SISTEM INFORMASI PEMESANAN RESTORAN BERBASIS ANDROID DAN *CLIENT SERVER* PADA JARINGAN LOKAL

SKRIPSI

Untuk memenuhi sebagian persyaratan mencapai gelar sarjana komputer

Disusun Oleh :

MAFTUHRIZA AFRUL YUMIDA

NIM. 105060801111021

PROGRAM STUDI INFORMATIKA / ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2014

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

LEMBAR PERSETUJUAN

SISTEM INFORMASI PEMESANAN RESTORAN
BERBASIS ANDROID DAN *CLIENT SERVER*
PADA JARINGAN LOKAL

SKRIPSI
Untuk Memenuhi Sebagian Persyaratan Mencapai Gelar Sarjana Komputer

Disusun Oleh :
MAFTUHRIZA AFRUL YUMIDA
NIM. 105060801111021

Telah diperiksa dan disetujui oleh:

Pembimbing I	Pembimbing II,
Aryo Pinandito, S.T., M.MT.	**Agi Putra Kharisma, S.T., MT.**
NIK. 83051916110374	

LEMBAR PENGESAHAN

SISTEM INFORMASI PEMESANAN RESTORAN
BERBASIS ANDROID DAN *CLIENT SERVER*
PADA JARINGAN LOKAL

SKRIPSI

Untuk Memenuhi Sebagian Persyaratan Mencapai Gelar Sarjana Komputer

Disusun Oleh :
Maftuhriza Afrul Yumida
NIM. 105060801111021

Skripsi ini telah diuji dan dinyatakan lulus pada tanggal 2 Juli 2014

Penguji I	Penguji II
Dr. Eng Herman Tolle, S.T., M.T.	**Eriq Muhammad Adams J, S.T.,M.Kom**
NIP. 19740823 200012 1 001	**NIK. 850410 06 1 1 0027**

Penguji III

Fajar Pradana, S.T., M.Eng
NIK. 87112116110371

Mengetahui
Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji, M.T.
NIP. 19670801 199203 1 001

PERNYATAAN
ORISINALITAS SKRIPSI

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 dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

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

Malang, Juni 2014

Mahasiswa,

Maftuhriza Afrul Yumida

NIM. 105060801111021

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Tuhan Yang Maha Esa karena hanya dengan rahmat dan karunia-Nya, penulis dapat menyelesaikan skripsi dengan judul "Sistem Informasi Pemesanan Restoranberbasis Android dan *Client-Server* Pada Jaringan Lokal".

Melalui kesempatan ini, penulis ingin menyampaikan rasa hormat dan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan bantuan dan dukungan selama penulisan skripsi, diantaranya:

1. Ayahanda Niam Afrosin, Ibunda Nurul Khoiriyah (Alm), Ibunda Sukarmi, Kakak Linda Widyawati, Adik Annisa Afrul Mufidah, Adik Anis Puju Lestari, Adik Ma'ruf Afrul Rifai dan seluruh keluaga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya dalam mendidik penulis serta memberikan doa dan semangat secara terus-menerus demi terselesaikannya skripsi ini.
2. Bapak Ir. Sutrisno, M.T., Bapak Ir. Heru Nurwasito, M.Kom., Bapak Himawat Aryadita, S.T., M.Sc., dan Bapak Eddy Santoso, S.Kom. selaku Ketua, Wakil Ketua 1, Wakil Ketua 2 dan Wakil Ketua 3 Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
3. Bapak Drs. Marji, M.T. dan Bapak Issa Arwani, S.Kom., M.Sc. selaku Ketua dan Sekretaris Program Studi Teknik Informatika Universitas Brawijaya.
4. Bapak Aryo Pinandito, S.T., M.MT. selaku dosen pembimbing I yang telah memberikan ilmu dan saran untuk menyelesaikan skripsi ini.
5. Bapak Agi Putra Kharisma, S.T., M.T. selaku dosen pembimbing II yang juga memberikan ilmu dan saran untuk menyelesaikan skripsi ini.
6. Bapak Budi Darma Setiawan, S.Kom., M.Cs. selaku dosen pembimbing akademik yang telah memberikan pengarahan selama penulis menempuh pendidikan di Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
7. Seluruh Dosen Teknik Informatika Universitas Brawijaya atas kesediaan membagi ilmunya kepada penulis.

8. Seluruh Civitas Akademika Teknik Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Teknik Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.
9. Mochtar Setya, Yulis Raga Priyono, Abdila Febrinianto, Happy Gagas T. A., Intan Rahmawati, Eko Aplilia, Candra Irwansyah, Mitta Testiasari yang membantu penulis dalam pengerjaan skripsi ini.
10. Teman-teman anggota LSO OPTIIK yang selalu memberi semangat kepada penulis untuk menyelesaikan skripsi ini.
11. Sahabat-sahabat penulis TIF 2010 yang telah memberikan doa, bantuan serta semangat kepada penulis selama menempuh studi di Teknik Informatika Universitas Brawijaya.
12. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun yang tidak langsung demi terselesaikannya skripsi ini.

Hanya doa yang bisa penulis berikan semoga Allah SWT memberikan pahala serta balasan kebaikan yang berlipat. Penulis menyadari bahwa skripsi ini masih banyak kekurangan. Oleh karena itu, kritik dan saran yang bersifat membangun sangat diharapkan untuk menyempurnakan skripsi ini. Semoga skripsi ini membawa manfaat bagi penyusun maupun pihak lain yang menggunakannya.

Malang, Juni 2014

Penulis

ABSTRAK

Maftuhriza Afrul Yumida.2014. : Sistem Informasi Pemesanan Berbasis Android dan *Client-Server* Pada Jaringan Lokal. Skripsi Program Studi Teknik Informatika, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang. Pembimbing : Aryo Pinandito, S.T., M.MT. dan Agi Putra Kharisma, S.T., MT.

Restoran merupakan bentuk usaha yang menyediakan berbagai jenis makanan dan minuman. Setiap restoran harus melayani jumlah pesanan yang banyak, apabila menerapkan proses pemesanan manual akan berdampak pada kesalahan dalam penulisan pesanan mengenai nama menu maupun jumlah pesanannya, selain itu *waiters* juga kesulitan dalam menjawab pertanyaan status pesanan dari pelanggan karena harus pergi ke dapur untuk menanyakannya. Oleh sebab itu dengan menerapkan teknologi yang berkembang saat ini kedalam sistem pemesanan restoran diharapkan mampu mengontrol kegiatan-kegiatan pada proses pemesanan untuk meminimalisir kesalahan. Observasi dan wawancara digunakan sebagai metode untuk mengumpulkan data yang akan dijadikan sebagai daftar kebutuhan. Dari daftar kebutuhan tersebut dianalisa untuk menghasilkan proses bisnis yang sesuai, kemudian dilakukan perancangan terhadap sistem yang nantinya akan diimplementasikan. Teknologi yang diterapkan adalah sistem informasi berbasis Android dan *web service* menggunakan metode *client-server* pada jaringan lokal, dengan ini pesanan akan dimasukkan melalui perangkat Android sebagai *client* lalu dikirim melalui jaringan ke *server* dan dibaca oleh dapur, koki dan kasir. Dengan proses yang sudah didigitalisasi dapat mengurangi kesalahan dalam memasukkan pesanan, mobilitas *waiters* tetap terjaga karena menggunakan perangkat *mobile* berbasis Android, antrian dapat terkontrol karena setiap pesanan memiliki data waktu pesanan, selain itu mampu memberikan informasi mengenai status pesanan dan ketersediaan menu.

Kata Kunci : Restoran, Sistem Informasi, Android, *client-server, web service*

repository.ub.ac.id

ABSTRACT

Maftuhriza Afrul Yumida.2014. : Restaurant Ordering Information System Base on Android and Client-Server In Local Network. Undergraduate Thesis of Informatic Engineering Study Program, Information Technology and Computer Science Program, Brawijaya University, Malang. Advisor : Aryo Pinandito, S.T., M.MT. and Agi Putra Kharisma, S.T., MT.

Restaurant is a business that provides a variety of foods and beverages. Every restaurant must serve the many orders, if applying manual ordering process will affect the order of the error in writing the name of the menu and the order amount, and then the waiters also difficulty in answering the question the status of orders from customers so must go to the kitchen to ask. Therefore, by applying the technology for restaurant ordering system is expected to control the ordering process to minimize error. Observation and interviews are used as a method to collect data that will be list of requirements. From the list of requirements were analyzed to generate the appropriate business processes, and then to design the system that will be implemented. The technology applied is an information system based on Android and a web service using client-server methods on the local network, this order would be put through Android device as a client and then sent over the network to the server and read by the kitchen, chef and cashiers. With a process that has been digitized, can reduce errors in entering the order, mobility the waiters is assured because using Android-based devices, the queue can be controlled for each order because has a time order, furthermore able to provide information of availability menu and orders status.

Keyword : Restaurant, Information Systems, Android, client-server, web services

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR KODE

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Bisnis kuliner dirasakan semakin menjamur, ditinjau dari munculnya berbagai restoran dan kafe dengan inovasi serta variasi baru untuk mendongkrak jumlah pengunjung. Berdasarkan data dari Kemenrian Pariwisata dan Ekonomi Kreatif (BUPDAR) jumlah usaha dibidang tersebut terus mengalami peningkatan khususnya dibidang restoran. Perkembangan usaha tersebut mencapai angka 200 per tahunnya dengan rata-rata tenaga kerja yang diangkat adalah 27 orang tiap usaha [BUPDAR]. Hal ini dapat diisyaratkan bahwa usaha restoran memiliki prospek yang baik kedepannya sekaligus memberi lapangan pekerjaan untuk masyarakat.

Pada umumnya dalam usaha restoran dalam praktek pemesanan dilakukan secara manual seperti yang ditemukan pada Cafe Monopoli Malang, yaitu dengan cara mencatat pesanan pada kertas oleh *waiters* yang natinya diserahkan ke dapur dan kasir, hal ini terkesan tidak fleksibel dan memungkinkan untuk terjadinya kesalahan penulisan yang berakibat pesanan tidak sesuai. Selain itu, dengan proses pemesanan manual seperti ini pelanggan sulit mengetahui status pesanan dan juga sering terjadi kesalahan dalam urutan pemesanan, hal tersebut berdampak negatif pada kepuasan pelanggan, terlebih setiap restoran harus melayani banyak pesanan sekaligus. Oleh karena itu dibutuhkan sistem yang mampu mempermudah proses pemesanan.

Pada era teknologi seperti saat ini, telah banyak banyak perangkat komputer beserta aplikasinya untuk membantu dan membermudah berbagai aktifitas, terlebih komputer yang dikemas dalam bentuk *mobile* sehingga mudah dibawa. Contohnya seperti *Tablet PC* dan *Smarphone,* saat ini telah banyak digunakan dalam berbagai sektor masyarakat baik dunia hiburan, pendidikan serta bisnis. Salah sati *platform* yang paling banyak berada dipasaran yaitu Android.

Dari survey yang dilakukan oleh CIRP (*Costumer Intelegence Research Partner*) pada awal tahun 2013, *platform* Android yang merupakan asuhan dari

Google mendapat prosentase 51,2% dalam penjualan, berada diatas iOS milik Apple dengan nilai 43,5% dan jauh diatas platform lainnya [TIME]. Android dan iOS banyak diminati karena memiliki berbagai kelebihan yaitu sudah dilengkapi layar sentuh untuk mempermudah penggunaanya serta fasilitas *web service* sehingga mampu bertukar informasi melalui jaringan dengan mudah dengan menerapkan konsep *client-server*. Metode *client-server* merupakan pilihan yang tepat untuk mengatasi permasalahan dimana banyak pengguna membutuhkan data yang sama, oleh karena itu pengguna diasumsikan sebagai *client* dan penyedia data sebagai *server* lalu terhubung melalui jaringan. Namun dengan segala fitur tersebut perangkat dengan platform Android dirasa lebih terjangkau dibandingkan perangkat dengan platform iOS dalam segi harga.

Saat ini perangkat berbasis Android dan *client-server* sudah banyak diterapkan dalam berbagai sistem informasi seperti sistem informasi geografis [SSM-14]. Oleh karena itu perlu dikembangkan sebuah sistem pemesanan yang sudah digitalisasi menggunakan perangkat Android dan terhubung dengan *web service* untuk menerapkan konsep *client-server* serta untuk mengatur proses pemesanan. Sehingga mampu membantu mengurangi kesalahan data pesanan dan urutan antrian serta mampu memberikan informasi status pesanan. Dengan demikian kepuasan pelanggan dapat dinaikan dan memberikan pengaruh positif pada restoran.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, dapat dirumuskan beberapa masalah sebagai berikut. Bagaimana menganalisis, merancang, mengimplementasi dan menguji sistem informasi pemesanan yang mampu :

1. Berjalan pada perangkat mobile berbasis Android untuk *waiters* dan berbasis *web* untuk admin, dapur, koki dan kasir.
2. Menampilkan menu makanan atau minuman pada perangkat Android serta status ketersediaanya.
3. Menambah pesanan dan mengirimkannya dari aplikasi berbasis Android ke aplikasi berbasis *web* menggunakan fasilitas *web service*.

4. Menampilkan daftar pesanan beserta status pesanannya dan berubah apabila terdapat pembaruan.
5. Mengontrol antrian pesanan dan merubah status pesanan, diakses oleh dapur menggunakan *web browser*.

1.3 Batasan Masalah

Oleh karena luasnya bidang yang dihadapi maka ruang lingkup masalah akan dibatasi sebagai berikut:

1. Hanya akan membahas mengenai sistem yang mencakup lima sisi user yaitu admin, *waiters*, dapur,koki dan kasir.
2. Pelanggan baru ditambahkan berdasarkan meja yang tersedia, dan diatur secara statis sebanyak enam buah meja.
3. Daftar menu yang ditampilkan pada *waiters* merupakan data masukan dari dapur dan dikelompokkan berdasarkan tipe seperti makanan, munuman dan lainnya. Tiap menu merupakan data tunggal berisi nama, harga dan status ketersediaannya. Menu tidak dapat dikelompokkan menjadi paket.
4. Pendataan pesanan dilakukan oleh *waiters*, dilanjutkan oleh dapur yang akan mengontrol status pesanan berdasarkan tahapan pesanan.
5. Pembaruan status ketersediaan menu makanan dilakukan sebelum proses pemesanan dimulai.
6. Proses pesanan berakhir ketika kasir menyatakan bahwa pelanggan telah membayar dan memperbarui status pelanggan.

1.4 Tujuan

Tujuan penelitian ini adalah :

1. Membuat aplikasi yang mampu mempermudah dalam proses pemesanan dengan data pesanan yang lebih akurat.
2. Membuat aplikasi yang mampu mempermudah *waiters* dalam memberikan informasi tentang daftar pesanan beserta statusnya kepada pelanggan.
3. Membuat aplikasi yang mampu mempermudah *waiters* dalam memberikan informasi tentang status ketersediaan menu yang dapat dipesan kepada pelanggan.
4. Membuat aplikasi yang mampu mempermudah dapur dalam mengamati dan mongontrol semua pesanan.

1.5 Manfaat

Manfaat dari penelitian ini adalah untuk memenuhi aspek utama dalam mencapai kepuasan pelanggan seperti.

1. Kesalahan pada proses pemesanan berkurang dengan data pesanan yang lebih akurat.
2. Terdapat waktu pemesanan sehingga proses pelayanan lebih terkontrol dan pelanggan menerima layanan sesuai dengan urutannya.
3. *Waiters* dengan mudah mengetahui status pesanan apabila ada pelanggan yang menanyakan tentang pesanannya.

1.6 Sistematika Penulisan

Untuk sistematika pnulisan dalam skripsi ini sebagai berikut:

BAB I Pendahuluan

Membahas tentang latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat dan sistematika penulisan.

BAB II Tinjauan Pustaka

Membahas tentang dasar teori dan referensi yang mendasari dalam pembuatan sistem.

BAB III Metodologi Penelitian

Membahas tentang metode yang digunakan dalam penulisan.

BAB IV Analisis dan Perancangan

Membahas tentang analisis kebutuhan dan perancangan sistem.

BAB V Implementasi dan Pengujian

Membahas tentang implementasi dari sistem dan proses pengujian terhadap sistem.

BAB VI Penutup

Membahas tentang kesimpulan serta saran untuk pengembangan sistem lebih lanjut.

BAB II
TINJAUAN PUSTAKA

2.1 Restoran

Menurut Suarthana (2006 : 23) restoran adalah: ”tempat usaha yang komersial yang ruang lingkup kegiatannya menyediakan pelayanan makanan dan minuman untuk umum di tempat usahanya”.

Sedangkan menurut Sihite (2000 : 16) restoran adalah: “suatu tempat dimana seseorang yang datang menjadi tamu yang akan mendapatkan pelayanan untuk menikmati makanan, baik pagi, siang, ataupun malam sesuai dengan jam bukanya dan oleh tamu yang menikmati hidangan itu harus membayar sesuai dengan harga yang ditentukan sesuai daftar yang disediakan di restoran itu” [RNH-10].

Dari dua pendapat tersebut dapat disimpulkan bahwa restoran merupakan tempat usaha yang ruang lingkup kegiatannya adalah menyediakan makanan dan minuman kepada tamu yang datang dan dikomersilkan

Penggunaan media alat tulis dan kertas untuk pemesanan makanan dan minuman di rumah makan masih digunakan sampai saat ini namun banyak menemui kendala- kendala antara lain adanya pemesanan yang rangkap (redudansi), tidak urutnya pembuatan pemesanan akibat bertumpuknya nota pemesanan terutama pada saat ramai pengunjung, juga kesalahan pencatatan akibat sulitnya membaca tulisan tangan [ADS-14:1].

2.2 Sistem Informasi

Menurut Edhy Sutanta (2003:10) “Sistem Informasi didefinisikan sebagai kumpulan subsistem yang saling behubungan, berkumpul bersama-sama dan membentuk satu kesatuan, saling berinteraksi dan bekerjasama antara bagian satu dengan yang lainnya dengan cara-cara tertentu untuk melakukan fungsi pengolahan data, menerima masukan (*input*) berupa data-data, kemudian mengolahnya, dan menghasilkan keluaran berupa informasi sebagai dasar bagi pengambilan keputusan yang berguna dan mempunyai nilai nyata yang dapat dirasakan akibatnya baik pada saat itu juga maupun di masa mendatang, mendukung kegiatan

operasional, manajerial, dan strategis organisasi, dengan memanfaatkan berbagai sumber daya yang ada dan tersedia bagi fungsi tersebut guna mencapai tujuan " [OCN-09:10].

2.3 *Client-Server*

Konsep dari *client server* adalah sebagai Sebuah aplikasi yang dapat dianggap sebagai requestor (*client*) atau dapat juga dianggap sebagai *provider* (*server*) terhubung melalui *web service.*

Biasanya jumlah *client* jauh lebih banyak daripada jumlah *server* dan mampu memberikan layanan kepada banyak *client* dengan kemampuan yang sama sebagaimana ketika hanya melayani sebuah *client* dari sisi suatu arsitektur *client server*, bahwa *client* adalah sebuah aplikasi yang berjalan pada komputer pribadi dan bergantung pada *server* untuk mengerjakan oprerasi. Sedangkan *server* adalah *node* yang memungkinkan *node* lain pada jaringan untuk mengakses sumbernya. *Server* ini bersifat terdedikasi yang artinya *node* tersebut dapat dipakai dengan cara lain [GGS-09:27].

Contoh aplikasi yang telah menerapkannya adalah "*Matlab/Simulink Based Remote Robotics Experiments*" yang dikerjakan oleh Ali Turan, Seta Bogosyan dan Metin Gokasan dimana robot bertindak sebagai *client* tanpa perintah apapun, dan akan bergerak sesuai perintah yang dimasukkan pada *server* kemudian dikirimkan ke *client* melalui internet (*web service*) [ASM-06:1]

2.4 *Web Service*

Konsep *Web service* adalah sebuah *software* yang dirancang untuk mendukung interoperabilitas interaksi mesin-ke-mesin melalui sebuah jaringan. *Web service* secara teknis memiliki mekanisme interaksi antar sistem sebagai penunjang interoperabilitas, baik berupa agregasi (pengumpulan) maupun sindikasi (penyatuan). *Web service* memiliki layanan terbuka untuk kepentingan integrasi data dan kolaborasi informasi yang bisa diakses melalui internet oleh berbagai pihak menggunakan teknologi yang dimiliki oleh masing- masing pengguna. Dalam perkembangannya, *model web service* memiliki dua metode yang berorientasi pada layanan dan sumberdaya informasi, yaitu: SOAP (*Simple Object Access Protocol*) dan REST (*REpresentational State Transfer*) [EDH-12].

2.4.1 REST

Metode REST lebih sederhana dibandingkan dengan SOAP karena menggunakan operasi untuk memanipulasi data menggunakan PUT, GET, POST, dan DELETE serta format standar (HTTP, HTML, XML, URI, MIME), namun jika diperlukan proses pertukaran data, maka konten berupa teks dari hasil eksekusi *web service* dapat diolah dalam format teks (seperti XML atau HTML) dengan menggunakan utilitas komunikasi data berupa koneksi *socket* protokol HTTP [EDH-12:2].

2.4.2 *Local Area Network* (LAN)

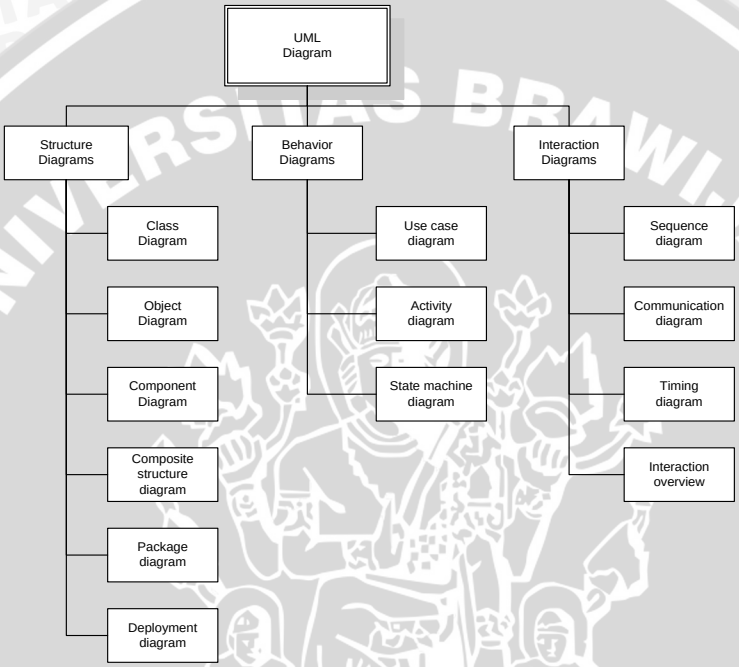
Local Area Network (LAN), merupakan jaringan milik pribadi di dalam sebuah gedung atau kampus yang berukuran sampai beberapa kilometer. LAN seringkali digunakan untuk menghubungkan komputer-komputer pribadi dan workstation dalam kantor suatu perusahaan atau pabrik-pabrik untuk memakai bersama sumberdaya (*resource*, misalnya *printer*) dan saling bertukar informasi [HBA-13].

2.5 Android

Android merupakan generasi baru *platform Mobile*, *platform* yang memberikan pengembang untuk melakukan pengembangan sesuai yang diharapkannya. Sistem operasi yang mendasari Android dilisensikan di bawah GNU, *general Public Lisensi* Versi 2 (GPLv2), yang lebih dikenal dengan istilah *copyleft*, lisensi di mana setiap perbaikan pihak ketiga harus terus dibawah syarat (*terms*). Android di distribusikan di bawah Lisensi *Apache Software* (ASL/Apache 2), yang memungkinkan untuk distribusi kedua dan seterusnya. Komersialisasi pengembang (produsen *handset* khususnya) dapat memilih untuk meningkatkan *platform* tanpa harus memberikan perbaikan mereka ke masyarakat *open source*. Sebaliknya, pengembang dapat keuntungan dari perangkat tambahan seperti perbaikan dan mendistribusikan ulang pekerjaan mereka di bawah lisensi apapun yang mereka inginkan. Pengembang aplikasi Android diperbolehkan untuk mendistribusikan aplikasi mereka di bawah skema lisensi apapun yang mereka inginkan [IBY-13:3].

2.6 Unified Modelling Language

Unified Modelling Language (UML) merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram teks-teks pendukung. UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikkan, menggambarkan, membangun, dan mendokumentasikan dari sistem perangkat lunak [ROS-11:117].



Gambar 2.1 Diagram UML
Sumber : [ROS-11:121]

Structure diagrams yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem [ROS-11 :121].

2.4.1 *Use case* Diagram

Diagram *Use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu [ROS-11:130].

Syarat penamaan pada diagram *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Simbol aktor adalah gambar orang, namun aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

No.	Gambar	Nama	Keterangan
1.		Aktor / Actor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
2.	<<include>> ------>	Menggunakan / include / uses	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan

			fungsinya atau sebagai syarat dijalankan use case ini.
3.	<<extend>>	Ekstensi/ Extend	Relasi use case tambahan ke sebuah use case,dimana use case yg ditambahkan dpt berdiri sendiri walaupun tanpa use case tambahan itu.
4.		Asosiasi/ Association	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interksi dengan aktor.
5.		*Use case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case.*

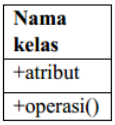
Tabel 2.1 Keterangan simbol - simbol *Use case* Diagram

Sumber : [ROS-11 :130]

2.4.2 Class Diagram

Class diagram atau diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas [ROS-11:122].

Jenis-jenis kelas dapat digabungkan satu sama lain sesuai dengan pertimbangan yang dianggap baik asalkan fungsi-fungsi pada struktur kelas tetap ada. Susunan kelas juga dapat ditambahkan kelas utilitas seperti koneksi ke basis data, membaca *file teks*, dan lain sebagainya sesuai kebutuhan [ROS-11:122].

No.	Gambar	Nama	Keterangan
1.		Kelas / *Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
7.		*Composite*	Bagian dari (*part-of*).

Tabel 2.2 Keterangan simbol - simbol *Class Diagram*

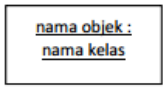
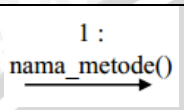
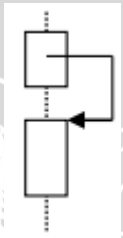
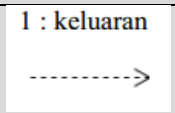
Sumber : [ROS-11:123]

2.4.3 Sequence Diagram

Sequence diagram atau diagram interaksi menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Penggambaran diagram *sequence* harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu [ROS-11:137].

Diagram *sequence* yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri.

No.	Gambar	Nama	Keterangan
1.		Aktor / *Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.

2.		Garis hidup/ *Lifeline*	Menyatakan kehidupan suatu objek.
3.		Objek / *Object*	Menyatakan objek yang berinteraksi.
4.		Pesan tipe *call*	Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri.  1 : nama_metode() Arah panah mengarah pada objek yang memiliki operasi atau metode, karena ini memanggil operasi atau metode maka operasi atau metode yang dipanggil harus ada pada diagram kelas sesuai dengan objek yang berinteraksi.
5.		Pesan tipe *return*	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.

Tabel 2.3 Keterangan simbol - simbol Sequence Diagram

Sumber : [ROS-11:138]

Penggambaran letak pesan harus berurutan, pesan yang lebih atas dari lainnya adalah pesan yang berjalan terlebih dahulu [ROS-11 :137].

2.7 MVC

MVC adalah singkatan dari *Model, View dan Controller* yang merupakan sebuah arsitektur untuk membuat sebuah program. Arsitektur ini menekankan kepada pembagian dari komponen-komponen program menjadi tiga bagian utama yaitu *Model, View, dan Controller*. Tujuan dari pembagian program ke dalam tiga bagian besar ini adalah untuk memisahkan fokus perhatian, tanggung jawab, dan logika ke dalam bagian masing-masing [CSM-12:2]:

1. Model

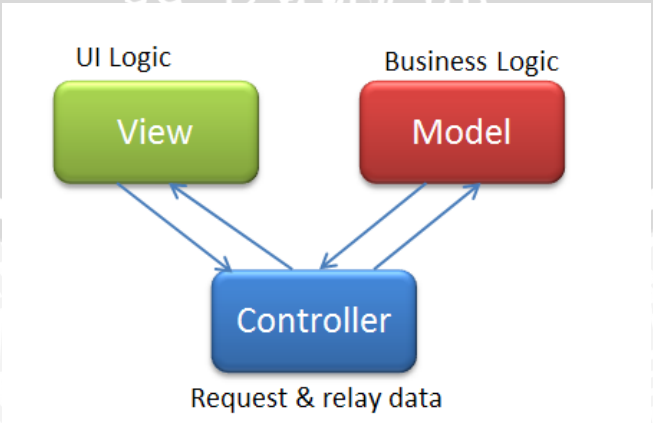
Model yang merepresentasikan data yang digunakan oleh aplikasi sebagaimana proses bisnis yang diasosiasikan terhadapnya.

