah kasir (MPBB_010_00)	
Aktor	Admin
Objective	Fungsi untuk menambah kasir
Precondition	Aktor sudah berada di halaman *index admin*
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar kasir 2. Sistem menampilkan halaman daftar kasir yang berisi id kasir, nama kasir, nomor telpon dan alamat 3. Aktor memerintahkan sistem untuk menambahkan kasir 4. Aktor mengisi kolom nama kasir, kolom nomor telpon dan kolom alamat 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar kasir
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *delete* kasir (MPBB_011_00) 2. b Aktor memerintahkan sistem untuk melakukan *update* kasir (MPBB_012_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar kasir
Postcondition	Aktor berhasil menambah kasir

Tabel 4.13 *Use case scenario* hapus kasir

Hapus kasir (MPBB_011_00)	
Aktor	Admin
Objective	Fungsi untuk menghapus kasir
Precondition	Aktor sudah berada di halaman *index admin*
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar kasir 2. Sistem menampilkan halaman daftar kasir yang berisi id kasir, nama kasir, nomor telpon dan alamat 3. Aktor memilih kasir yang akan dihapus 4. Sistem menampilkan halaman konfirmasi hapus kasir 5. Aktor memerintahkan sistemm untuk melakukan hapus kasir 6. Sistem menampilkan halaman daftar kasir
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *update* kasir (MPBB_012_00) 2. b Aktor memerintahkan sistem untuk melakukan tambah kasir (MPBB_010_00) 5. a Aktor memerintahkan sistem untuk melakukan batal hapus kasir
Postcondition	Aktor berhasil menghapus kasir

Tabel 4.14 *Use case scenario* ubah kasir

Ubah kasir (MPBB_012_00)	
Aktor	Admin
Objective	Fungsi untuk mengubah kasir
Precondition	Aktor sudah berada di halaman *index admin*
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar kasir 2. Sistem menampilkan halaman daftar kasir yang berisi id kasir, nama kasir, nomor telpon dan alamat 3. Aktor memilih kasir yang akan diubah 4. Aktor mengisi kolom nama kasir, kolom nomor telpon dan kolom alamat 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar kasir
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *delete* kasir (MPBB_011_00) 2. b Aktor memerintahkan sistem untuk melakukan tambah kasir (MPBB_010_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar kasir
Postcondition	Aktor berhasil mengubah kasir

Tabel 4.15 *Use case scenario* tambah menu

Tambah menu (MPBB_013_00)	
Aktor	Pemilik
Objective	Fungsi untuk menambah menu rumah makan
Precondition	Aktor sudah berada di halaman *index* pemilik
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar menu 2. Sistem menampilkan halaman daftar menu yang berisi id menu, nama menu, jumlah menu, harga menu, status menu, tombol *update*, tombol *delete* dan tombol tambah menu 3. Aktor memerintahkan sistem untuk menambahkan menu 4. Aktor mengisi kolom nama menu, kolom harga menu dan kolom bahan baku 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar menu
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *update* menu (MPBB_014_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar menu
Postcondition	Aktor berhasil menambah menu rumah makan

Tabel 4.16 *Use case scenario* ubah menu

Ubah menu (MPBB_014_00)	
Aktor	Pemilik
Objective	Fungsi untuk mengubah menu rumah makan
Precondition	Aktor sudah berada di halaman *index* pemilik
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar menu 2. Sistem menampilkan halaman daftar menu yang berisi id menu, nama menu, jumlah menu, harga menu dan status menu 3. Aktor memilih menu yang akan diubah 4. Aktor mengisi kolom harga menu dan kolom status menu 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar menu
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan tambah menu (MPBB_013_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar menu
Postcondition	Aktor berhasil mengubah menu rumah makan

Tabel 4.17 *Use case scenario* tambah bahan baku

Tambah bahan baku (MPBB_015_00)	
Aktor	Pemilik
Objective	Fungsi untuk menambah bahan baku
Precondition	Aktor sudah berada di halaman *index* pemilik
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar bahan baku 2. Sistem menampilkan halaman daftar bahan baku yang berisi id bahan baku, nama bahan baku, jumlah bahan baku dan jumlah persediaan 3. Aktor memerintahkan sistem untuk menambahkan bahan baku 4. Aktor mengisi kolom nama bahan baku, kolom jumlah bahan baku dan kolom jumlah persediaan 5. Aktor memerintahkan sistem untuk memproses data 6. Sistem menampilkan halaman daftar bahan baku
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *delete* bahan baku (MPBB_016_00) 2. b Aktor memerintahkan sistem untuk melakukan *update* bahan baku (MPBB_004_00) 3. a Aktor memerintahkan sistem untuk kembali ke halaman daftar bahan baku
Postcondition	Aktor berhasil menambah bahan baku

Tabel 4.18 *Use case scenario* hapus bahan baku

Hapus bahan baku (MPBB_016_00)	
Aktor	Pemilik
Objective	Fungsi untuk menghapus bahan baku
Precondition	Aktor sudah berada di halaman *index* pemilik
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar bahan baku 2. Sistem menampilkan halaman daftar bahan baku yang berisi id bahan baku, nama bahan baku, jumlah bahan baku dan jumlah persediaan 3. Aktor memilih bahan baku yang akan dihapus 4. Sistem menampilkan halaman konfirmasi hapus bahan baku 5. Aktor memerintahkan sistem untuk melakukan hapus bahan baku 6. Sistem menampilkan halaman daftar bahan baku
Alternative Flows	2. a Aktor memerintahkan sistem untuk melakukan *update* bahan baku (MPBB_004_00) 2. b Aktor memerintahkan sistem untuk melakukan tambah bahan baku (MPBB_015_00) 5. a Aktor memerintahkan sistem untuk melakukan batal hapus bahan baku
Postcondition	Aktor berhasil menghapus bahan baku

Tabel 4.19 *Use case scenario* cek pesanan

Cek pesanan (MPBB_017_00)	
Aktor	Kasir
Objective	Fungsi untuk melihat daftar pesanan harian
Precondition	Aktor sudah berada di halaman *index* kasir
Main Flow	1. Aktor memerintahkan sistem untuk menampilkan informasi daftar pesanan 2. Sistem menampilkan halaman pesanan yang berisi id pesanan, nama pemesan, jumlah, menu, nomor meja, tanggal, nama kasir dan total harga 3. Aktor melihat daftar pesanan harian
Alternative Flows	2. a Jika daftar pesanan kosong maka sistem menampilkan pesan "Data pesanan kosong"
Postcondition	Aktor berhasil melihat daftar pesanan harian

BAB 5 PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan

Perancangan merupakan proses yang digunakan untuk menghasilkan rancangan dari sistem yang akan dibuat. Pada proses pengembangan sistem ini perancangan dilakukan dengan beberapa tahap, yaitu: perancangan arsitektur, perancangan data, perancangan komponen dan perancangan antarmuka pengguna.

5.1.1 Perancangan arstitektur

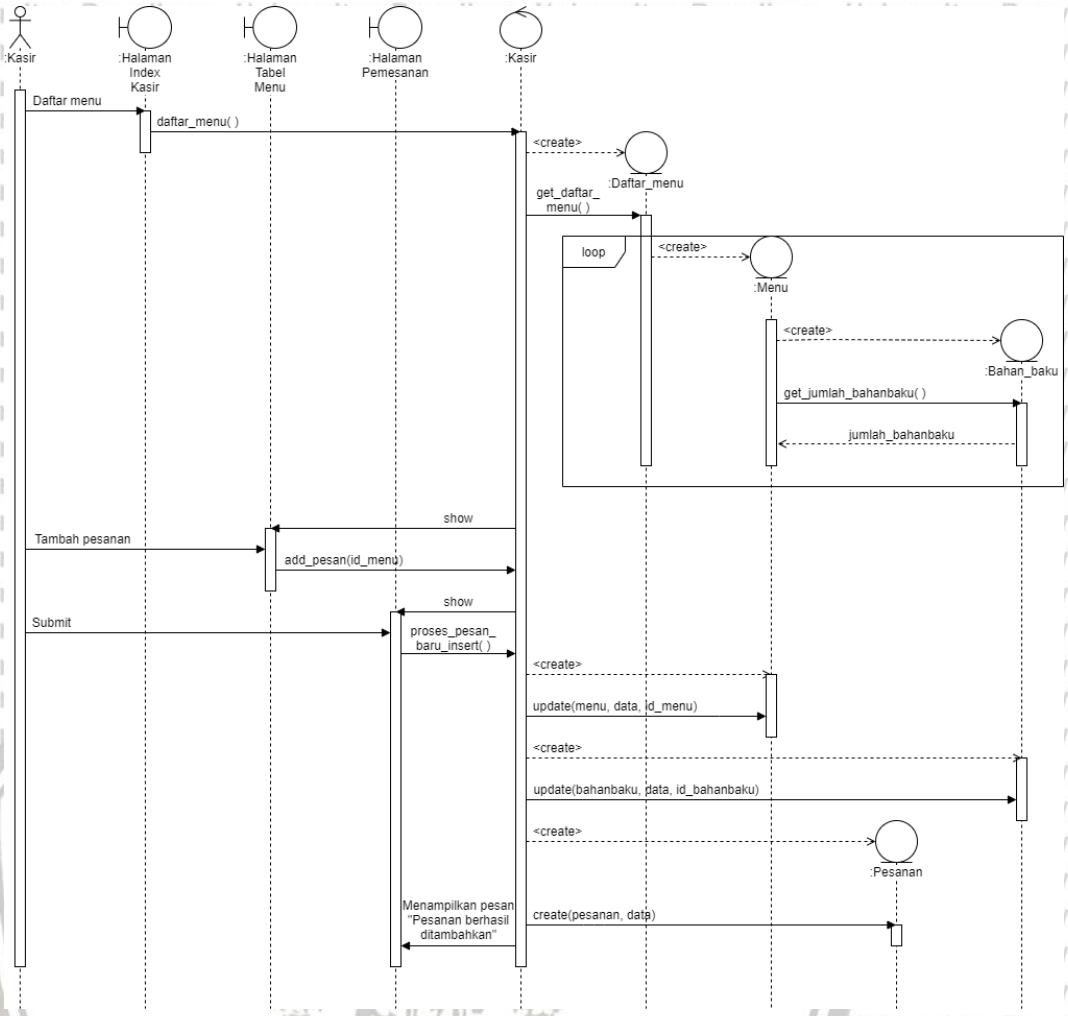
Perancangan arsitektur digunakan untuk mengetahui gambaran umum dari sistem yang akan dikembangkan. Perancangan arsitektur dilakukan dengan dua tahap, yaitu pemodelan *sequence diagram* dan pemodelan *class diagram*.

5.1.1.1 Pemodelan *sequence diagram*

Pemodelan *sequence diagram* digunakan untuk mengidentifikasi objek apa saja yang ada dan bagaimana hubungan antar objek. Pemodelan *sequence diagram* dilakukan berdasarkan *use case scenario* yang telah dibuat sebelumnya. Pemodelan *sequence diagram* dilakukan dengan menggunakan tiga fungsi yang telah dipilih dari sistem, fungsi tersebut yaitu: Buat pesanan, Tambah menu dan Ubah persediaan bahan baku.

1. *Sequence diagram* Buat pesanan

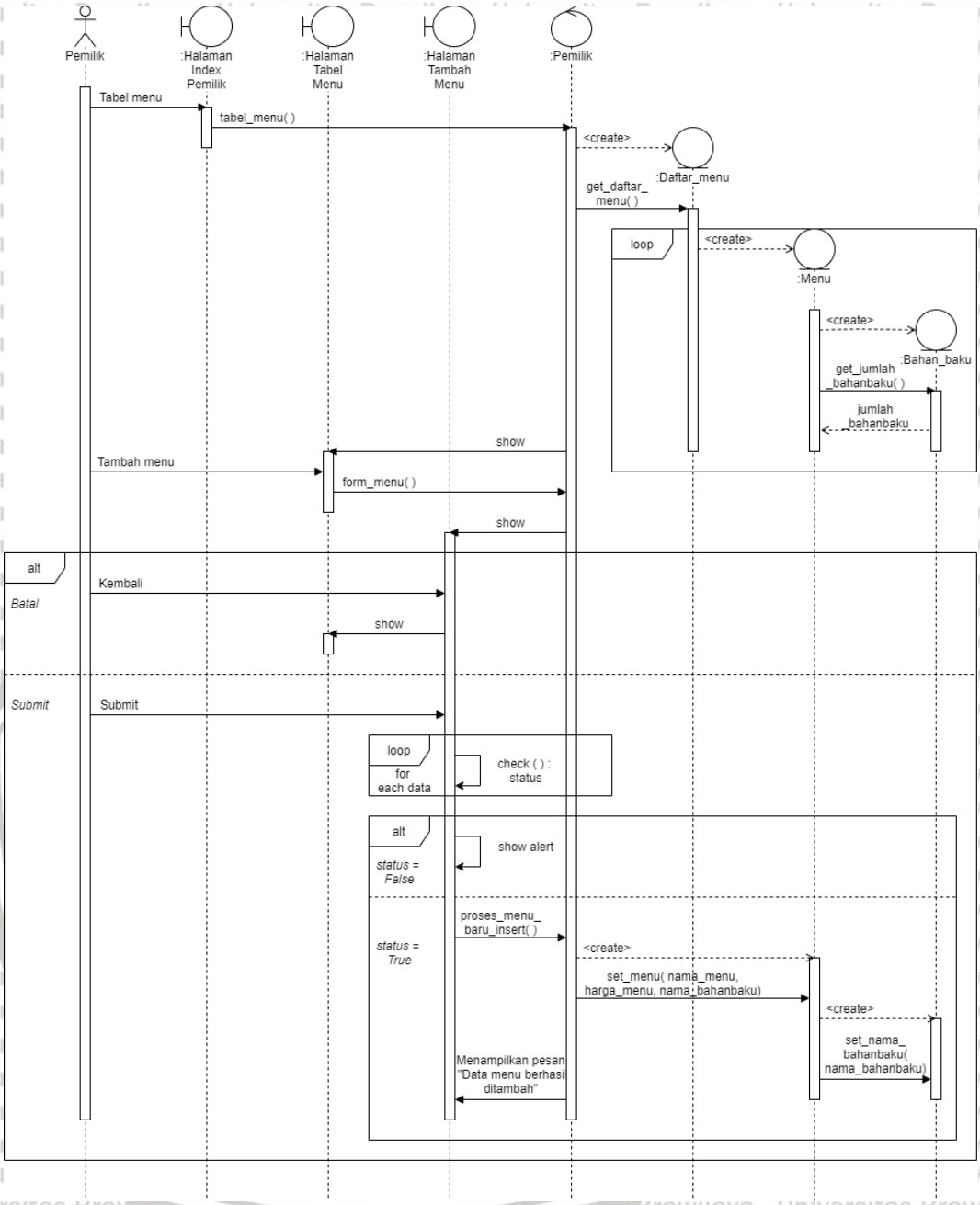
Pemodelan *sequence diagram* Buat pesanan diilustrasikan pada Gambar 5.1. Fungsi Buat pesanan dilakukan oleh aktor Kasir dan terdiri dari tiga objek *boundary*, satu objek *controller* dan empat objek *entity*. Objek *boundary* terdiri dari Halaman *Index*, Halaman Menu dan Halaman Pemesanan. Objek *controller* terdiri dari Kasir yang berfungsi untuk menerima perintah dari *boundary* dan mengirimkan ke *model*. Objek *entity* terdiri dari Daftar_menu, Menu, Pesanan dan Bahan_baku.



