APLIKASI *FOOD DELIVERY ORDER RESTAURANT* BERBASIS ANDROID

TUGAS AKHIR

Disusun dan Diajukan sebagai Salah Satu Syarat Menempuh Ujian Akhir
Memperoleh Gelar Ahli Madya pada Fakultas Sains dan Teknologi
Universitas Pembangunan Panca Budi
Medan

Disusun oleh:

Nama	**: Aditya**
NPM	**: 1614373115**
Program Studi	**: Teknik Komputer**

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS PEMBANGUNAN PANCA BUDI
MEDAN
2020

HALAMAN PENGESAHAN

APLIKASI FOOD DELIVERY ORDER RETAURANT BERBASIS ANDROID

Dipersiapkan dan disusun oleh

ADITYA

1614373115

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Meja Hijau

Program Studi Diploma III Teknik Komputer

Fakultas Sains dan Teknologi Universitas Pembangunan Panca Budi Medan

pada Hari Rabu, Tanggal 01 Juli 2020

DOSEN PEMBIMBING

SRI WAHYUNI, S.Kom., M.Kom

Tugas akhir ini telah diterima sebagai salah satu persyaratan

untuk memperoleh gelar Ahli Madya Komputer

Medan, 01 Juli 2020

DEKAN FAKULTAS SAINS DAN TEKNOLOGI

Hamdani, ST., MT.

KETUA PROGRAM STUDI

Akhyar Lubis, S.Kom., M.Kom

KATA PENGANTAR

Bismillahirrahmanirrahim,

Puji beserta syukur saya panjatkan kehadirat Allah *Subhanahu Wata'ala,* atas segala rahmat dan karunia-Nya. Shalawat beserta salam tidak lupa pula kita panjatkan kepada Rasulullah Muhammad *Shalallahu'alaihi Wa Sallam*. Yang telah membawa kita dari masa kebodohan ke masa yang penuh dengan ilmu pengetahuan. Akhirnya saya dapat menyelesaikan laporan tugas akhir ini dengan judul "**Aplikasi *Food Delivery Order Restaurant* Berbasis Android**". Penyusunan laporan tugas akhir ini bertujuan untuk menyelesaikan program diploma tiga (D3) pada Fakultas Sains Dan Teknologi Universitas Pembangunan Panca Budi jurusan Teknik Komputer.

Dalam kesempatan ini, saya ingin menyampaikan rasa hormat dan terima kasih kepada semua pihak yang telah memberikan bimbingan dan bantuan yang sangat berharga dalam menyelesaikan laporan tugas akhir ini :

1. Penulis mengucapkan rasa terimakasih yang sebesar-besarnya dan mempersembahkan tugas akhir ini kepada orang tua tersayang yang telah mendidik,membesarkan, membiayai kuliah, sehingga penulis dapat menyelesaikan tugas terakhir ini berkat semangat dan doa dari Ayah Zulkifli dan Ibunda tercinta Ratnawati, beserta seluruh keluarga.
2. Bapak Dr. H. Muhammad Isa Indrawan, S.E., M.M. selaku Rektor Universitas Pembangunan Panca Budi.

3. Bapak Hamdani, ST., MT. selaku Dekan Fakultas Sains Dan Teknologi Universitas Pembangunan Panca Budi.
4. Bapak Akhyar Lubis, S.Kom., M.Kom. selaku Ketua Prodi Teknik Komputer Universitas Pembangunan Panca Budi.
5. Ibu Sri Wahyuni, S.Kom., M.Kom. selaku Pembimbing utama tugas akhir penulis dan meluangkan watunya untuk memberikan arahan sehingga penyelesaian tugas akhir ini dapat berjalan dengan lancar.
6. Ibu Rita Warni, S.Kom., M.Kom. selaku pembimbing II yang telah membantu dan memberikan arahan sehingga terselesainya tugas akhir ini dengan baik.
7. Seluruh dosen dan karyawan di LP3I College Banda Aceh yang telah membantu dan menyediakan fasilitas selama proses pembelajaran dan penyelesaian tugas akhir penulis
8. Seluruh dosen dan karyawan di Fakultas Sains Dan Teknologi yang telah banyak memberikan pengetahuan selama belajar di Universitas Panca Budi.
9. Seluruh teman-teman seangkatan, seperjuangan yang telah sama-sama melewati masa-masa pembelajaran.

Rasa hormat dan terimakasih bagi semua pihak atas dukungan dan doa, semoga Allah *Subhanahu Wata'ala* membalas segala kebaikan mereka. Penulis memohon maaf apabila ada kesalahan-kesalahan selama ini yang tak luput dari seorang manusia biasa.