2. View

Layer ini berisi keseluruhan dari implementasi user interface. Disini, komponen grafis menyediakan representasi proses internal aplikasi dan menuntun alur interaksi user terhadap aplikasi.

3. Controller

Layer ini berisi keseluruhan alur program dan transisi layer, dan juga bertanggungjawab akan penampungan *events* yang dibuat oleh user dari *View* dengan melakukan update terhadap komponen *Model* menggunakan data yang dimasukkan oleh user.



Gambar 2.2 Pola Perancangan MVC

2.8 HTML (*Hyper Text Markup Language*)

HTML yang merupakan singkatan dari Hyper Text Markup Language adalah serangkaian kode program yang merupakan dasar dari representasi visual sebuah halaman Web. Didalamnya berisi kumpulan informasi yang disimpan dalam tag-tag tertentu, dimana tag-tag tersebut digunakan untuk melakukan format terhadap informasi yang dimaksud.

Berbagai pengembangan telah dilakukan terhadap kode HTML dan telah melahirkan teknologi-teknologi baru di dalam dunia pemrograman web. Kendati demikian, sampai sekarang HTML tetap berdiri kokoh sebagai dasar dari bahasa web seperti PHP, ASP, JSP dan lainnya. Bahkan secara umum, mayoritas situs web yang ada di Internet pun masih tetap menggunakan HTML sebagai teknologi utama mereka [FCB-11].

2.8.1 CSS (*Cascading Style Sheets*)

Cascading Style Sheet merupakan kepanjangan dari CSS. Penggunaan CSS membuat pemrograman *web* menjadi lebih mudah karena kita dapat melakukan penyeragaman format terhadap elemen-elemen yang sama dalam situs dengan cepat. Saat ini hampir semua situs berbasis HTML menggunakan CSS untuk meningkatkan keluwesan tampilan. CSS dapat disimpan dalam file terpisah dengan ekstensi .css, dan setiap perubahan yang dilakukan pada *file* tersebut akan mempengaruhi seluruh dokumen HTML yang terkait padanya. Dengan demikian, waktu untuk melakukan perubahan terhadap situs dengan jumlah halaman yang banyak dapat dikurangi berkat bantuan CSS [FCB-11].

2.9 PHP

PHP adalah sebuah bahasa pemograman yang berjalan dalam sebuah web-server (*server side*). PHP diciptakan oleh programmer unix dan Perl yang bernama Rasmus Lerdoft pada bulan Agustus-September 1994. Pada awalnya, Rasmus mencoba menciptakan sebuah *script* dalam *wesite* pribadinya dengan tujuan untuk melihat siapa saja yang pernah mengunjungi *website*-nya [PHP-08].

Script PHP adalah bahasa program yang berjalan pada sebuah *webserver*, atau sering disebut *server-side*. Oleh karena itu, PHP dapat melakukan apa saja

yang bisa dilakukan program CGI lain, yaitu mengolah data dengan tipe apapun, menciptakan halaman web yang dinamis, serta menerima dan menciptakan *cookies*, dan bahkan PHP bisa melakukan lebih dari itu. Arti *script server-side* adalah, agar dapat menjalankan script ini dibutuhkan tiga program utama, yaitu *web-server* (dapat berupa IIS dari *windows* atau *apache*), modul PHP dan juga *web browser*.

Sistem kerja dari PHP diawali dengan permintaan yang beasal dari halaman website oleh browser. Berdasarkan URL atau alamat website dalam jaringan internet, browser akan menemukan sebuah alamat dari webserver, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *webserver*.

Selanjutnya *web server* akan mencarikan berkas yang diminta dan menampilkan isinya di browser. Browser yang mendapatkan isinya segera menerjemahkan kode HTML dan menampilkannya.

Pada prinsipnya apabila yang halaman dipanggil mengandung *script* PHP adalah sama dengan memanggil kode HTML, namun pada saat permintaan dikirim ke *web-server, web-server* akan memeriksa tipe file yang diminta user. Jika tipe *file* yang diminta adalah PHP, maka akan memeriksa isi *script* dari halaman PHP tersebut.

Apabila dalam *file* tersebut tidak mengandung *script* PHP, permintaan *user* akan langsung ditampilkan ke *browser*, namun jika dalam file tersebut mengandung *script* PHP, maka proses akan dilanjutkan ke modul PHP sebagai mesin yang menerjemahkan *script* PHP dan mengolah *script* tersebut, sehingga dapat dikonversikan ke kode-kode HTML lalu ditampilkan ke *browser user* [PHP-08].

2.9.1 CodeIgniter

CodeIgniter adalah aplikasi *open source* yang berupa *framework* dengan pola MVC (*Model*, *View*, *Controller*) untuk membangun *website* dinamis dengan menggunakan PHP. Code Igniter memudahkan developer untuk membuat aplikasi *web* dengan cepat dan mudah dibandingkan dengan membuatnya dari awal [CSM-12].

2.10 JAVA

Java adalah bahasa pemrograman tingkat tinggi yang berorientasi objek dan program java tersusun dari bagian yang disebut kelas. Kelas terdiri atas metode-metode yang melakukan pekerjaan dan mengembalikan informasi setelah melakukan tugasnya. Para pemrogram Java banyak mengambil keuntungan dari kumpulan kelas di pustaka kelas Java, yang disebut dengan *Java Application Programming Interface* (API). Kelas-kelas ini diorganisasikan menjadi sekelompok yang disebut paket (*package*). Java API telah menyediakan fungsionalitas yang memadai untuk menciptakan applet dan aplikasi canggih. Jadi ada dua hal yang harus dipelajari dalam Java, yaitu mempelajari bahasa Java dan bagaimana mempergunakan kelas pada Java API. Kelas merupakan satu-satunya cara menyatakan bagian eksekusi program, tidak ada cara lain. Pada Java khususnya untuk android, dalvik merupakan program untuk mengkompilasi *file* kode sumber Java menjadi kelas-kelas *bytecode*. *File* kode sumber mempunyai ekstensi *.java. Kompilator dalvik menghasilkan file *bytecode* kelas dengan ekstensi *.dex [BOD-08] .

Java merupakan bahasa berorientasi objek (OOP) yaitu cara ampuh dalam pengorganisasian dan pengembangan perangkat lunak. Pada OOP, program komputer sebagai kelompok objek yang saling berinteraksi. Deskripsi ringkas OOP adalah mengorganisasikan program sebagai kumpulan komponen, disebut objek. Objek-objek ini ada secara independen, mempunyai aturan-aturan berkomunikasi dengan objek lain dan untuk memerintahkan objek lain guna meminta informasi tertentu atau meminta objek lain mengerjakan sesuatu. Kelas bertindak sebagai modul sekaligus tipe. Sebagai tipe maka pada saat jalan, program menciptakan objek-objek yang merupakan instan-instan kelas. Kelas dapat mewarisi kelas lain [NSH-12].

2.11 Pengujian Perangkat Lunak

Pengujian (*Testing)* Arsitektur dari perangkat lunak berorientasi objek menghasilkan sekumpulan *layered subsistems* yang mengenkapsulasi kelas-kelas yang berkolaborasi. Setiap elemen sistem (*subsistem* dan *class*) melakukan fungsi yang membantu untuk mencapai kebutuhan sistem. Setiap elemen sistem

sangat penting untuk menguji sebuah *object oriented* sistem pada berbagai macam level yang berbeda dalam sebuah usaha untuk menemukan kesalahan-kesalahan yang mungkin terjadi dari kolaborasi kelas-kelas dan komunikasi subsistem melewati architetural layer [CNP-13].

2.11.1 Teknik Pengujian

Pengujian perangkat lunak memerlukan perancangan kasus uji *(test case)* agar dapat menemukan kesalahan dalam waktu singkat dan usaha minimum. Teknik atau metode perancangan kasus uji yang digunakan adalah *white-box dan black-box testing* [CNP-13].

2.11.2 *Black-Box Testing*

Black-box testing atau *behavioral testing* berfokus pada persyaratan fungsional perangkat lunak [CNP-13]. Pengujian *black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk semua program.

2.11.3 Strategi Pengujian

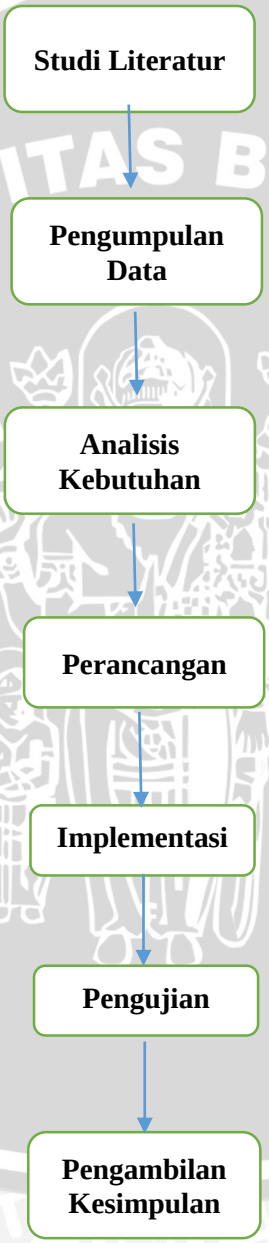
Strategi untuk pengujian perangkat lunak mengintegrasikan metode desain *test case* perangkat lunak ke dalam sederetan langkah yang direncanakan dengan baik, dan hasilnya adalah konstruksi perangkat lunak yang berhasil [CNP-13].

2.11.4 Pengujian Validasi

Pada kulminasi pengujian terintegrasi, perangkat lunak secara lengkap dirakit sebagai suatu paket; kesalahan *interfacing* telah diungkap dan dikoreksi, dan seri akhir dari pengujian perangkat lunak, yaitu pengujian validasi dapat dimulai. Validasi dapat ditentukan dengan berbagai cara, tetapi definisi yang sederhana adalah bahwa validasi berhasil bila perangkat lunak berfungsi dengan cara yang dapat diharapkan secara bertanggung jawab oleh pelanggan. Validasi perangkat lunak dicapai melalui sederetan pengujian *black-box* yang memperlihatkan konformitas dengan persyaratan [CNP-13].

BAB III
METODOLOGI PENELITIAN

Pada bab ini dijelaskan langkah-langkah yang akan dilakukan dalam perancangan, implementasi dan pengujian dari sistem yang kan dibuat.



Gambar 3.1 Metodologi Penelitian

3.1 Studi Literatur

Merupakan penelusuran literatur yang bertujuan dalam menyusun dasar teori yang digunakan untuk menunjang pembuatan sistem. Penelusuran dapat bersumber dari buku, media, pakar maupun hasil penelitian orang lain. Teori-teori pendukung meliputi:

1. *Unified Modelling Language*
   - 1.1 *Use Case* Diagram
   - 1.2 *Activity* Diagram
   - 1.3 *Class* Diagram
   - 1.4 *Sequence* Diagram
2. Pengujian Perangkat Lunak
   - 2.1 Teknik Pengujian
   - 2.2 Strategi Pengujian
3. Bahasa Pemrograman
   - 3.1 PHP
   - 3.2 Java
4. *Database MySQL*
5. *Web Service*
6. *Android*
7. *JavaScript Object Notation (JSON)*

3.2 Pengumpulan Data

3.2.1 Observasi

Observasi adalah pengamatan langsung para pengguna sistem dan atau pengamatan langsung terhadap sistem yang sedang berjalan. Pada tahap ini penulis melakukan observasi pada restoran yang masih menggunakan cara manual dalam proses pemesanan terhadap konsumennya, penulis mengamati dan mentatat berbagai kegiatan di lingkungan restoran dan sistem manual yang sedang berjalan. Kegiatan tersebut meliputi proses penerimaan pelanggan baru yang ditempatkan di meja tertentu, pencatatan pesanan, pengiriman data pesanan ke dapur, tindakan *waiters* ketika ada pertanyaan

dari pelanggan dan proses pelanggan saat membayar. Adapun restoran yang menjadi bahan pemngamatan adalah Cafe Monopoli Malang.

3.2.2 Wawancara

Pada tahap ini penulis melakukan wawancara dengan salah satu pelayan yang yang bekerja di restoran tertentu untuk memperoleh keterangan mengenai objek penelitian dan berbagai kebutuhan user yang akan menggunakan sistem pemesanan. Penulis meminta keterangan mengenai format data menu restoran dan berbagai kegiatan dalam sirkulasi pelayanan restoran.

3.3 Analisis Kebutuhan

Proses Analisis kebutuhan ini diawali dengan penjabaran gambaran umum aplikasi, penjabaran tentang daftar kebutuhan yang kemudian dimodelkan kedalam diagram use case berdasarkan data yang telah dikumpulkan pada tahap sebelumnya. Analisis kebutuhan ini bertujuan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh sistem.

3.3.1 Kebutuhan Sistem

Terdapat dua jenis kebutuhan, yaitu kebutuhan fungsional dan nonfungsional. Berikut penjelasan dari kebutuhan tersebut :

1. Kebutuhan Fungsional

   Merupakan kebutuhan yang harus disediakan oleh sistem, meliputi fitur-fitur utama untuk melakukan proses pemesanan.

2. Kebutuhan Nonfungsional

   Merupakan kebutuhan tambahan yang harus disediakan oleh sistem, meliputi fitur pendukung untuk melakukan proses pemesanan.

3.3.2 Perancangan *Use case*

Sistem dapat dipandang dari tiga aspek yaitu pengguna sistem, kegiatan yang dilakukan dan hubungan antar kegiatan tersebut. Penggunanya antara lain *waiters* yang akan mengakses aplikasi berbasis Android untuk

melakukan proses pemesanan, kemudian admin, dapur, koki, dan kasir yang akan mengakses aplikasi berbasis web untuk mengatur pesanan yang telah terkirim, pelanggan dan pegawai.

3.3.3 Perancangan Aktifitas

Menggambarkan alur kerja secara berurutan menggunakan *activity diagram*. Diagram ini bermanfaat menganalisis *use case* melalui penggambaran aktifitas-aktifitas yang dilakukan, penggambaran algoritma berurutan, dan pemodelan aplikasi dengan proses paralel. Adapun kegiatan yang dapat dilakukan oleh sistem dan pengguna adalah memesan menu dan mengubah status pesanan.

3.4 Perancangan

Pada tahap ini komponen-komponen yang telah ditetapkan akan diterapkan kedalam sistem melalui diagram kelas.

3.4.1 Perancangan Arsitektural

Perancangan arsitektural pada sistem ini menggunakan metode MVC (*Model View Controler*). *View* merupakan antarmuka yang digunakan untuk berkomunikasi dengan pengguna, *Model* adalah yang bertugas memanipulasi data yang dibutuhkan, sedangkan *Controler* bertugas menjembatani antara *View* dan *Model*. Dalam sistem tiap pengguna diterapkan dalam bentuk Controler untuk mengatur fitur-fitur yang akan dimuat. Terdapat dua aplikasi, yaitu aplikasi pemesanan berbasis Android yang nantinya digunakan oleh *waiters* dan aplikasi pemesanan berbasis *web* yang nantinya akan digunakan oleh admin, dapur, koki dan kasir menggunakan *web browser*.

Client-server merupakan konsep yang diterapkan untuk memecahkan masalah mobilitas, aplikasi berbasis Android bertindak sebagai *client* yang nantinya akan terhubung ke aplikasi berbasis *web* yang bertindak sebagai *server* dan penyedia data, menggunakan jaringan *wireless* dan metode REST. *Web service* diimplementasikan menggunakan *framework* codeIgniter.

3.4.2 Perancangan Interaksi

Perancangan interaksi atau perancangan *sequence* antar elemen (objek) dalam sistem yang telah teridentifikasi dimodelkan dalam *sequence* diagram. Diagram ini berupa skenario atau rangkaian langkah-langkah yang dilakukan dari sebuah kegiatan untuk menghasilkan *output* tertentu.

3.4.3 Perancangan Kelas

Merupakan perancangan yang memodelkan sistem dalam bentuk class diagram. Diagram ini digunakan untuk mengidentifikasi kelas-kelas serta paket-paket yang terdapat dalam sistem, kemudian dilanjutkan dengan mengidentifikasi hubungan antar kelas.

3.4.4 Perancangan Basis Data

Perancangan basis data dimodelkan dalam *Entity Relationship Diagram*. Diagram ini berupa hubungan antar entitas yang ada pada sistem, serta atribut yang disimpan oleh tiap entitas.

3.4.5 Perancangan Antarmuka

- Mendesain tampilan yang yang sesuai pada perangkat *mobile* berbasis Android.
- Mendesain tampilan yang sesuai pada perangkat berbasis *web service* yang nantinya diakses menggunakan *web browser*.

3.5 Implementasi

Implementasi perangkat lunak mengacu pada perancangan. Dilakukan menggunakan bahasa pemrograman PHP untuk proses pengambilan data serta bahasa pemrograman Java yang diterapkan pada perangkat berbasis Android.

3.6 Pengujian

Dilakukan untuk mengetahui apakah kinerja dan performa sistem telah memenuhi spesifiksi kebutuhan yang melandasinya. Strategi pengujian yang akan digunakan yaitu pengujian unit *(Unit Testing)* dan pengujian validasi *(validation testing)*. Analisis juga dilakukan terhadap hasil yang didapatkan pada tahap pengujian untuk mendapatkan kesimpulan.

3.7 Pengambilan Kesimpulan

Pengambilan kesimpulan dilakukan setelah semua tahap dilalui. Kesimpulan diambil untuk menjawab rumusan masalah yang ditetapkan sebelumnya. Tahap terakhir dari penulisan adalah saran yang dimaksudkan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta memberikan pertimbangan atas pengembangan sistem.

BAB IV
ANALISIS DAN PERANCANGAN

Bab ini membahas mengenai analisis sistem dan desain dari sistem pemesanan restoran berbasis Android dan *client-server* pada jaringan lokal. Analisis sistem dan desain sistem memiliki dua tahap. Tahap pertama adalah proses analisis kebutuhan yang merupakan analisis kebutuhan yang dilihat dari sudut pandang pengguna. Dilanjutkan dengan tahap kedua yaitu proses perancangan perangkat lunak dilakukan setelah semua kebutuhan perangkat lunak didapatkan melalui tahap analisis kebutuhan. Tahap analisis kebutuhan terdiri atas lima langkah yaitu melakukan penjabaran gambaran umum aplikasi, penjabaran tentang daftar kebutuhan, kemudian memodelkannya ke dalam diagram *use case* dan *activity diagram*. Proses perancangan perangkat lunak memiliki lima tahap, yaitu perancangan arsitektural, perancangan *sequence diagram*, perancangan *class diagram* untuk menggambarkan perancangan struktur *class-class* yang menyusun aplikasi, perancangan basis data untuk menggambarkan interaksi antar entitas di dalam sistem dan perancangan antarmuka.

4.1 Analisis Kebutuhan Sistem

Proses analisis kebutuhan ini diawali dengan penjabaran gambaran umum aplikasi, penjabaran tentang daftar kebutuhan dan kemudian dimodelkan kedalam diagram *use case*. Analisis kebutuhan ini bertujuan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh sistem agar dapat memenuhi kebutuhan pengguna.

4.1.1 Gambaran umum sistem

Sistem pemesanan restoran ini merupakan sistem yang dirancang untuk mempermudah proses pemesanan pada restoran dari pengiriman pesanan ke dapur, sehingga meminimalisir kesalahan dalam pemesanan akibat kesalahan penulisan pesanan. Selain itu, sistem dilengkapi waktu pemesanan sehingga pihak dapur dalam melayani pelanggan dapat terjadwal dan memperbarui status pesanan yang natinya akan dikirimkan ke *waiters* sehingga pelanggan dapat mengetahui dengan cepat saat memverifikasi status pesanannya.

4.1.2 Identifikasi Sistem

Pada sistem terdapat empat jenis pengguna yaitu admin, *waiters*, dapur dan kasir. Sistem untuk *waiters* berfungsi untuk mencatat pesanan oleh pelanggan menggunakan perangkat Android untuk dikirim ke *server* yang natinya akan dibaca oleh dapur dan menampilkan status pesanan. Sistem untuk dapur berfungsi untuk memasukkan, mengubah dan menghapus data menu serta mengubah status pesanan dan ketersediaan menu. Sistem untuk kasir berfungsi untuk menampilkan tagihan tiap pelanggan dan mengubah status dari pelanggan dan meja. Sedangkan untuk sistem berfungsi untuk mengatur data pegawai.

4.1.3 Daftar Aktor

4.1.3.1 *Waiters*

Bertindak sebagai pelayan pada restoran, bertugas untuk menyambut pelanggan, menawari menu makanan, mencatat pesanan dan menyerahkannya pada dapur untuk diproses lebih lanjut. Serta bertugas untuk memberikan informasi kepada pelanggan mengenai pesananannya. Jabatan *Waiters* diidentifikasikan sama yaitu sebagai *Waiters.*

4.1.3.2 Admin

Bertugas mengontrol pegawai yang berhak mengakses aplikasi. Berhak menambah, mengubah, maupun menghapus pegawai. Jabatan Admin diidentifikasikan sebagai *Administrator*.

4.1.3.3 Dapur

Bertindak sebagai jembatan antara *waiters* dengan koki. Bertugas untuk menerima pesanan dari *waiters* dan menyalurkannya pada koki untuk dimasak. Jabatan Dapur diidentifikasikan sebagai *Kitchen*.

4.1.3.4 Koki

Bertugas untuk memasak pesanan yang telah diserahkan ke dapur. Jabatan Koki diidentifikasikan sebagai *Koki*.

4.1.3.5 Kasir

Bertugas menerima pembayaran oleh pelanggan dan merupakan tahapan akhir proses pemesanan. Jabatan Dapur diidentifikasikan sebagai *Cashier*.

4.1.4 Daftar Kebutuhan

Daftar kebutuhan terdiri dari kebutuhan fungsional dan non-fungsional. Daftar kebutuhan ini disebut dengan *Software Requirement Specification* (SRS). Daftar kebutuhan fungsional ditunjukkan dengan penomoran SRSF. Pada Tabel 4.1 kebutuhan fungsional ditunjukkan dengan penomoran FW untuk *waiters*, FA untuk admin, FD untuk dapur, FI untuk koki dan FK untuk kasir, sedangkan kebutuhan non-fungsional ditunjukkan dengan penomoran NF.

Tabel 4.1 Tabel kebutuhan fungsional

Identifikasi	Kebutuhan	Use case
FW01	Sistem dapat menampilkan meja yang tersedia	Melihat meja tersedia
FW02	*Waiters* dapat menambahkan pelanggan baru	Tambah pelanggan
FW03	*Waiters* dapat memilih pelanggan	Pilih pelanggan
FW04	Sistem dapat menampilkan menu	Melihat menu makanan
FW05	*Waiters* dapat menambah pesanan dengan perangkat Android	Tambah pesanan
FW06	Sistem dapat menampilkan status pesanan	Melihat daftar pesanan
FW07	*Waiters* dapat mengubah dan menghapus pesanan	Edit pesanan, Delete pesanan
FW08	*Waiters* dapat mengirim pesanan	Kirim pesanan
FA01	Admin dapat menambah pegawai	Tambah pegawai
FA02	Admin dapat mengubah, dan menghapus pegawai	Edit pegawai, Delete pegawai
FD01	Dapur dapat melihat daftar pesanan	Melihat daftar pesanan
FD02	Dapur dapat mengubah status pesanan	Ubah status pesanan
FD03	Dapur dapat melihat daftar menu	Melihat menu makanan
FD04	Dapur dapat menambah daftar menu	Tambah menu makanan

FD05	Dapur dapat mengubah status ketersediaan menu	Update status menu makanan
FD06	Dapur dapat mengubah, dan menghapus data menu	Edit menu makanan, Delete menu makanan
FI01	Koki dapat melihat daftar pesanan yang antri	Tampil Daftar Antrian
FK01	Kasir dapat melihat daftar pelanggan	Melihat daftar pelanggan
FK02	Kasir dapat melihat daftar tagihan	Melihat tagihan
FK03	Kasir dapat mengubah status pelanggan	Ubah status Pelanggan

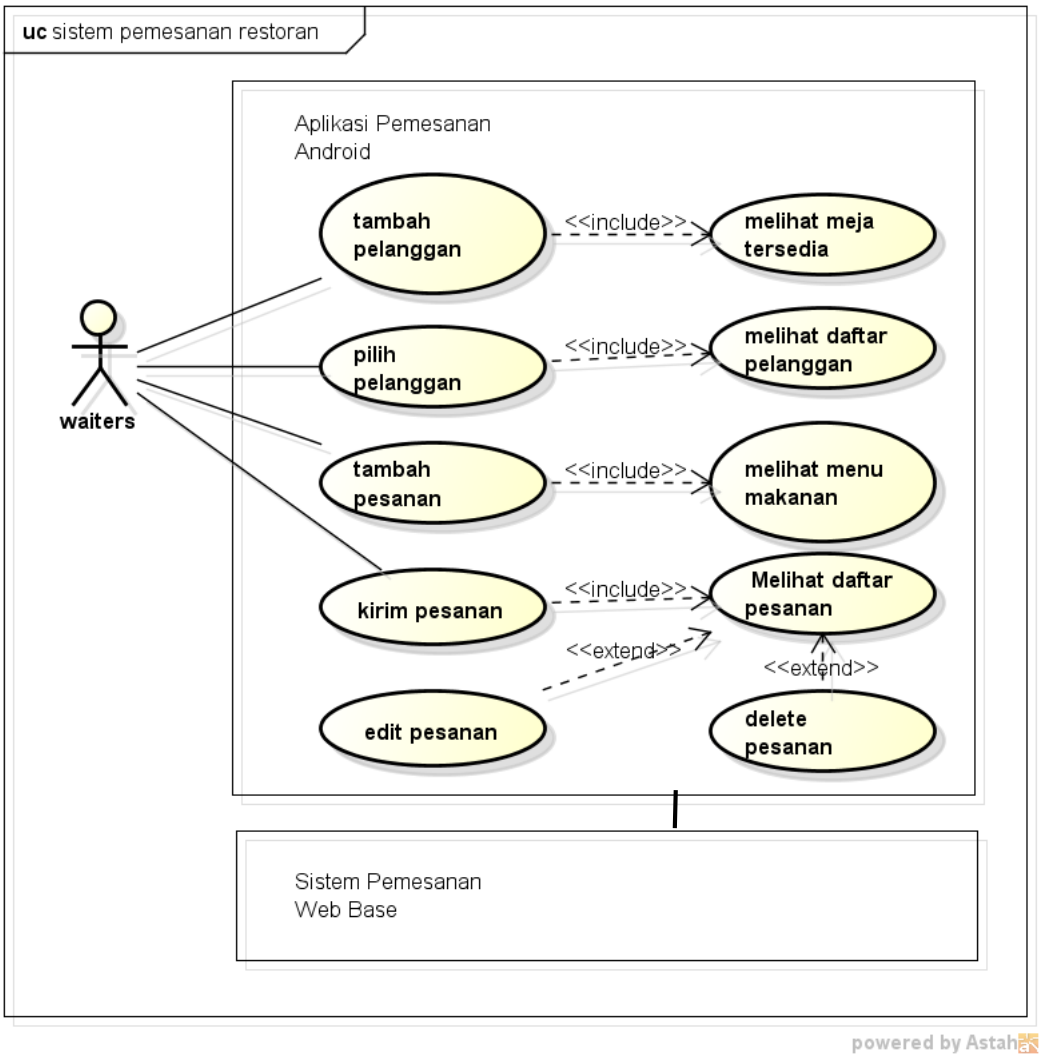
Tabel 4.1 merupakan tabel kebutuhan fungsional sistem. Masing–masing kebutuhan sistem harus dapat dipenuhi oleh sistem. Setiap kebutuhan sistem dimodelkan ke dalam bentuk *use case*, dan setiap kebutuhan fungsional harus dilakukan pengujian terlebih dahulu sebelum sistem dipublikasikan. Tabel 4.2 merupakan daftar kebutuhan non-fungsional.