Gambar 5.1 *Sequence diagram* buat pesanan

2. *Sequence diagram* Tambah menu

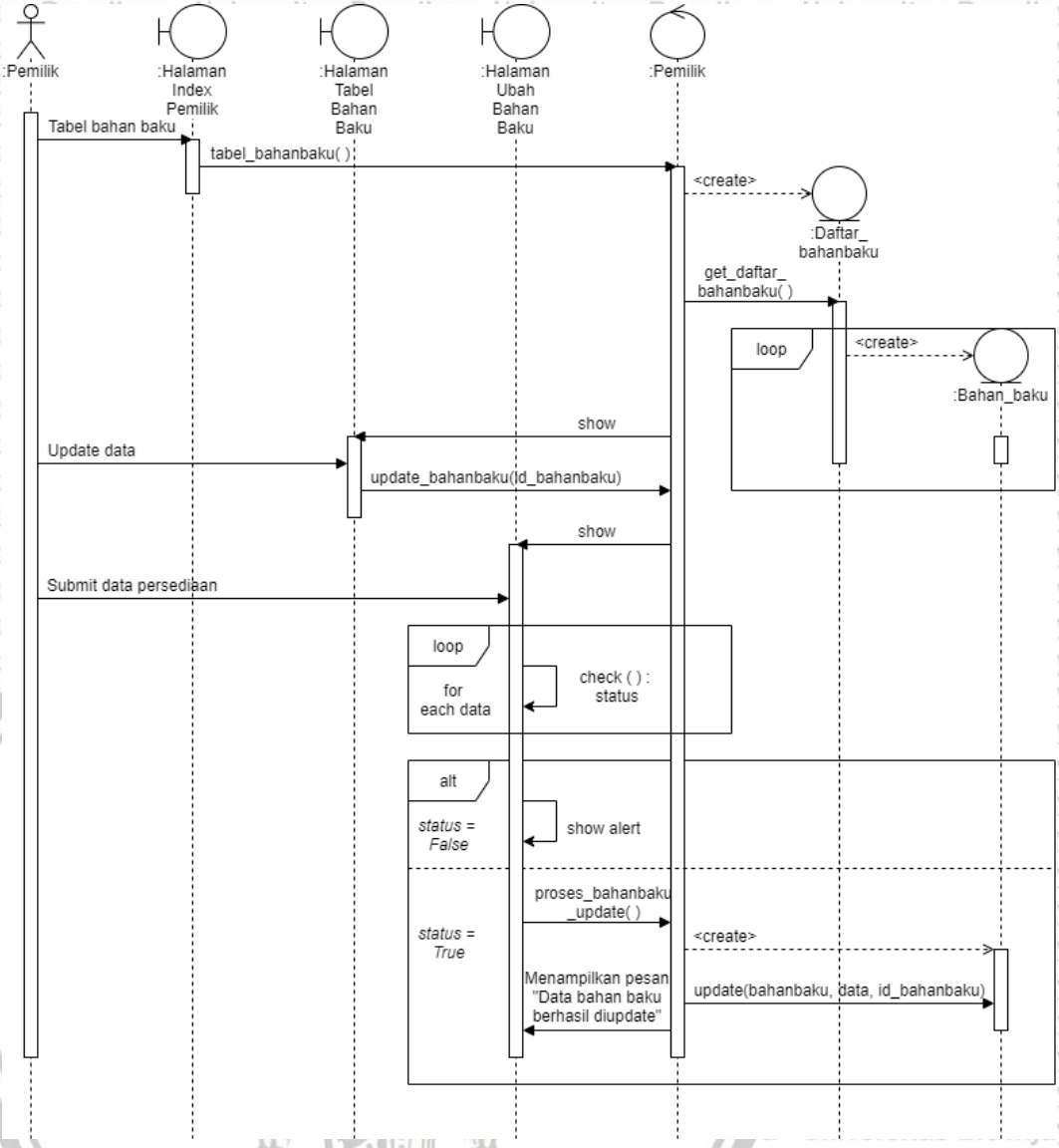
Pemodelan *sequence diagram* Tambah menu diilustrasikan pada Gambar 5.2. Fungsi Tambah menu dilakukan oleh aktor Pemilik dan terdiri dari tiga objek *boundary*, satu objek *controller* dan tiga objek *entity*. Objek *boundary* terdiri dari Halaman *Index*, Halaman Tabel Menu dan Halaman Tambah Menu. Objek *controller* terdiri dari Pemilik yang berfungsi untuk menerima perintah dari *boundary* dan mengirimkan ke *model* sedangkan objek *entity* terdiri dari Daftar_menu, Menu dan Bahan_baku



Gambar 5.2 *Sequence diagram* tambah menu

3. *Sequence diagram* Ubah persediaan bahan baku

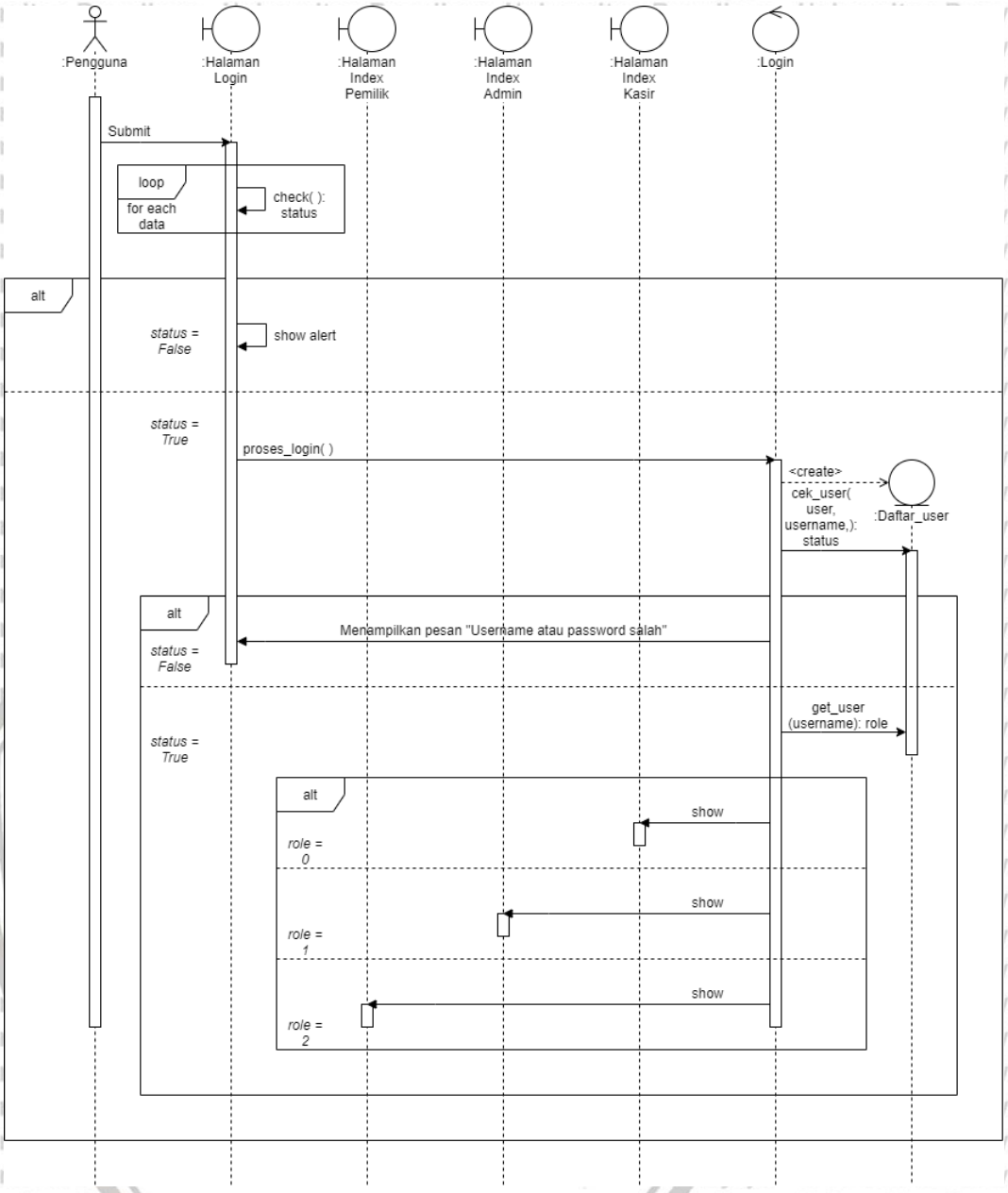
Pemodelan *sequence diagaram* Ubah persediaan bahan baku diilustrasikan pada Gambar 5.3. Fungsi Buat pesanan dilakukan oleh aktor Pemilik dan terdiri dari tiga objek *boundary*, satu objek *controller* dan dua objek *entity*. Objek *boundary* terdiri dari Halaman *Index*, Halaman Bahan Baku dan Halaman Ubah Bahan Baku. Objek *controller* terdiri dari Pemilik yang berfungsi untuk menerima perintah dari *boundary* dan mengirimkan ke *model*. Objek *entity* yang terdiri dari Bahan_baku dan Daftar_bahan_baku



Gambar 5.3 *Sequence diagram* ubah persediaan bahan baku

4. *Sequence diagram Login*

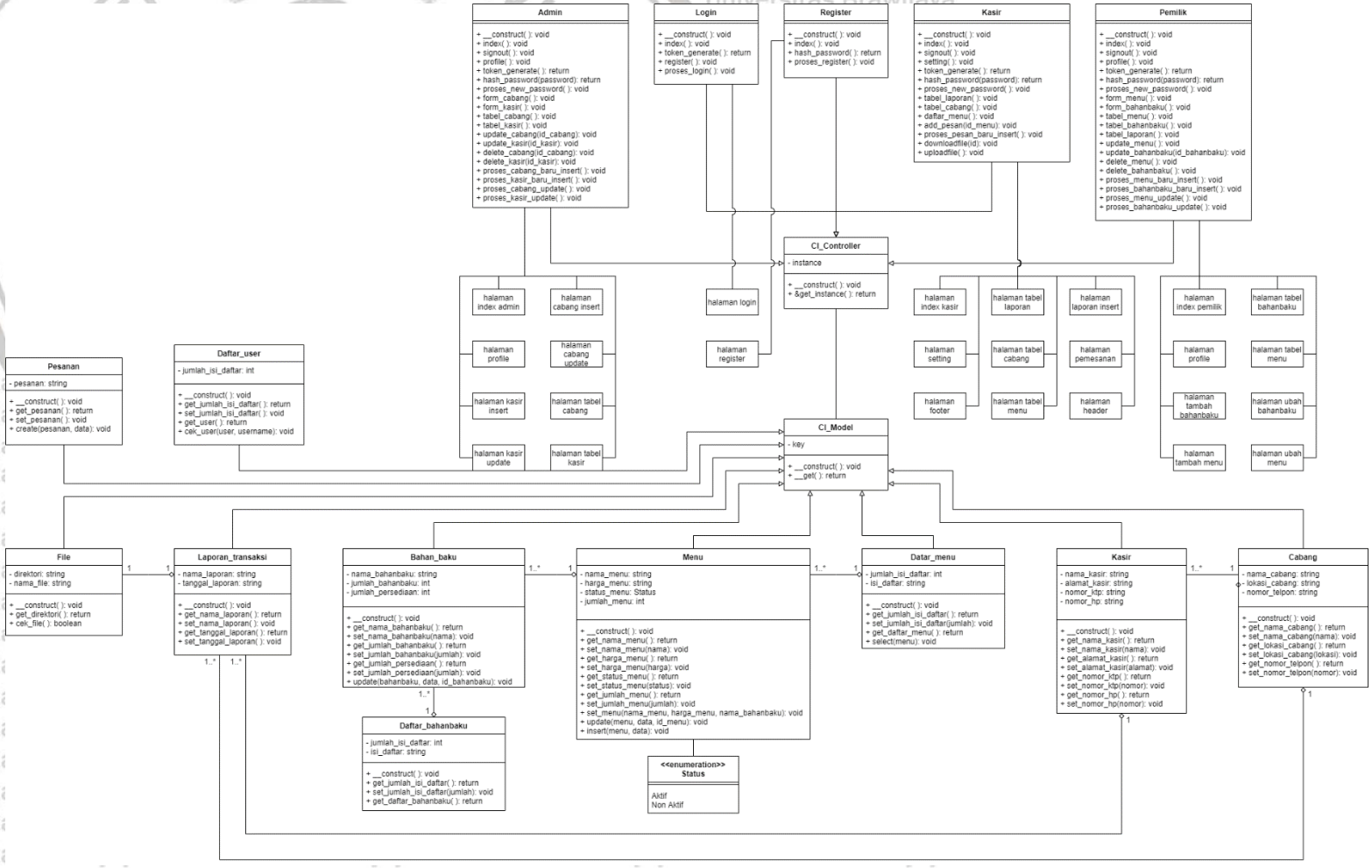
Pemodelan *sequence diagaram login* diilustrasikan pada Gambar 5.4. Fungsi *Login* dilakukan oleh aktor Pengguna dan terdiri dari empat objek *boundary*, satu objek *controller* dan satu objek *entity*. Objek *boundary* terdiri dari Halaman Login, Halaman *Index* Pemilik, Halaman *Index Admin* dan Halaman *Index* Kasir. Objek *controller* terdiri dari Login yang berfungsi untuk menerima perintah dari *boundary* dan mengirimkan ke *model*. Objek *entity* yang terdiri dari Daftar_*user*.



Gambar 5.4 *Sequence diagram login*

5.1.1.2 Pemodelan *class diagram*

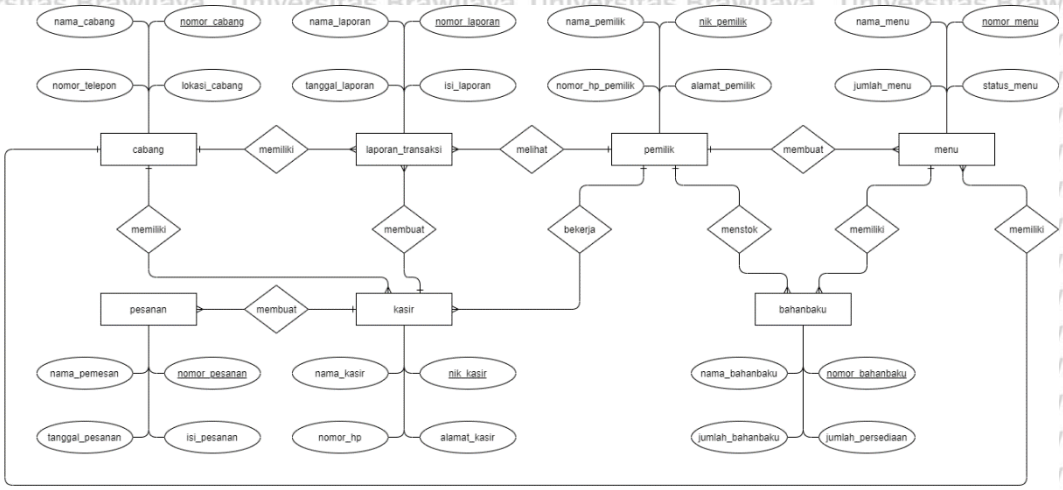
Pemodelan *class diagram* mengacu pada *sequence diagram* yang telah didefinisikan sebelumnya. Perangkat lunak dibangun oleh klas-klas *boundary*, klas-klas *control* yang merupakan extend dari klas CI_Controller dan klas-klas *entity* yang merupakan *extend* dari klas CI_Model. Setiap klas tersebut merujuk pada objek-objek yang ada pada pemodelan *sequence diagram*. Klas *control* yang ada pada *class diagram* mempunyai hubungan dengan klas *boundary* sehingga satu klas *control* berasosiasi dengan satu klas *boundary*. Pemodelan *class diagram* ini menggunakan konsep ECB sehingga klas *control* berhubungan langsung dengan klas *entity* dan *boundary*. Pemodelan *class diagram* dapat dilihat pada Gambar 5.4.