Medan, 1 Juli 2020

Penulis

Aditya

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

INTISARI

Perkembangan aplikasi *mobile* sangatlah pesat ditinjau dari perkembangan teknologi yang serba praktis, namun berbanding terbalik dengan Sistem yang berjalan sekarang mengharuskan pembeli datang langsung ke restoran untuk melihat dan memesan sendiri secara manual tanpa adanya media atau perangkat yang dapat memesan dari jarak jauh. Membangun aplikasi pemesanan makanan di restoran yang dapat mempercepat kinerja dalam pemesanan menu yang lebih efisien terhadap waktu dan kemudahan pelanggan memesan dan melakukan pembayaran. Aplikasi Pemesanan Makanan Berbasis Android ini dirancang dengan menggunakan metode *Unified Modeling Language (UML)*, dibuat menggunakan Android Studio, dan mengunakan *Firebase Database*. Hal ini dilakukan untuk mendefinisikan *requirement* aplikasi, membuat analisis dan desain serta menggambarkan arsitektur sistem tersebut. Aplikasi yang berbasis android ini dapat membantu pelayan suatu restoran untuk melakukan pemesanan makanan dan menyampaikan informasi yang terkini kepada konsumen mengenai menu makanan. Aplikasi yang dibuat dapat menangani pesanan yang banyak dengan tingkat pengunjung yang banyak. Juga dapat mempercepat dalam penyajian makanan ke konsumen.

Kata kunci: *Food Delivery*, Aplikasi, Android, *Restaurant*

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dewasa ini, aplikasi mobile mengakibatkan kecanduan pada smartphone yang tentu tidak bisa kita pungkiri, berkembangnya teknologi di era digital sangat berpengaruh pada kehidupan manusia yang menginginkan sesuatu hal yang instan. Penggunaan teknologi oleh manusia diawali dengan pengubahan sumber daya alam menjadi alat-alat sederhana. Penemuan prasejarah tentang kemampuan mengendalikan api telah menaikkan ketersediaan sumber-sumber pangan, sedangkan penciptaan roda telah membantu manusia dalam bepergian dan mengendalikan lingkungan mereka. Perkembangan teknologi terbaru, termasuk di antaranya mesin cetak, telepon, dan Internet, telah memperkecil hambatan fisik terhadap komunikasi dan memungkinkan manusia untuk berinteraksi secara bebas dalam skala global.

Metode pemesanan makanan yang masih menggunakan metode lama, yang mengharuskan pembeli datang langsung ke restoran untuk melihat dan memesan sendiri secara manual tanpa adanya media atau perangkat yang dapat memesan dari jarak jauh merupakan permasalahan yang ada di masa ini. Pesan antar (*delivery order*) bukanlah hal yang baru, layanan yang disediakan oleh restoran di kota-kota besar merupakan metode unggulan dalam meningkatkan minat pasar konsumen. Ketergantungan terhadap koneksi jaringan internet sebagai metode yang

menghubungkan satu sama lain di dunia sangat memudahkan pelayanan yang dapat dilakukan secara *online* dibandingkan penggunaan pulsa yang memakan tarif relatif lebih mahal. Sebuah inovasi yang berkembang seperti Pemesanan makanan *online delivery* dapat membantu penjualan serta mampu mengangkat potensi promosi dan metode penjualan yang lebih mudah, dengan media smartphone pelanggan dapat dengan mudah memesan makanan tanpa harus ke restoran.

Aplikasi restoran dengan sistem *delivery order* mampu berperan dengan baik didalam kehidupan di era digital yang dibuat dengan system interaktif langsung dari *system platform* Android sehingga aplikasi ini sangat membantu dengan *smartphone* yang sudah dikenal secara luas baik muda maupun tua. Sebuah restoran yang menginginkan *brand*-nya sendiri, aplikasinya dengan nama restoran pribadi miliknya, serta dengan produk-produk yang ingin mereka promosikan melalui aplikasi ini, sehingga dapat dikenal dengan sebuah restoran yang memiliki sistem yang memudahkan pelanggan memesan makanan secara online dan menyediakan layanan *delivery order* yang dapat dikelola sendiri oleh manajemen restoran tersebut yang tentunya akan memudahkan pekerja melalui aplikasi ini.

Sehingga penulis membuat aplikasi yang berjudul **"Aplikasi *Food Delivery Order Restaurant* Berbasis Android"** dengan harapan aplikasi ini dapat bermanfaat bagi restaurant yang menerapkan sistem digital pemesanan online delivery.