Tabel 4.2 Tabel kebutuhan non-fungsional

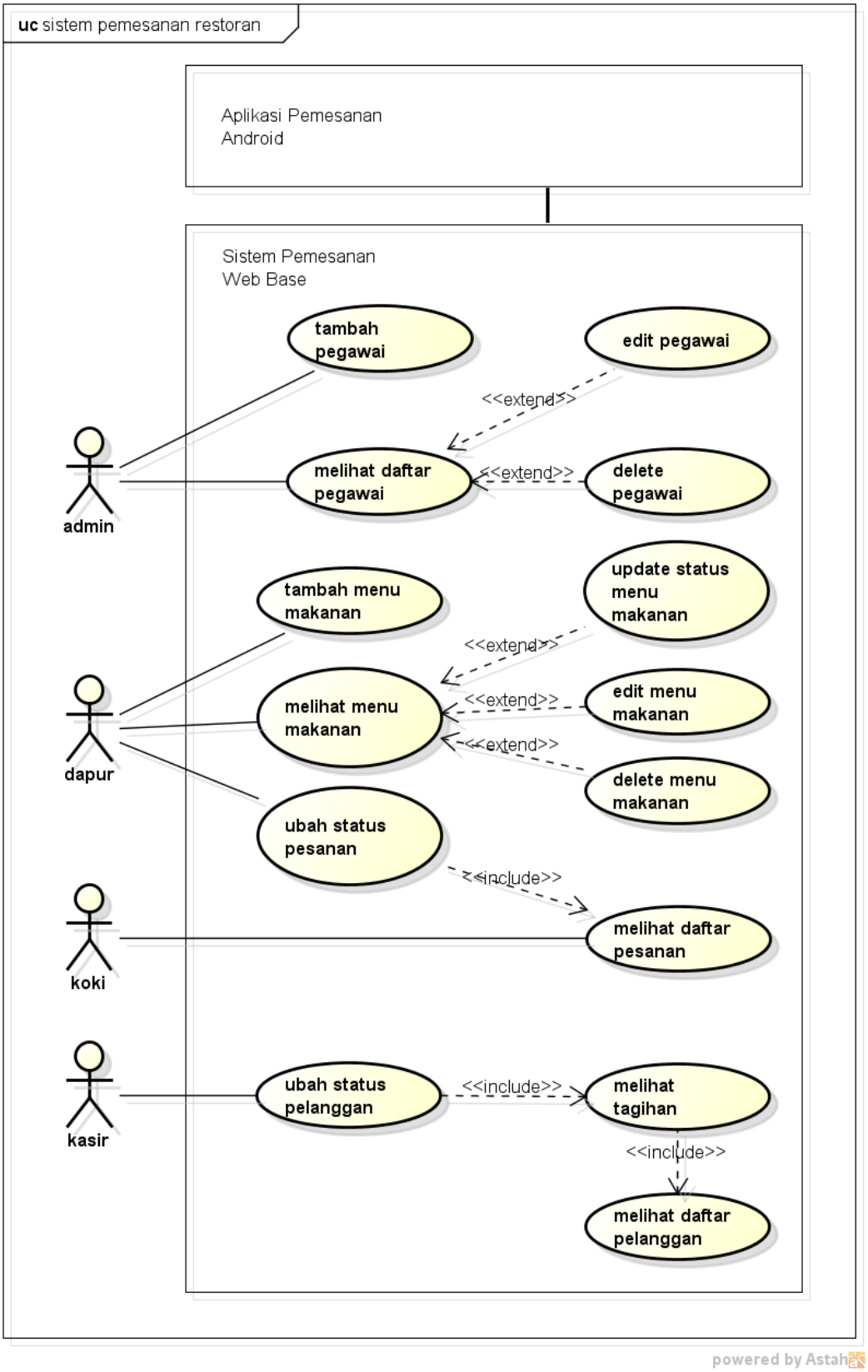
Parameter	Kode	Deskripsi Kebutuhan
Compatibility	NF01	Aplikasi dapat digunakan pada perangkat dengan perangkat berbasis Android untuk *waiters.* Dan perangkat dengan fasilitas web browser untuk admin, dapur, koki dan kasir
Authentication	NF02	Terdapat mekanisme login sebagai syarat untuk mengakses aplikasi.
Authorization	NF03	Setiap pegawai memiliki jabatan yang digunakan untuk mengakses fitur tertentu.

4.1.5 Diagram *Use Case*

Pemodelan *use case* sistem diperoleh dari kebutuhan fungsional yang digunakan untuk menggambarkan interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat.



Gambar 4.1 *Use Case Diagram Waiters*



Gambar 4.2 *Use Case Diagram* Admin, Dapur, Koki dan Kasir

Gambar 4.1 merupakan *use case diagram* untuk *waiters* dimana pengguna dapat melakukan mekanisme pemesanan pada aplikasi pemesanan berbasis Android. Mekanisme tersebut meliputi penambahan pelanggan baru dan melihat meja yang tersedia, dinotasikan dengan dependensi *include* karena penambahan pelanggan baru dapat dilakukan ketika ada meja yang tersedia, memilih pelanggan dan melihat daftar pelanggan yang menjadi tanggung jawab *waiters* tersebut, dinotasikan dengan dependensi *include* karena memilih pelanggan bisa dijalankan setelah menampilkan daftar pelanggan, penambahan pesanan baru dan melihat menu makanan yang juga dinotasikan dengan dependensi *include* karena penambahan pesanan baru hanya bisa dilakukan setelah melihat menu makanan, selain itu *waiters* juga dapat melihat daftar pesanan yang nantinya dapat di hapus atau diubah, dinotasikan dengan dependensi *extend*. *Use cese* tersebut juga menjelaskan bahwa aplikasi pemesanan android juga berinteraksi data sistem pemesanan *web base*.

Sedangkan pada Gambar 4.2 merupakan *use case diagram* untuk admin, dapur dan admin yang melakukan mekanisme pemesanan dengan sistem pemesanan *web base*. Untuk admin meliputi penambahan pegawai baru yang nantinya dapat mengakses sistem pemesanan dan melihat daftar pegawai serta mengubah maupun menghapusnya. Untuk dapur dapat mengubah status pesanan setelah melihat daftar pesanan yang ada, dinotasikan dengan dependensi *include*, menambahkan menu makanan baru, dan melihat daftar menu serta mengubah maupun menghapusnya. Untuk koki hanya dapat melihat daftar pesanan yang antri. Sedangkan untuk kasir dapat mengubah status pada pelanggan setelah melihat tagihan dan menerima pembayaran dari pelanggan yang dipilih dari daftar pelanggan aktif, dinotasikan pula dengan dependensi *include*.

Setiap *use case* yang ada pada Gambar 4.1 dan 4.2 dapat dijelaskan oleh skenario *use case*.

4.1.6 Skenario *Use Case*

Tabel 4.3 Skenario *Use Case* tambah pesanan

Nomor *Use Case*	FW05
Nama *Use Case*	Tambah pesanan
Tujuan	Menambahkan pesanan pada pelanggan
Prakondisi	Aplikasi sudah menunjuk pada seorang pelanggan
Kondisi Akhir	Pesanan ditambahkan
Aktor	*Waiters*
Alur Utama	1. *Waiters* melihat daftar menu dan memilih salah satu menu 2. *Waiters* memasukkan jumlah pesanan yang ditambahkan 3. Pesanan ditambahkan

Pada Tabel 4.3 menjelaskan tentang proses yang dalam melakukan tambah pesanan. Nomor *use case* merupakan kode dari tabel kebutuhan fungsional yang harus terpenuhi pada sistem. Nama *use case* merupakan fitur yang tertera pada *use case diagram* untuk *waiters*. Tujuan merupakan maksud dari diadakannya fitur tersebut yaitu untuk menambahkan pesanan baru. Prakondisi yaitu keadaan awal sebelum fitur dapat dijalankan yaitu sudah ada pelanggan dan aplikasi sudah menunjuk pada pelanggan tersebut. Kondisi akhir merupakan kondisi dimana fitur telah dijalankan yaitu terdapat pesanan baru yang ditambahkan pada daftar pesanan pelanggan yang ditunjuk. Aktor yaitu pengguna yang menjalankan fitur yaitu *waiters*. Alur utama yaitu garis besar langkah-langkah untuk menjalankan fitur, pertama aktor yaitu *waiters* melihat daftar menu dan memilih salah satu untuk ditambahkan setelah itu memasukkan jumlah yang akan dipesan lali menekan tombol tambah, jika proses ini berhasil maka akan terdapat pesanan baru yang ditambahkan.

Tabel 4.4 Skenario *Use Case* Mengubah Status pesanan

Nomor *Use Case*	FD02
Nama *Use Case*	Mengubah status pesanan
Tujuan	Melakukan pengubahan pada status pesanan
Prakondisi	Sudah ada pesanan yang dikirimkan oleh *waiters*
Kondisi Akhir	Status pesanan berubah
Aktor	Dapur
Alur Utama	1. Dapur melihat daftar pesanan dan memilih satu untuk diubah, lalu pilih status yang tersedia sesuai dengan pesanan. 2. Status pesanan sudah diperbarui

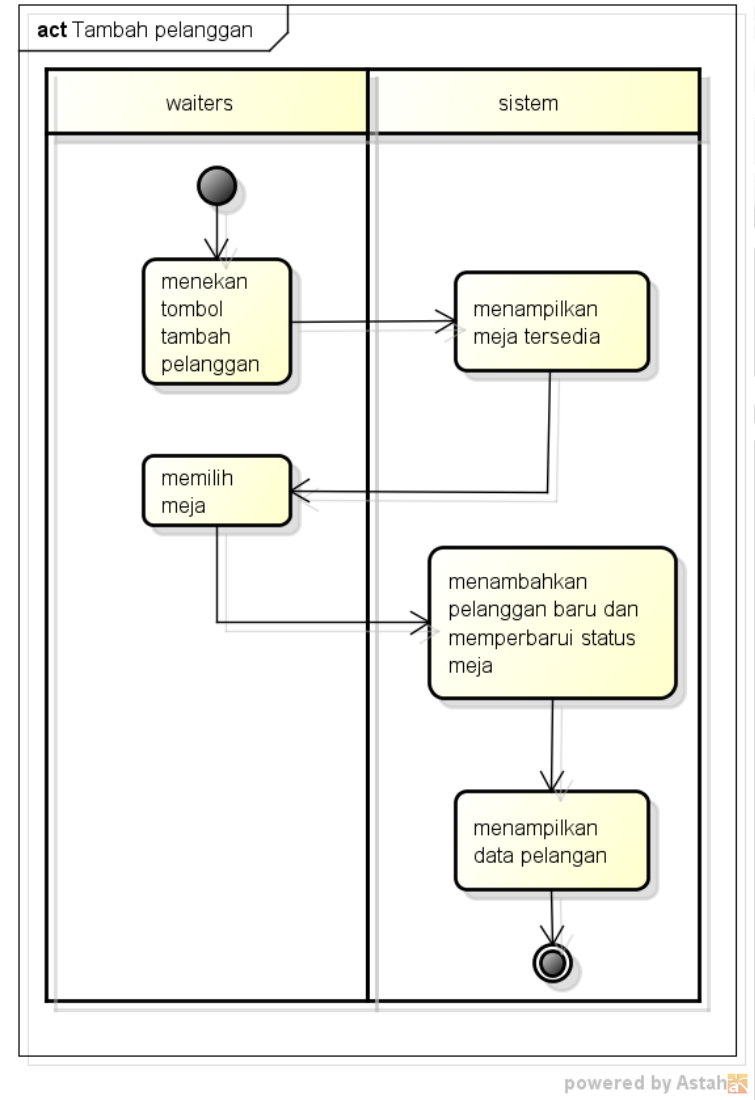
Pada Tabel 4.4 menjelaskan tentang proses yang dalam melakukan pengubahan status pesanan. Nomor *use case* merupakan kode dari tabel kebutuhan fungsional yang harus terpenuhi pada sistem. Nama *use case* merupakan fitur yang tertera pada *use case diagram.* Tujuan merupakan maksud dari diadakannya fitur tersebut yaitu untuk mengubah status pesanan. Prakondisi yaitu keadaan awal sebelum fitur dapat dijalankan yaitu sudah ada pesanan yang dikirimkan. Kondisi akhir merupakan kondisi dimana fitur telah dijalankan yaitu status pesanan berubah. Aktor yaitu pengguna yang menjalankan fitur yaitu dapur. Alur utama yaitu garis besar langkah-langkah untuk menjalankan fitur, pertama aktor yaitu dapur melihat daftar pesanan yang sudah dikirimkan dan memilih salah satu pesanan setelah menunjuk ke salah satu pesanan selanjutnya memilih status yang baru, jika proses ini berhasil maka status pesanan yang ditunjuk akan diperbarui.

Untuk tabel skenario yang selanjutnya terlampir pada Lampiran 1, menggunakan komponen yang sama dengan tabel skenario sebelumnya dan dijelaskan dengan cara yang sama.

4.1.7 Diagram *Activity*

Diagram *activity* adalah diagram untuk memodelkan aktivitas antara pengguna dan sistem yang berjalan berdasarkan pada skenario *use case.* Skenario

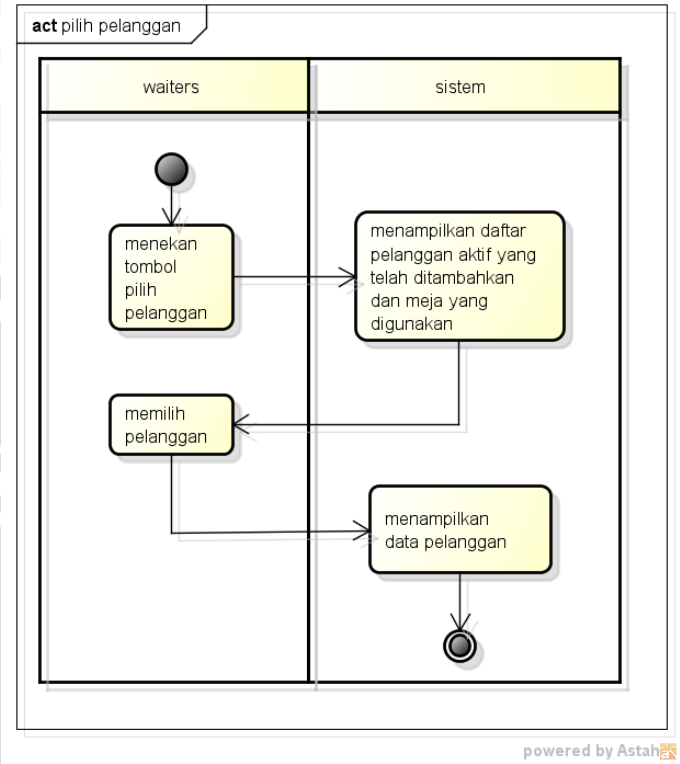
use case yang digambarkan akan dimodelkan kedalam diagram *activity*. Dengan diagram *activity*, dapat terlihat aktor yang melakukan proses tiap langkahnya.



Gambar 4.3 Diagram Aktifitas Menambah Pelanggan Baru

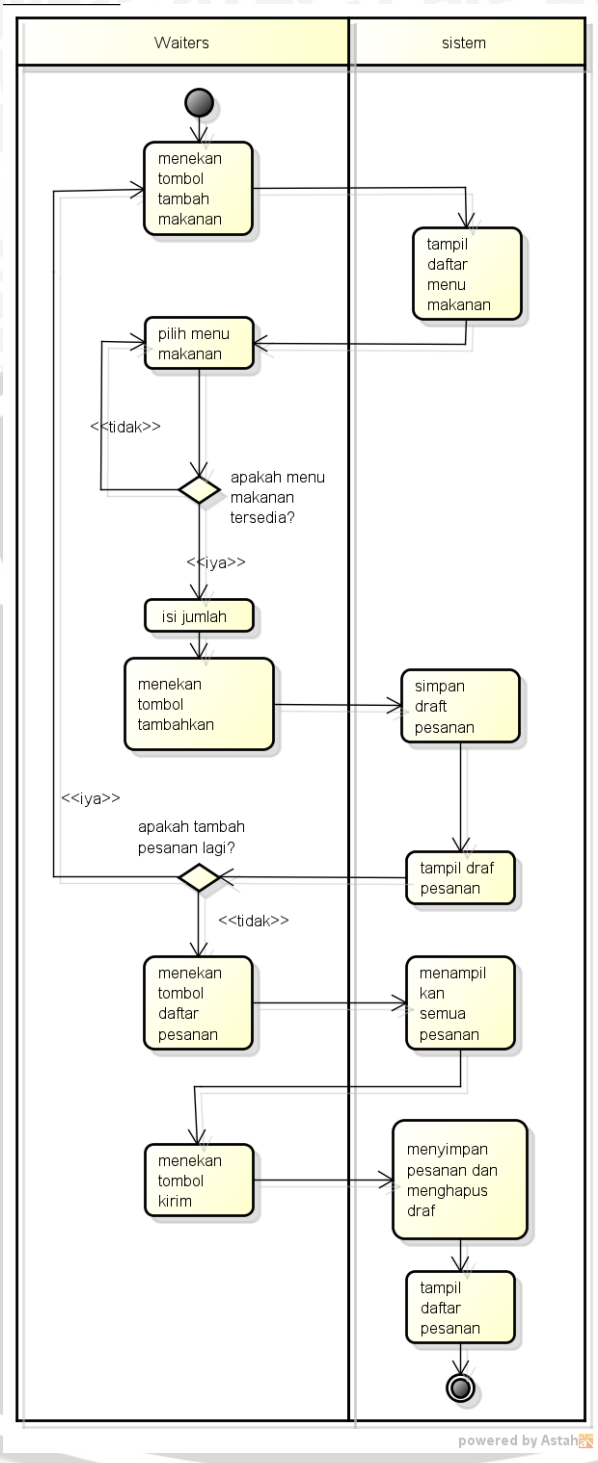
Gambar 4.3 merupakan diagram aktifitas yang menjelaskan proses *waiters* dalam menembahkan pelanggan baru. Saat ada pelanggan baru yang memasuki restoran yang pertama adalah *waiters* menekan tombol tambah pelanggan lalu sistem akan menampilkan daftar meja yang masih tersedia dan *waiters* memilih salah satu meja yang nantinya dapat ditempati oleh pelanggan, setelah itu sistem

akan menambahkan pelanggan baru dan memperbarui status meja yang ditempati menjadi tidak tersedia, lalu sistem akan menampilkan data pelananggan.



Gambar 4.4 Diagram Aktifitas Pilih Pelanggan

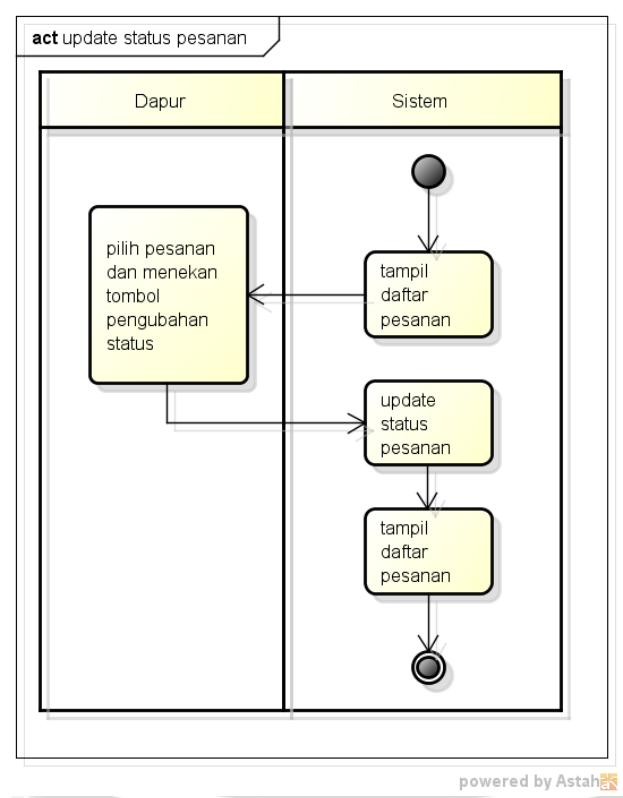
Gambar 4.4 merupakan diagram aktifitas yang menjelaskan proses *waiters* dalam memilih pelanggan tertentu untuk menampilkan datanya. Fitur ini berfungsi ketika seorang *waiters* telah menambahkan lebih dari satu pelanggan dan hendak menampilkan data pelanggan tertentu. Dimulai dengan *waiters* menekan tombol tambah pelanggan lalu sistem akan menampilkan daftar pelanggan, setelah itu *waiters* memilih salah satu pelanggan dan sistem akan menampilkan data pelanggan.



Gambar 4.5 Diagram Aktifitas tambah pesanan

Gambar 4.5 merupakan diagram aktifitas yang menjelaskan proses *waiters* dalam memasukkan pesanan pelanggan ke dalam sistem. Diawali dengan *waiters* menekan tombol tambah makanan lalu sistem akan menampilkan daftar menu

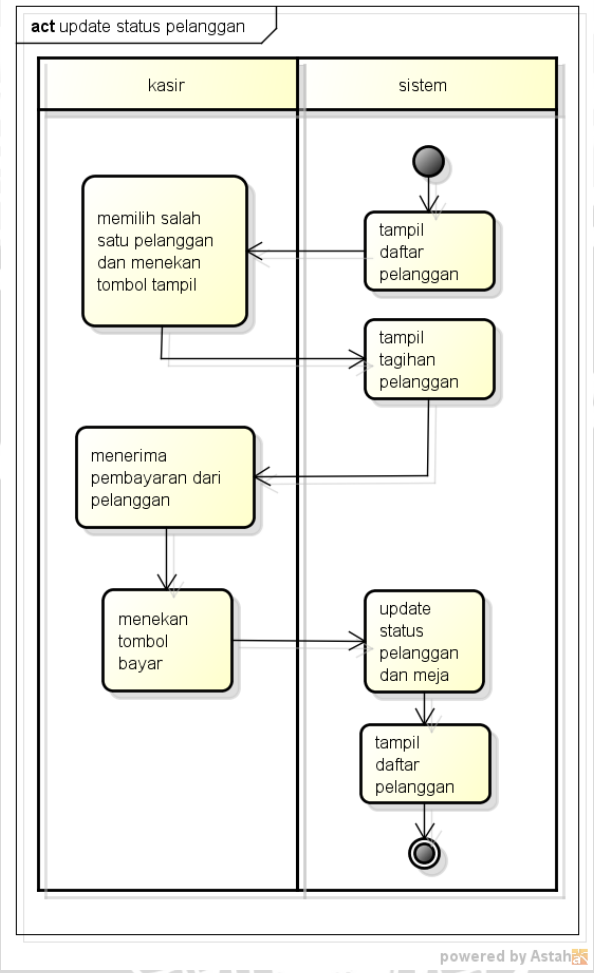
makanan dilanjutkan oleh *waiters* memilih menu makanan untuk dipesan, apabila menu makanan yang dipilih tidak tersedia atau habis maka *waiters* memilih menu makanan yang lainnya namun apabila tersedia akan muncul *form* yang diisi *waiters* dengan jumlah pesanan dan menekan tombol tambahkan. Setelah itu sistem akan menampilkan daftar draf pesanan, jika masih akan menambahkan pesanan lagi *waiters* menekan kembali tombol tambah makanan dan sistem akan menampilkan kembali daftar menu makanan namun jika sudah selesai maka *waiters* menekan tampil daftar pesanan maka sistem akan menampilkan semua pesanan baik yang sudah terkirim maupun masih *draft* setelah itu *waiters* menekan tombol kirim untuk menyimpan pesanan *draft* dan sistem akan menyimpan pesanan dan menampilkan daftar pesanan secara keseluruhan. Diagram aktifitas tambah pesanan ini juga berlaku untuk tambah minuman dan jenis menu lainnya.



Gambar 4.6 Diagram Aktifitas *update* status pesanan

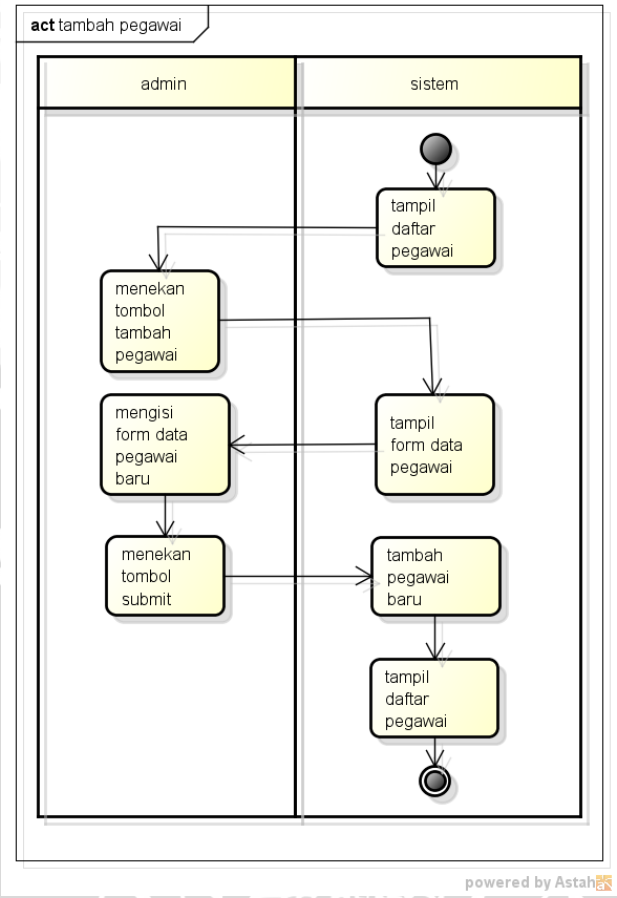
Gambar 4.6 merupakan diagram aktifitas yang menjelaskan proses dapur dalam memperbarui status pesanan ke dalam sistem yang nantinya dapat dibaca oleh *waiters*. Proses diawali dari sistem yang menampilkan daftar pesanan yang tersimpan kemudian dapur menekan tombol pengubahan status yang menjadi status

baru dari pesanan, setelah itu sistem akan meperbarui status pesanan dan menyimpannya lalu menampilkan kembali daftar pesanan.



Gambar 4.7 Diagram Aktifitas *update* status pelanggan dan meja

Gambar 4.7 merupakan diagram aktifitas yang menjelaskan proses kasir dalam memperbarui status pelanggan yang sudah menyelesaikan proses pemesanan ke dalam sistem, ditandai oleh pelanggan yang membayar sesuai tagihan, serta memperbarui status meja menjadi tersedia. Diawali dengan sistem menampilkan daftar pelanggan kemudian kasir memilih salah satu pelanggan yang hendak membayar dan menekan tompol tampil dan sistem akan menampilkan tagihan dari pelanggan tersebut, lalu kasir akan menerima uang pembayaran dan menekan tombol bayar maka sistem akan memperbarui status pelanggan menjadi tidak aktif dan status meja menjadi tersedia kembali dan dapat digunakan oleh pelanggan yang lain. Setelah itu sistem akan menampilkan kembali daftar pelanggan.