Gambar 5.5 *Class diagram* umum

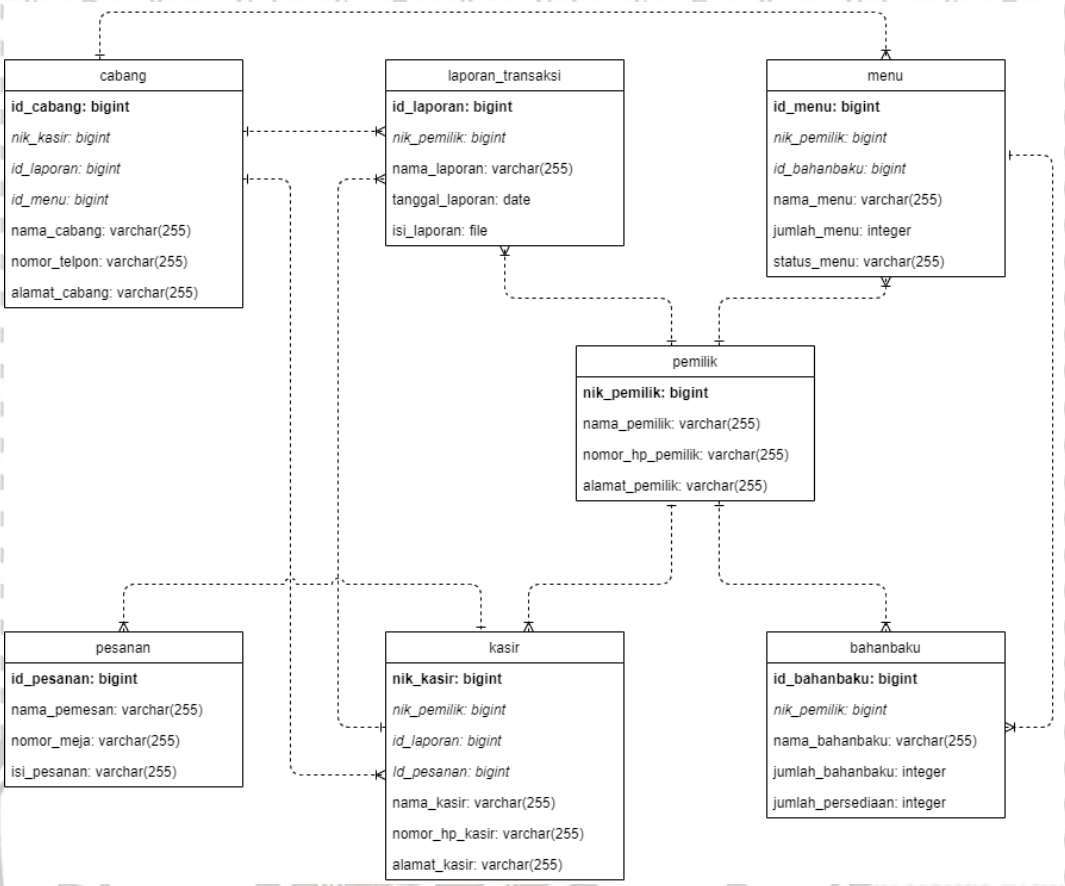
5.1.2 Perancangan data

Perancangan data dilakukan untuk mendefinisikan struktur data secara konsep dan fisik. Perancangan data menggunakan basis data sehingga pemodelan data dilakukan dalam bentuk *conceptual data model* (CDM) dan *physical data model* (PDM). CDM direpresentasikan dalam bentuk *entity relationship diagram* (ERD). Entity yang terdapat dalam ERD berjumlah tujuh. CDM dapat dilihat pada Gambar 5.5.



Gambar 5.6 *Conceptual data model*

CDM kemudian ditransformasikan menjadi PDM. Proses transformasi CDM menjadi PDM dilakukan dengan memetakan *entity* dan relasi menjadi kumpulan tabel. Nama *entity* akan menjadi nama tabel dan atribut pada *entity* akan menjadi kolom. Atribut yang memiliki garis bawah akan menjadi *primary key*. *Primary key* akan digunakan sebagai *foreign key* untuk tabel lain yang berelasi dengan tabel yang memiliki *primary key*. Atribut yang terdapat pada CDM akan ditentukan tipe data dan panjang karakternya. PDM dapat dilihat pada Gambar 5.6.



Gambar 5.7 *Physical data model*

5.1.3 Perancangan komponen

Perancangan komponen dilakukan untuk mendefinisikan algoritme dalam suatu *method*. Algoritme suatu *method* dituliskan dalam bentuk *pseudocode*. Perancangan komponen berisi tiga *method*, buat pesanan, tambah menu dan ubah persediaan bahan baku. Perancangan komponen dapat dilihat pada Tabel 5.1 hingga Tabel 5.3.

Tabel 5.1 *Pseudocode* buat pesanan

No.	*Pseudocode*
1	*START*
2	*nama_pemesan = input dari field nama_pemesan*
3	*jumlah_pesanan = input dari field jumlah_pesanan*
4	*nomor_meja = input dari field nomor_meja*
5	*data = array(*
6	*'nama_pemesan' => nama_pemesan,*
7	*'jumlah_pesanan' => jumlah_pesanan,*
8	*'nomor_meja' => nomor_meja*

9	*)*
10	*insert data ke database pesanan*
11	*menampilkan pesan "Data Pesanan Berhasil Ditambahkan"*
12	*END*

Tabel 5.2 *Pseudocode* tambah menu

No.	*Pseudocode*
1	*START*
2	*nama_menu = input dari field nama_menu*
3	*harga_menu = input dari field harga_menu*
4	*id_bahanbaku = input dari field bahanbaku*
5	*data = array(*
6	*'nama_menu' => nama_menu,*
7	*'harga_menu' => harga_menu,*
8	*'id_bahanbaku' => id_bahanbaku*
9	*)*
10	*insert data ke database menu*
11	*menampilkan pesan "Data Menu Berhasil Ditambahkan"*
12	*END*

Tabel 5.3 *Pseudocode* ubah persediaan bahan baku

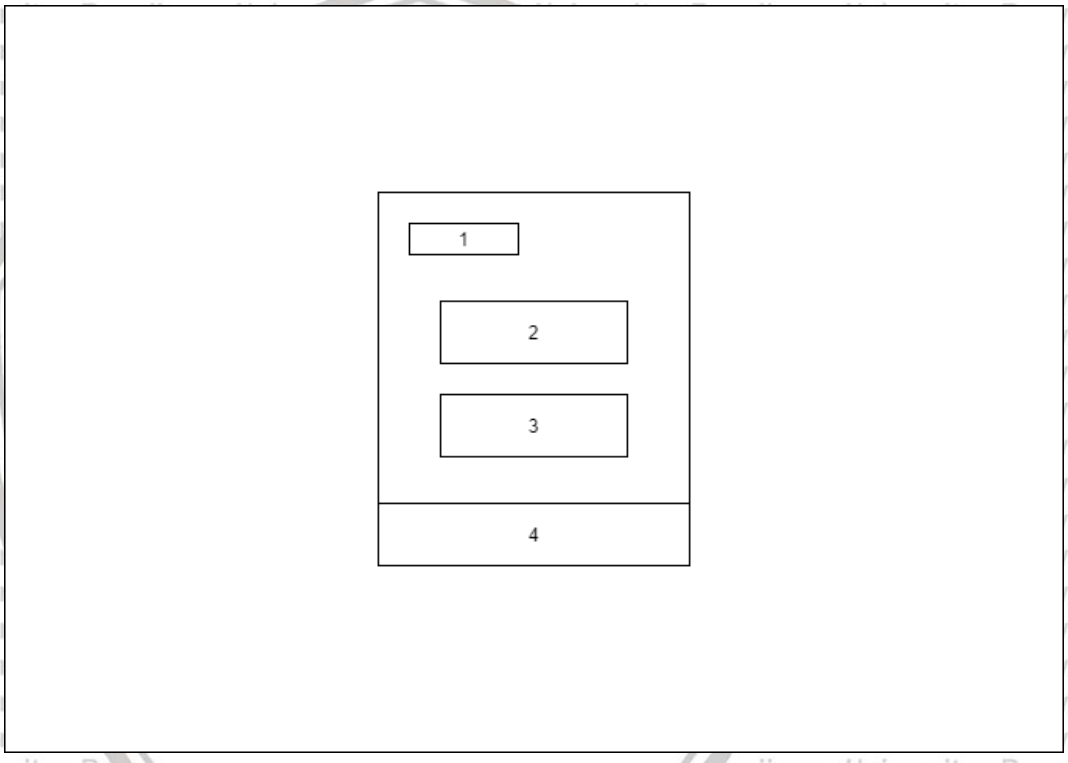
No.	*Pseudocode*
1	*START*
2	*nama_bahanbaku = input dari field nama_bahanbaku*
3	*jumlah_persediaan = input dari field jumlah_persediaan*
4	*data = array(*
5	*'nama_bahanbaku' => nama_bahanbaku,*
6	*'jumlah_persediaan' => jumlah_persediaan*
7	*)*
8	*insert data ke database bahanbaku*
9	*menampilkan pesan "Data Bahan Baku Berhasil Diupdate"*
10	*END*

5.1.4 Perancangan antarmuka

Perancangan antarmuka dilakukan dengan menggambar rancangan antarmuka yang akan diimplementasikan. Perancangan antarmuka memuat lima sampel yang terdiri dari halaman *login*, halaman pemesanan, halaman tambah menu, halaman hapus kasir, dan halaman ubah bahan baku. Perancangan antarmuka dapat dilihat pada Gambar 5.7 hingga Gambar 5.11. Keterangan antarmuka dapat dilihat pada Tabel 5.4 hingga Tabel 5.8.

5.1.4.1 Perancangan antarmuka halaman *login*

Halaman *login* merupakan halaman yang pertama kali ditampilkan saat pengguna mengakses sistem. Rancangan antarmuka dapat dilihat pada Gambar 5.7. Keterangan gambar dapat dilihat pada Tabel 5.4.



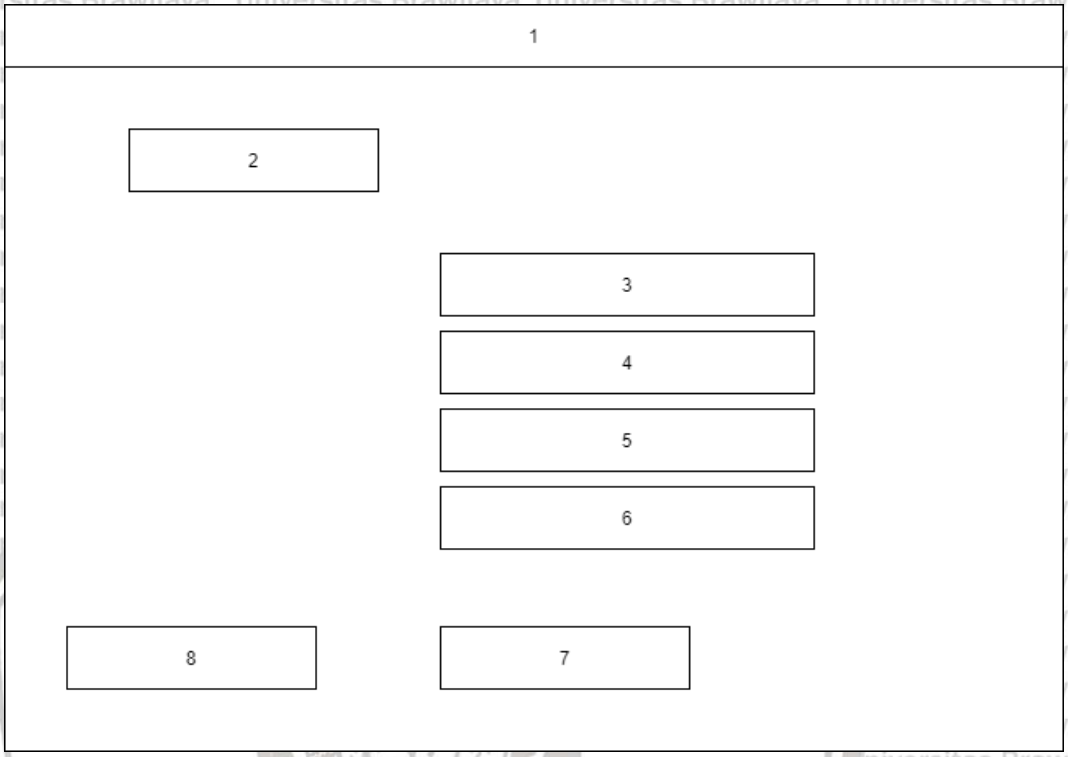
Gambar 5.8 Rancangan antarmuka halaman *login*

Tabel 5.4 Keterangan antarmuka halaman *login*

No	Nama objek	Tipe	Keterangan
1	*Log in*	*Text*	Menampilkan judul form
2	*Username*	*Text field*	Area untuk mengisi nama pengguna
3	*Password*	*Text field*	Area untuk mengisi kata sandi
4	*Log in*	*Button*	Tombol untuk masuk ke sistem

5.1.4.2 Perancangan antarmuka halaman pemesanan

Halaman pemesanan merupakan halaman yang berisi *form* pesan saat pengguna akan melakukan pemesanan. Rancangan antarmuka dapat dilihat pada Gambar 5.8. Keterangan gambar dapat dilihat pada Tabel 5.5.