1.2. Rumusan Masalah

Rumusan yang di dapat dari latar belakang masalah sebagai berikut:

1. Bagaimana merancang aplikasi *food delivery order restaurant* berbasis android?

2. Bagaimana menerapkan aplikasi *food delivery order restaurant* berbasis android di restoran?

1.3. Batasan Masalah

Berbekal dari rumusan masalah, ruang lingkupnya meliputi:

1. Aplikasi *food delivery order restaurant* dibuat menggunakan bahasa pemograman *java*.
2. Aplikasi *food delivery order restaurant* menampilkan jenis-jenis makanan dalam bentuk kategori dan informasi-informasi yang dibutuhkan pelanggan dan pengelola aplikasi servernya yaitu restoran yang bersangkutan.
3. Aplikasi *food delivery order restaurant* menggunakan *realtime database* berbasis *Firebase*.

1.4. Tujuan Penelitian

Berdasarkan rumusan masalah di atas yang menjadi tujuan peneliti adalah:

1. Menganalisa permaslahan-permasalahan yang ada pada restoran.
2. Mempermudah restoran dalam mengelola pesanan secara *online*.

1.5. Manfaat Penelitian

Penelitian ini diharapkan dapat memberi manfaat terhadap iptek maupun manfaat yang dirasakan jika diterapkan. Adapun manfaatnya sebagai berikut:

1. Manfaat penelitian yang dilakukan terhadap iptek yaitu mampu megubah sistem pemesanan manual menjadi pemesanan yang digital menggunakan sistem yang terstruktur, memudahkan kinerja dan mengenalkan kepada

masyarakat awam teknologi yang mampu membuat semuanya terasa lebih mudah.

2. Manfaat yang dapat dirasakan oleh perusahaan akan lebih mudah mengelola pesanan sehingga *delivery* tiba tepat waktu dan dengan adanya aplikasi ini yang merupakan *brand* restoran sendiri mampu memangkas biaya yang diterapkan oleh penyedia jasa atau pihak ketiga. Pelanggan yang memakai aplikasi ini mampu melihat dan menilai menu yang ada di aplikasi retoran secara lebih detail.

1.6. Sistematika Penulisan

Sistematika penulisan yang menjelaskan masing-masing isi bab secara ringkas asalah sebagai berikut:

1. BAB I PENDAHULUAN

   Bab I menjelaskan tentang latar belakang lahirnya gagasan tentang aplikasi pemesanan *online delivery* makanan pada *restaurant*, yaitu aplikasi berbasis android yang mudah dioperasikan dan memeiliki beberapa keuntungan bagi pemilik dan pelanggan. Adapaun sistematika penulisan sebagai berikut: Latar Belakang Masalah, Rumusan Masalah, Batasan Masalah, Tujuan Penelitian, Manfaat Penelitian, dan Sistematika Penulisan.

2. BAB II LANDASAN TEORI

   Bab II menjelaskan Landasan Teori dengan penyajian data dari aplikasi yang memberikan pemahaman dan bayangan yang jelas terhadap sistem penulisan dan data yang diperlukan.

3. BAB III PERANCANGAN

Bab ini juga menguraikan tentang gambaran perancangan *project*, misalnya gambaran umum rancangan *project* yang akan dibangun, gambaran umum produk, serta data yang dipergunakan untuk memecahkan masalah-masalah yang dihadapi yang berkaitan dengan implementasi *project*. Adapun sistematika penulisannya yaitu : Bahan/Data Perancangan Aplikasi, Peralatan, Prosedur dan Pengumpulan Data.

4. BAB IV IMPLEMENTASI DAN PEMBAHASAN

Bab IV merupakan tahap implementasi dan uji coba yang akan dibahas bagaimana sistem aplikasi bekerja dan proses apa saja yang akan dilalui oleh aplikasi sehingga mendapatkan sebuah hasil. Sistematika penulisan bab ini meliputi: Implementasi dan Uji Coba Sistem dan Pembahasan.

5. BAB V PENUTUP

Bab V merupakan penutup dan kesimpulan yang dibahas secara jelas dan singkat tentang hasil penelitian yang dilakukan pada aplikasi. Adapun sistematika penulisan bab ini yaitu: Kesimpulan dan Saran.