Gambar 4.8 Diagram Aktifitas *menambah* pegawai baru

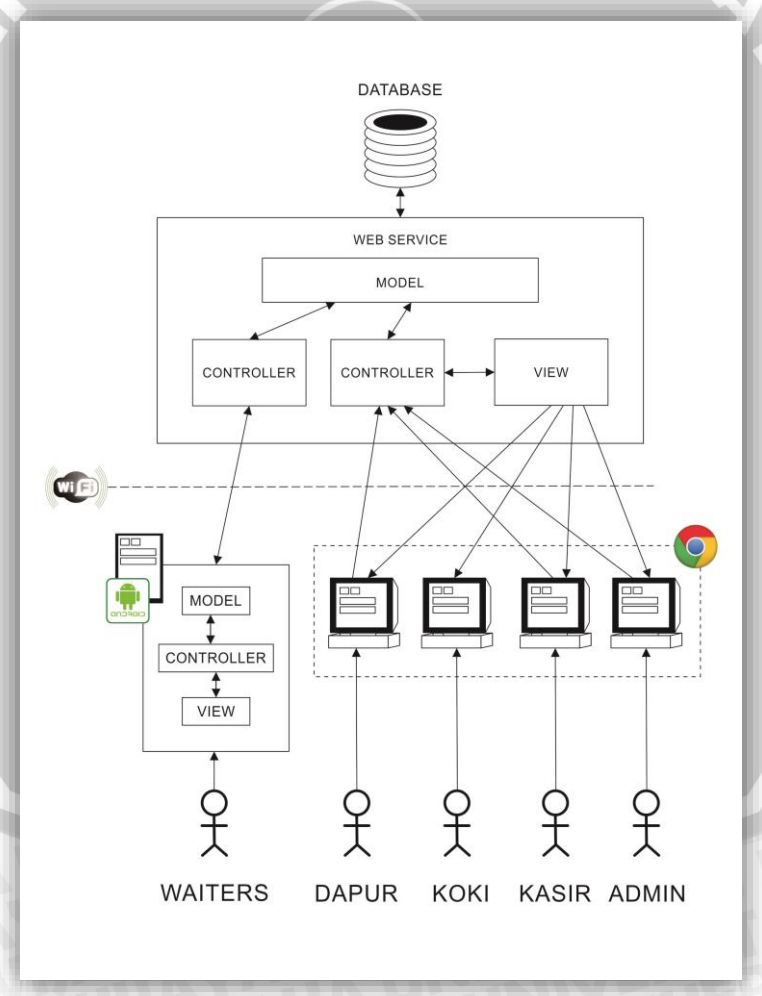
Gambar 4.8 merupakan diagram aktifitas yang menjelaskan proses admin dalam menambahkan pegawai baru. Diawali oleh sistem menampilkan daftar pegawai lalu admin menekan tombol tambah pegawai maka sistem akan menampilkan *form* mengenai data pegawai yang harus diisi oleh admin, setelah form diisi oleh admin lalu menekan tombol *submit* maka sistem akan menyimpan pegawai baru tersebut dan menampilkan daftar pegawai.

4.2 Perancangan

Pada sistem pemesanan ini terdiri dari Perancangan Arsitektural, Perancangan *Sequence*, Perancangan Kelas, Perancangan Basis Data dan Perancangan Antar muka.

4.2.1 Perancangan Arsitektural

Perancangan arsitektural pada sistem ini menggunakan metode MVC *(Model View Controler). View* merupakan antarmuka yang digunakan untuk berkomunikasi dengan pengguna, *Model* adalah yang bertugas menipulasi data yang dibutuhkan, sedangkan *Controler* bertugas menjembatani antara *View* dan *Model*.



Gambar 4.9 Arsitektur Sistem

Dari Gambar 4.9 dapat diketahui bahwa terdapat dua sistem yaitu aplikasi pemesanan berbasis android yang digunakan oleh *waiters* dan sistem pemesanan berbasis *web (web service)* yang digunakan oleh dapur, koki, kasir dan admin. Untuk aplikasi yang digunakan oleh *waiters*, dirancang menggunakan pola perancangan MVC (*Model View Controller*) diterapkan pada perangkat *mobile* berbasis android yaitu *smartphone* atau *tablet PC*. Prosesnya adalah *waiters* akan berinteraksi dengan sistem menggunakan *view* (*layout*) pada perangkat Android, *controller* (*activity*) akan melakukan manipulasi terhadap data yang diambil oleh *model.*

Untuk sistem pemesanan berbasis *web* juga dirancang menggunakan pola perancangan MVC. *Model* yang terdapat pada aplikasi pemesanan berbasis android terhubung dengan *controller* yang terdapat pada *web service* melalui *wireless (wifi)* dan saling bertukar data, *controller* sendiri bertugas mengatur data yang diterima untuk disalurkan ke *model* dan mengirim data yang didapatkan dari model. Data yang dikirimkan melalui jaringan (*wifi*) menggunakan format json.

Kode 4.1 Format Json yang digunakan

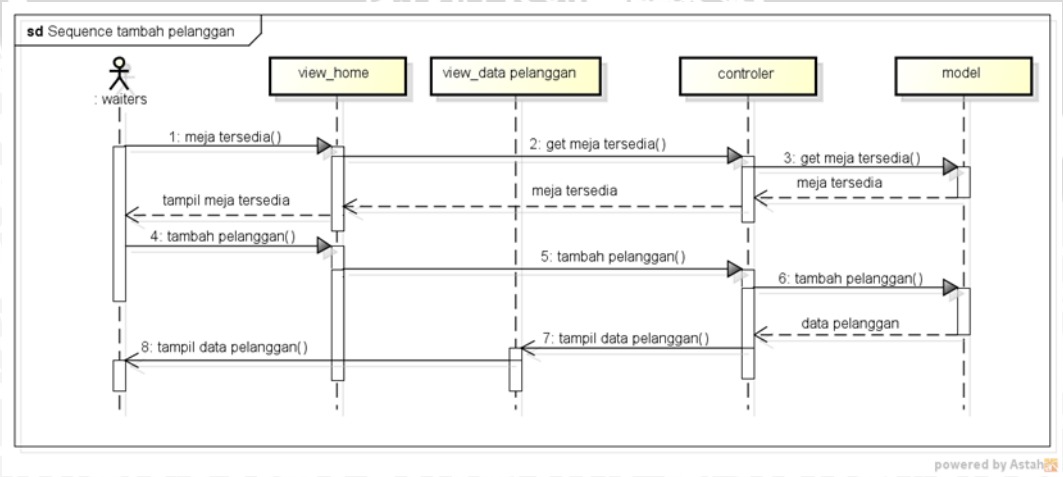
```
{"items":[{"id_meja":"2","status":"tersedia"},{"id_meja":"3","status":"tersedia"}
,{"id_meja":"5","status":"tersedia"},{"id_meja":"6","status":"tersedia"}]}
```

Terdapat dua jenis *controller* pada *web service*, yang pertama yaitu *controller* tanpa *view*, yaitu yang berhubungan dengan aplikasi pemesanan berbasis android dan yang kedua *controller* dengan *view* yaitu yang digunakan oleh dapur, kasir dan admin diakses menggunakan aplikasi *web browser* yang terpasang pada pernagkat komputer. *View* bertugas untuk menampilkan antarmuka kepada pengguna ketika pengguna mengakses ke *controller*. *Controller* dengan *view* bertugas mengatur data masuk lalu dikirimkan ke *model* dan menampilkan data dari *model* ke *view*. *Model* pada web service bertugas untuk mengatur data yang didapat dari kedua *controller* untuk disimpan ke *database* dan mengambil data untuk dikirimkan ke *controller*. Untuk koki hanya mampu melihat data pesanan yang antri yang dikirimkan melalui *view*.

Proses pemesanan dimulai dari pelanggan baru yang datang disambut oleh *waiters* dan mendaftarkannya pada sistem dan mengubah status meja yang ditempati pelanggan menjadi aktif. Dilanjutkan dengan *waiters* yang mencatat pesanan menggunakan perangkat android, setelah pencatatan selesai dan daftar pesanan disetujui oleh pelanggan maka daftar pesanan dikirimkan ke dapur. Daftar pesanan yang dikirim dapat dilihat oleh dapur dan koki. Koki melihat daftar pesanan melalui layar tersendiri, ketika hendak memasak salah satu pesanan maka melapor ke dapur agar dirubah statusnya, begitu pula ketika selesai memasaknya. Pihak dapur dapat mengatur antrian pesanan berdasarkan waktu dikirimkannya pesanan yang sudah diurutkan, untuk pengelompokan proses memasaknya dapur dapat menentukan sesuai kebijakan restoran setempat. Setelah proses pemesanan pelanggan membayar dan kasir mengubah status pelanggan menjadi tidak aktif dan status meja menjadi tersedia kembali.

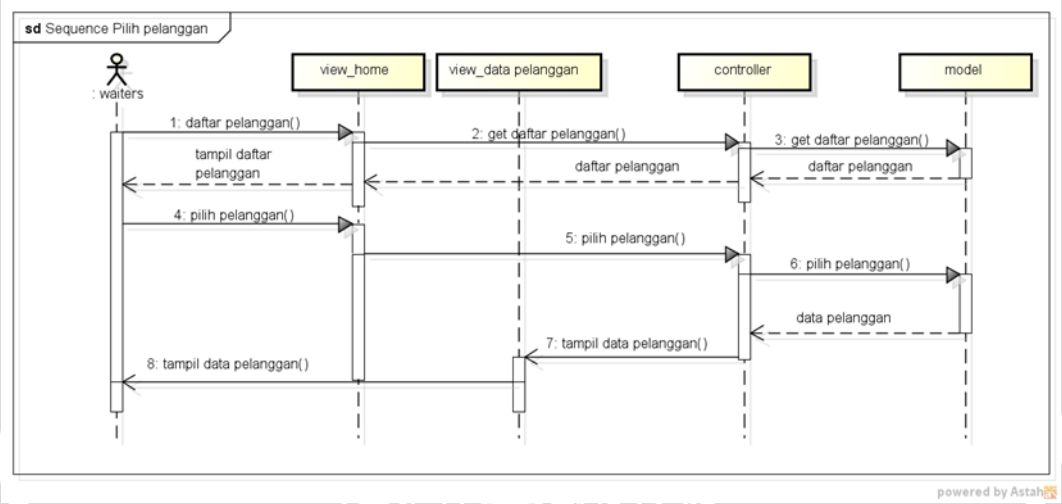
4.2.2 Perancangan *Sequence Diagram*

Diagram ini berupa skenario atau rangkaian langkah-langkah yang dilakukan dari sebuah kegiatan untuk menghasilkan output tertentu.



Gambar 4.10 *Sequence Diagram* tambah pelanggan

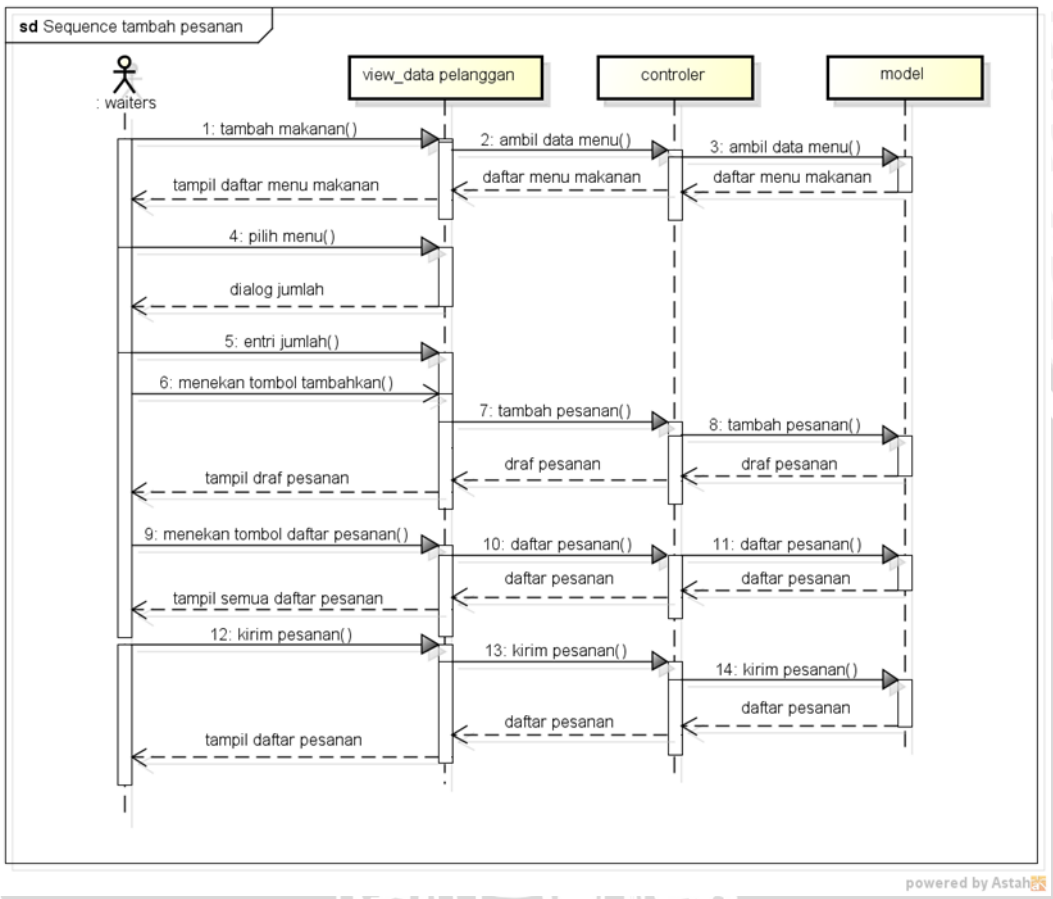
Gambar 4.10 merupakan *Sequence Diagram* tambah pelanggan menjelaskan tentang proses *waiters* saat menambahkan pelanggan baru sesuai meja yang masih dalam keadaan tersedia dan bisa digunakan. Dimulai dengan *waiters* melihat meja yang tersedia dengan mengakses fungsi meja_tersedia() pada *view_home* dan merespon dengan menampilkan dialog yang berisi daftar meja yang tersedia, didapatkan dari *model* melalui *controler* menggunakan fungsi get meja tersedia(), kemudian *waiters* memilih salah satu meja untuk ditempati dengan mengakses fungsi tambah_pelanggan() pada *view_home* lalu diteruskan ke *controller* melalui fungsi tambah_pelanggan() dan diteruskan kembali ke *model* untuk disimpan melalui fungsi tambah_pelanggan(). Setelah menyimpan data, *model* mengembalikan data pelanggan ke *controller* dan diteruskan ke *view_data_pelanggan* untuk ditampilkan.



Gambar 4.11 *Sequence Diagram* Pilih Pelanggan

Gambar 4.11 merupakan *Sequence Diagram* pilih pelanggan menjelaskan *waiters* saat memilih pelanggan untuk ditampilkan datanya. Dimulai dengan *waiters* mengakses fungsi daftar_pelanggan() pada *view_home* dan merespon dengan menampilkan dialog berisi daftar pelanggan aktif dan meja yang digunakan yang didapatkan dari *model* melalui *controller* menggunakan fungsi get daftar pelanggan(), kemudian *waiters* memilih salah satu dari daftar untuk ditampilkan dengan mengakses fungsi pilih pelanggan() pada *view_home* lalu diteruskan ke *controller* melalui fungsi pilih pelanggan() dan diteruskan kembali ke *model* untuk

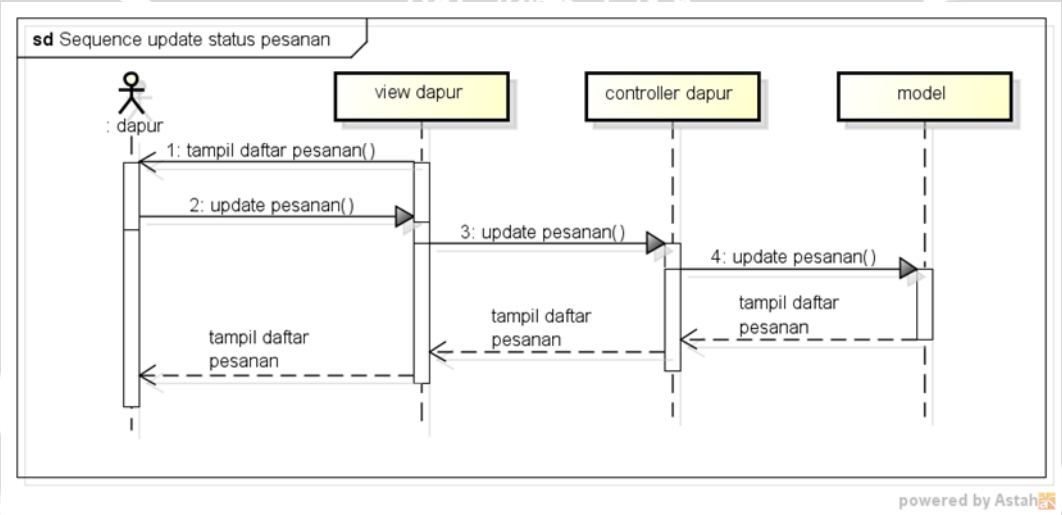
disimpan melalui fungsi pilih pelanggan(). Setelah mengambil data, *model* mengembalikan data pelanggan ke *controller* dan diteruskan ke *view_data_pelanggan* untuk ditampilkan.



Gambar 4.12 *Sequence Diagram* tambah pesanan

Gambar 4.12 merupakan *Sequence Diagram* tambah pesanan menjelaskan *waiters* saat menambahkan pesanan baru kepada pelanggan yang ditunjuk. Dimulai dengan *waiters* yang mengakses fungsi tambah_makanan() pada *view_data_pelanggan* dan merespon dengan menampilkan dialog berupa daftar makanan yang ditawarkan, didapatkan dari *model* melalui *controller* menggunakan fungsi ambil data menu(), lalu *waiters* memilih salah satu untuk dipesan dan sistem merespon dengan menampilkan form yang diisi jumlah yang akan dipesan, kemudian *waiters* memasukkan jumlahnya pada form dan menyimpannya dengan mengakses fungsi entri_jumlah() dan menekan tombol tambahkan() pada *view_data_pelanggan*, setelah itu diteruskan ke *controller* melalui fungsi

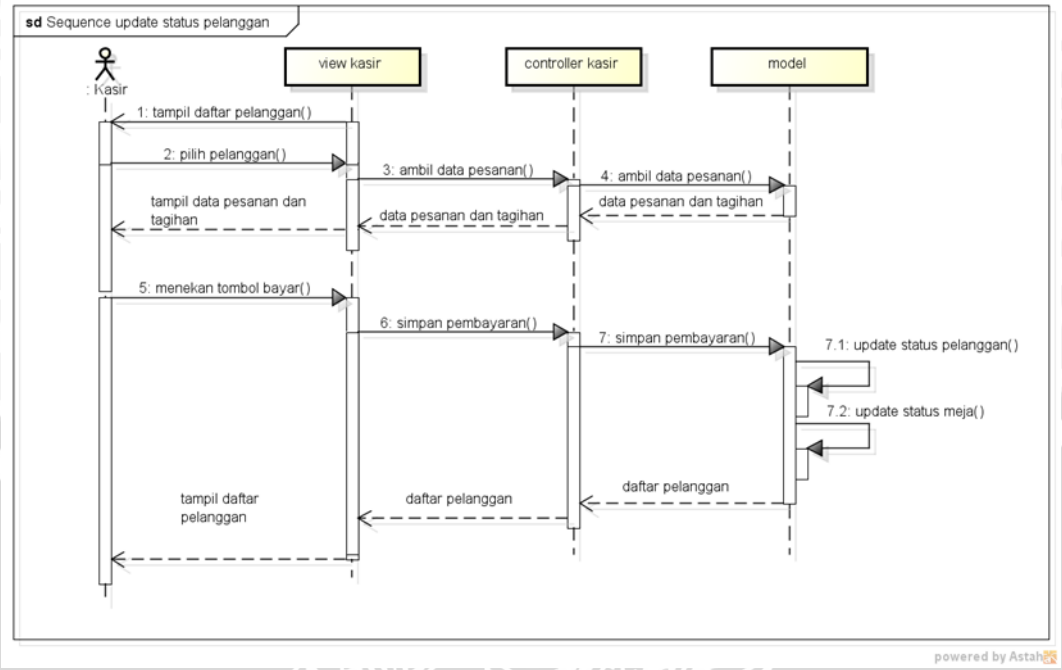
tambah_pesanan() dan diteruskan kembali ke *model* untuk disimpan melalui fungsi tambah_pesanan(). Setelah menyimpan data, *model* mengembalikan daftar *draft* pesanan ke *controller* untuk ditampilkan. *Draft* pesanan merupakan pesanan yang belum dikirimkan untuk dilihat oleh pihak dapur. Setelah *draft* siap dan disepakati untuk dikirim *waiters* dengan cara pertama menekan tombol daftar pesanan yang nantinya *controller* akan mengambil semua daftar pesann yang sudah terkirim maupun belum dari *model* untuk ditampilkan ke *view_data_pelanggan* setelah itu *waiters* mengirim pesanan dengan mengakses fungsi kirim_pesanan() lalu diteruskan ke *controller* kemudian *model* untuk disimpan. Setelah menyimpan data, *model* mengembalikan daftar pesanan yang ke *controller* dan diteruskan ke *view_data_pelanggan* untuk ditampilkan.



Gambar 4.13 *Sequence Diagram* update status pesanan

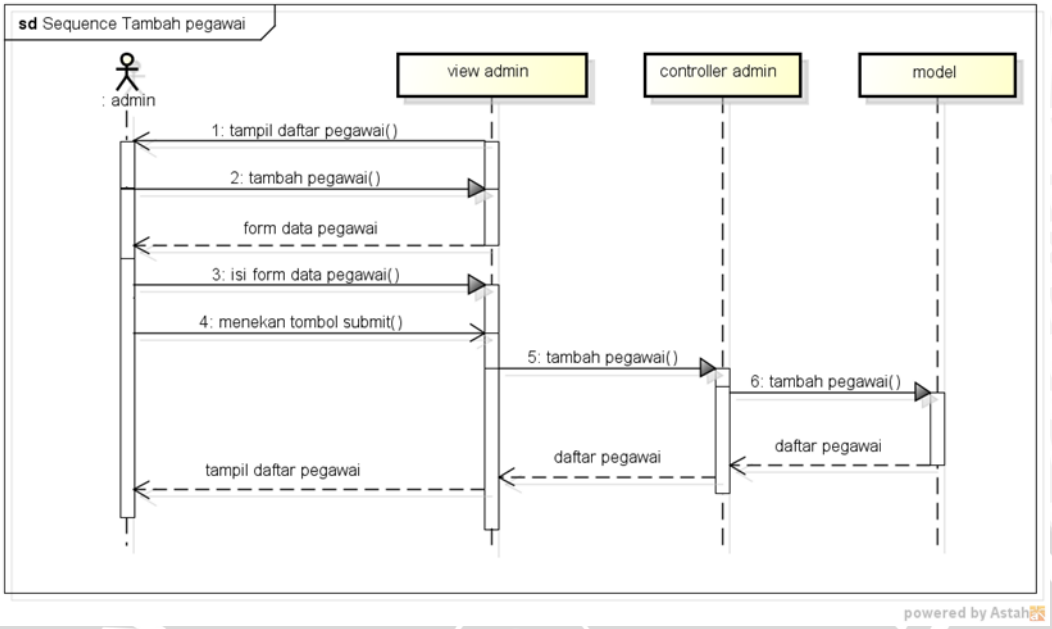
Sequence Diagram update status pesanan menjelaskan saat dapur mengubah status dapi pesanan pelanggan yang dikirim oleh *waiters*. *Update* dapat berupa status proses untuk pesanan yang sedang diproses dan *done* untuk pesanan yang sudah selesai diproses dan dihidangkan. Dimulai dengan sistem yang menampilkan daftar pesanan lalu dapur memilih salah satu dari daftar pesanan untuk diubah dan memilih status yang baru dengan mengakses fungsi update_pesanan() pada *view_dapur* lalu diteruskan ke *controller* melalui fungsi update_pesanan() dan diteruskan kembali ke *model* untuk disimpan melalui fungsi update_pesanan(). Setelah memperbarui dan menyimpan status yang baru, *model*

mengembalikan daftar pesanan ke *controller* yang akan diteruskan ke view_dapur untuk ditampilkan



Gambar 4.14 *Sequence Diagram* update status pelanggan dan meja

Sequence Diagram update status pelanggan dan meja menjelaskan tentang proses kasir dalam memperbarui status pelanggan yang sudah membayar bserta mengubah status meja yang sebelumnya ditempati menjadi tersedia. Dimulai dengan sistem menampilkan daftar pelanggan yang aktif lalu kasir yang memilih salah satu pelanggan untuk ditampilkan tagihan melalui fungsi pilih_pelanggan() pada view_kasir dan meresponnya dengan mengembalikan data tagihan pelanggan yang dipilih yang didapatkan dari *model* melalui *controller* kasir dengan mengakses fungsi ambil_data_pesanan(), kemudian setelah kasir menerima pembayaran kasir menekan tombol bayar() pada view_kasir lalu diteruskan ke *controller* melalui fungsi simpan_pembayaran() dan diteruskan kembali ke *model* untuk disimpan melalui fungsi simpan_pembayaran(). Pada fungsi simpan_pembayaran() terdapat pemanggilan fingsi yang lain yaitu update status pelanggan() dan update status meja(), kemudian mengembalikan daftar pelanggan ke *controller* yang akan diteruskan ke view_kasir untuk ditampilkan.

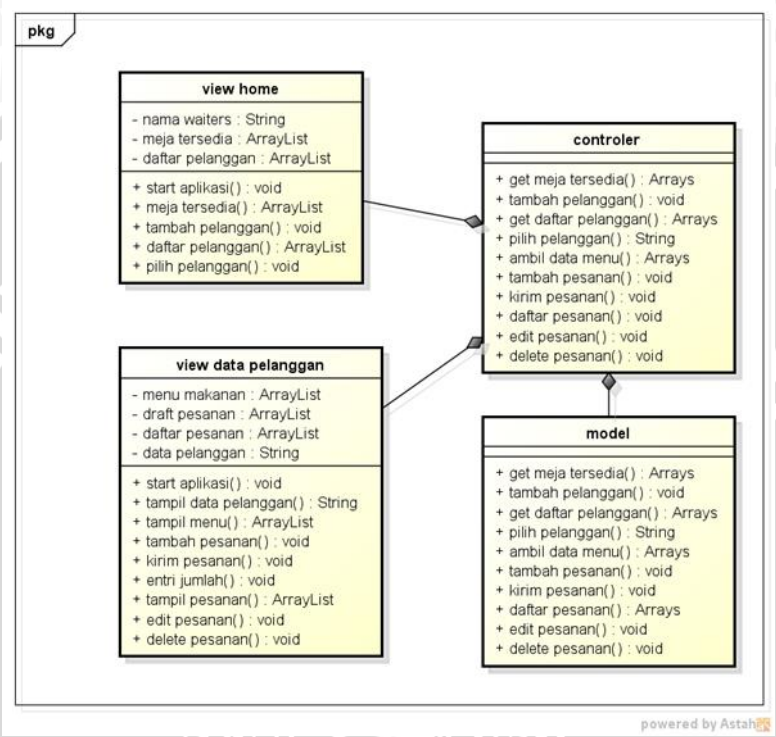


Gambar 4.15 *Sequence Diagram* tambah pegawai

Sequence Diagram tambah pegawai menjelaskan saat admin menambahkan pegawai baru yang nantinya dapat login sesuai jabatannya yaitu *waiters*, dapur, kasir atau admin itu sendiri. Dimulai dengan sistem menampilkan daftar pegawai lalu admin yang mengakses fungsi tambah_pegawai() pada *view_admin* dan merespon dengan menampilkan dialog berupa form data pegawai, kemudian admin memasukkan data pegawai yang baru pada form dan menyimpannya dengan mengakses fungsi submit() pada *view_admin* lalu diteruskan ke *controller* melalui fungsi tambah_pegawai() dan diteruskan kembali ke *model* untuk disimpan melalui fungsi tambah_pegawai(). Setelah menyimpan data, *model* mengembalikan daftar pegawai ke *controller* dan diteruskan ke view_admin untuk ditampilkan.