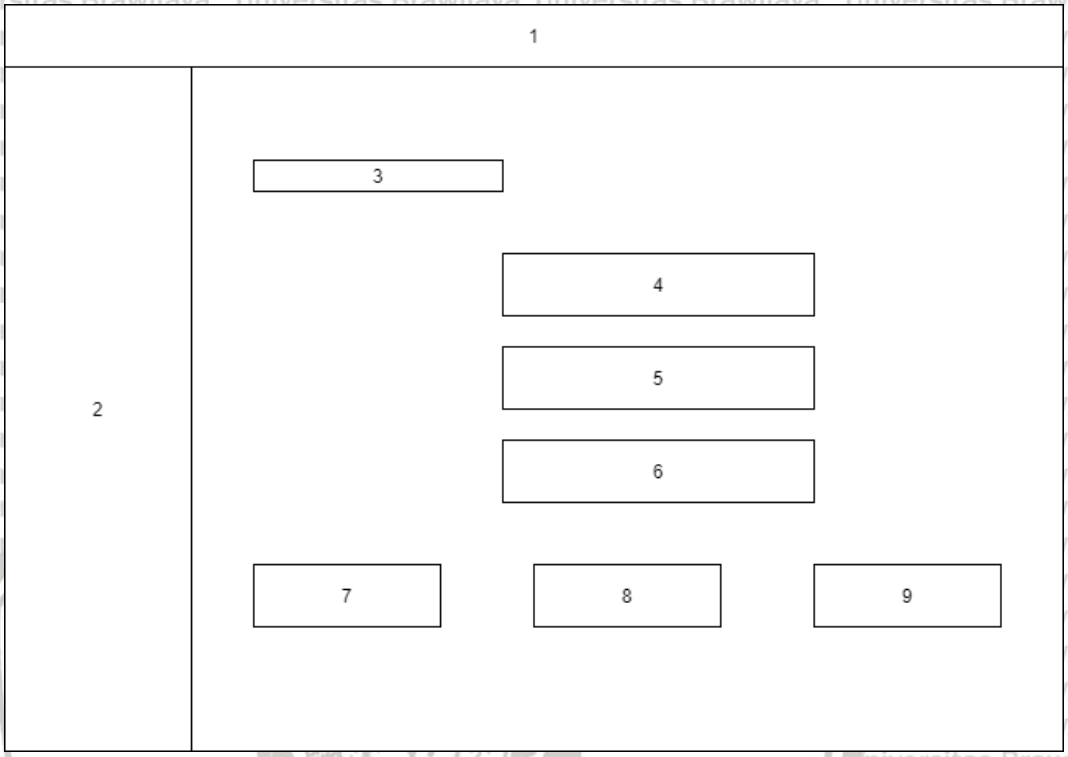
Gambar 5.9 Rancangan antarmuka halaman pemesanan

Tabel 5.5 Keterangan antarmuka halaman pemesanan

No	Nama objek	Tipe	Keterangan
1	Rumah makan	*Header*	Menampilkan *header* sistem
2	Tambah data	*Text*	Menampilkan judul form
3	Nama menu	*Text*	Menampilkan nama menu
4	Nama pemesan	*Text field*	Area untuk mengisi nama
5	Nomor meja	*Text field*	Area untuk mengisi nomor meja
6	Jumlah pesanan	*Text field*	Area untuk mengisi jumlah
7	*Submit*	*Button*	Tombol untuk menambah pesanan
8	Kembali	*Button*	Tombol untuk kembali

5.1.4.3 Perancangan antarmuka halaman tambah menu

Halaman tambah menu merupakan halaman yang berisi rincian suatu menu saat pengguna akan melakukan tambah menu. Rancangan antarmuka dapat dilihat pada Gambar 5.9. Keterangan gambar dapat dilihat pada Tabel 5.6.



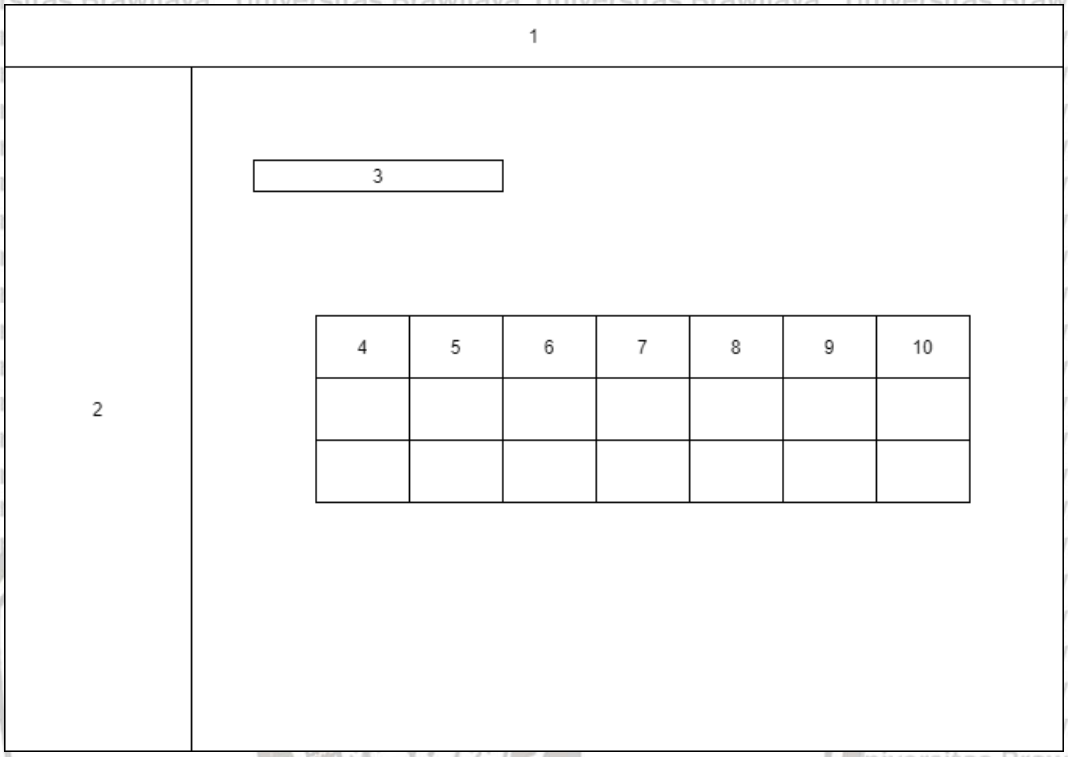
Gambar 5.10 Rancangan antarmuka halaman tambah menu

Tabel 5.6 Keterangan antarmuka halaman tambah menu

No	Nama objek	Tipe	Keterangan
1	Pemilik	*Header*	Menampilkan *header* sistem
2	*Navigation*	*Sidebar*	Menampilkan *sidebar* sistem
3	Tambah data	*Text*	Menampilkan judul *form*
4	Nama menu	*Text field*	Area untuk mengisi nama menu
5	Harga menu	*Text field*	Area untuk mengisi harga menu
6	Bahan baku	*Text field*	Area untuk mengisi bahan baku
7	Kembali	*Button*	Tombol untuk kembali
8	*List* menu	*Button*	Tombol untuk melihat menu
9	*Submit*	*Button*	Tombol untuk menambah menu

5.1.4.4 Perancangan antarmuka halaman hapus kasir

Halaman hapus kasir merupakan halaman yang berisi tabel kasir saat pengguna akan melakukan hapus kasir. Rancangan antarmuka dapat dilihat pada Gambar 5.10. Keterangan gambar dapat dilihat pada Tabel 5.7.



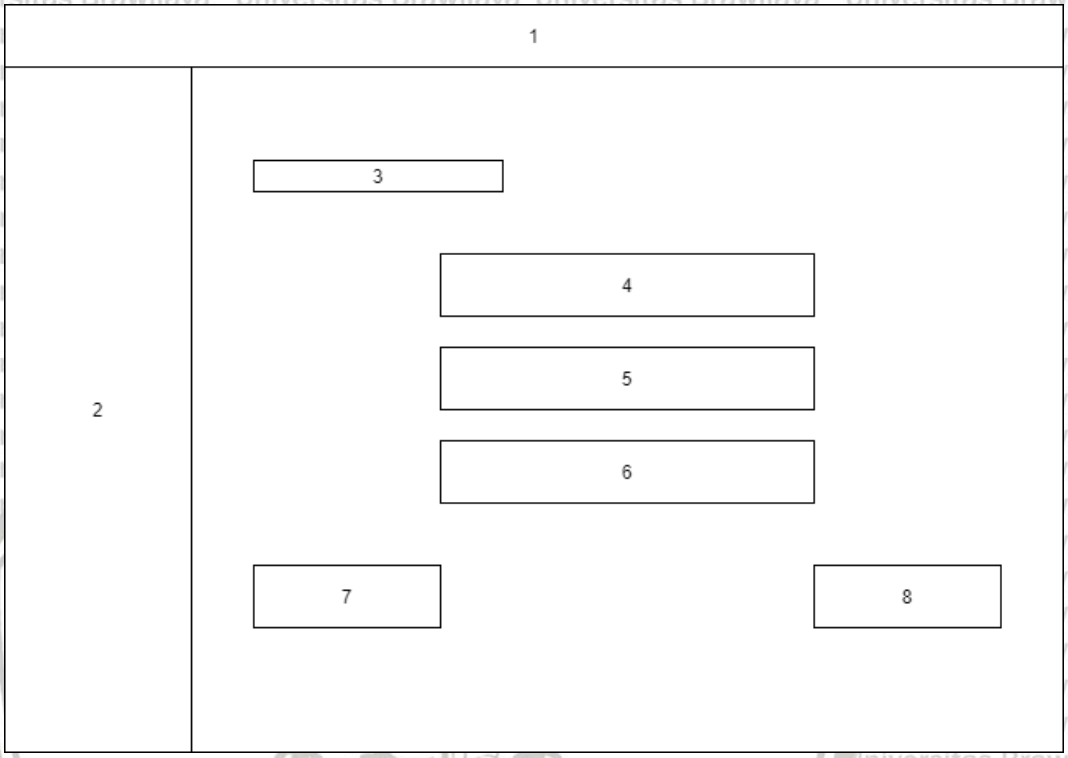
Gambar 5.11 Rancangan antarmuka halaman hapus kasir

Tabel 5.7 Keterangan antarmuka halaman hapus kasir

No	Nama objek	Tipe	Keterangan
1	Admin	*Header*	Menampilkan *header* sistem
2	*Navigation*	*Sidebar*	Menampilkan *sidebar* sistem
3	Kasir	*Text*	Menampilkan judul tabel
4	No	*Text*	Menampilkan nomor tabel
5	Id kasir	*Text*	Menampilkan id kasir
6	Nama kasir	*Text*	Menampilkan nama kasir
7	Nomor telpon	*Text*	Menampilkan nomor telpon
8	Alamat	*Text*	Menampilkan alamat
9	*Update*	*Button*	Tombol untuk update kasir
10	*Delete*	*Button*	Tombol untuk delete kasir

5.1.4.5 Perancangan antarmuka halaman ubah bahan baku

Halaman ubah bahan baku merupakan halaman yang berisi *form* ubah persediaan bahan baku saat pengguna akan melakukan ubah persediaan. Rancangan antarmuka dapat dilihat pada Gambar 5.11. Keterangan gambar dapat dilihat pada Tabel 5.8.



Gambar 5.12 Rancangan antarmuka halaman ubah bahan baku

Tabel 5.8 Keterangan antarmuka halaman ubah bahan baku

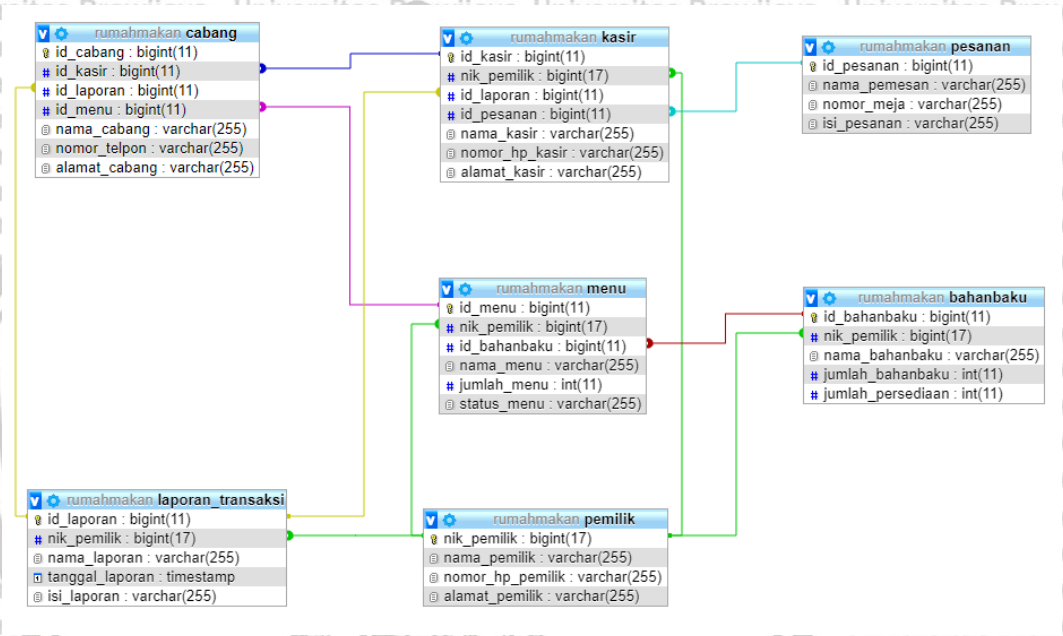
No	Nama objek	Tipe	Keterangan
1	Pemilik	*Header*	Menampilkan *header* sistem
2	*Navigation*	*Sidebar*	Menampilkan *sidebar* sistem
3	Ubah data	*Text*	Menampilkan judul *form*
4	Id bahan baku	*Text*	Menampilkan id bahan baku
5	Nama bahan baku	*Text field*	Area untuk mengisi nama bahan baku
6	Jumlah persediaan	*Text field*	Area untuk mengisi jumlah persediaan
7	Kembali	*Button*	Tombol untuk kembali
8	*Submit*	*Button*	Tombol untuk simpan perubahan

5.2 Implementasi

Implementasi merupakan proses yang digunakan untuk mengubah hasil rancangan yang telah dibuat menjadi bentuk kode. Implementasi dilakukan dengan menerapkan seluruh rancangan yang ada. Pada penelitian ini, implementasi dilakukan dengan menggunakan beberapa tahapan, yaitu implementasi basis data, implementasi kode dari sistem dan implementasi antarmuka untuk tampilan dari sistem.

5.2.1 Implementasi data

Implementasi data dilakukan dengan mengacu pada PDM yang telah dibuat pada perancangan data. Gambar 5.12 merupakan hasil implementasi basis data.



Gambar 5.13 Implementasi basis data

5.2.2 Implementasi kode program

5.2.2.1 Implementasi kode program buat pesanan

Tabel 5.9 merupakan implementasi kode program *method* proses_pesan_baru_insert( ) dari klas Pesanan.

Tabel 5.9 Implementasi kode program buat pesanan

Pesanan.php	
1	`public function proses_pesan_baru_insert( ){`
2	`$nama_pemesan = $this->input->post('nama_pemesan',TRUE);`
3	`$jumlah_pesanan = $this->input->post('jumlah_pesanan',TRUE);`
4	`$nomor_meja = $this->input->post('nomor_meja',TRUE);`
5	`$menu_pesan = $this->input->post('menu_pesan',TRUE);`
6	

```
7       $data = array(
8             'nama_pemesan'   => $nama_pemesan,
9             'jumlah_pesanan' => $jumlah_pesanan,
10            'nomor_meja'     => $nomor_meja,
11            'menu_pesan'     => $menu_pesan,
12      );
13      $this->M_user->insert('pesanan',$data);
14      $this->session->set_flashdata('msg_berhasil','Data Pesanan
15  Berhasil Ditambahkan');
16      $this->load->view('user/templates/header.php');
17      $this->load->view('user/index');
18      $this->load->view('user/templates/footer.php');
19    }
20  }
```

5.2.2.2 Implementasi kode program tambah menu

Tabel 5.10 merupakan implementasi kode program *method* proses_menu_baru_insert( ) dari klas Pemilik.

Tabel 5.10 Implementasi kode program tambah menu

Pemilik.php

```
1   public function proses_menu_baru_insert( ){
2         $nama_menu      = $this->input->post('nama_menu',TRUE);
3         $harga_menu     = $this->input->post('harga_menu',TRUE);
4         $id_bahanbaku   = $this->input->post('bahanbaku',TRUE);
5
6         $data = array(
7               'nama_menu'      => $nama_menu,
8               'harga_menu'     => $harga_menu,
9               'id_bahanbaku'   => $id_bahanbaku,
10        );
11        $this->M_pemilik->insert('menu',$data);
12        $this->session->set_flashdata('msg_berhasil','Data Menu
13  Berhasil Ditambahkan');
14        redirect(base_url('pemilik/form_menu'));
```

5.2.2.3 Implementasi kode program ubah persediaan bahan baku

Tabel 5.11 merupakan implementasi kode program *method* proses_bahanbaku_update( ) dari klas Pemilik.

Tabel 5.11 Implementasi kode program ubah persediaan bahan baku

Pemilik.php

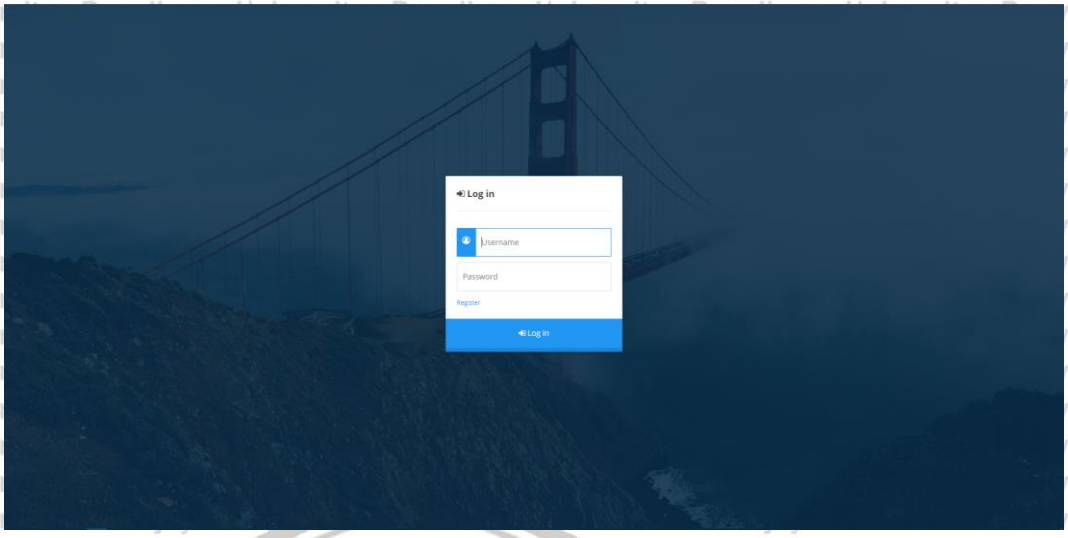
```
1   public function proses_bahanbaku_update( ){
2       $id_bahanbaku     = $this->input->post
3   ('id_bahanbaku',TRUE);
4       $nama_bahanbaku    = $this->input->post
5   ('nama_bahanbaku',TRUE);
6       $jumlah_persediaan = $this->input->post
7   ('jumlah_persediaan',TRUE);
8
9       $where = array('id_bahanbaku' => $id_bahanbaku);
10      $data = array(
11              'id_bahanbaku'       => $id_bahanbaku,
12              'nama_bahanbaku'     => $nama_bahanbaku,
13              'jumlah_persediaan'  => $jumlah_persediaan,
14      );
15      $this->M_pemilik->update('bahanbaku',$data,$where);
16      $this->session->set_flashdata('msg_berhasil','Data Bahan
17  Baku Berhasil Diupdate');
18      redirect(base_url('pemilik/tabel_bahanbaku'));
```

5.2.3 Implementasi antarmuka

Implementasi antarmuka dilakukan berdasarkan hasil perancangan antarmuka. Terdapat lima sampel implementasi antarmuka, yaitu halaman *login*, halaman pemesanan, halaman tambah menu, halaman hapus kasir, dan halaman ubah bahan baku.

5.2.3.1 Halaman *login*

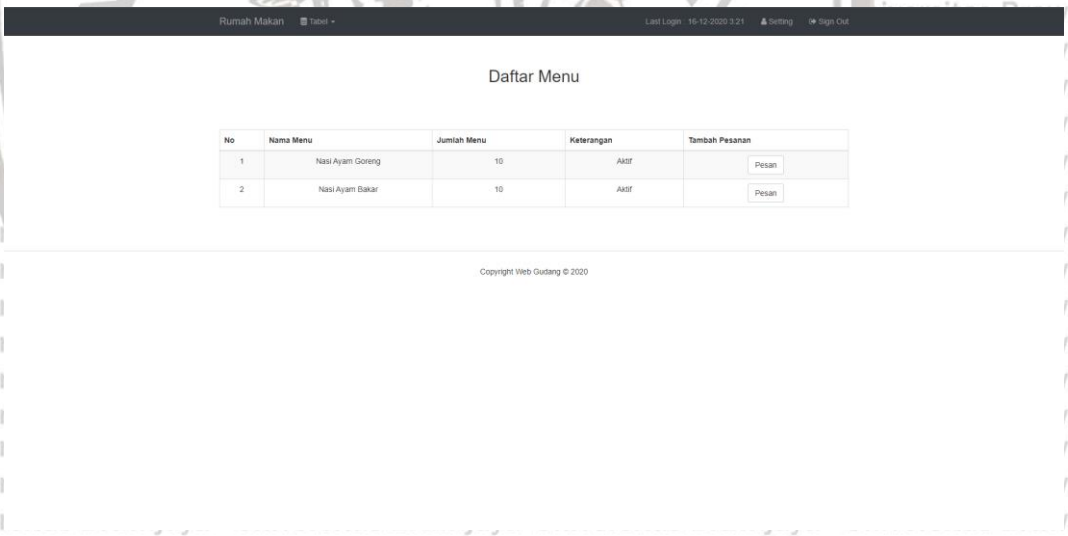
Gambar 5.13 merupakan hasil implementasi antarmuka halaman *login*. Pada halaman *login* terdapat satu formulir dalam *panel* yang berisi dua *textfield* untuk masukkan *username* dan *password*, serta satu tombol *login* yang berfungsi untuk mengarahkan pengguna masuk kedalam sistem sesuai peranannya.



Gambar 5.14 Implementasi antarmuka halaman *login*

5.2.3.2 Halaman pemesanan

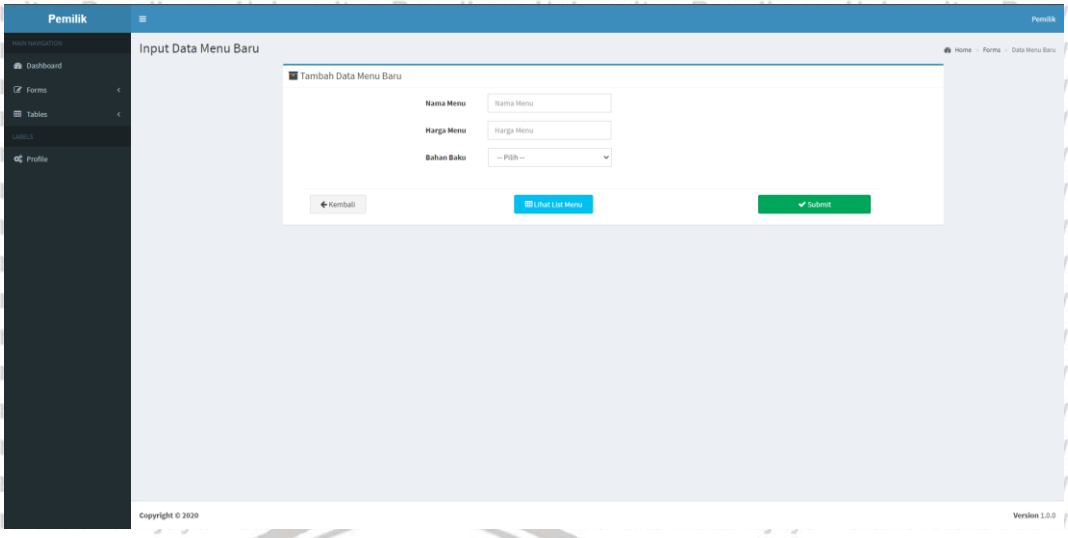
Gambar 5.14 merupakan hasil implementasi antarmuka halaman pemesanan. Pada halaman ini terdapat *header* berupa nama aplikasi dan tombol *logout*, serta *body* yang berisi daftar menu beserta informasi nama menu dan jumlah menu.



Gambar 5.15 Implementasi antarmuka halaman pemesanan

5.2.3.3 Halaman tambah menu

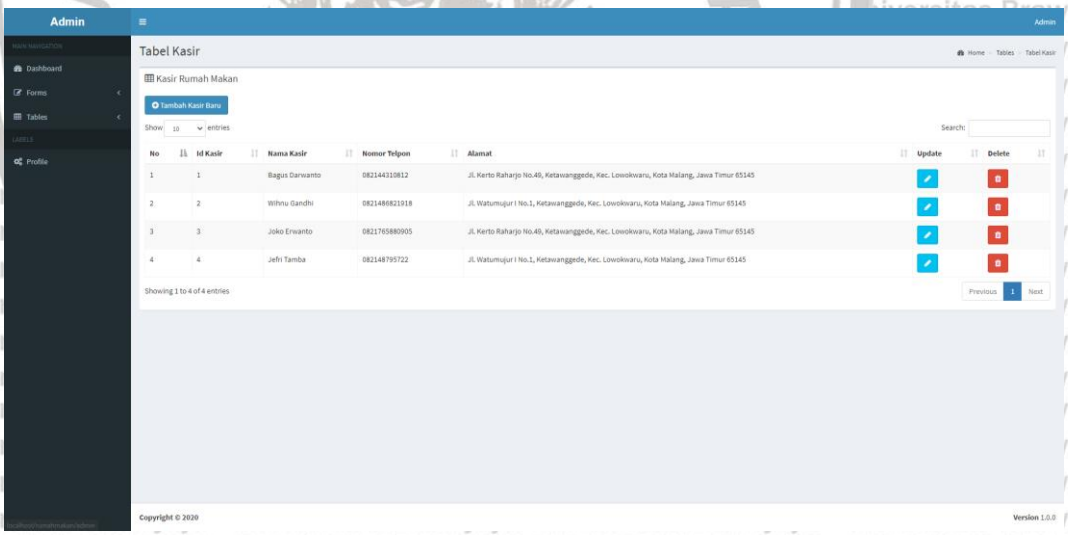
Gambar 5.15 merupakan implementasi antarmuka halaman tambah menu. Pada halaman ini terdapat *header* berupa nama pengguna dan tombol *logout*, serta *sidebar menu* yang berisi navigasi beserta informasi nama dan keterangan peran pegguna. Terdapat satu panel formulir yang terdiri dari *textfield* nama menu dan harga menu serta sebuah *options* untuk bahan baku. Ada tiga buah tombol yaitu tombol kembali, tombol lihat *list* menu dan tombol *submit*.



Gambar 5.16 Implementasi antarmuka halaman tambah menu

5.2.3.4 Halaman hapus kasir

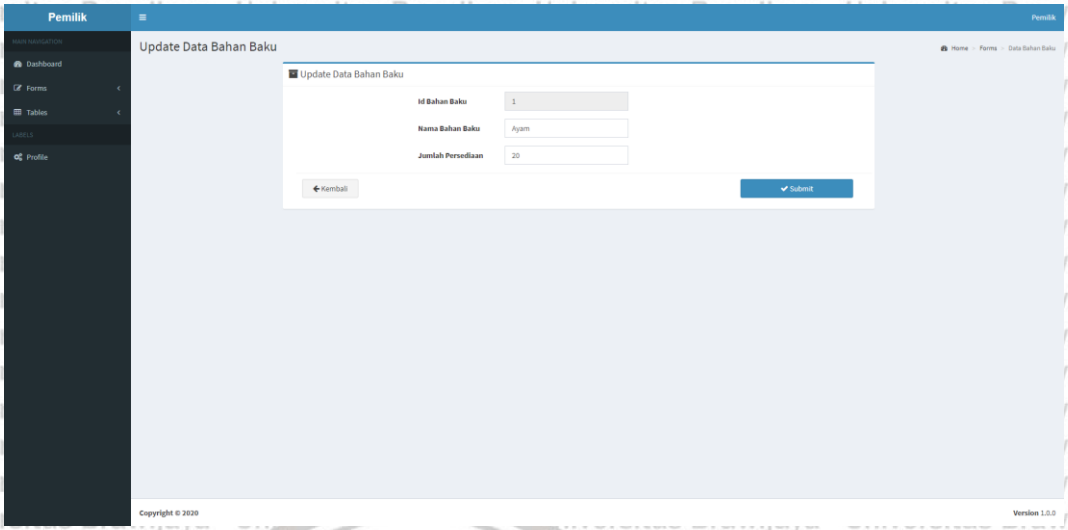
Gambar 5.16 merupakan implementasi antarmuka halaman hapus kasir. Pada halaman ini terdapat *header* berupa nama pengguna dan tombol *logout*, serta *sidebar menu* yang berisi navigasi beserta informasi nama dan keterangan peran pegguna. Terdapat satu panel tabel yang terdiri dari *textfield* nomor, id kasir, nama kasir, nomor telpon dan alamat serta tombol untuk *update* dan *delete*.



Gambar 5.17 Implementasi antarmuka halaman hapus kasir

5.2.3.5 Halaman ubah bahan baku

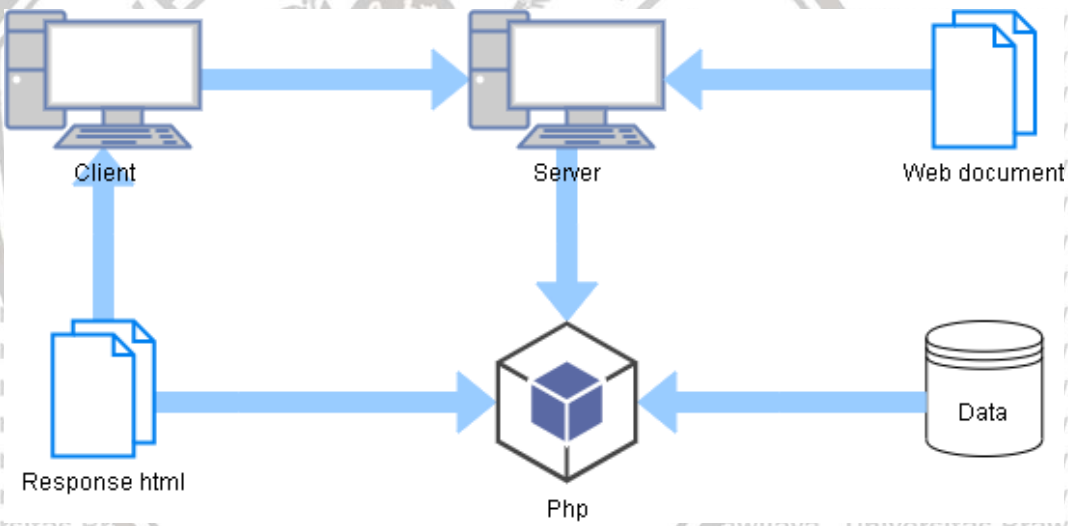
Gambar 5.17 merupakan implementasi antarmuka halaman ubah bahan baku. Pada halaman ini terdapat *header* berupa nama pengguna dan tombol *logout*, serta *sidebar menu* yang berisi navigasi beserta informasi nama dan keterangan peran pegguna. Terdapat satu panel formulir yang terdiri dari *textfield* nama bahan baku dan jumlah persediaan. Ada dua buah tombol yaitu tombol kembali dan tombol *submit*.