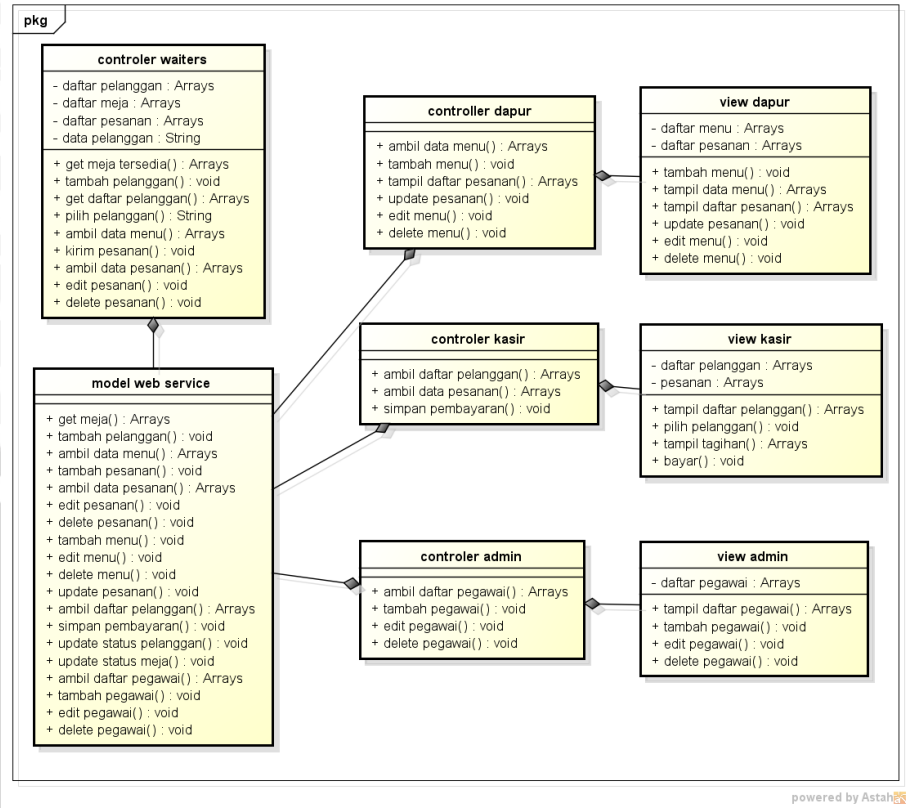
4.2.3 Perancangan Kelas

Diagram ini digunakan untuk mengidentifikasi kelas-kelas serta paket-paket yang terdapat dalam sistem, kemudian dilanjutkan dengan mengidentifikasi hubungan antar kelas.



Gambar 4.16 *Class Diagram* aplikasi

Pada gambar 4.16 diatas menjelaskan *class diagram* yang terdapat pada aplikasi perangkat Android yang nantinya akan dipegang oleh *waiters*. *View* merupakan kelas yang berisi fungsi-fungsi yang akan berinteraksi langsung dengan *user* yaitu *waiters* dan bertugas untuk mengelola tampilan dan data inputan dari *user*. Terdapat dua buah *view* yaitu *view_home* dan *view_data_pelanggan*. *Controler* yaitu kelas yang berisi fungsi-fungsi untuk mengolah dan mengontrol data dan bertugas menjembatani antara *view* dan *model*, sedangkan *model* yaitu kelas yang berisi fungsi-fungsi yang berhubungan dengan media penyimpanan bertugas untuk mengirim dan mengambil data. *View* dan *model* merupakan kelas yang diinisialisasi di dalam *controller* oleh karena itu dihubungkan dengan notasi composition.

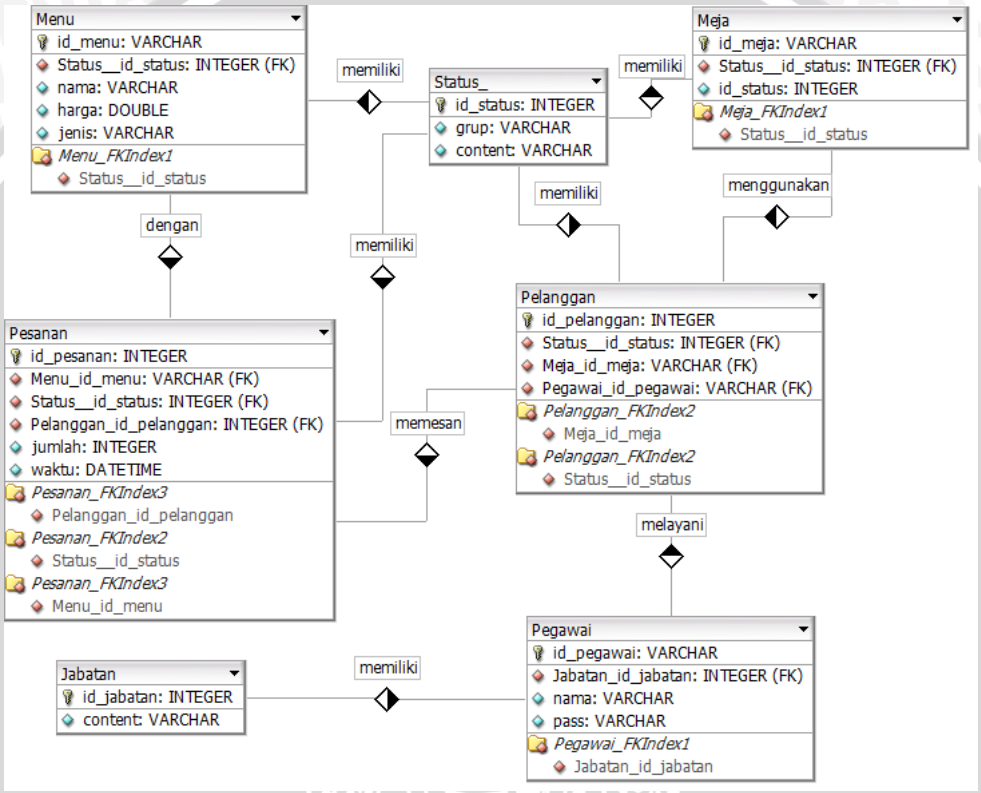


Gambar 4.17 Class Diagram web server

Pada gambar 4.17 diatas menjelaskan *class diagram* yang terdapat pada *web server*. *Waiters* yang menggunakan aplikasi perangkat Android akan mengakses *cotroller* milik *waiters*, sedangkan untuk dapur, kasir dan admin yang mengakses sistem menggunakan *browser* pada personal komputer akan mengakses *view* masing-masing. Semua *view* dikontrol oleh *controller* masing-masing yang terhubung ke *model*. *Model* berisi fungsi-fungsi yang berhubungan dengan media penyimpanan secara langsung yaitu *database*, bertugas untuk menyimpan dan mengambil data. *View* dan *model* merupakan kelas yang diinisialisasi di dalam *controller* oleh karena itu dihubungkan dengan notasi composition.

4.2.4 Perancangan Basis Data

Basis data (Database) merupakan tempat penyimpanan paling optimal, oleh karena itu diperlukan Perancangan Basis Data untuk menentukan susunan data. Tujuan dari perancangan ini adalah untuk mengoplimalisasi pengambilan dan penyimpanan data. Rancangan Basis Data yang digunakan adalah sebagai berikut:



Gambar 4.18 Perancangan Basis Data

Gambar 4.18 merupakan rancangan basis data, merupakan rancangan hubungan antar entitas yang nantinya disimpan dalam database. Terdapat tujuh entitas yang berhubungan. Pertama yaitu Status, berfungsi menyimpan data status. Kedua Menu, berfungsi menyimpan menu makanan dan minuman. Selanjutnya Meja, berfungsi menyimpan daftar meja yang disediakan. Kemudian Pelanggan, berfungsi menyimpan daftar pelanggan dan memiliki hubungan dengan entitas meja yaitu "menggunakan". Lalu Pesanan, berfungsi menyimpan daftar pesanan yang dipesan oleh pelanggan hubungan dengan entitas pelanggan adalah "memesan"

sedangkan dengan menu adalah “dengan”. Entitas Menu, Meja, Pelanggan dan Pesanan memiliki hubungan dengan entitas Status adalah “memiliki”. Selanjutnya Jabatan, berfungsi menyimpan daftar jabatan dari pegawai . Dan yang terakhir adalah Pegawai, berfungsi menyimpan daftar pegawai yang berhak mengakses aplikasi, memiliki hubungan dengan entitas pelanggan adalah “melayani” dan memiliki hubungan dengan entitas jabatan yaitu “memiliki”. Pada garis penghubung antar entitas terdapat segiempat dengan dua sisi warna yang berbeda, diartikan hubungannya yaitu satu dan banyak, satu untuk entitas dengan warna terang dan banyak untuk warna gelap.

4.2.5 Perancangan Antarmuka

Perancangan antarmuka merupakan kerangka pembuatan sebuah aplikasi. Tujuan dari perancangan antar muka untuk membuat sebuah tampilan dari aplikasi yang sederhana agar memudahkan user dalam mengoperasikan aplikasi.

4.2.5.1 Antarmuka *Waiters*

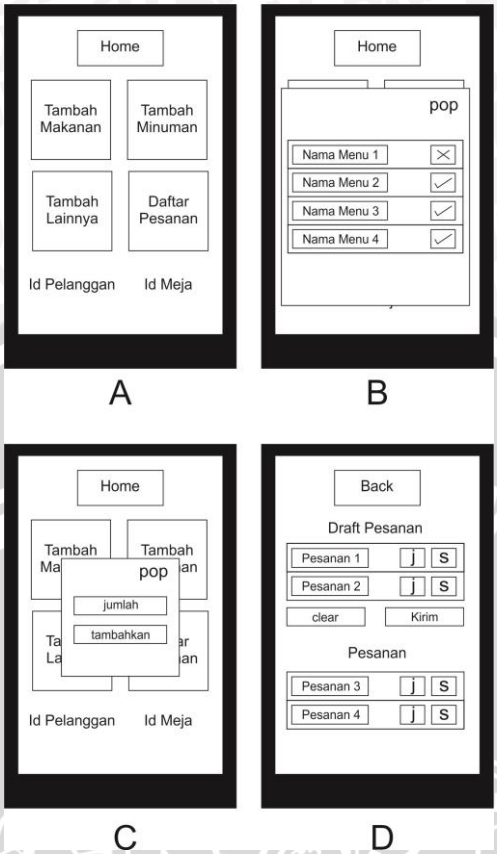
Gambar 4.19 Antarmuka *Home* untuk *Waiters*

Pada gambar 4.19 merupakan rancangan antarmuka untuk halaman *home* pada perangkat Android yang dioperasikan oleh *waiters*. Pada halaman

ini terdapat judul aplikasi, nama *waiters*, fungsi untuk menambahkan pelanggan baru dan memilih pelanggan serta tombol *logout*. Berikut keterangan mengenai antarmuka *home waiters:*

Tabel 4.5 Keterangan Antarmuka Halaman *Home Waiters*

Nama	Keterangan
Restoholic	Merupakan judul dari aplikasi
Tambah Pelanggan	Tombol berfungsi untuk menambah pelanggan baru
Pilih Pelanggan	Tombol yang berfungsi untuk memilih pelanggan yang merupakan tanggung jawab dari *waiters* yang *login*
Nama *waiters*	Menampilkan nama *waiters* yang sedang *login*
Logout	Tombol yang berfungsi untuk keluar dari akun *waiters*



Gambar 4.20 Antarmuka Data Pelanggan Untuk *Waiters*

Pada gambar 4.20 merupakan rancangan antarmuka untuk halaman menampilkan data pelanggan pada perangkat Android yang dioperasikan oleh *waiters*. Gambar A merupakan tampilan awal data pelanggan yang berisi tombol untuk kembali ke halaman home, tombol tambah makanan, tombol tambah minuman, tombol tambah lainnya, tombol tampil daftar pesanan, id pelanggan dan meja yang ditempati pelanggan. Gambar B merupakan tampilan daftar menu apabila *waiters* menekan salah satu dari tombol tambah makanan, tombol tambah minuman atau tombol tambah lainnya, maka muncul *popup dialog* yang berisi daftar menu yang bisa dipesan oleh pelanggan serta status ketersediaannya. Gambar C merupakan tampilan dialog pesanan apabila salah satu menu dipilih, maka akan muncul *popup* dialog yang berisi form jumlah yang akan dipesan dan tombol untuk mengirim pesanan. Gambar D merupakan tampilan daftar apabila tombol daftar pesanan ditekan, maka akan muncul

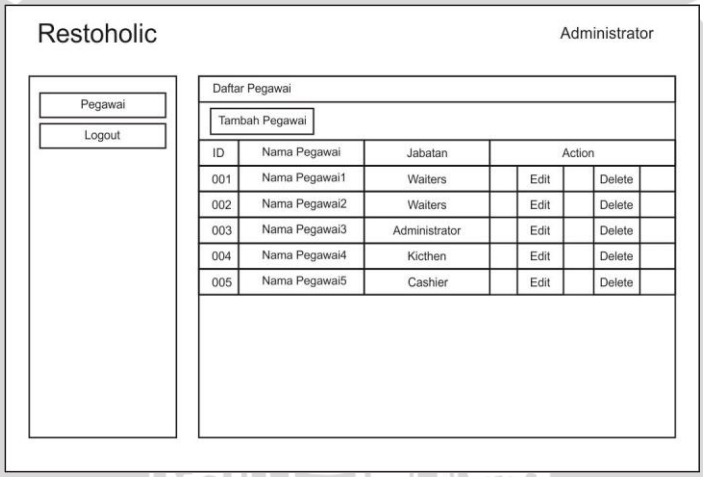
daftar yang berisi semua daftar pesanan. Terdapat dua jenis daftar pesanan yaitu *draft* pesanan yang merupakan pesanan yang belum bisa dilihat oleh dapur dan masih berupa konsep dan daftar pesanan yang sudah bisa dilihat dapur. Untuk mengubah dari *draft* pesanan agar bisa dilihat oleh dapur terdapat tombol kirim. Berikut keterangan mengenai antarmuka menu *waiters:*

Tabel 4.6 Keterangan Antarmuka Halaman Menu *Waiters*

Nama	Keterangan
Home	Tombol yang berfungsi untuk berpindah ke halaman home
Back	Tombol yang berfungsi untuk berpindah ke halaman sebelumnya
Tambah Makanan	Tombol yang berfungsi untuk menampilkan daftar menu makanan yang disediakan
Tambah Makanan	Tombol yang berfungsi untuk menampilkan daftar menu minuman yang disediakan
Tambah Lainnya	Tombol yang berfungsi untuk menampilkan daftar menu lainnya
Daftar pesanan	Tombol yang berfungsi untuk menampilkan daftar pesanan
Nama menu	Berisi keterangan nama menu yang akan dipilih
√ / X	Berisi keterangan status ketersediaan dari menu
Pop	Merupakan kotak dialog
Jumlah	Form yang diisi untuk menentukan jumlah yang akan dipesan

Tambahkan	Tombol yang berfungsi untuk menyimpan pesanan baru
Kirim	Tombol yang berfungsi untuk mengubah dari *draft* agar bisa dilihat oleh dapur
Clear	Tombol yang berfungsi untuk menghapus semu pesanan *draft*
J	Berisi keterangan jumlah pesanan
S	Berisi keterangan status pesanan

4.2.5.2 Antarmuka Admin



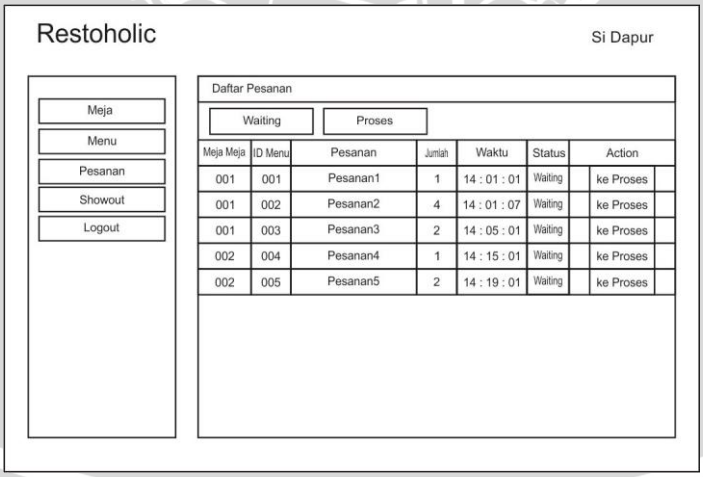
Gambar 4.21 Antarmuka Admin Halaman Daftar Pegawai

Pada gambar 4.21 merupakan rancangan antarmuka untuk halaman admin melihat daftar pegawai. Pada halaman ini terdapat judul aplikasi, pemakai yang sedang *login* yaitu administrator, tombol pegawai untuk melihat daftar pegawai, tombol logout, tombol tambah pegawai dan daftar pegawai. Pada daftar pegawai menampilkan id pegawai, nama pegawai, jabatan dan tombol untuk mengubah atau menghapus pegawai. Berikut keterangan mengenai antarmuka Daftar Pegawai:

Tabel 4.7 Keterangan Antarmuka Admin Halaman Daftar Pegawai

Nama	Keterangan
Restoholic	Merupakan judul dari aplikasi
Administrator	Nama pegawai yang sedang login
Tombol Pegawai	Untuk melihat daftar pegawai
Tombol Tambah Pegawai	Tombol yang berfungsi untuk menambahkan pegawai baru
Id	Berisi id_pegawai
Nama Pegawai	Berisi nama lengkap pegawai
Jabatan	Berisi Jabatan dari pegawai yaitu *waiters*, admin (*administrator*), dapur (*kitchen)*, kasir (*cashier*)
Action : Edit	Tombol untuk mengubah data pegawai
Action : Delete	Tombol untuk menghapus data pegawai

4.2.5.3 Antarmuka Dapur



Gambar 4.22 Antarmuka Dapur Halaman Daftar Pesanan

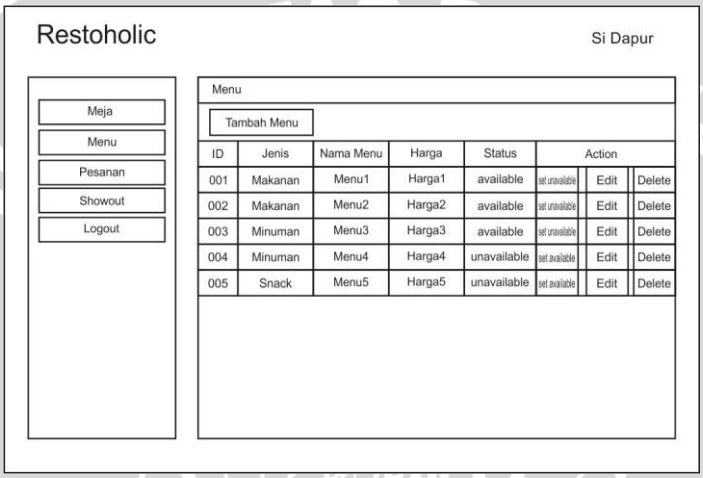
Pada gambar 4.22 merupakan rancangan antarmuka untuk halaman dapur melihat daftar pesanan. Pada halaman ini terdapat judul aplikasi, pemakai yang sedang *login* yaitu si dapur, tombol pesanan untuk melihat daftar

pesanan, tombol menu untuk menampilkan halaman daftar menu, tombol *logout*, tombol *waiting* untuk menampilkan pesanan yang berstatus *waiting*, tombol proses untuk menampilkan pesanan yang berstatus proses, dan daftar pesanan. Pada daftar pesanan menampilkan id pesanan, nama menu, jumlah pesanan, meja pemesan, waktu antri, dan tombol untuk mengubah status pesanan, apabila status pesanan *waiting* maka pada *action* berupa tombol ke proses dan apabila status pesanan proses pada action berupa tombol *done*. Berikut keterangan mengenai antarmuka Daftar Pesanan:

Tabel 4.8 Keterangan Antarmuka Dapur Halaman Daftar Pesanan

Nama	Keterangan
Restoholic	Merupakan judul dari aplikasi
Si Dapur	Nama pegawai yang sedang login
Tombol Pesanan	Untuk melihat daftar pesanan
Tombol Menu	Untuk melihat daftar menu
Tombol Meja	Untuk melihat daftar meja
Tombol *Showout*	Untuk menampilkan satu layar penuh berisi daftar pesanan, ditunjukkan untuk membuka tampilan koki
Tombol *Waiting*	Untuk melihat daftar pesanan berstatus *waiting*
Tombol Proses	Untuk melihat daftar pesanan berstatus proses
Id meja	Berisi id_meja pemesan
Id menu	Berisi id_menu makanan
Pesanan	Berisi nama menu yang dipesan
Jumlah	Berisi jumlah pesanan
Waktu	Berisi waktu *waiters* mengirimkan pesanan untuk diantrikan
Status	Menampilkan status pesanan

Meja	Berisi keterangan meja yang memesan
Action : ke Proses	Untuk mengubah status pesanan menjadi proses menandakan sedang proses pembuatan.
Action : *done*	Untuk mengubah status pesanan menjadi *done* menandakan pesanan sudah jadi dan siap diantar



Gambar 4.23 Antarmuka Dapur Halaman Daftar Menu

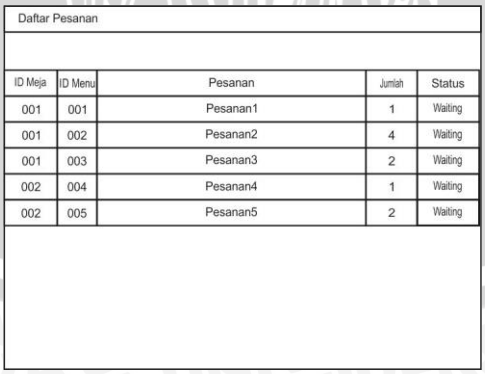
Pada gambar 4.23 merupakan rancangan antarmuka untuk halaman dapur melihat daftar menu. Pada halaman ini terdapat judul aplikasi, pemakai yang sedang *login* yaitu si dapur, tombol pesanan untuk melihat daftar pesanan, tombol menu untuk menampilkan halaman daftar menu, tombol *logout*, tombol tambah menu untuk menambah menu baru dan daftar menu. Pada daftar menu menampilkan id menu, nama menu, jenis menu dan tombol untuk mengubah atau menghapus menu. Berikut keterangan mengenai antarmuka Daftar Menu:

Tabel 4.9 Keterangan Antarmuka Dapur Halaman Daftar Menu

Nama	Keterangan
Restoholic	Merupakan judul dari aplikasi
Si Dapur	Nama pegawai yang sedang login

Tombol Pesanan	Untuk melihat daftar pesanan
Tombol Menu	Untuk melihat daftar menu
Tombol Meja	Untuk melihat daftar meja
Tombol *Showout*	Untuk menampilkan satu layar penuh berisi daftar pesanan, ditunjukkan untuk membuka tampilan koki
Tombol Pesanan	Untuk melihat daftar pesanan
Tombol Menu	Untuk melihat daftar menu
Id	Berisi id_pesanan
Jenis	Berisi jenis menu
Nama menu	Berisi nama menu
Harga	Berisi harga dari menu
Status	Berisi status ketersediaan menu
Action : Set Unavailable	Mengubah status pesanan dari tersedia ke tidak tersedia
Action : Set Available	Mengubah status pesanan dari tidak tersedia ke tersedia
Action : Edit	Tombol untuk mengubah data pegawai
Action : Delete	Tombol untuk menghapus data pegawai

4.2.5.4 Antarmuka Koki



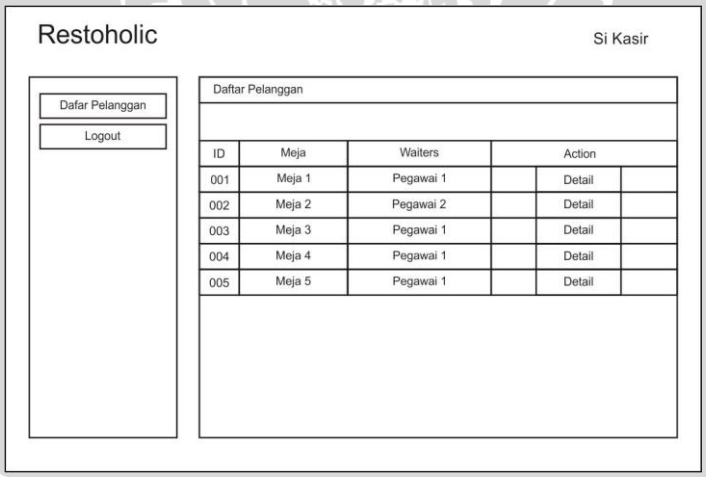
Gambar 4.24 Antarmuka Koki Halaman Daftar Pesanan

Pada gambar 4.24 merupakan rancangan antarmuka untuk halaman Koki melihat daftar pesanan. Pada halaman ini terdapat daftar pesanan yang antri dan harus dimasak oleh koki. Pada daftar pelanggan menampilkan id meja, id menu, nama menu yang dipesan, jumlah dan status pesanan. Berikut keterangan mengenai antarmuka Daftar Pesanan:

Tabel 4.10 Keterangan Antarmuka Koki Halaman Daftar Pesanan

Nama	Keterangan
Id Meja	Berisi id meja yang ditempati pelanggan pemesan
Id menu	Id menu yang dipesan
Pesanan	Nama menu yang dipesan
Jumlah	Jumlah memesan
Status	Berisi status pesanan

4.2.5.5 Antarmuka Kasir



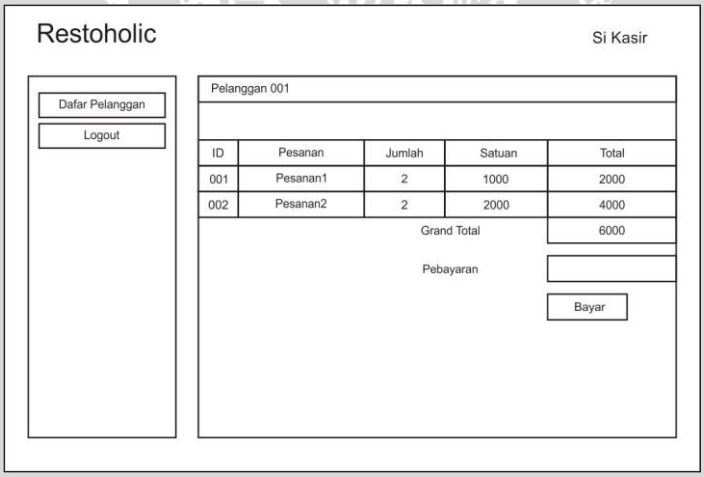
Gambar 4.25 Antarmuka Kasir Halaman Daftar Pelanggan

Pada gambar 4.25 merupakan rancangan antarmuka untuk halaman kasir melihat daftar pelanggan. Pada halaman ini terdapat judul aplikasi, pemakai yang sedang *login* yaitu si kasir, tombol daftar pelanggan untuk melihat daftar pelanggan dan daftar pelanggan. Pada daftar pelanggan

menampilkan id pelanggan, meja yang ditempati dan tombol untuk melihat data tagihan pelanggan. Berikut keterangan mengenai antarmuka Daftar Pelanggan:

Tabel 4.11 Keterangan Antarmuka Kasir Halaman Daftar Pelanggan

Nama	Keterangan
Restoholic	Merupakan judul dari aplikasi
Si Kasir	Nama pegawai yang sedang login
Daftar pelanggan	Untuk melihat daftar pelanggan yang aktif
Id	Berisi id_pelanggan
Meja	Berisi meja yang ditempati pelanggan
Waiters	Nama *waiters* yang melayani
Action : Detail	Untuk melihat tagihan pelanggan



Gambar 4.26 Antarmuka Kasir Halaman Tagihan

Pada gambar 4.26 merupakan rancangan antarmuka untuk halaman kasir melihat tagihan pelanggan. Pada halaman ini terdapat judul aplikasi, pemakai yang sedang *login* yaitu si kasir, tombol daftar pelanggan untuk melihat daftar pelanggan, id pelanggan dan daftar tagihan. Pada daftar pelanggan menampilkan id pesanan, nama menu yang dipesan, jumlah

pesanan, harga menu, total dari tiap menu yang dipesan, total semua yang harus dibayar, form pembayaran dan tombol bayar. Berikut keterangan mengenai antarmuka Daftar Pelanggan:

Tabel 4.12 Keterangan Antarmuka Kasir Halaman Tagihan

Nama	Keterangan
Restoholic	Merupakan judul dari aplikasi
Si Kasir	Nama pegawai yang sedang login
Daftar pelanggan	Untuk melihat daftar pelanggan yang aktif
Pelanggan 001	Berisi id pelanggan
Id	Berisi id_pesanan
Pesanan	Berisi nama menu yang dipesan
Jumlah	Berisi jumlah pesanan
Satuan	Berisi harga satuan
Total	Berisi jumlah yang harus dibayarkan tiap menunya
Grand total	Berisi jumlah yang harus dibayarkan
Pembayaran	Form yang diisikan uang pembayaran
Bayar	Tombol untuk mengubah status pelanggan menjari sudah bayar

BAB V
IMPLEMENTASI DAN PENGUJIAN

Bab ini membahas mengenai tahapan implementasi dan pengujian Sistem Informasi Pemesanan Restoran Berbasis Android dan *client-server* pada Jaringan Lokal berdasarkan dari hasil analisis kebutuhan dan perancangan aplikasi. Pembahasan terdiri dari implementasi basis data, implementasi *class*, dan implementasi antarmuka. Kemudian dilanjutkan dengan pengujian unit dan pengujian validasi pada aplikasi. Setelah pengujian selesai, akan dilanjutkan dengan proses analisis untuk mendapatkan kesimpulan dari hasil pengujian aplikasi.

5.1 Implementasi Basis Data

Pada Sistem Pemesanan *Web Base* menggunakan basis data sebagai tempat untuk menyimpan data, merupakan implementasi dari Perancangan Basis Data Gambar 4.18. Tabel 5.1 menjelaskan penerapan perancangan pada aplikasi.

Tabel 5.1 Implementasi Tabel Basis Data Status

Status				
No	Kolom	Key	Type Data	Deskripsi
1	Id_status	PK	Int	Kode unik tiap status menggunakan penomoran data (1,2,3,...)
2	Grup		Varchar(10)	Jenis status (Menu)
3	Content		Varchar(10)	Deskripsi status (Tersedia)

Pada Tabel 5.1 menjelaskan mengenai kolom-kolom yang diimplementasikan pada Tabel Status digunakan untuk menyimpan data status. Kolom id_status menjadi *primary key*, kolom *grup* untuk mengelompokkan jenis status, dan kolom *content* berisi deskripsi status.

Tabel 5.2 Implementasi Tabel Basis Data Menu

Menu				
No	**Kolom**	**Key**	**Type Data**	**Deskripsi**
1	Id_menu	PK	Varchar(10)	Kode unik dari tiap menu (MK01)
2	nama		Varchar(50)	Nama dari tiap menu (Nasi Goreng)
3	harga		Double	Harga dari tiap menu (8000)
4	Jenis		Varchar(10)	Jenis menu (Makanan)
5	Id_status	FK	Int	Status menu

Pada Tabel 5.2 menjelaskan mengenai kolom-kolom yang diimplementasikan pada Tabel Menu digunakan untuk menyimpan daftar Menu yang disediakan. Kolom id_menu menjadi *primary key*, kolom nama berisi tentang nama dari tiap menu, kolom harga berisi harga yang harus dibayar untuk memesan menu tersebut, kolom jenis berisi jenis dari menu, dan id_status merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Status digunakan untuk menyimpan status menu.

Tabel 5.3 Implementasi Tabel Basis Data Meja

Meja				
No	Kolom	Key	Type Data	Deskripsi
1	id_meja	PK	Varchar(1)	Kode unik tiap meja menggunakan penomoran data (1,2,3,...)
2	id_status		Int	Status meja

Pada Tabel 5.3 menjelaskan mengenai kolom-kolom yang diimplementasikan pada Tabel Meja digunakan untuk menyimpan daftar Meja yang disediakan. Kolom id_meja menjadi *primary key* sedangkan id_status merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Status digunakan untuk menyimpan status meja.

Tabel 5.4 Implementasi Tabel Basis Data Jabatan

Jabatan				
No	**Kolom**	**Key**	**Type Data**	**Deskripsi**
1	Id_jabatan	PK	Int	Kode unik tiap jabatan menggunakan penomoran data (1,2,3,...)
2	Content		Varchar(10)	Deskripsi mengenai jabatan (Kasir)

Pada Tabel 5.4 menjelaskan mengenai kolom-kolom yang diimplementasikan pada Tabel Jabatan digunakan untuk menyimpan daftar Jabatan. Kolom id_jabatan menjadi *primary key* sedangkan kolom content berisi keterangan mengenai jabatan tersebut.

Tabel 5.5 Implementasi Tabel Basis Data Pegawai

Pegawai				
No	**Kolom**	**Key**	**Type Data**	**Deskripsi**
1	Id_pegawai	PK	Varchar(10)	Kode unik tiap pegawai (subek)
2	Id_jabatan	FK	Int	Jabatan dari pegawai
3	Nama		Varchar(30)	Nama lengkap pegawai (Subekti)
4	Pass		Varchar(30)	Password untuk *login*

Pada Tabel 5.5 menjelaskan mengenai kolom-kolom yang diimplementasikan pada Tabel Pegawai digunakan untuk menyimpan daftar Pegawai. Kolom id_pegawai menjadi *primary key* sebagai kode unik tiap pegawai digunakan untuk *login,* kolom pass untuk menyimpan *password* digunakan untuk *login.* Kemudian kolom nama untuk menyimpan nama lengkap pegawai sedangkan id_jabatan merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Jabatan digunakan untuk menyimpan jabatan pegawai.

Tabel 5.6 Implementasi Tabel Basis Data Pelanggan

Pelanggan				
No	**Kolom**	**Key**	**Type Data**	**Deskripsi**
1	Id_pelanggan	PK	Int	Kode unik tiap pelanggan menggunakan penomoran data (1,2,3,...)
2	Id_meja	FK	Varchar(1)	id dari meja yang dipakai
3	Id_pegawai	FK	Varchar(10)	Id pegawai yang melayani
4	Id_status	FK	Int	Status pelanggan

Pada Tabel 5.6 menjelaskan mengenai kolom-kolom yang diimplementasikan pada Tabel Pelanggan digunakan untuk menyimpan daftar Pelanggan. Kolom id_pelanggan menjadi *primary key* sebagai kode unik tiap pelanggan, kolom id_meja merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Meja digunakan untuk menyimpan meja yang digunakan, kolom id_pegawai merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Pegawai digunakan untuk menyimpan pegawai yang melayani dan kolom id_status merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Status digunakan untuk menyimpan status yang pelanggan.

Tabel 5.7 Implementasi Tabel Basis Data Pesanan

Pesanan				
No	**Kolom**	**Key**	**Type Data**	**Deskripsi**
1	Id_pesanan	PK	Int	Kode unik tiap pesanan menggunakan penomoran data (1,2,3,...)
2	Id_pelanggan	FK	Int	id dari pelanggan yang memesan
3	Id_menu	FK	Varchar(10)	Id menu yang dipesan
4	jumlah		Int	Jumlah pesanan (2)
5	Id_status	FK	Int	Status pesanan

6	waktu		Datetime (DD:MM:YYYY H:M:S)	Berisi waktu memesan (20:05:2014 08:08:08)

Pada Tabel 5.7 menjelaskan mengenai kolom-kolom yang diimplementasikan pada Tabel Pesanan digunakan untuk menyimpan daftar Pelanggan. Kolom id_pesanan menjadi *primary key* sebagai kode unik tiap pesanan, kolom id_pelanggan merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Pelanggan digunakan untuk menyimpan pelanggan yang memesan, kolom id_menu merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Menu digunakan untuk menyimpan Menu yang dipesan, kolom id_status merupakan kolom yang menjadi *foreign key* yang menunjuk tabel Status digunakan untuk menyimpan status yang pesanan, kolom jumlah berisi jumlah pelanggan memesan untuk menu tersebut dan kolom waktu untuk menyimpan waktu pesanan tersebut dipesan berfungsi sebagai antrian untuk diproses.

5.2 Implementasi *Class*

Setiap *class* yang telah dirancang pada proses perancangan dalam Gambar 4.16 dan 4.17 direalisasikan pada sebuah *file*. Dalam Gambar 4.16 direalisasikan dengan ekstensi .java dan *file layout* dengan ekstensi .xml sebagai tampilan dari *class* tersebut. Pada tabel 5.8 menjelaskan mengenai pasangan antara *class* dengan *file* program dan *layout* yang digunakan untuk mengimplementasikan *use case*.

Tabel 5.8 Implementasi perancangan class Gambar 4.16

No	Package	Objek	Nama Class	Nama File Program	Nama File *Layout*
1	com.project.restoholic2	*View* dan *controller*	View_home, *controller*	DashboardActivity.java	dashboard.xml
2	com.project.restoholic2	*View* dan *controller*	View_data_pelanggan, *controller*	WaitersDoActivity.java	menu.xml

2	com.project.restoholic2.library	*Model*	*Model*	UserFunction.java	-

Pada Tabel 5.8 package merupakan nama folder tempat dimana file program disimpan di dalam aplikasi, terdapat dua package yaitu com.project.restoholic2 untuk menyimpan *class activity* dan com.project.restoholic2.library untuk menyimpan *class* pendukung. *Class activity* pada Android merupakan berperan sebagai *class main* pada pemrograman java pada umumnya. Objek merupakan keterangan mengenai posisi class dalam konsep MVC, posisi *class* dapat berupa *view*, *controller* atau *model*. Nama *class* diambil dari perancangan kelas, merupakan sumber dari implementasi kelas. Nama file merupakan nama dari file dimana *class* tersebut disimpan. Dan nama *file layout*, merupakan nama file dimana kode yang merepresentasikan tampilan disimpan.

Perancangan *class* dalam Gambar 4.17 direalisasikan pada Tabel 5.9 dengan mengimplementasikan *class* tersebut kedalam beberapa file.php.

Tabel 5.9 Implementasi perancangan class Gambar 4.17

No	Objek	Nama Class	Nama File Program
1	*Controller*	*Controller waiters*	c_pemesanan.php
2	*Controller*	*Controller dapur*	c_dapur.php
3	*Controller*	*Controller kasir*	c_kasir.php
4	*Controller*	*Controller admin*	c_dashboard.php
5	*View*	*View dapur*	_d_menu.php _d_pesanan.php _d_showout.php

5	*View*	*View kasir*	_k_pelanggan.php _k_bill.php
5	*View*	*View admin*	_get_pegawai.php
6	*Model*	*Model web service*	data.php

Pada tabel 5.8 terdapat objek *controller* yang bertugas mengatur data yang didapatkan dari *model* untuk ditampilkan di *view*. Sedangkan pada Tabel 5.9 terdapat beberapa *controller* untuk menampilkan mengatur tampilan dan data yang ditampilkan untuk masing-masing aktor yang didapat dari *model* yang terkoneksi dengan *database*. File UserFunction.java pada tabel 5.8 terhubung dengan file c_pemesanan.php pada tabel 5.9 dalam pendistribusian data. Pada class *view dapur* terdapat pemisahan penyimpanan kode program berdasarkan fungsinya, _d_menu.php digunakan untuk menyimpan fungsi yang ada kaitanya dengan menu seperti tampil_menu(), tambah_menu(), update_menu(), dan delete_menu(), untuk file _d_pesanan.php digunakan untuk menyimpan fungsi yang ada kaitanya dengan pesanan seperti tampil_daftar_pesanan() dan update_pesanan(), dan untuk file _d_showout.php digunakan untuk menyimpan fungsi tampil_daftar_pesanan() yang diakses oleh Koki. Pada class *view koki* juga terdapat pemisahan penyimpanan kode program berdasarkan fungsinya, _k_pelanggan.php digunakan untuk menyimpan fungsi tampil_daftar_pelanggan() dan pilih_pelanggan() dan file _k_bill.php untuk menyimpan fungsi tampil_tagihan() dan bayar().

5.3 Implementasi *code* program

Implementasi kode program untuk Gambar 4.16 *class diagram* aplikasi untuk *waiters* terdapat beberapa *class* yaitu *view_home, view_data_pelanggan* , *controler* dan *model*. Dari masing-masing *class* terdapat beberapa *method* yang akan diimplementasikan kedalam kode berikut :

Kode 5.2 Implementasi kode *method* start_aplikasi()

```
package com.project.restoholic2;

public class DashboardActivity extends Activity {

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
}
}
```

Pada Kode 5.1 merupakan kode implementasi dari *class view_home* dan *view_data_pelanggan* untuk *method* start_aplikasi() yang diimplementasikan dengan *method* onCreate() untuk memulai menjalankan aplikasi.

Kode 5.2 Implementasi kode *method* tampil_menu()

```
package com.project.restoholic2;

public class WaitersDoActivity extends Activity implements
OnItemClickListener {
setContentView(R.layout.menu);
}
```

Pada Kode 5.2 merupakan kode implementasi dari *class view_data_pelanggan* untuk *method* tampil_menu() dengan menggunakan *method* `setContentView(R.layout.menu)`. Ketika aplikasi berjalan, aplikasi memanggil *layout* menu.xml sebagai antarmuka untuk aplikasi yang di dalam *layout* tersebut terdapat daftar menu makanan yang disediakan oleh *controller*.

Kode 5.3 Implementasi kode *method* ambil_data_menu() pada *controller*

```
package com.project.restoholic2;

public class WaitersDoActivity extends Activity implements
OnItemClickListener {
```

```
JSONArray json = (JSONArray)
userFunctions.getMenu("makanan").getJSONArray("items");
}
```

Pada Kode 5.3 merupakan kode implementasi dari *class controller* untuk *method* ambil_data_menu() yang diimplementasikan dengan method `userFunctions.getMenu("makanan")` untuk mengambil data dari *model* yaitu userFunction, dalam format JSON.

Kode 5.4 Implementasi kode *method* ambil_data_menu() pada *model*

```
package com.project.restoholic2.library;

public JSONObject getMenu(String jenis){

List<NameValuePair> params = new ArrayList<NameValuePair>();

if (jenis.equals("makanan")){
      params.add(new BasicNameValuePair("jenis", "makanan"));}
else if (jenis.equals("minuman")) {
      params.add(new BasicNameValuePair("jenis", "minuman"));}
else if (jenis.equals("lain")) {
      params.add(new BasicNameValuePair("jenis", "lain"));}
JSONObject json =
jsonParser.getJSONFromUrl(URL+"get_menu/", params);

return json;
}
```

Pada Kode 5.4 merupakan kode implementasi dari *class model* untuk *method* ambil_data_menu() yang diimplementasikan dengan *method* getMenu() untuk mengambil data menu dari *web service*. Sebelum mengambil data, terlebih dahulu mempersiapkan parameter yaitu jenis menu yang hendak diambil dapat berupa makanan, minuman dan lain, parameter disimpan dalam format List. Fungsi `jsonParser.getJSONFromUrl()` merupakan implementasi dari metode REST yaitu GET, bertujuan untuk mendapatkan data dari URI.

Kode 5.5 Implementasi kode kirim_pesanan() pada *controller*

```
btnKirim.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
    try {
      JSONObject json = userFunctions.kirimPesanan
                    (getApplicationContext(),id_pelanggan);
    } catch (JSONException e) {}
    }
});
```

Pada Kode 5.5 merupakan kode implementasi dari *class controller* untuk *method* kirim_pesanan() yang diimplementasikan dengan method `userFunctions.kirimPesanan` untuk mengirim data ke *model* yaitu userFunction. Bertujuan untuk mengirim data pesanan *draf* dan masih disimpan pada perangkat Android ke *web service*.

Kode 5.6 Implementasi kode daftar_pesanan()

```
public JSONObject getPesanan(String id_pelanggan){

List<NameValuePair> params = new ArrayList<NameValuePair>();
params.add(new  BasicNameValuePair("id_pelanggan",  id_pelanggan
));
JSONObject  json  =  jsonParser.getJSONFromUrl(URL+"get_pesanan/",
params);

return json;
}
```

Pada Kode 5.6 merupakan kode implementasi dari *class model* untuk *method* daftar_pesanan() yang diimplementasikan dengan *method* getPesanan() untuk mengambil data pesanan dari *web service*. Sebelum mengambil data, terlebih dahulu mempersiapkan parameter yaitu id pelanggan dan disimpan dalam format List. Id pelanggan merupakan kode unik yang dimiliki pelanggan untuk mengidentifikasi pesanannya. Fungsi `jsonParser.getJSONFromUrl()`

merupakan implementasi dari metode REST yaitu GET, bertujuan untuk mendapatkan data dari URI.

Implementasi kode program untuk Gambar 4.17 *class diagram web server* terdapat beberapa *class* yaitu *view_dapur, view_kasir, view_admin, controler waiters, controler dapur, controler kasir, controler admin,* dan *model web service*. Dari masing-masing *class* terdapat beberapa *method* yang akan diimplementasikan kedalam kode berikut :

Kode 5.7 Implementasi kode ambil_data_menu()

```
public function get_menu(){
            $jenis = $this->input->post('jenis');
            $data = $this->Data->get_menu_byjenis($jenis);
            echo '{"items":'. json_encode($data).'}';
}
```

Pada Kode 5.7 merupakan kode implementasi dari *class controller waiters* untuk *method ambil_data_menu*() yang diimplementasikan dengan *method* `get_menu()` untuk mengambil daftar menu dari *model web service*. Fungsi `get_menu_byjenis($jenis)` bertujuan untuk mengambil data menu yang disediakan berdasarkan jenisnya dari *model web service*. Fungsi `echo '{"items":'. json_encode($data).'}';` digunakan untuk mengkonversi dari array ke json dan merupakan nilai kembalian apabila dipanggil oleh fungsi `getMenu()` pada Kode 5.4.

Kode 5.8 Implementasi kode kirim_pesanan()

```
public function kirim_pesanan(){

$var['id_menu'] =  $this->input->post('id_menu');
$var['jumlah'] =  $this->input->post('jumlah');
$var['id_pelanggan'] =  $this->input->post('id_pelanggan');
$data = $this->Data->add_pesanan($var);
echo '{"items":'. json_encode($data).'}';
}
```

Pada Kode 5.8 merupakan kode implementasi dari *class controller waiters* untuk *method kirim_pesanan*() yang diimplementasikan dengan *method* `kirim_pesanan()` untuk menerima kiriman pesanan dari fungsi `kirim_pesanan()` pada Kode 5.5 dan melanjutkan ke *model web service* pada fungsi `add_pesanan($var).`

Kode 5.9 Implementasi kode ambil_data_pesanan()

```
public function get_pesanan(){
$data    =    $this->Data->get_pesanan_bypelanggan($this->input->
post('id_pelanggan'));
echo '{"items":'. json_encode($data).'}';
}
```

Pada Kode 5.9 merupakan kode implementasi dari *class controller waiters* untuk *method ambil_data_pesanan*() yang diimplementasikan dengan *method* `get_pesanan()` untuk mengambil daftar pesanan dari *model web service*. Fungsi `get_pesanan_ bypelanggan()` untuk mengambil data pesanan berdasarkan id pelanggan tertentu dari *model web service*. Fungsi `echo '{"items":'. json_encode($data).'}';` digunakan untuk mengkonversi dari array ke json dan merupakan nilai kembalian apabila dipanggil oleh fungsi `getPesanan()` pada Kode 5.6.

Kode 5.10 Implementasi kode update_pesanan()

```
function action($action = null, $id_pesanan = null){
if        ('KITCHEN'        !=        strtoupper($this->session->
                                userdata('jabatan')))
    redirect(site_url('c_dasboard'));

if (!empty ($action)){
     $this->Data->update_pesanan_to($action,$id_pesanan);
}

redirect(site_url('c_dapur/dapur_pesanan'));
}
```

Pada Kode 5.10 merupakan kode implementasi dari *class controller dapur* untuk *method update_pesanan*() yang diimplementasikan dengan *method* `action()` untuk mengubah status pesanan. Fungsi `update_pesanan_to()` digunakan untuk memanggil fungsi pada *model web service* untuk mengubah status pesanan.

Kode 5.11 Implementasi kode ambil_data_menu()

```
function get_menu(){

 $query = $this->db->query("SELECT `id_menu`, `nama`, `harga`,
                  j.content as jenis , s.content as status
                  FROM `menu` join jenis j using (id_jenis)
                  join status s using (id_status)");
        if ($query->num_rows() > 0){
          $res = $query->result_array();
          return $res;
        }
return FALSE;

}
```

Pada Kode 5.11 merupakan kode implementasi dari *class model web service* untuk *method ambil_data_menu*() yang diimplementasikan dengan *method* `get_menu()` untuk mengambil data menu makanan dari *database* menggunakan *query*. `$this->db->query()` merupakan fungsi untuk mengeksekusi query yang ada didalamnya.

5.4 Implementasi Antarmuka

Terdapat lima aktor yang memiliki desain antar muka untuk diimplementasikan yaitu *waiters*, dapur, koki, kasir dan admin.

5.4.1 Implementasi Antarmuka *Waiters*

Gambar 5.1 Tampilan Antarmuka *Home* untuk *Waiters*

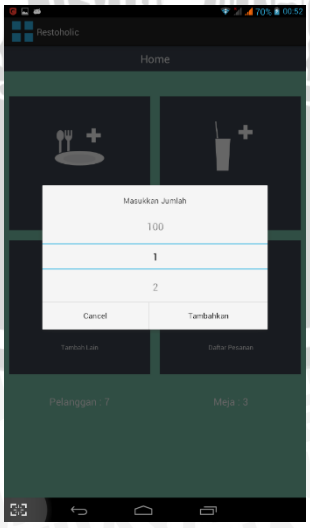
Gambar 5.1 merupakan implementasi dari Gambar 4.19, digunakan sebagai halaman awal untuk *waiters*. Berisi tombol untuk menambah pelanggan baru, memilih pelanggan yang sedang dilayani dan tampilan nama *waiters* waing sedang *login.*

Gambar 5.2 Tampilan Antarmuka Halaman Data Pelanggan untuk *Waiters*

Gambar 5.2 merupakan implementasi dari Gambar 4.20 A, digunakan sebagai halaman data pelanggan berisi tampilan id dan meja yang digunakan pelanggan, tombol untuk menampilkan menu yang disediakan dan tombol untuk melihat daftar pesanan.

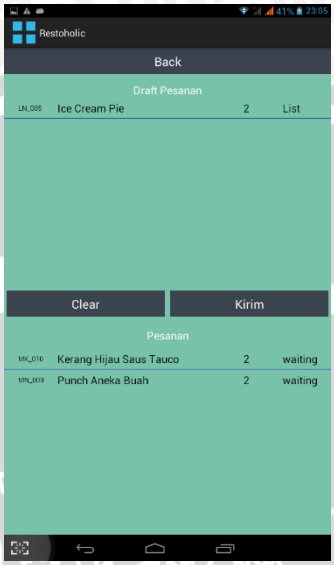
Gambar 5.3 Tampilan Antarmuka Daftar Menu untuk *Waiters*

Gambar 5.3 merupakan implementasi dari Gambar 4.20 B, digunakan sebagai halaman menu yang disediakan, berisi id dan nama dari daftar menu serta status ketersediaannya.



Gambar 5.4 Tampilan Antarmuka Dialog Pesanan untuk *Waiters*

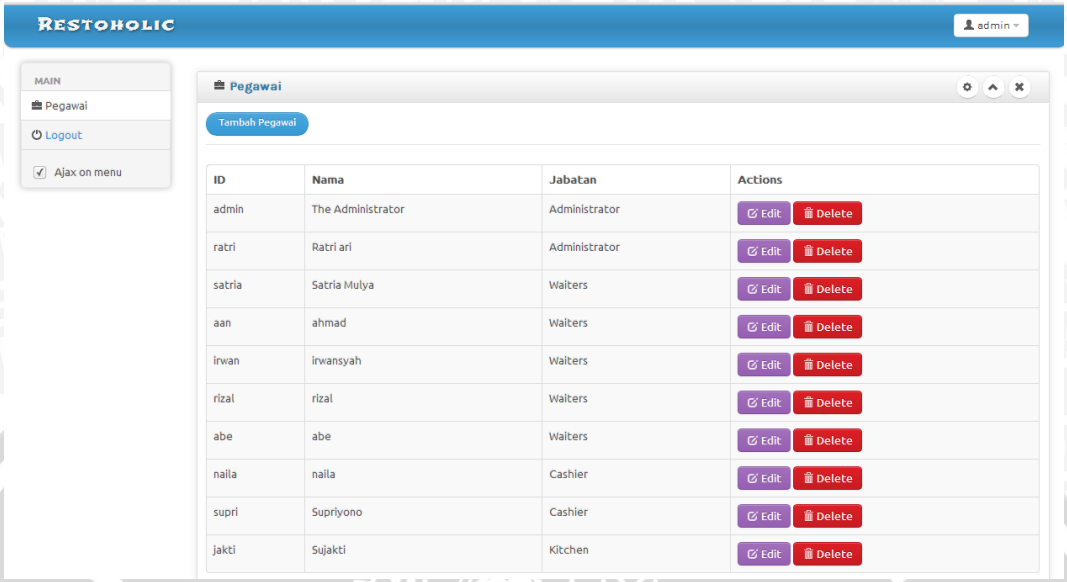
Gambar 5.4 merupakan implementasi dari Gambar 4.20 C, digunakan sebagai halaman pengisian jumlah pesanan. berisi form yang harus diisi jumlah pesanan dan tombol tambahkan.



Gambar 5.5 Tampilan Antarmuka Daftar Pesanan untuk *Waiters*

Gambar 5.5 merupakan implementasi dari Gambar 4.20 D, digunakan sebagai halaman daftar pesanan, berisi id dan nama menu yang dipesan, jumlah dan status pemesanannya. Terdapat pengelompokan antara pesanan yang masih *draft* berada di atas dan pesanan yang sudah dikirim ke *web service* berada di bawah.

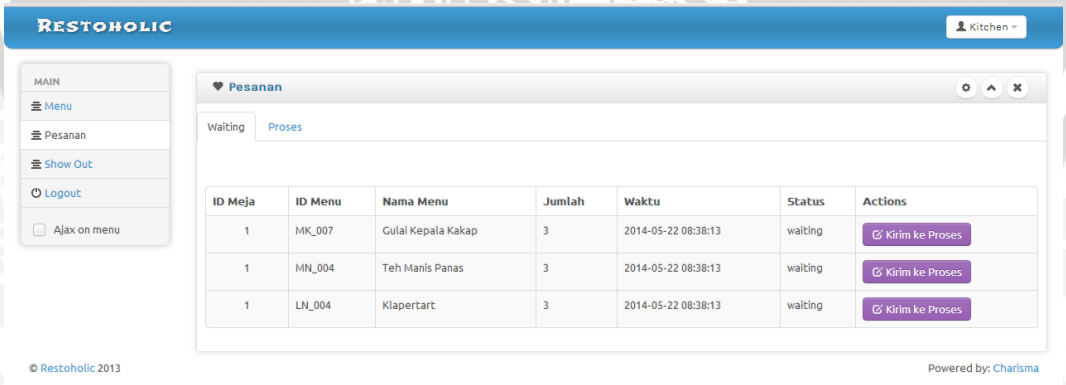
5.4.2 Implementasi Antarmuka Admin



Gambar 5.6 Tampilan Antarmuka Daftar Pegawai untuk *Waiters*

Gambar 5.6 merupakan implementasi dari Gambar 4.21, digunakan sebagai halaman admin melihat daftar pegawai. Daftar berisi id, nama dan jabatan dari pegawai serta tombol aksi untuk mengubah atau menghapus data pegawai.

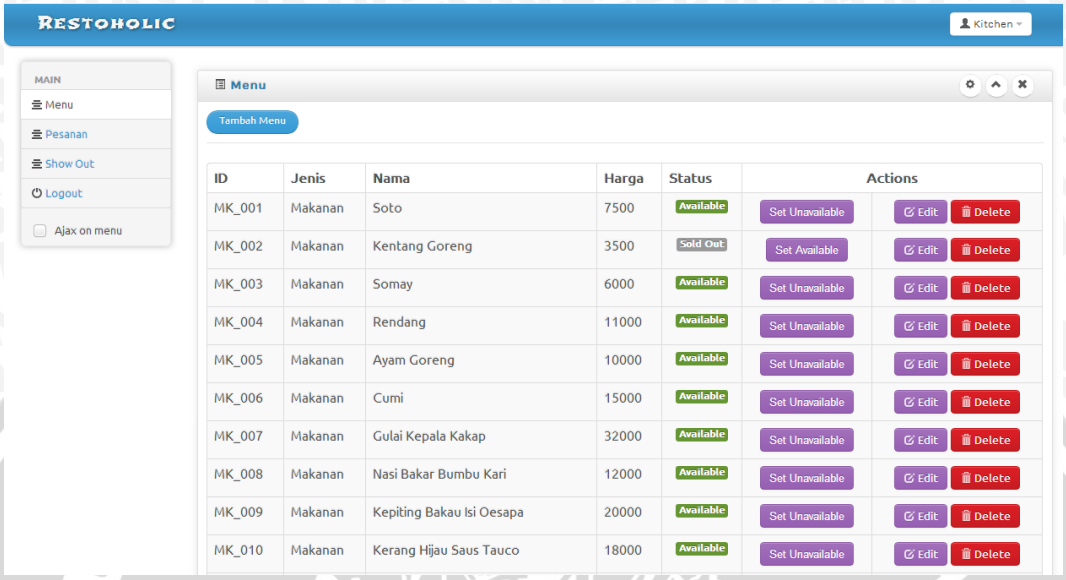
5.4.3 Implementasi Antarmuka Dapur



Gambar 5.7 Tampilan Antarmuka Daftar Pesanan untuk Dapur

Gambar 5.7 merupakan implementasi dari Gambar 4.22, digunakan sebagai halaman dapur melihat daftar pesanan yang telah dikirim oleh *waiters*. Daftar berisi id meja pemesan, id *waiters* yang melayani, nama menu, jumlah pemesanan waktu pengiriman, status pemesanan dan aksi untuk mengubah status

pemesanan. Terdapat pengelompokan pesanan berdasarkan status yaitu status *waiting* dan status *done* pada *tab* yang berbeda.