Gambar 5.18 Implementasi antarmuka halaman ubah bahan baku

5.2.4 Implementasi arsitektur

Implementasi arsitektur dilakukan berdasarkan hasil perancangan arsitektur. Gambar 5.19 merupakan hasil implementasi arsitektur.



Gambar 5.19 Implementasi arsitektur

BAB 6 PENGUJIAN

6.1 Pengujian Unit

Pengujian unit dilakukan pada tiga sampel algoritme, yaitu algortime *method* proses_pesan_baru_insert ( ) dari klas Kasir, algoritme *method* proses_menu_baru_insert ( ) dari klas Pemilik dan algoritme *method* proses_bahanbaku_update ( ) dari klas Pemilik. Klas driver digunakan sebagai klas pengganti karena klas-klas yang akan diuji tidak memiliki *main method*.

6.1.1 Pengujian algoritme *method* proses_pesan_baru_insert ( )

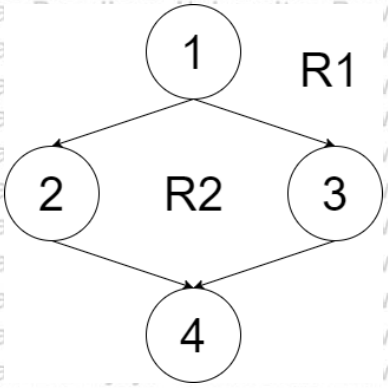
Method proses_pesan_baru_insert ( ) berfungsi untuk menambahkan pesanan ke basis data. Tabel 6.1 adalah algoritme pada *method* proses_pesan_baru_insert ( ).

Tabel 6.1 Algoritme *method* proses_pesan_baru_insert ( )

```
START
    data = array(
        'nama_pemesan' => $nama_pemesan,
        'jumlah_pesanan' => $jumlah_pesanan,
        'menu_pesan' => $nama_menu,
        'nomor_meja' => $nomor_meja,
    );
    query insert data
    cek data
    if data exist {
        return true
    } else {
        return false
    }
END
```

1
2
3
4

1. Flow graph



Gambar 6.1 *Flow graph* algoritme *method* proses_pesan_baru_insert ( )

2. *Cyclomatic complexity*

   V (G) = R = 2

   V (G) = E – N + 2 = 4 – 4 + 2 = 2

   V (G) = P + 1 = 1 + 1 = 2

3. *Independet path*

   i. Jalur 1: 1 – 2 – 4

   ii. Jalur 2: 1 – 3 – 4

Tabel 6.2 adalah kasus uji coba algoritme pada *method* proses_pesan_baru_insert ( ).

Tabel 6.2 Kasus uji algoritme pada *method* proses_pesan_baru_insert ( )

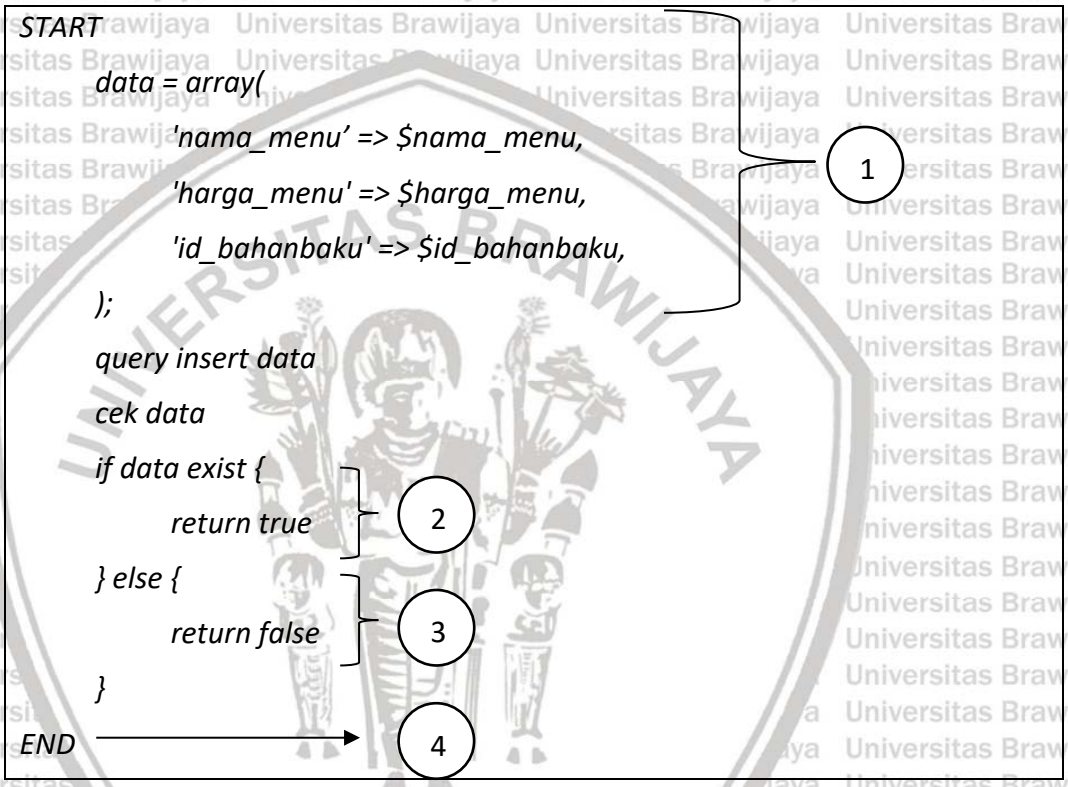
Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	Klas *driver* menjalankan *method* proses_pesan_baru_insert ( ) dengan nilai: nama_pemesan = 'Budi', jumlah_pesanan = '2', menu_pesan = 'Nasi Ayam Bakar ', nomor_meja = '1' *data exist* = *true*	Mengembalikan nilai *true*	Mengembalikan nilai *true*	Valid
2	Klas *driver* menjalankan *method* proses_pesan_baru_insert ( ) dengan nilai: nama_pemesan = 'Budi', jumlah_pesanan = '2',	Mengembalikan nilai *false*	Mengembalikan nilai *false*	Valid

	menu_pesan = 'Nasi Ayam Bakar ', nomor_meja = '1' *data exist = false*			

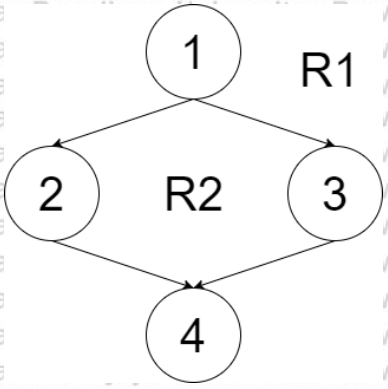
6.1.2 Pengujian algoritme *method* proses_menu_baru_insert ( )

Method proses_menu_baru_insert ( ) berfungsi untuk menambahkan menu ke basis data. Tabel 6.3 adalah algoritme pada *method* proses_menu_baru_insert ( ).

Tabel 6.3 Algoritme *method* proses_menu_baru_insert ( )



1. *Flow graph*



Gambar 6.2 *Flow graph* algoritme *method* proses_menu_baru_insert ( )

2. *Cyclomatic complexity*

   V (G) = R = 2

   V (G) = E – N + 2 = 4 – 4 + 2 = 2

   V (G) = P + 1 = 1 + 1 = 2

3. *Independet path*

   i. Jalur 1: 1 – 2 – 4

   ii. Jalur 2: 1 – 3 – 4

Tabel 6.2 adalah kasus uji coba algoritme pada *method* proses_menu_baru_insert ( ).

Tabel 6.4 Kasus uji algoritme pada *method* proses_menu_baru_insert ( )

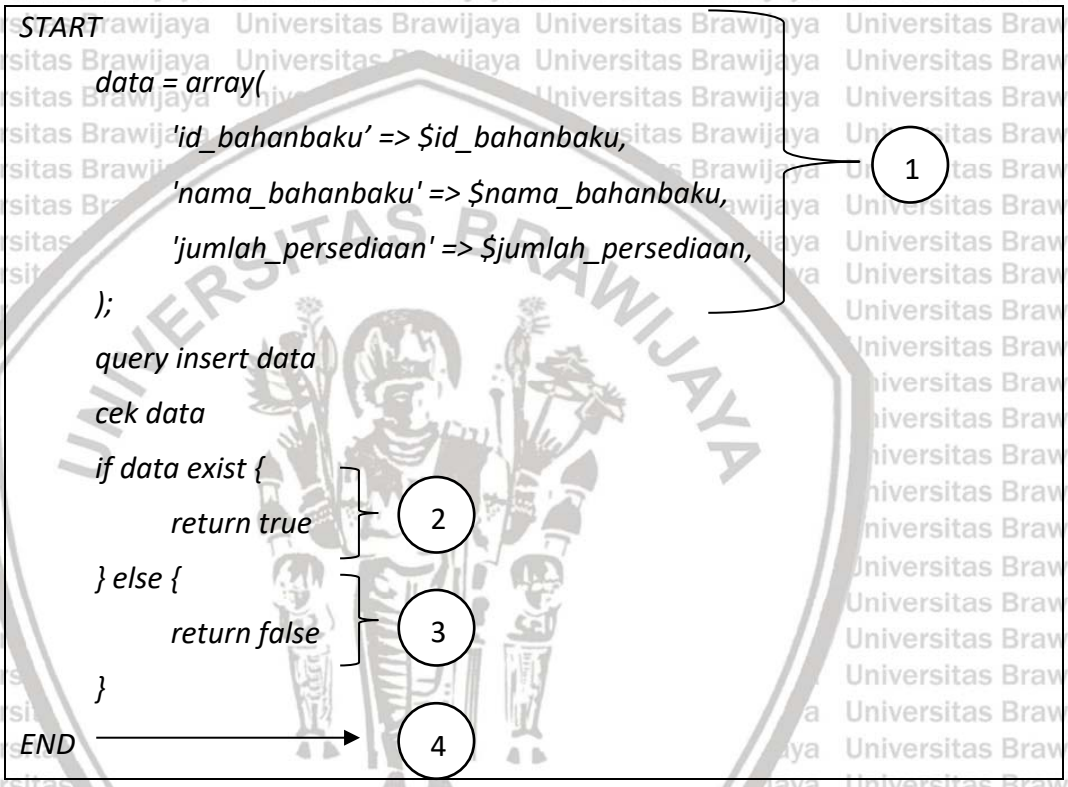
Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	Klas *driver* menjalankan *method* proses_menu_baru_insert ( ) dengan nilai: nama_menu = 'Nasi Telur', harga_menu = '10000', id_bahanbaku = '3' *data exist* = *true*	Mengembalikan nilai *true*	Mengembalikan nilai *true*	Valid
2	Klas *driver* menjalankan *method* proses_menu_baru_insert ( ) dengan nilai: nama_menu = 'Nasi Telur', harga_menu =	Mengembalikan nilai *false*	Mengembalikan nilai *false*	Valid

	'10000', id_bahanbaku = '3' *data exist = false*				

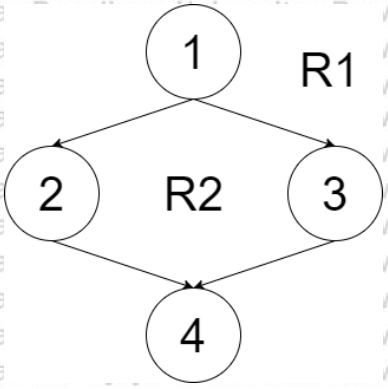
6.1.3 Pengujian algoritme *method* proses_bahanbaku_update ( )

Method proses_bahanbaku_update ( ) berfungsi untuk mengubah bahan baku di basis data. Tabel 6.5 adalah algoritme pada *method* proses_bahanbaku_update ( ).

Tabel 6.5 Algoritme *method* proses_bahanbaku_update ( )



1. *Flow graph*



Gambar 6.3 *Flow graph* algoritme *method* proses_bahanbaku_update ( )

2. *Cyclomatic complexity*

   V (G) = R = 2

   V (G) = E – N + 2 = 4 – 4 + 2 = 2

   V (G) = P + 1 = 1 + 1 = 2

3. *Independet path*

   i. Jalur 1: 1 – 2 – 4

   ii. Jalur 2: 1 – 3 – 4

Tabel 6.6 adalah kasus uji coba algoritme pada *method* proses_bahanbaku_update ( ).

Tabel 6.6 Kasus uji algoritme pada *method* proses_bahanbaku_update ( )

Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	Klas *driver* menjalankan *method* proses_bahanbaku_update ( ) dengan nilai: id_bahanbaku = ‘1 ’, nama_bahanbaku = ‘Ayam’, jumlah_persediaan = ‘20’ *data exist* = *true*	Mengembalikan nilai *true*	Mengembalikan nilai *true*	Valid
2	Klas *driver* menjalankan *method* proses_bahanbaku_update ( ) dengan nilai: id_bahanbaku = ‘1 ’, nama_bahanbaku =	Mengembalikan nilai *false*	Mengembalikan nilai *false*	Valid

	‘Ayam’, jumlah_persediaan = ‘20’ *data exist = false*			

6.2 Pengujian Integrasi

Pengujian integrasi dilakukan pada 3 sampel algoritme, yaitu algoritme pada *method* get_daftar_menu ( ) dari klas Daftar_menu, *method* get_daftar_bahanbaku ( ) dari klas Daftar_bahanbaku dan *method* get_jumlah_menu dari klas Menu.

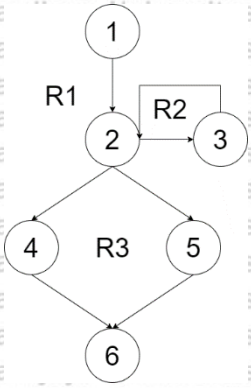
6.2.1 Pengujian algoritme *method* get_daftar_menu ( )

Tabel 6.7 merupakan algoritme *method* get_daftar_menu ( ) dari klas Daftar_menu.

Tabel 6.7 Algoritme *method* get_daftar_menu ( )

```
START
    inisialisasi variable jumlah_isi_daftar = jumlah baris tabel menu      1
    inisialisasi variabel i = ‘1’
    while i kurang dari sama dengan jumlah_isi_daftar {                    2
        instansiasi objek menu
        inisialisasi array isi_daftar = objek menu                         3
        penambahann nilai variabel i
    }
    if status berhasil {                                                   4
        pesan berhasil
    } else {                                                               5
        Pesan gagal
    }
END                                                                        6
```

1. Flow graph



Gambar 6.4 *Flow graph* algoritme *method* get_daftar_menu ( )

2. *Cyclomatic complexity*

V (G) = R = 3

V (G) = E – N + 2 = 7 – 6 + 2 = 3

V (G) = P + 1 = 2 + 1 = 3

3. *Independet path*

i. Jalur 1: 1 – 2 – 4 – 6

ii. Jalur 2: 1 – 2 – 3 – 2 – 4 – 6

iii. Jalur 2: 1 – 2 – 3 – 2 – 5 – 6

Tabel 6.8 adalah kasus uji coba algoritme pada *method* get_daftar_menu ( ).