Gambar 5.8 Tampilan Antarmuka Daftar Menu untuk Dapur

Gambar 5.8 merupakan implementasi dari Gambar 4.23, digunakan sebagai halaman dapur untuk melihat daftar menu yang disediakan. Daftar berisi id, jenis, nama, harga dan status ketersediaanya, serta terdapat tombol aksi untuk mengubah status ketersediaan dan mengubah maupun menghapus data menu.

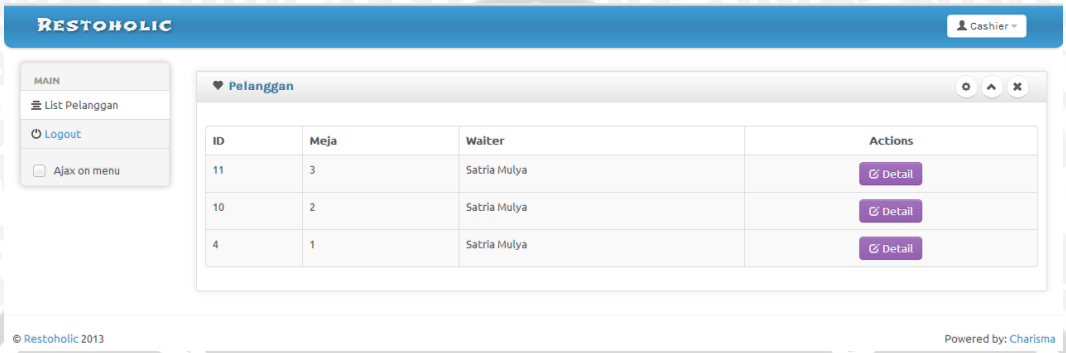
5.4.4 Implementasi Antarmuka Koki

Gambar 5.9 Tampilan Antarmuka Daftar Pesanan untuk Koki

Gambar 5.9 merupakan implementasi dari Gambar 4.24, digunakan sebagai halaman koki dalam melihat daftar pesanan. Daftar berisi id meja pemesan, id menu dan nama menu yang dipesan, jumlah yang dipesan serta status

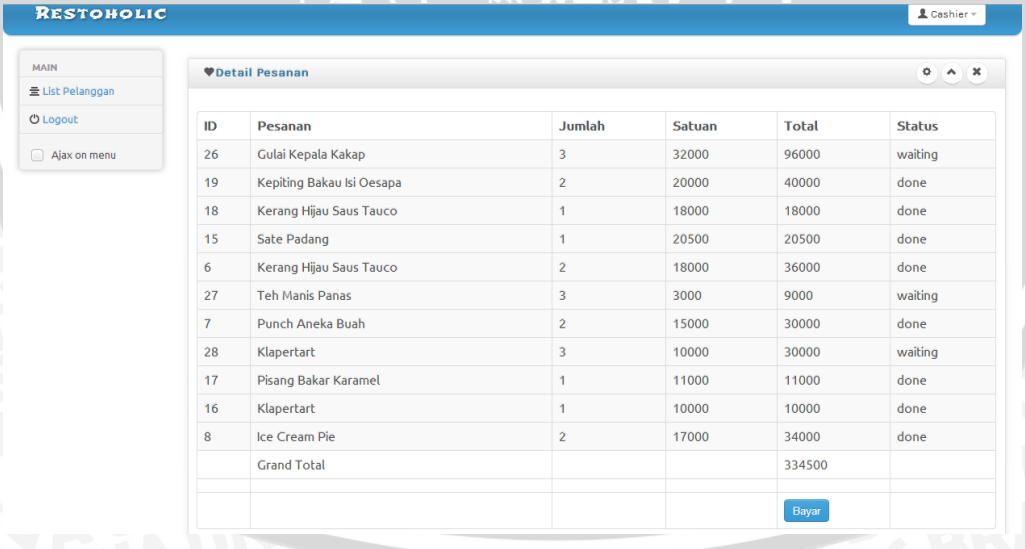
pemesanannya. Tidak terdapat tombol aksi apapuk karena koki hanya dapat melihat saja, dan yang berhak untuk memberikan aksi adalah dapur.

5.4.5 Implementasi Antarmuka Kasir



Gambar 5.10 Tampilan Antarmuka Daftar Pelanggan untuk Kasir

Gambar 5.10 merupakan implementasi dari Gambar 4.25, digunakan sebagai halaman kasir dalam melihat daftar pelanggan. Daftar berisi id pelanggan dan meja pelanggan, *waiters* yang melayani serta tombol aksi untuk melihat tagihan pelanggan.



Gambar 5.11 Tampilan Antarmuka Tagihan untuk Kasir

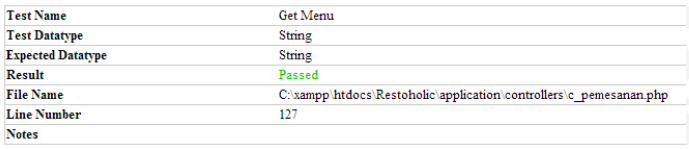
Gambar 5.11 merupakan implementasi dari Gambar 4.26, digunakan sebagai halaman kasir untuk melihat daftar tagihan. Daftar berisi id pesanan, nama menu yang dipesan beserta jumlahnya, harga satuan beserta total harga tiap pesanan

dan status pemesanannya, juga terdapat total tagihan yang harus dibayarkan untuk semua pesanan dan tombol bayar untuk mengubah status pelanggan menjadi tidak aktif dan status meja yang ditempati menjadi tersedia kembali.

5.5 Pengujian Unit

Pengujian ini untuk mengetahui apakah implementasi fungsi pada sistem dapat berjalan sesuai dengan yang diharapkan. Pengujian unit menggunakan pengujian *White-Box*, karena diperlukan konsentrasi terhadap kode program dan hasil keluaran tiap fungsinya. Strategi pengujiannya adalah menggunakan *library* unit_test yang terdapat pada Codeigniter.

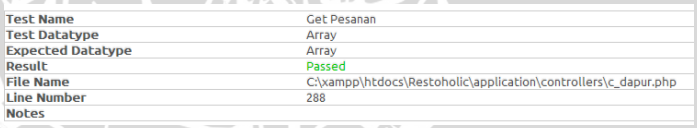
Tabel 5.10 Tabel kasus uji untuk pengujian unit *waiters* ambil_data_menu()

Kode Uji	KU01
Nama Kasus Uji	Kasus uji *waiters* fungsi ambil_data_menu()
Hasil yang diharapkan	{"items":[{"id_menu":"LN_001","nama":"Puding Pisang Emping Jagung","harga":"13000", "jenis": "Dessert","status":"tersedia"},{"id_menu":"LN_002" ,"nama":"Pisang Bakar Karamel","harga":"11000", "jenis":"Dessert","status":"tersedia"},{"id_menu":"L N_003","nama":"Puding Apel Strawberry","harga": "13000","jenis":"Dessert","status":"habis"},{"id_men u":"LN_004","nama":"Klapertart","harga":"10000","j enis":"Dessert","status":"tersedia"},{"id_menu":"LN _005","nama":"Ice Cream Pie","harga":"17000", "jenis":"Dessert","status":"tersedia"}]}
Hasil Pengujian	

Test Name	Get Menu
Test Datatype	String
Expected Datatype	String
Result	Passed
File Name	C:\xampp\htdocs\Restoholic\application\controllers\c_pemesanan.php
Line Number	127
Notes	

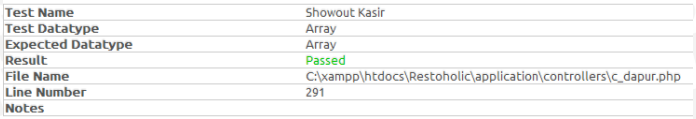
Status Validitas	Valid

Tabel 5.11 Tabel kasus uji untuk pengujian unit dapur ambil_data_pesanan ()

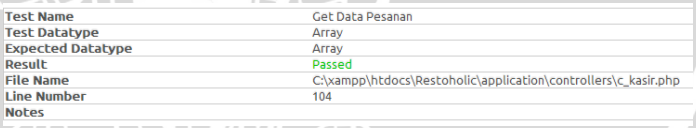
Kode Uji	KU02
Nama Kasus Uji	Kasus uji dapur fungsi ambil_data_ pesanan ()
Hasil yang diharapkan	Array ([alert] => [waiting] => Array ([0] => Array ([id_pesanan] => 31 [id_pelanggan] => 14 [id_meja] => 1 [id_menu] => MN_002 [nama] => Iced Coffee [harga] => 3000 [status] => waiting [jenis] => Minuman [jumlah] => 1 [waktu] => 2014-07-02 13:32:38)) [proses] =>)
Hasil Pengujian	
Status Validitas	Valid

Tabel 5.12 Tabel kasus uji untuk pengujian unit koki ambil_data_pesanan ()

Kode Uji	KU03
Nama Kasus Uji	Kasus uji koki fungsi ambil_data_ pesanan ()
Hasil yang diharapkan	Array ([alert] => [waiting] => Array ([0] => Array ([id_pesanan] => 31 [id_pelanggan] => 14 [id_meja] => 1 [id_menu] => MN_002 [nama] => Iced Coffee [harga] => 3000 [status] => waiting [jenis] =>

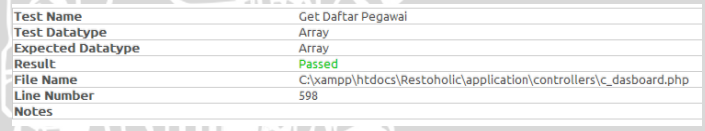
	Minuman [jumlah] => 1 [waktu] => 2014-07-02 13:32:38)))
Hasil Pengujian	
Status Validitas	Valid

Tabel 5.13 Tabel kasus uji untuk pengujian unit kasir ambil_data_pesanan ()

Kode Uji	KU04
Nama Kasus Uji	Kasus uji kasir fungsi ambil_data_ pesanan ()
Hasil yang diharapkan	Array ([alert] => [data] => Array ([0] => Array ([id_pesanan] => 31 [id_pelanggan] => 14 [id_menu] => MN_002 [nama] => Iced Coffee [harga] => 3000 [status] => waiting [jenis] => Minuman [jumlah] => 1)) [id_pelanggan] => 14)
Hasil Pengujian	
Status Validitas	Valid

Tabel 5.14 Tabel kasus uji untuk pengujian unit admin ambil_daftar_pegawai()

Kode Uji	KU05
Nama Kasus Uji	Kasus uji admin fungsi ambil_daftar_pegawai()

Hasil yang diharapkan	Array ([alert] => [data] => Array ([0] => Array ([id_pegawai] => admin [nama] => The Administrator [jabatan] => Administrator) [1] => Array ([id_pegawai] => ratri [nama] => Ratri ari [jabatan] => Administrator) [2] => Array ([id_pegawai] => satria [nama] => Satria Mulya [jabatan] => Waiters) [3] => Array ([id_pegawai] => aan [nama] => ahmad [jabatan] => Waiters) [4] => Array ([id_pegawai] => irwan [nama] => irwansyah [jabatan] => Waiters) [5] => Array ([id_pegawai] => rizal [nama] => rizal [jabatan] => Waiters) [6] => Array ([id_pegawai] => abe [nama] => abe [jabatan] => Waiters) [7] => Array ([id_pegawai] => naila [nama] => naila [jabatan] => Cashier) [8] => Array ([id_pegawai] => supri [nama] => Supriyono [jabatan] => Cashier) [9] => Array ([id_pegawai] => jakti [nama] => Sujakti [jabatan] => Kitchen)))
Hasil Pengujian	
Status Validitas	Valid

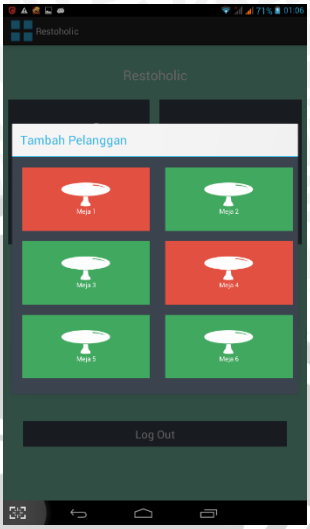
5.6 Pengujian Fungsional

Pengujian ini untuk mengetahui apakah sistem yang dibangun sudah benar sesuai dengan perancangan. Pengujian validasi menggunakan metode pengujian *Black-Box*, karena tidak diperlukan konsentrasi terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan kenyamanan antara kinerja sistem dengan daftar kebutuhan. Pada skripsi ini dilakukan pengujian validasi terhadap aplikasi.

5.5.1 Kasus Uji *Waiters*

Tabel 5.15 Tabel kasus uji untuk pengujian validasi menampilkan meja yang tersedia untuk *waiters*

Kode Uji	KW01
Nama Kasus Uji	Kasus uji *waiters* menampilkan meja yang tersedia untuk *waiters*
Objek Uji	Kebutuhan Fungsional (FW01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional untuk menampilkan meja yang tersedia
Data	{"items":[{"id_meja":"2","status":"tersedia"},{"id_meja":"3","status":"tersedia"},{"id_meja":"5","status":"tersedia"},{"id_meja":"6","status":"tersedia"}]}
Prosedur Uji	1. *Waiters* menjalankan aplikasi restoran dengan menekan *icon* aplikasi restoran pada perangkat bergerak. 2. *Waiters* menekan tombol tambah pelanggan
Hasil yang Diharapkan	Aplikasi dapat menampilkan meja yang tersedia yaitu meja 2, meja 3, meja 5 dan meja 6 dengan warna hijau dan meja yang tidak tersedia yaitu meja 1 dan meja 4 dengan warna merah.

Hasil uji	 Aplikasi menampilkan meja meja 2, meja 3, meja 5 dan meja 6 dengan warna hijau sebagai tanda bahwa tersedia dan meja 1 dan meja 4 dengan warna merah sebagai tanda tidak tersedia.
Status Validitas	Valid

5.5.2 Hasil Uji *Waiters*

Tabel 5.16 Hasil pengujian fungsional *waiters*

No.	Kode Uji	Nama Kasus Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status Validitas
1	KW01	Kasus uji Sistem menampilkan meja yang tersedia untuk *waiters*	Aplikasi dapat menampilkan meja yang tersedia yaitu meja 2, meja 3, meja 5 dan meja 6 dengan warna hijau dan meja yang tidak	Aplikasi menampilkan meja meja 2, meja 3, meja 5 dan meja 6 dengan warna hijau sebagai tanda bahwa tersedia dan meja 1 dan meja 4	Valid

			tersedia yaitu meja 1 dan meja 4 dengan warna merah.	dengan warna merah sebagai tanda tidak tersedia.	
2	KW02	Kasus *Waiters* dapat menambahkan pelanggan baru	Aplikasi dapat menambahkan pelanggan baru sesuai meja yang dimaksud yaitu meja 3 dan melanjutkan ke halaman data pelanggan.	Aplikasi dapat menambahkan pelanggan baru sesuai meja yang dimaksud yaitu meja 3 dan melanjutkan ke halaman data pelanggan.	Valid
3	KW03	Kasus uji *Waiters* dapat memilih pelanggan	Aplikasi dapat menampilkan daftar meja sesuai yang digunakan oleh pelanggan aktif yaitu meja 1,meja 3 dan meja 4 dengan warna hijau dan meja yang tidak aktif dengan warna abu-abu yaitu meja 2,meja 5 dan meja 6. Kemudian dapat melanjutkan ke halaman data pelanggan yang	Aplikasi dapat menampilkan daftar meja sesuai yang digunakan oleh pelanggan aktif yaitu meja 1,meja 3 dan meja 4 dengan warna hijau dan meja yang tidak aktif dengan warna abu-abu yaitu meja 2,meja 5 dan meja 6. Kemudian dapat melanjutkan ke halaman data pelanggan yang	Valid

			sesuai yaitu meja 3.	sesuai yaitu meja 3.	
4	KW04	Kasus uji Sistem dapat menampilkan menu untuk *waiters*	Aplikasi dapat menampilkan daftar menu yang dapat dipesan beserta status ketersediannya. Tanda centang untuk menu tersedia dan silang untuk menu tidak tersedia.	Aplikasi dapat menampilkan daftar menu yang dapat dipesan beserta status ketersediannya. Tanda centang untuk menu tersedia dan silang untuk menu tidak tersedia.	Valid
5	KW05	Kasus uji *Waiters* dapat menambah pesanan	Aplikasi dapat menambahkan pesanan baru yaitu Ice Cream Pie beserta jumlahnya 1 dan dapat menampilkan daftar pesanan *draft*.	Aplikasi dapat menambahkan pesanan baru yaitu Ice Cream Pie beserta jumlahnya 1 dan dapat menampilkan daftar pesanan *draft*.	Valid
6	KW06	Kasus uji Sistem dapat menampilkan status pesanan untuk *waiters*	Aplikasi dapat menampilkan daftar pesanan yaitu Ice Cream Pie dengan jumlah 1 dan berstatus draft.	Aplikasi dapat menampilkan daftar pesanan yaitu Ice Cream Pie dengan jumlah 1 dan berstatus draft.	Valid

7	KW07	Kasus uji *waiters* dapat mengubah dan menghapus pesanan	Aplikasi dapat mengubah jumlah pesanan dari 1 menjadi 2 dan menghapus pesanan pada draft	Aplikasi dapat mengubah jumlah pesanan dari 1 menjadi 2 dan menghapus pesanan pada draft	Valid
8	KW08	Kasus uji *Waiters* dapat mengirim pesanan	Aplikasi dapat mengirim daftar pesanan Ice Cream Pie dengan jumlah 2 dan berstatus *draft* ke *web service* dan mengubah statusnya menjadi *waiting*	Aplikasi dapat mengirim daftar pesanan Ice Cream Pie dengan jumlah 2 dan berstatus *draft* ke *web service* dan mengubah statusnya menjadi *waiting*	Valid

5.5.3 Kasus Uji Admin

Tabel 5.17 Tabel kasus uji untuk pengujian validasi Admin dapat menambah pegawai

Kode Uji	KA01
Nama Kasus Uji	Kasus uji Admin dapat menambah pegawai
Objek Uji	Kebutuhan Fungsional (FA01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional Admin dapat menambah pegawai
Data	Id : rifki

	Jabatan : Cashier Nama : Rifki Hasyim Pasword : rifki
Prosedur Uji	1. Admin menjalankan aplikasi restoran dengan mengakses alamat URL aplikasi pada web *browser*. 2. Admin menekan tombol tambah pegawai 3. Admin mengisi form pegawai baru 4. Admin menekan tombol submit 5. Sistem menampilkan daftar pegawai
Hasil yang Diharapkan	Aplikasi dapat menambahkan pegawai baru. Dengan id rifki, jabatan Cashier, nama Rifki Hasyim dan password rifki
Hasil Uji	Penampakan pada daftar pegawai :  Aplikasi dapat menambahkan pegawai baru. Dengan id rifki, jabatan Cashier, nama Rifki Hasyim dan password rifki
Status Validitas	Valid

5.5.4 Hasil Uji Admin

Tabel 5.18 Hasil pengujian fungsional admin

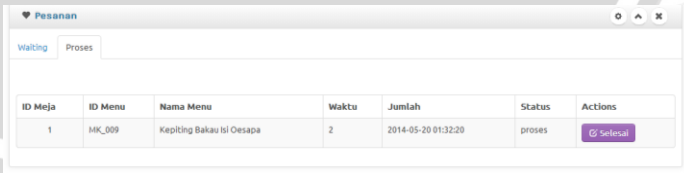
No.	Kode Uji	Nama Kasus Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status Validitas
1	KA01	Kasus uji Admin dapat menambah pegawai	Aplikasi dapat menambahkan pegawai baru. Dengan id rifki, jabatan Cashier,	Aplikasi dapat menambahkan pegawai baru. Dengan id rifki, jabatan Cashier,	Valid

			nama Rifki Hasyim dan password rifki	nama Rifki Hasyim dan password rifki	
2	KA02	Kasus uji Admin dapat mengubah, dan menghapus pegawai	Aplikasi dapat mengubah data pegawai dengan id rifki dari nama Rifki Hasyim menjadi Rifki Hasyim Ashari dan menghapus pegawai tersebut.	Aplikasi dapat mengubah data pegawai dengan id rifki dari nama Rifki Hasyim menjadi Rifki Hasyim Ashari dan menghapus pegawai tersebut.	Valid

5.5.5 Kasus Uji Dapur

Tabel 5.19 Tabel kasus uji untuk pengujian validasi Dapur dapat melihat daftar pesanan

Kode Uji	KD01
Nama Kasus Uji	Kasus uji Dapur dapat melihat daftar pesanan
Objek Uji	Kebutuhan Fungsional (FD01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional Dapur dapat melihat daftar pesanan
Data	`$data['waiting'] =( ( [0] => Array ( [id_pesanan] => 22 [id_pelanggan] => 8 [id_meja] => 3 [id_menu] => LN_005 [nama] => Ice Cream Pie [harga] => 17000 [status] => waiting [jenis] => Dessert [jumlah] => 2`

	[detail] => [waktu] => 2014-05-22 01:43:22)) $data['proses'] =([0] => Array ([id_pesanan] => 19 [id_pelanggan] => 4 [id_meja] => 1 [id_menu] => MK_009 [nama] => Kepiting Bakau Isi Oesapa [harga] => 20000 [status] => proses [jenis] => Makanan [jumlah] => 2 [detail] => [waktu] => 2014-05-20 01:32:20))
Prosedur Uji	1. Dapur menjalankan aplikasi restoran dengan mengakses alamat URL aplikasi pada web *browser*. 2. Dapur menekan tombol Pesanan 3. Sistem menampilkan daftar pesanan
Hasil yang Diharapkan	Aplikasi menampilkan pesanan yang telah dikirim *waiters* yang berstatus *waiting* yaitu Ice Cream Pie dan proses yaitu Kepiting Bakau Isi Oesapa. Dan dikelompokkan pada tab yang berbeda.
Hasil Uji	Penampakan pesanan dengan status waiting :  Penampakan pesanan dengan status proses :  Aplikasi menampilkan pesanan yang telah dikirim *waiters* yang berstatus *waiting* yaitu Ice Cream Pie

	dan proses yaitu Kepiting Bakau Isi Oesapa. Dan dikelompokkan pada tab yang berbeda.
Status Validitas	Valid

5.5.6 Hasil Uji Dapur

Tabel 5.20 Hasil pengujian fungsional dapur

No.	Kode Uji	Nama Kasus Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status Validitas
1	KD01	Kasus uji Dapur dapat melihat daftar pesanan	Aplikasi menampilkan pesanan yang telah dikirim *waiters* yang berstatus *waiting* yaitu Ice Cream Pie dan proses yaitu Kepiting Bakau Isi Oesapa. Dan dikelompokkan pada tab yang berbeda.	Aplikasi menampilkan pesanan yang telah dikirim *waiters* yang berstatus *waiting* yaitu Ice Cream Pie dan proses yaitu Kepiting Bakau Isi Oesapa. Dan dikelompokkan pada tab yang berbeda.	Valid
2	KD02	Kasus uji Dapur dapat mengubah status pesanan	Aplikasi dapat mengubah status pesanan Ice Cream Pie dari waiting menjadi proses dan menampilkannya.	Aplikasi dapat mengubah status pesanan Ice Cream Pie dari waiting menjadi proses dan menampilkannya	Valid

3	KD03	Kasus uji Dapur dapat melihat daftar menu	Aplikasi menampilkan daftar menu yang telah disimpan	Aplikasi menampilkan daftar menu yang telah disimpan	Valid
4	KD04	Kasus uji Dapur dapat menambah daftar menu	Aplikasi dapat menambahkan daftar menu baru dengan id MK_88, jenis makanan, nama Ayam Penyet, harga 10000 dan status tersedia kemudian menampilkannya.	Aplikasi dapat menambahkan daftar menu baru dengan id MK_88, jenis makanan, nama Ayam Penyet, harga 10000 dan status tersedia kemudian menampilkannya.	Valid
5	KD05	Kasus uji Dapur dapat mengubah status ketersediaan menu	Aplikasi dapat mengubah status menu ayam penyet dari tersedia menjadi tidak tersedia	Aplikasi dapat mengubah status menu ayam penyet dari tersedia menjadi tidak tersedia	Valid
6	KD06	Kasus uji Dapur dapat mengubah, dan menghapus data menu	Aplikasi dapat mengubah data makanan dengan nama Ayam penyet dari harga 10000 menjadi 70000 dan menghapus data menu makanan tersebut.	Aplikasi dapat mengubah data makanan dengan nama Ayam penyet dari harga 10000 menjadi 70000 dan menghapus data menu makanan tersebut.	Valid

5.5.7 Kasus Uji Koki

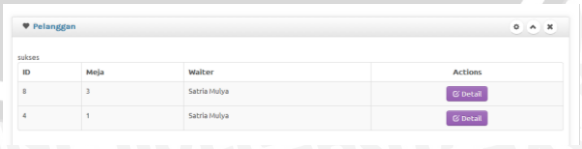
Tabel 5.21 Tabel kasus uji untuk pengujian validasi Koki dapat melihat daftar pesanan yang antri

Kode Uji	KI01
Nama Kasus Uji	Kasus uji Koki dapat melihat daftar pesanan yang antri
Objek Uji	Kebutuhan Fungsional (FI01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional Koki dapat melihat daftar pesanan yang antri
Data	`$data['waiting'] =( ( [0] => Array ( [id_pesanan] => 22 [id_pelanggan] => 8 [id_meja] => 3 [id_menu] => LN_005 [nama] => Ice Cream Pie [harga] => 17000 [status] => waiting [jenis] => Dessert [jumlah] => 2 [detail] => [waktu] => 2014-05-22 01:43:22 ) )`
Prosedur Uji	1. Dapur menekan tombol *showout* dan ditampilkan pada layar sehingga Koki bisa melihat daftar pesanan. 2. Aplikasi menampilkan daftar pesanan yang antri
Hasil yang Diharapkan	Aplikasi menampilkan daftar pesanan dengan status *waiting*, dipesan oleh pelanggan yang menempati meja 3 dengan menu Ice Cream Pie berjumlah 2
Hasil Uji	 Aplikasi menampilkan daftar pesanan dengan status *waiting*, dipesan oleh pelanggan yang menempati meja 3 dengan menu Ice Cream Pie berjumlah 2

Status Validitas	Valid

5.5.8 Kasus Uji Kasir

Tabel 5.22 Tabel kasus uji untuk pengujian validasi kasir dapat melihat daftar pelanggan

Kode Uji	KK01
Nama Kasus Uji	Kasus uji Kasir dapat melihat daftar pelanggan
Objek Uji	Kebutuhan Fungsional (FK01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional Kasir dapat melihat daftar pelanggan
Data	`$pelanggan = Array ( [0] => Array ( [id_pelanggan] => 8 [id_meja] => 3 [nama] => Satria Mulya [content] => aktif) [1] => Array ( [id_pelanggan] => 4 [id_meja] => 1 [nama] => Satria Mulya [content] => aktif) )`
Prosedur Uji	1. Kasir menekan tombol *List* Pelanggan 2. Aplikasi menampilkan daftar pelanggan yang aktif
Hasil yang Diharapkan	Aplikasi menampilkan daftar pelanggan yang aktif. Yaitu pelanggan yang menempati meja 1 dan meja 3.
Hasil Uji	
Status Validitas	Valid

5.5.9 Hasil Uji Kasir

Tabel 5.23 Hasil pengujian fungsional Kasir

No.	Kode Uji	Nama Kasus Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status Validitas
1	KK01	Kasus uji Kasir dapat melihat daftar pelanggan	Aplikasi menampilkan daftar pelanggan yang aktif. Yaitu pelanggan yang menempati meja 1 dan meja 3.	Aplikasi menampilkan daftar pelanggan yang aktif. Yaitu pelanggan yang menempati meja 1 dan meja 3.	Valid
2	KK02	Kasus uji Kasir dapat melihat daftar tagihan	Aplikasi menampilkan tagihan yang harus dibayar oleh pelanggan yang menempati meja 3. Yaitu Es Kelapa Campur dengan harga 15500 berjumlah 1, Es Jeruk dengan harga 4000 berjumlah 1 dan Ice Cream Pie dengan harga 17000 berjumlah 2. Serta menampilkan jumlah total yang harus dibayarkan	Aplikasi menampilkan tagihan yang harus dibayar oleh pelanggan yang menempati meja 3. Yaitu Es Kelapa Campur dengan harga 15500 berjumlah 1, Es Jeruk dengan harga 4000 berjumlah 1 dan Ice Cream Pie dengan harga 17000 berjumlah 2. Serta menampilkan jumlah total yang harus dibayarkan	Valid

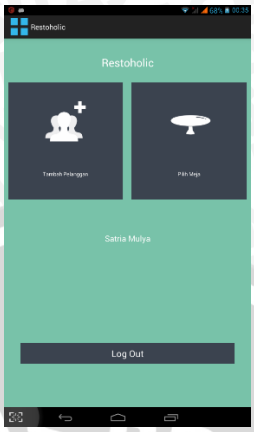
3	KK03	Kasus uji Kasir dapat mengubah status pelanggan	Aplikasi mengubah status pelanggan yang menempati meja 3 menjadi tidak aktif dan status meja 3 menjadi tersedia	Aplikasi mengubah status pelanggan yang menempati meja 3 menjadi tidak aktif dan status meja 3 menjadi tersedia	Valid

5.6 Pengujian Non Fungsional

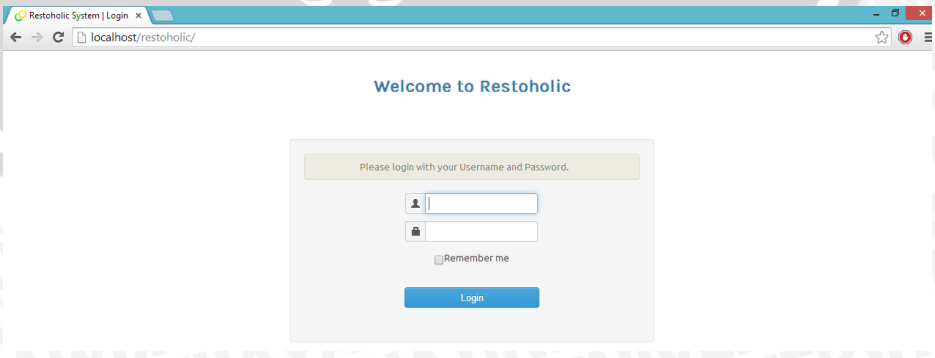
Pada pengujian non fungsional ini sistem akan diuji dengan blackbox yaitu pengujian dari sisi pengguna. NF01 adalah untuk pengujian *compatibility* hasil implementasi ke perangkat lunak. Dari hasil analisis kebutuhan aplikasi akan *compatible* pada *platform* android versi 4.0 untuk kebutuhan minimalnya. Pengujainnya dilakukan dengan cara memasang aplikasi ke perangkat dengan *platform* android versi 4.2.2, 4.1.1 dan 4.4. hasil dari pengujian dapat dilihat pada Tabel 5.19.

Tabel 5.24 Tabel Pengujian *Compatibility* Android

Platform Android	4.1.1
Merk	Treq Turbo
Dimensi Layar	1024×600 *pixels*, 7.0 *inches*
Hasil yang diharapkan	Berhasil dipasang pada perangkat berbasis android dan dapat berjalan dengan lancar

Hasil Uji	 Berhasil dipasang pada perangkat berbasis android dan dapat berjalan dengan lancar
Status Validitas	Valid

Tabel 5.25 Tabel Pengujian *Compatibility Web browser*

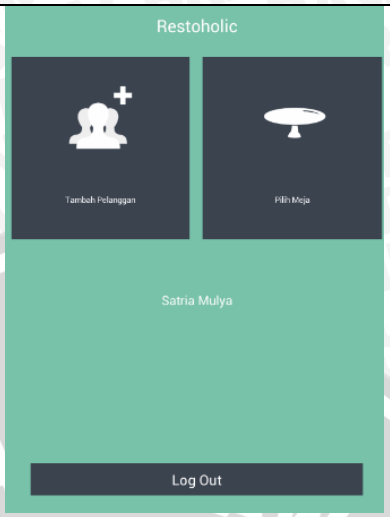
Aplikasi *web browser*	Google Chrome
Hasil yang diharapkan	Aplikasi berhasil berhasil diakses menggunakan url dari aplikasi, dan berjalan normal
Hasil Uji	

	Aplikasi berhasil berhasil diakses menggunakan url dari aplikasi, dan berjalan normal
Status Validitas	Valid

Untuk pengujian *Authorization* pada pengujian nonfungsional NF03 dengan cara menjalakan aplikasi dan login sesuai jabatan. Untuk pengujianya dapat dilihat pada Tabel 5.26.

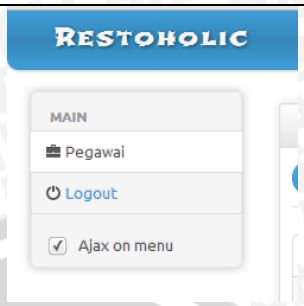
Tabel 5.26 Kasus uji untuk pengujian validasi *Authorization Waiters*

Kode Uji	T02
Nama Kasus Uji	Kasus Uji *Authorization Waiters*
Objek Uji	Kebutuhan Non Fungsional (NF03)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi menyediakan fitur sesual jabatan pegawai.
Data	*Username* : satria *Password* : 123
Prosedur Uji	Admin *login* dengan membuka aplikasi pada perangkat Android
Hasil yang Diharapkan	Aplikasi dapat mengidentifikasi *waiters* dengan id satria, password 123 lalu menampilkan fitur untuk *waiters*
Hasil Uji	Kondisi setelah login :

	
Status Validitas	Valid

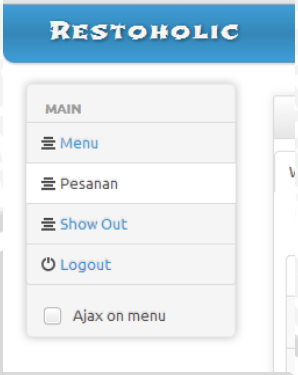
Tabel 5.27 Kasus uji untuk pengujian *Authorization* Admin

Kode Uji	T03
Nama Kasus Uji	Kasus Uji *Authorization* Admin
Objek Uji	Kebutuhan Non Fungsional (NF03)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi menyediakan fitur sesual jabatan pegawai.
Data	*Username* : admin *Password* : the admin
Prosedur Uji	Admin *login* dengan mengakses alamat URL aplikasi yang telah dibuat
Hasil yang Diharapkan	Aplikasi dapat mengidentifikasi admin dengan id admin, password the admin lalu menampilkan fitur untuk admin yaitu melihat daftar pegawai
Hasil Uji	Kondisi setelah login :

	
Status Validitas	Valid

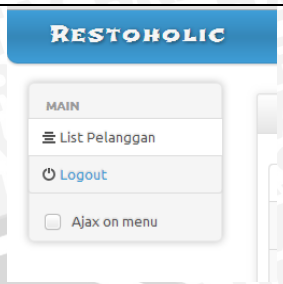
Tabel 5.28 Kasus uji untuk pengujian *Authorization* Dapur

Kode Uji	T04
Nama Kasus Uji	Kasus Uji *Authorization* Dapur
Objek Uji	Kebutuhan Non Fungsional (NF03)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi menyediakan fitur sesual jabatan pegawai.
Data	*Username* : jakti *Password* : sujak
Prosedur Uji	Dapur *login* dengan mengakses alamat URL aplikasi yang telah dibuat
Hasil yang Diharapkan	Aplikasi dapat mengidentifikasi dapur dengan id jakti, password sujak lalu menampilkan fitur untuk dapur yaitu melihat daftar menu, daftar pesanan dan daftar pesanan untuk koki.
Hasil Uji	Kondisi setelah login :

	
Status Validitas	Valid

Tabel 5.29 Kasus uji untuk pengujian *Authorization Kasir*

Kode Uji	T05
Nama Kasus Uji	Kasus Uji *Authorization* Kasir
Objek Uji	Kebutuhan Non Fungsional (NF03)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi menyediakan fitur sesual jabatan pegawai.
Data	*Username* : naila *Password* : nail
Prosedur Uji	Kasir *login* dengan mengakses alamat URL aplikasi yang telah dibuat
Hasil yang Diharapkan	Aplikasi dapat mengidentifikasi Kasir dengan id naila, password nail lalu menampilkan fitur untuk kasir yaitu melihat daftar pelanggan
Hasil Uji	Kondisi setelah login :

	
Status Validitas	Valid

5.7 Analisa Pengujian

Proses analisis bertujuan untuk mendapatkan kesimpulan dari hasil pengujian sistem informasi pemesanan yang telah dilakukan. Proses analisis mengacu pada hasil pengujian yang didapatkan. Analisis dilakukan terhadap hasil pengujian disetiap tahap pengujian. Proses analisis yang dilakukan meliputi analisis hasil pengujian unit dan pengujian validasi.

Proses analisis terhadap hasil pengujian unit dilakukan dengan melihat kesesuaian hasil yang didapatkan pada kode dengan hasil yang diharapkan. Pada Tabel 5.10, Tabel 5.11, Tabel 5.12, Tabel 5.13 dan Tabel 5.14 merupakan hasil dari pengujian unit dan dapat disimpulkan bahwa hasil implementasi kode telah sesuai yang diharapkan.

Proses analisis terhadap hasil pengujian validasi dilakukan dengan melihat kesesuaian antara hasil kinerja sistem dengan daftar kebutuhan. Pada Tabel 5.16, Tabel 5.18, Tabel 5.20, Tabel 5.21 dan Tabel 5.23 merupakan hasil dari pengujian validasi dan dapat disimpulkan bahwa implementasi kebutuhan fungsional sistem informasi pemesanan telah memenuhi kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan.

Proses analisis terhadap hasil pengujian non-fungsional dilakukan dengan melihat hasil dari pengujian non-fungsional. Dari hasil pengujian compatibility pada Tabel 5.24 dapat disimpulkan bahwa aplikasi dapat dipasang pada perangkat berbasis Android. Proses analisis dilanjutkan pada pengujian *Authorization* yang ditunjukan pada Tabel 5.26, Tabel 5.27, Tabel 5.28 dan Tabel 5.29. Pengujian ini bertujuan untuk mengetahui bahwa sistem telah memberikan fitur yang tepat

kepada pengguna sesuai jabatannya, karena jabatan merupakan identifikasi dari tiap aktor. Hasil pada pengujian ini membuktikan bahwa sistem telah memenuhi kebutuhan *Authorization* karena sistem telah memberikan fitur yang sesuai kepada penggunanya yaitu *waiters*, admin, dapur dan kasir.

BAB VI
PENUTUP

6.1 Kesimpulan

Dari tahapan-tahapan penulisan yang dilalui dapat disimpulkan bahwa dengan melakukan observasi dan wawancara dapat dihasilkan daftar kebutuhan yang akan dianalisa dan dimodelkan dalam diagram *usecase* dan diagram aktifitas, dilanjutkan dengan melakukan perancangan arsitektur, *sequence*, kelas, *database* dan antarmuka kemudian diimplementasikan pada kode, *database* dan antarmuka maka dihasilkan sistem informasi pemesanan restoran berbasis Android dan *client-server* yang mampu :

1. Berjalan pada perangkat *mobile* berbasis Android untuk *waiters* dan berbasis *web* untuk admin, dapur, koki dan kasir.
2. Menampilkan menu makanan atau minuman pada perangkat Android serta status ketersediaanya.
3. Menambah pesanan dan mengirimkannya dari aplikasi berbasis Android ke aplikasi berbasis *web* menggunakan fasilitas *web service.*
4. Menampilkan daftar pesanan beserta status pesanannya dan berubah apabila terdapat pembaruan.
5. Mengontrol antrian pesanan berdasarkan waktu dikirimkannya pesanan ke dapur dan merubah status pesanan, diakses oleh dapur menggunakan *web browser.*

Dari tahapan pengujian didapatkan bahwa sistem 100% valid terhadap kasus yang diujikan meliputi pengujian unit serta pengujian validasi kebutuhan fungsional dan non fungsional, oleh karena itu dapat disimpulkan bahwa sistem dapat memecahkan permasalahan yang dirumuskan.

6.2 Saran

Saran untuk pengembangan sistem informasi pemesanan restoran ini antara lain :

1. Dalam pengembangan selanjutnya diharapkan dapat memberikan fitur notifikasi dari *server* ke *client* ketika ada perubahan status pemesanan. Sehingga *waiters* dapat segera mengetahui perubahan yang terjadi.
2. Dalam pengembangan selanjutnya diharapkan dapat memberikan fitur cetak bill dan nota untuk daftar tagihan pelanggan.
3. Dalam pengembangan selanjutnya diharapkan dapat memberikan penambahan jumlah meja sesuai dengan keadaan yang sebenarnya pada restoran dan dipetakan sesuai posisinya.
4. Dalam pengembangan selanjutnya diharapkan dapat menambahakan fitur *reporting* penjualan pada sisi kasir.

DAFTAR PUSTAKA

[ADS-14]. Sasongko, Adi. 2014. "Aplikasi Pemesanan Makanan Dan Minuman Pada Rumah Makan"

[AGK-12]. Kusumawaty, Anggia. 2012. "Aplikasi Pemesanan Makanan Pada Restoran Berbasis Android dan Php Menggunakan Protokol Json".

[ASM-06]. Turan, Ali , Bogosyan, Seta dan Gokasan, Metin. 2006. "*Development of a Client-Server Communication Method for Matlab/Simulink Based Remote Robotics Experiments*"

[BOD-08]. Bornstein, Dan (2008-05-29). "*Presentation of Dalvik VM Internals*"

[BUPDAR] http://www.parekraf.go.id/asp/detil.asp?c=114&id=1429, diakses pada 21 Oktober 2013

[CNP-13]. Previana, Cantika Nur.2013."Aplikasi *Sms Gateway* Pada Sistem Antrian Pasien Di Poli Rumah Sakit Lavalette"

[CSM-12]. Simangunsong, Chandra. 2012. "Implementasi Model MVC pada Sistem Absensi SMAN 4 Batam melalui penerapan *Framework* CodeIgniter"

[EDH-12]. Sutanta, Edhy dan Mustofa, Khabib. 2012. "Kebutuhan *Web Service* Untuk Sinkronisasi Data Antar Sistem Informasi Dalam E-Gov Di Pemkab Bantul Yogyakarta".

[FCB-11]. Constantianus, Frederick dan Suteja, Bernard Renaldy.2011."Analisa dan Desain Sistem Bimbingan Tugas Akhir Berbasis *Web* dengan Studi Kasus Fakultas Teknologi Informasi"

[GGS-09]. Sukmana, Gugun. 2009. "Sistem Informasi Penjualan dan Pembelian Bahan Baku Bangunan Di. Pd. Pembangunan Raya Berbasis *Client Server*".

[HBA-13]. Saputra, Hariyadi Bagus dan Basten, Army. 2013. "Sistem Jaringan Komputer".

[IBY-13]. Adnyana, Ida Bagus Made Yogie. 2013. "Rancang Bangun Sistem Informasi Geografis Persebaran Lokasi Obyek Pariwisata Berbasis *Web* Dan *Mobile* Android"

[NSH-12] Safaat, Nazruddin .2012. “Android Pemograman Aplikasi *Mobile Smartphone* dan *Tablet* Berbasis Android” . Informatika. Bandung

[OCN-09]. Nurrahman,Ocky.2009. "Membangun Aplikasi Sistem Informasi Data pegawai PT. Indonesia Mastite Gasket dengan Intranet".

[PHP-08] Powers, David (2008).“PHP Object Oriented Solution”. Springer-Verlag, New York.

[RNH-10]. Haryanto, Ryan Nur. 2010. “Analisis Pengaruh Harga, Produk, Kebersihan, dan Kualitas Layanan Terhadap Kepuasan Pelanggan”.

[ROS-11]. Rumbaugh, James, Jacobson, Ivar, Booch, Grady. 199, “*The Unified Modeling Language Reference Manual*”, Addison Wesley, Canada.

[SSM-14]. Siahaan, Richard RF and Satoto, Kodrat Iman and Martono, Kurniawan Teguh. 2014. “Implementasi Sistem Informasi Geografis Daerah Pariwisata Kota Semarang Berbasis Android Dengan *Global Positioning System* (Gps)”. Undergraduate thesis, Diponegoro University.

[TIME] http://techland.time.com/2013/04/16/ios-vs-android/ , diakses pada 21 Oktober 2013

LAMPIRAN 1

Tabel Skenario *Use Case* Melihat Meja Tersedia

Nomor *Use Case*	FW01
Nama *Use Case*	Melihat meja tersedia
Tujuan	Menampilkan meja yang tersedia yang dapat digunakan pelanggan baru
Prakondisi	Aplikasi sudah terbuka
Kondisi Akhir	Meja yang tersedia ditampilkan
Aktor	*Waiters*
Alur Utama	4. *Waiters* menjalankan aplikasi restoran dengan menekan *icon* aplikasi restoran pada perangkat bergerak. 5. *Waiters* menekan tombol tambah pelanggan

Tabel Skenario *Use Case* Menambahkan Pelanggan Baru

Nomor *Use Case*	FW02
Nama *Use Case*	Tambah Pelanggan
Tujuan	Menambahkan pelanggan baru
Prakondisi	Aplikasi sudah terbuka dan menampilkan daftar meja tersedia, serta terdapat meja yang bisa digunakan
Kondisi Akhir	Pelanggan baru ditambahkan
Aktor	*Waiters*
Alur Utama	1. *Waiters* menekan salah satu dari meja yang tersedia 2. Sistem menambahkan pelanggan baru dan menampilkan data pelanggan

Tabel Skenario *Use Case* Memilih Pelanggan

Nomor *Use Case*	FW03
Nama *Use Case*	Pilih pelanggan
Tujuan	Memilih pelanggan untuk ditampilkan data pelanggan
Prakondisi	Aplikasi sudah terbuka dan terdapat pelanggan yang sedang aktif
Kondisi Akhir	Menampilkan data pelanggan yang dipilih
Aktor	*Waiters*
Alur Utama	1. *Waiters* menekan tombol pilih pelanggan 2. *Waiters* menekan salah satu pelanggan yang aktif (berwarna hijau) 3. Sistem menampilkan data pelanggan

Tabel Skenario *Use Case* Menampilkan Menu

Nomor *Use Case*	FW04
Nama *Use Case*	Melihat menu makanan
Tujuan	Menampilkan daftar makanan yang disediakan
Prakondisi	Aplikasi sudah menampilkan data pelanggan tertentu
Kondisi Akhir	Aplikasi menampilkan daftar menu makanan yang disediakan
Aktor	*Waiters*
Alur Utama	1. *Waiters* menekan salah satu dari tombol tambah makanan, tambah minuman atau tambah lainnya. Kasus menggunakan tombol tambah laiinya. 2. Sistem menampilkan daftar menu yang dapat dipesan.

Tabel Skenario *Use Case* Menampilkan Status Pesanan

Nomor *Use Case*	FW06
Nama *Use Case*	Melihat daftar pesanan
Tujuan	Menampilkan daftar pesanan serta status pesanan
Prakondisi	Pelanggan sudah memesan menu makanan
Kondisi Akhir	Aplikasi menampilkan daftar pesanan serta status pesanan
Aktor	*Waiters*
Alur Utama	1. *Waiters* menekan tombol daftar pesanan. 2. Sistem menampilkan daftar pesanan

Tabel Skenario *Use Case* Mengubah Dan Menghapus Pesanan

Nomor *Use Case*	FW07
Nama *Use Case*	Edit pesanan, Delete Pesanan
Tujuan	Mengubah pesanan atau menghapus pesanan
Prakondisi	Pelanggan sudah memesan menu makanan
Kondisi Akhir	Pesanan berubah atau terhapus
Aktor	*Waiters*
Alur Utama	update : 1. *Waiters* menekan salah satu daftar pesanan. 2. *Waiters* mengisikan jumlah terbaru 3. *Waiters* menekan tombol ubah 4. Sistem menampilkan daftar pesanan delete : *1.* *Waiters* menekan salah satu daftar pesanan. *2.* *Waiters* menekan tobol hapus *3.* Sistem menampilkan daftar pesanan

Tabel Skenario *Use Case* Mengirim Pesanan

Nomor *Use Case*	FW08
Nama *Use Case*	Kirim pesanan
Tujuan	Mengirim pesanan ke *web service*
Prakondisi	Pelanggan sudah memesan menu makanan
Kondisi Akhir	Status pesanan berubah menjadi *waiting*
Aktor	*Waiters*
Alur Utama	1. *Waiters* menekan tombol daftar pesanan 2. *Waiters* menekan tombol kirim 3. Sistem menampilkan daftar pesanan

Tabel Skenario *Use Case* Menambah Pegawai

Nomor *Use Case*	FA01
Nama *Use Case*	Tambah pegawai
Tujuan	Menambahkan pegawai baru yang dapat mengakses sistem
Prakondisi	Admin telah membuka aplikasi
Kondisi Akhir	Terdapat pegawai baru yang ditambahkan
Aktor	Admin
Alur Utama	1. Admin menjalankan aplikasi restoran dengan mengakses alamat URL aplikasi pada web *browser*. 2. Admin menekan tombol tambah pegawai 3. Admin mengisi form pegawai baru 4. Admin menekan tombol submit 5. Sistem menampilkan daftar pegawai

Tabel Skenario *Use Case* Mengubah Dan Menghapus Pegawai

Nomor *Use Case*	FA02
Nama *Use Case*	Edit Pegawai, Delete Pegawai
Tujuan	Mengubah data pegawai atau menghapusnya
Prakondisi	Admin telah membuka aplikasi dan sudah ada pegawai yang terdaftar
Kondisi Akhir	Data pegawai berubah atau terhapus
Aktor	Admin
Alur Utama	update : *1.* Admin menekan tombol *edit* pada salah satu pegawai *2.* Admin mengubah isi form *3.* Admin menekan tombol submit *4.* Sistem menampilkan daftar pegawai delete: 1. Admin menekan tombol *delete* pada salah satu pegawai 2. Sistem menampilkan daftar pegawai

Tabel Skenario *Use Case* Melihat Daftar Pesanan

Nomor *Use Case*	FD01
Nama *Use Case*	Melihat daftar pesanan
Tujuan	Melihat daftar pesanan yang telah terkirim
Prakondisi	Dapur telah membuka aplikasi dan telah ada pesanan yang terkirim
Kondisi Akhir	Menampilkan daftar pesanan yang telah terkirim
Aktor	Dapur
Alur Utama	*4.* Dapur menjalankan aplikasi restoran dengan mengakses alamat URL aplikasi pada web *browser*. *5.* Dapur menekan tombol Pesanan *6.* Sistem menampilkan daftar pesanan

Tabel Skenario *Use Case* Melihat Daftar Menu

Nomor *Use Case*	FD03
Nama *Use Case*	Melihat menu makanan
Tujuan	Melihat daftar menu makanan
Prakondisi	Dapur telah membuka aplikasi dan telah ada menu yang terdaftar
Kondisi Akhir	Menampilkan daftar menu yang disediakan
Aktor	Dapur
Alur Utama	*1.* Dapur menekan tombol menu *2.* Sistem menampilkan daftar menu

Tabel Skenario *Use Case* Menambah Daftar Menu

Nomor *Use Case*	FD04
Nama *Use Case*	Tambah menu makanan
Tujuan	Menambahkan menu baru
Prakondisi	Dapur telah membuka aplikasi
Kondisi Akhir	Terdapat menu baru yang ditambahkan
Aktor	Dapur
Alur Utama	*1.* Dapur menekan tombol tambah menu *2.* Dapur mengisi form menu baru *3.* Dapur menekan tombol submit *4.* Sistem menampilkan daftar menu

Tabel Skenario *Use Case* Mengubah Status Ketersediaan Menu

Nomor *Use Case*	FD05
Nama *Use Case*	Update status menu makanan
Tujuan	Memperbarui status ketersediaan menu
Prakondisi	Dapur telah membuka aplikasi dan telah terdapat menu yang terdaftar
Kondisi Akhir	Status menu diperbarui
Aktor	Dapur
Alur Utama	*1.* Dapur menekan *set unavailable* atau *set available* pada salah satu daftar menu *2.* Sistem menampilkan daftar menu

Tabel Skenario *Use Case* Mengubah Menu Atau Menghapus Menu

Nomor *Use Case*	FD04
Nama *Use Case*	Edit menu makanan, delete menu makanan
Tujuan	Mengubah menu atau menghapus menu
Prakondisi	Dapur telah membuka aplikasi dan telah terdapat menu yang terdaftar
Kondisi Akhir	Data pesanan diperbarui atau dihapus
Aktor	Dapur
Alur Utama	update: *1.* Dapur menekan tombol *edit* pada salah satu daftar menu *2.* Dapur mengisi form data menu *3.* Dapur menekan tombol *submit* delete : 1. Sistem menampilkan daftar menu 2. Dapur menekan tombol delete pada salah satu daftar menu 3. Sistem menampilkan daftar menu

Tabel Skenario *Use Case* Melihat Daftar Pesanan Yang Antri

Nomor *Use Case*	FI01
Nama *Use Case*	Tampil daftar antrian
Tujuan	Manampilkan daftar pesanan yang antri
Prakondisi	Terdapat pesanan yang telah terkirim dan dapur telah membuka tampilan *showout* untuk koki
Kondisi Akhir	Aplikasi menampilkan daftar pesanan yang antri
Aktor	Koki
Alur Utama	1. Dapur menekan tombol *showout* dan ditampilkan pada layar sehingga Koki bisa melihat daftar pesanan. 2. Aplikasi menampilkan daftar pesanan yang antri

Tabel Skenario *Use Case* Melihat Daftar Pelanggan

Nomor *Use Case*	FK01
Nama *Use Case*	Melihat daftar pelanggan
Tujuan	Manampilkan daftar pelanggan yang aktif
Prakondisi	Kasir telah membuka aplikasi dan terdapat pelanggan yang masih aktif
Kondisi Akhir	Aplikasi menampilkan daftar pelanggan yang aktif
Aktor	kasir
Alur Utama	1. Kasir menekan tombol *List* Pelanggan 2. Aplikasi menampilkan daftar pelanggan yang aktif

Tabel Skenario *Use Case* Melihat Daftar Tagihan

Nomor *Use Case*	FK02
Nama *Use Case*	Melihat tagihan
Tujuan	Melihat daftar tagihan pelanggan
Prakondisi	Kasir telah membuka aplikasi dan terdapat pelanggan yang masih aktif
Kondisi Akhir	Aplikasi menampilkan daftar tagihan pelanggan tertentu
Aktor	Kasir
Alur Utama	1. Kasir menekan tombol *detail* pada salah satu pelanggan pada daftar. 2. Aplikasi menampilkan daftar tagihan yang harus dibayarkan

Tabel Skenario *Use Case* Mengubah Status Pelanggan

Nomor *Use Case*	FK03
Nama *Use Case*	Ubah status pelanggan
Tujuan	Melihat daftar tagihan pelanggan
Prakondisi	Aplikasi menampilkan daftar tagihan pelanggan tertentu
Kondisi Akhir	Aplikasi menampilkan daftar tagihan pelanggan tertentu
Aktor	Kasir
Alur Utama	1. Kasir menekan tombol bayar. 2. Aplikasi menampilkan daftar pelanggan aktif