Tabel 6.8 Kasus uji algoritme *method* get_daftar_menu ( )

Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	Klas *driver* menjalankan *method* get_daftar_menu ( ) dengan nilai: jumlah_isi_daftar = '0 ', *status* = *true*	Mengembalikan nilai *true*	Mengembalikan nilai *true*	Valid
2	Klas *driver* menjalankan *method* get_daftar_menu ( ) dengan nilai: jumlah_isi_daftar = '1', *status* = *true*	Mengembalikan nilai *true*	Mengembalikan nilai *true*	Valid
3	Klas *driver* menjalankan *method*	Mengembalikan nilai *false*	Mengembalikan nilai *false*	Valid

	get_daftar_menu ( ) dengan nilai: jumlah_isi_daftar = '1', *status = false*			

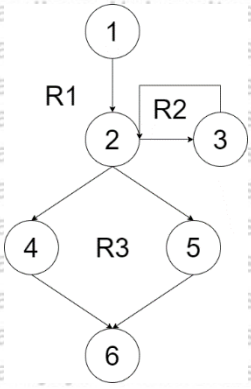
6.2.2 Pengujian algoritme *method* get_daftar_bahanbaku ( )

Tabel 6.9 merupakan algoritme *method* get_daftar_bahanbaku ( ) dari klas Daftar_bahanbaku.

Tabel 6.9 Algoritme *method* get_daftar_bahanbaku ( )

```
START
    inisialisasi variable jumlah_isi_daftar =
    jumlah baris tabel bahan baku                                    1
    inisialisasi variabel i = '1'
    while i kurang dari sama dengan jumlah_isi_daftar {              2
        instansiasi objek bahan baku
        inisialisasi array isi_daftar = objek bahan baku             3
        penambahann nilai variabel i
    }
    if status berhasil {                                             4
        pesan berhasil
    } else {                                                         5
        Pesan gagal
    }
END                                                                  6
```

1. *Flow graph*



Gambar 6.5 *Flow graph* algoritme *method* get_daftar_bahanbaku ( )

2. *Cyclomatic complexity*

V (G) = R = 3

V (G) = E – N + 2 = 7 – 6 + 2 = 3

V (G) = P + 1 = 2 + 1 = 3

3. *Independet path*

i. Jalur 1: 1 – 2 – 4 – 6

ii. Jalur 2: 1 – 2 – 3 – 2 – 4 – 6

iii. Jalur 2: 1 – 2 – 3 – 2 – 5 – 6

Tabel 6.10 adalah kasus uji coba algoritme pada *method* get_daftar_bahanbaku ( ).

Tabel 6.10 Kasus uji algoritme *method* get_daftar_bahanbaku ( )

Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	Klas *driver* menjalankan *method* get_daftar_bahanbaku ( ) dengan nilai: jumlah_isi_daftar = '0 ', *status* = *true*	Mengembalikan nilai *true*	Mengembalikan nilai *true*	Valid
2	Klas *driver* menjalankan *method* get_daftar_bahanbaku ( ) dengan nilai: jumlah_isi_daftar = '1', *status* = *true*	Mengembalikan nilai *true*	Mengembalikan nilai *true*	Valid

3	Klas *driver* menjalankan *method* get_daftar_bahanbaku ( ) dengan nilai: jumlah_isi_daftar = '1', *status = false*	Mengembalikan nilai *false*	Mengembalikan nilai *false*	Valid

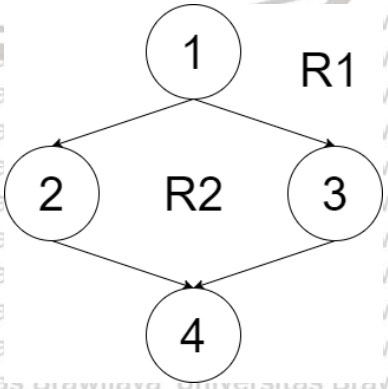
6.2.3 Pengujian algoritme *method* get_jumlah_menu ( )

Tabel 6.11 merupakan algoritme *method* get_jumlah_menu ( ) dari klas Menu.

Tabel 6.11 Algoritme *method* get_jumlah_menu ( )

```
START
    instansiasi objek bahan baku
    inisialisasi variabel jumlah_menu =
    objek bahan baku memanggil method get_jumlah_bahanbaku   (1)
    query select
    cek data
    if data exist {                                          (2)
        return true
    } else {                                                 (3)
        return false
    }                                                        (4)
END
```

1. *Flow graph*



Gambar 6.6 *Flow graph* algoritme *method* get_jumlah_menu ( )

2. *Cyclomatic complexity*

V (G) = R = 2

V (G) = E – N + 2 = 4 – 4 + 2 = 2

V (G) = P + 1 = 1 + 1 = 2

3. *Independet path*

i. Jalur 1: 1 – 2 – 4

ii. Jalur 2: 1 – 3 – 4

Tabel 6.12 adalah kasus uji coba algoritme pada *method* get_jumlah_menu ( ).

Tabel 6.12 Kasus uji algoritme pada *method* proses_bahanbaku_update ( )

Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	Klas *driver* menjalankan *method* get_jumlah_menu ( ) dengan nilai: Jumlah_menu = '1' *data exist = true*	Mengembalikan nilai *true*	Mengembalikan nilai *true*	Valid
2	Klas *driver* menjalankan *method* get_jumlah_menu ( ) dengan nilai: Jumlah_menu = '0' *data exist = false*	Mengembalikan nilai *false*	Mengembalikan nilai *false*	Valid

6.3 Pengujian Validasi

Pengujian validasi dilakukan untuk mengetahui apakah fungsional sistem sudah sesuai dengan analisis kebutuhan atau tidak. Pengujian validasi dilakukan pada semua kebutuhan fungsional sistem. Kebutuhan fungsional pada sistem berjumlah enam belas kebutuhan fungsional.

6.3.1 Pengujian validasi *login*

Kasus uji pada pengujian validasi *login* berjumlah tiga kasus uji, yaitu kasus uji jalur utama dan kasus uji alternative 2a dan 2b. Pengujian validasi *login* dapat dilihat pada Tabel 6.13 hingga Tabel 6.15.

Tabel 6.13 Kasus uji *login* jalur utama

Kode kebutuhan	MPBB_001_00
Nama kasus uji	*Login* jalur utama
Prosedur pengujian	1. Mengakses halaman *login* 2. Mengisi form *login* dengan *username* = Admin dan *password* = 123456 3. Menekan tombol *login*
Hasil yang diharapkan	Aktor berhasil masuk ke halaman *index* sesuai dengan peranan masing-masing
Hasil yang didapatkan	Aktor berhasil masuk ke halaman *index* sesuai dengan peranan masing-masing
Status validasi	Valid

Tabel 6.14 Kasus uji login jalur alternatif 2a

Kode kebutuhan	MPBB_001_00
Nama kasus uji	*Login* jalur alternatif 2a
Prosedur pengujian	1. Mengakses halaman *login* 2. Mengisi form *login* dengan *username* = Admin dan *password* = admin 3. Menekan tombol *login*
Hasil yang diharapkan	Sistem menampilkan pesan gagal
Hasil yang didapatkan	Sistem menampilkan pesan gagal
Status validasi	Valid

Tabel 6.15 Kasus uji login jalur alternatif 2b

Kode kebutuhan	MPBB_001_00
Nama kasus uji	*Login* jalur alternatif 2b
Prosedur pengujian	1. Mengakses halaman *login* 2. Mengisi form *login* dengan *username* = Admin dan *password* = 3. Menekan tombol *login*
Hasil yang diharapkan	Sistem menampilkan pesan *required*
Hasil yang didapatkan	Sistem menampilkan pesan *required*
Status validasi	Valid

6.3.2 Pengujian validasi keluar

Kasus uji pada pengujian validasi keluar berjumlah satu kasus uji, yaitu kasus uji untuk jalur utama. Pengujian validasi keluar dapat dilihat pada Tabel 6.16.

Tabel 6.16 Kasus uji keluar jalur utama

Kode kebutuhan	MPBB_002_00
Nama kasus uji	Keluar jalur utama
Prosedur pengujian	1. Mengakses halaman *login* 2. Melakukan proses *login* dengan benar 3. Menekan tombol keluar
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

6.3.3 Pengujian validasi buat pesanan

Kasus uji pada pengujian validasi buat pesanan berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi buat pesanan dapat dilihat pada Tabel 6.17 dan Tabel 6.18.

Tabel 6.17 Kasus uji buat pesanan jalur utama

Kode kebutuhan	MPBB_003_00
Nama kasus uji	Buat pesanan jalur utama
Prosedur pengujian	1. Mengakses halaman daftar menu 2. Memilih menu 3. Mengisi *form* pesan 4. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.18 Kasus uji buat pesanan jalur alternatif 4a

Kode kebutuhan	MPBB_003_00
Nama kasus uji	Buat pesanan jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman daftar menu 2. Memilih menu 3. Mengisi *form* pesan 4. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.4 Pengujian validasi ubah persediaan bahan baku

Kasus uji pada pengujian validasi ubah persediaan bahan baku berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi ubah persediaan bahan baku dapat dilihat pada Tabel 6.19 dan Tabel 6.20.

Tabel 6.19 Kasus uji ubah persediaan bahan baku jalur utama

Kode kebutuhan	MPBB_004_00
Nama kasus uji	Ubah persediaan bahan baku jalur utama
Prosedur pengujian	1. Mengakses halaman daftar bahan baku 2. Memilih bahan baku 3. Mengisi *form* bahan baku 4. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.20 Kasus uji ubah persediaan bahan baku jalur alternatif 4a

Kode kebutuhan	MPBB_004_00
Nama kasus uji	Ubah persediaan bahan baku jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman daftar bahan baku 2. Memilih bahan baku 3. Mengisi *form* bahan baku 4. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.5 Pengujian validasi tambah cabang

Kasus uji pada pengujian validasi tambah cabang berjumlah tiga kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a dan 4b. Pengujian validasi tambah cabang dapat dilihat pada Tabel 6.21 dan Tabel 6.23.

Tabel 6.21 Kasus uji tambah cabang jalur utama

Kode kebutuhan	MPBB_005_00
Nama kasus uji	Tambah cabang jalur utama
Prosedur pengujian	1. Mengakses halaman tabel cabang 2. Menekan tombol tambah 3. Mengisi *form* tambah 4. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.22 Kasus uji tambah cabang jalur alternatif 4a

Kode kebutuhan	MPBB_005_00
Nama kasus uji	Tambah cabang jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel cabang 2. Menekan tombol tambah 3. Mengisi *form* tambah 4. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

Tabel 6.23 Kasus uji tambah cabang jalur alternatif 4b

Kode kebutuhan	MPBB_005_00
Nama kasus uji	Tambah cabang jalur alternatif 4b
Prosedur pengujian	1. Mengakses halaman tabel cabang 2. Menekan tombol tambah 3. Mengisi *form* tambah dengan nama cabang = Cabang 3, nomor telpon = 085816613787 dan alamat = 4. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan *required*
Hasil yang didapatkan	Sistem menampilkan pesan *required*
Status validasi	Valid

6.3.6 Pengujian validasi hapus cabang

Kasus uji pada pengujian validasi hapus cabang berjumlah satu kasus uji, yaitu kasus uji jalur utama. Pengujian validasi hapus cabang dapat dilihat pada Tabel 6.24.

Tabel 6.24 Kasus uji hapus cabang jalur utama

Kode kebutuhan	MPBB_006_00
Nama kasus uji	Hapus cabang jalur utama
Prosedur pengujian	1. Mengakses halaman tabel cabang 2. Memilih cabang 3. Menekan tombol *delete*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

6.3.7 Pengujian validasi ubah cabang

Kasus uji pada pengujian validasi ubah cabang berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi ubah cabang dapat dilihat pada Tabel 6.25 dan Tabel 6.26.

Tabel 6.25 Kasus uji ubah cabang jalur utama

Kode kebutuhan	MPBB_007_00
Nama kasus uji	Ubah cabang jalur utama
Prosedur pengujian	1. Mengakses halaman tabel cabang 2. Memilih cabang 3. Menekan tombol *update* 4. Mengisi form *update* 5. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.26 Kasus uji ubah cabang jalur alternatif 4a

Kode kebutuhan	MPBB_007_00
Nama kasus uji	Ubah cabang jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel cabang 2. Memilih cabang 3. Menekan tombol *update* 4. Mengisi form *update* 5. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.8 Pengujian validasi unduh laporan cabang

Kasus uji pada pengujian validasi unduh laporan cabang berjumlah satu kasus uji, yaitu kasus uji jalur utama. Pengujian validasi unduh laporan cabang dapat dilihat pada Tabel 6.27.

Tabel 6.27 Kasus uji unduh laporan cabang jalur utama

Kode kebutuhan	MPBB_008_00
Nama kasus uji	Unduh laporan cabang jalur utama
Prosedur pengujian	1. Mengakses halaman tabel laporan transaksi 2. Memilih laporan transaksi 3. Menekan *link download*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

6.3.9 Pengujian validasi tambah laporan cabang

Kasus uji pada pengujian validasi tambah laporan cabang berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi tambah laporan cabang dapat dilihat pada Tabel 6.28 dan Tabel 6.29.

Tabel 6.28 Kasus uji tambah laporan cabang jalur utama

Kode kebutuhan	MPBB_009_00
Nama kasus uji	Tambah laporan cabang jalur utama
Prosedur pengujian	1. Mengakses halaman tabel laporan transaksi 2. Menekan tombol tambah 3. Mengisi form tambah 4. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.29 Kasus uji tambah laporan cabang jalur alternatif 4a

Kode kebutuhan	MPBB_009_00
Nama kasus uji	Tambah laporan cabang jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel laporan transaksi 2. Menekan tombol tambah 3. Mengisi form tambah 4. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.10 Pengujian validasi tambah kasir

Kasus uji pada pengujian validasi tambah kasir berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi tambah kasir dapat dilihat pada Tabel 6.30 dan Tabel 6.31.

Tabel 6.30 Kasus uji tambah kasir jalur utama

Kode kebutuhan	MPBB_010_00
Nama kasus uji	Tambah kasir jalur utama
Prosedur pengujian	1. Mengakses halaman tabel kasir 2. Menekan tombol tambah 3. Mengisi form tambah 4. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.31 Kasus uji tambah kasir jalur alternatif 4a

Kode kebutuhan	MPBB_010_00
Nama kasus uji	Tambah kasir jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel kasir 2. Menekan tombol tambah 3. Mengisi form tambah 4. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.11 Pengujian validasi hapus kasir

Kasus uji pada pengujian validasi hapus kasir berjumlah satu kasus uji, yaitu kasus uji jalur utama. Pengujian validasi hapus kasir dapat dilihat pada Tabel 6.32.

Tabel 6.32 Kasus uji hapus kasir jalur utama

Kode kebutuhan	MPBB_011_00
Nama kasus uji	Hapus kasir jalur utama
Prosedur pengujian	1. Mengakses halaman tabel kasir 2. Memilih kasir 3. Menekan tombol *delete*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

6.3.12 Pengujian validasi ubah kasir

Kasus uji pada pengujian validasi ubah kasir berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi ubah kasir dapat dilihat pada Tabel 6.33 dan Tabel 6.34.

Tabel 6.33 Kasus uji ubah kasir jalur utama

Kode kebutuhan	MPBB_012_00
Nama kasus uji	Ubah kasir jalur utama
Prosedur pengujian	1. Mengakses halaman tabel kasir 2. Memilih kasir 3. Menekan tombol *update* 4. Mengisi form *update* 5. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.34 Kasus uji ubah kasir jalur alternatif 4a

Kode kebutuhan	MPBB_012_00
Nama kasus uji	Ubah kasir jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel kasir 2. Memilih kasir 3. Menekan tombol *update* 4. Mengisi form *update* 5. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.13 Pengujian validasi tambah menu

Kasus uji pada pengujian validasi tambah menu berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi tambah menu dapat dilihat pada Tabel 6.35 dan Tabel 6.36.

Tabel 6.35 Kasus uji tambah menu jalur utama

Kode kebutuhan	MPBB_013_00
Nama kasus uji	Tambah menu jalur utama
Prosedur pengujian	1. Mengakses halaman tabel menu 2. Menekan tombol tambah 3. Mengisi form tambah 4. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.36 Kasus uji tambah menu jalur alternatif 4a

Kode kebutuhan	MPBB_013_00
Nama kasus uji	Tambah menu jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel menu 2. Menekan tombol tambah 3. Mengisi form tambah 4. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.14 Pengujian validasi ubah menu

Kasus uji pada pengujian validasi ubah menu berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi ubah menu dapat dilihat pada Tabel 6.37 dan Tabel 6.38.

Tabel 6.37 Kasus uji ubah menu jalur utama

Kode kebutuhan	MPBB_014_00
Nama kasus uji	Ubah menu jalur utama
Prosedur pengujian	1. Mengakses halaman tabel menu 2. Memilih menu 3. Menekan tombol *update* 4. Mengisi form *update* 5. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.38 Kasus uji ubah menu jalur alternatif 4a

Kode kebutuhan	MPBB_014_00
Nama kasus uji	Ubah menu jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel menu 2. Memilih menu 3. Menekan tombol *update* 4. Mengisi form *update* 5. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.15 Pengujian validasi ubah menu

Kasus uji pada pengujian validasi ubah menu berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi ubah menu dapat dilihat pada Tabel 6.39 dan Tabel 6.40.

Tabel 6.39 Kasus uji ubah menu jalur utama

Kode kebutuhan	MPBB_014_00
Nama kasus uji	Ubah menu jalur utama
Prosedur pengujian	1. Mengakses halaman tabel menu 2. Memilih menu 3. Menekan tombol *update* 4. Mengisi form *update* 5. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.40 Kasus uji ubah menu jalur alternatif 4a

Kode kebutuhan	MPBB_014_00
Nama kasus uji	Ubah menu jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel menu 2. Memilih menu 3. Menekan tombol *update* 4. Mengisi form *update* 5. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.16 Pengujian validasi tambah bahan baku

Kasus uji pada pengujian validasi tambah bahan baku berjumlah dua kasus uji, yaitu kasus uji jalur utama dan kasus uji alternatif 4a. Pengujian validasi tambah bahan baku dapat dilihat pada Tabel 6.41 dan Tabel 6.42.

Tabel 6.41 Kasus uji tambah bahan baku jalur utama

Kode kebutuhan	MPBB_015_00
Nama kasus uji	Tambah bahan baku jalur utama
Prosedur pengujian	1. Mengakses halaman tabel bahan baku 2. Menekan tombol tambah 3. Mengisi form tambah 4. Menekan tombol *submit*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

Tabel 6.42 Kasus uji tambah bahan baku jalur alternatif 4a

Kode kebutuhan	MPBB_015_00
Nama kasus uji	Tambah bahan baku jalur alternatif 4a
Prosedur pengujian	1. Mengakses halaman tabel bahan baku 2. Menekan tombol tambah 3. Mengisi form tambah 4. Menekan tombol kembali
Hasil yang diharapkan	Sistem menampilkan halaman sebelumnya
Hasil yang didapatkan	Sistem menampilkan halaman sebelumnya
Status validasi	Valid

6.3.17 Pengujian validasi hapus bahan baku

Kasus uji pada pengujian validasi hapus bahan baku berjumlah satu kasus uji, yaitu kasus uji jalur utama. Pengujian validasi hapus bahan baku dapat dilihat pada Tabel 6.43.

Tabel 6.43 Kasus uji hapus bahan baku jalur utama

Kode kebutuhan	MPBB_016_00
Nama kasus uji	Hapus bahan baku jalur utama
Prosedur pengujian	1. Mengakses halaman tabel bahan baku 2. Memilih bahan baku 3. Menekan tombol *delete*
Hasil yang diharapkan	Sistem menampilkan pesan berhasil
Hasil yang didapatkan	Sistem menampilkan pesan berhasil
Status validasi	Valid

BAB 7 KESIMPULAN DAN SARAN

7.1 Kesimpulan

Kesimpulan yang didapat dari penelitian ini yaitu:

1. Tahap rekayasa kebutuhan menghasilkan tiga aktor yaitu admin, pemilik dan kasir. Kebutuhan fungsional yang dihasilkan sebanyak enam belas fungsional. Beberapa kebutuhan fungsional utama yaitu buat pesanan, ubah persediaan bahan baku, unduh laporan cabang dan tambah cabang.
2. Tahap perancangan menghasilkan *sequence diagram* dan *class diagram* pada perancangan arsitektur, *pseudocode* pada perancangan komponen, *conceptual data model* (CDM) dan *physical data model* (PDM) pada perancangan data dan rancangan antarmuka pada perancangan antarmuka.
3. Tahap implementasi menghasilkan sistem manajemen persediaan bahan baku sesuai dengan kebutuhan dan perancangan. Sistem dibangun dengan menggunakan bahasa pemrograman PHP dengan *framework* CodeIgniter.
4. Tahap pengujian menggunakan strategi pengujian unit, integrasi dan validasi. Pengujian unit dan integrasi menggunakan teknik *white-box testing* sedangkan pengujian validasi menggunakan teknik *black-box testing*. Pengujian unit dilakukan pada tiga sampel dan mendapatkan hasil valid. Pengujian integrasi dilakukan pada tiga sampel dan mendapatkan hasil valid. Pengujian validasi dilakukan pada enam belas fungsional dan mendapatkan hasil valid.

7.2 Saran

Saran yang diberikan untuk penelitian berikutnya adalah mengembangkan sistem dengan menambahkan fungsi prediksi persediaan bahan baku untuk mempermudah pengguna mengelola persediaan bahan baku.

DAFTAR REFERENSI

Bassil, Y., 2012. *A Simulation Model for the Waterfall Software Development Life Cycle*. International Journal of Engineering & Technology (iJET), II (5)

Blanco, J. A. & Upton, D., 2009. *Codeigniter 1.7*. Birmingham: Packt Publishing.

Booch, G., 1998. *The unified modeling language user guide*. USA: Addison-Wesley Professional

Carter, A., 2001. *Evolving Beyond Requirements Creep: A Risk-Based Evolutionary Prototyping Model*. Raleigh: North Carolina State University

Damodaran, D. B., Shirin S. & Surekha M. V., 2016. *Performance Evaluation of MySQL and MongoDB Databases*. International Journal on Cybernetics & Informatics (IJCI). 5(2).

Date, C. J., 2010. *An Introduction to Database Systems 8th Edition*. USA: Darthmouth Publishing, Inc.

Davin, A., 2016. Sistem Informasi Penjualan Menu Makanan Dan Persediaan Bahan Baku Berbasis Web Pada Koffie Tijd Kafe Dan Resto. *UNIKOM Repository*, [online] Tersedia di: <https://repository.unikom.ac.id/16273/> [Diakses 1 Maret 2019]

Ebert, C., 2016. *Cyclomatic Complexity*. IEEE. 33(6): 27-29

Guru99, 2019. *What Is Data Modelling? Conceptual, Logical, & Physical Data Models*. [Online] Tersedia di: <https://www.guru99.com/data-modellingconceptual-logical.html> [Diakses 11 Januari 2020]

Hamilton, R., 2002. *Wink and Grow Rich*. New York: Wealth Dynamics

Handoko, H., 1999. *Manajemen*. Yogyakarta: BPFE Yogyakarta.

Handoko, H., 2000. *Manajemen Sumber Daya Manusia*. Yogyakarta: BPFE Yogyakarta,

Hutomo, A., 2014. *Pengembangan Aplikasi Android Kamus Command Line (FYComm) Sebagai Media Bantu Belajar Siswa SMK Negeri 1 Bantul Kompentensi Keahlian Teknik Komputer Dan Jaringan*. Universitas Negeri Yogyakarta

Jailia, M., dkk., 2016. *Behavior of MVC (Model View Controller) based Web Application developed in PHP and .NET framework.* India : Banasthali Vidyapith Rajasthan

Keputusan Menteri Pariwisata, Pos dan Telekomunikasi No.KN.73/PVVI05/MPPT-85 tentang Peraturan usaha Rumah Makan. Jakarta: Kementerian Sekretariat Negara Republik Indonesia.

Kurniawan, H., 2015. SISTEM APLIKASI PERSEDIAAN BAHAN BAKU PENYUSUN MAKANAN DIRUMAH MAKAN PODO JOYO. *Academia*, [online] Tersedia di: < http://www.academia.edu/15193611/ > [Diakses 1 Maret 2019]

Kurniawan, Tri A., 2018. *Pemodelan Use Case (UML): Evaluasi Terhadap Beberapa Kesalahan Dalam Praktik*. JTIIK. 5(1), pp.77–86.

Marsum, A., 2005. *Restoran dan Segala Permasalahannya*. Edisi Empat. Yogyakarta: Andi.

Mathiassen, L., dkk., 2000. *Object-oriented Analysis & Design*. Michigan: Marko Publishing

Mulyadi, 1999. *Akuntansi Biaya*, Edisi kelima. Yogyakarta: Universitas Gadjah Mada.

Mukherjee, M., 2016. *Object-Oriented Analysis and Design*. Scienpress Ltd. 1(1), pp.1-11.

Okamoto, S., Sergiu, D. & Dwight, E., 2006. *Web Interface Development Environment (WIDE): Software Tool for Automatic Generation of Web Application Interfaces*. Budapest: WAC

Onu, F. U., Uche N. E. O. & Chigbundu K. E., 2016. *The Role of Object-Oriented Programming (OOP) in Modeling of Geographic Information Systems (GIS)*. IOSR Journal of Mobile Computing & Application. 3(4), pp.40-46.

Pratama, N. W., 2010. *Codeigniter: Cara Mudah Membangun Aplikasi PHP*. Jakarta: Mediakita

Pressman, R. S., 2001. *Software Engineering: A Practitioner's Approach, Fifth Ed*. New York: McGraw-Hill Book Company.

Pressman, R. S., 2010. *Rekayasa Perangkat Lunak (Pendekatan Praktisi). 7 ed*. Yogyakarta: Andi.

Rumbaugh, J., Jacobson, I. & Booch, G., 2005. *The Unified Modeling Language reference manual*. 2nd ed. Boston: Addison-Wesley.

Siagian, M., 2005. *Supply Chain Management Dalam Dunia Bisnis*. Jakarta: PT. Grasindo.

Simarmata, J., 2010. Rekayasa Web. Yogyakarta: Andi Offset.

Sommerville, I., 2011. *Software Engineering 9th edition*. Boston: Pearson Education.

Sugiarto, E., & Sri, S., 2001. *Pengantar akomodasi dan restoran*. Jakarta: Gramedia Pustaka Utama

Yicheng, L., 2011. *Development of a Blog System Using Codeigniter Framework*, s.l.: Oulu University of Applied Sciences.

LAMPIRAN A WAWANCARA

Pewawancara: Timoti Agape Siahaan

Narasumber: Derwanto

Pertanyaan:

1. Apa saja menu yang disajikan Rumah Makan Lalapan ala Dewi?
2. Ada berapa cabang dan dimana cabang Rumah Makan Lalapan ala Dewi?
3. Apakah jumlah persediaan disetiap cabang berbeda dan mengapa?
4. Apakah jumlah kebutuhan dan persediaan setiap cabang bisa diprediksi untuk penyediaan bahan?
5. Bagaimana proses bisnis pada Rumah Makan Lalapan ala Dewi?
6. Apakah Rumah Makan Lalapan ala Dewi memiliki kendala dalam menjalankan proses bisnis?

Jawaban:

1. Menu yang disajikan oleh Rumah Makan Lalapan ala Dewi ada makanan dan minuman. Jenis makanan yang disediakan biasanya makanan yang digoreng seperti ayam, jamur, usus, tahu dan tempe serta beberapa aneka sambal. Untuk jenis minuman biasanya disediakan minuman dingin dan hangat seperti teh, jeruk dan minuman saset.
2. Rumah Makan Lalapan ala Dewi sampai saat ini sudah memiliki 5 cabang dengan lokasi yang tidak terlalu jauh. Untuk lokasi cabang 1, 2 dan 3 ada di jalan Kerto Raharjo, sedangkan lokasi cabang 4 di jalan Watumujur dan lokasi cabang 5 di jalan Kerto Leksono.
3. Setiap cabang memiliki pengurus cabang masing-masing sehingga jumlah persediaan disetiap cabang berbeda karena diurus dan dijalankan oleh pengurus cabangnya. Pengurus cabang akan melaporkan jumlah kebutuhan dan persediaan secara rutin seminggu sekali atau bisa lebih cepat jika kebutuhan mendesak.
4. Untuk saat ini pemilik Rumah Makan Lalapan ala Dewi masih mengandalkan catatan laporan setiap pengurus cabang untuk dijadikan evaluasi secara manual

sehingga masih susah untuk memprediksi dalam penyediaan bahan. Perbedaan lokasi sekitar cabang dan hari apa penjualan itu terjadi sangat mempengaruhi daya jual dan jumlah konsumen sehingga jumlah transaksi sering tidak menentu.

5. Ketika konsumen datang dan melakukan transaksi di lokasi Rumah Makan Lalapan ala Dewi maka pengurus cabang akan menyajikan menu yang dipesan. Setelah konsumen selesai bertransaksi maka hasil transaksi berupa laporan dari mesin kasir akan disimpan dan dilaporkan kepada pemilik Rumah Makan Lalapan ala Dewi. Pemilik nantinya akan mengetahui jumlah keuntungan setiap cabang dan secara rutin akan menyediakan kebutuhan setiap cabang berdasarkan laporan pengurus cabang.
6. Kendala yang sering terjadi biasanya pada pelaporan, pengurus cabang sering kali memberikan laporan tidak tepat waktu dikarenakan jumlah transaksi masih harus disesuaikan dengan catatan persediaan. Penyediaan bahan untuk setiap cabang bisa terhambat jika laporan dari pengurus cabang tidak tepat waktu dan mengakibatkan kekurangan persediaan atau telatnya penyediaan bahan.

LAMPIRAN B FOTO

Gambar 7.1. Foto nota pemesanan