BAB II

LANDASAN TEORI

2.1. Android

Pengertian Android adalah sistem operasi berbasis Linux yang dipergunakan sebagai pengelola sumber daya perangkat keras, baik untuk ponsel, *smartphone* dan juga PC tablet. Secara umum Android adalah *platform* yang terbuka (*Open Source*) bagi para pengembang untuk menciptakan aplikasi mereka sendiri yang digunakan oleh berbagai piranti bergerak. Telepon pertama yang memakai sistem operasi Android adalah *HTC Dream*, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan Android. Semenjak kehadirannya pada 9 Maret 2009, Android telah hadir dengan versi 1.1, yaitu sistem operasi yang sudah dilengkapi dengan pembaruan estetis pada aplikasinya, seperti jam *alarm*, *voice search*, pengiriman pesan dengan *Gmail*, dan pemberitahuan *email*.

Pada perkembangannya, sistem operasi Android telah mengalami beberapa perubahan dan perbaikan. Dan yang paling menarik adalah versi keluaran Android yang diberi nama seperti nama-nama makanan.

Menurut Teguh Arifianto (2011 : 1), android merupakan perangkat bergerak pada sistem operasi untuk telepon seluler yang berbasis linux. Menurut Hermawan (2011 : 1), Android merupakan *OS* (*Operating System*) *Mobile* yang tumbuh ditengah *OS* lainnya yang berkembang dewasa ini. *OS* lainnya seperti *Windows Mobile, i-Phone OS, Symbian,* dan masih banyak lagi. Akan tetapi, *OS* yang ada ini berjalan dengan memprioritaskan aplikasi inti yang dibangun sendiri tanpa melihat

potensi yang cukup besar dari aplikasi pihak ketiga. Oleh karena itu, adanya keterbatasan dari aplikasi pihak ketiga untuk mendapatkan data asli ponsel, berkomunikasi antar proses serta keterbatasan distribusi aplikasi pihak ketiga untuk platform mereka. Berdasarkan pendapat diatas, maka dapat ditarik kesimpulan bahwa android adalah sistem operasi berbasis linux yang sedang berkembang ditengah *OS* lainnya.

Nazrudin safaat H (2012:1) menyatakan bahwa android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencangkup sistem opreasi, *middleware* dan aplikasi. Android menyiapkan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka.

Nazrudin safaat H (2012:3) menyatakan bahwa Android dipuji sebagai "*platform mobile pertama yang lengkap, terbuka, dan bebas*".

2.1.1. Karakteristik Android

Android adalah sistem operasi *open source* untuk perangkat *mobile* dan proyek *open source* yang sesuai yang dipimpin oleh *Google*. Situs ini dan repositori *Android Open Source Project (AOSP)* menawarkan informasi dan *source code* yang diperlukan untuk membuat varian *custom* dari OS Android, perangkat *Port* dan aksesori ke *platform* Android, dan memastikan perangkat memenuhi persyaratan kompatibilitas yang menjaga ekosistem Android sebagai lingkungan yang sehat dan stabil bagi jutaan pengguna.

Sebagai proyek *open source*, tujuan Android adalah untuk menghindari titik pusat kegagalan di mana satu pemain industri dapat membatasi atau mengontrol inovasi dari pemain lain. Untuk itu, Android adalah sistem operasi penuh, kualitas produksi untuk produk konsumen, lengkap dengan kode sumber disesuaikan yang dapat *porting* ke hampir semua perangkat dan dokumentasi publik yang tersedia untuk semua orang. Berikut merupakan beberapa karakteristik android:

1. Lengkap *(Complete Platform)*

   Para desainer dapat melakukan pendekatan yang komprehensif ketika mereka sedang mengembangkan *platform* Android. Android merupakan sistem operasi yang aman dan banyak menyediakan *tools* dalam membangun *software* dan memungkinkan untuk peluang pengembangan aplikasi.

2. Terbuka (*Open Source Platform)*

   Platform Android disediakan melalui lisensi *open source*. Pengembang dapat dengan bebas untuk mengembangkan aplikasi. Android sendiri menggunakan Linux Kernel 2.6.

3. Free (Free Platform)

   Android adalah *platform*/Aplikasi yang bebas untuk *develop*. Tidak ada lisensi atau biaya *royalty* untuk dikembangkan pada platform Android. Tidak ada biaya keanggotaan diperlukan. Tidak diperlukan biaya pengujian. Tidak ada kontrak yang diperlukan. Aplikasi untuk android dapat didistribusikan dan diperdagangkan dalam bentuk apa pun.

2.1.2. Arsitektur Sistem Android

Google sebagai pencipta Android yang kemudian diasuh oleh *Handset Alliance* mengibaratkan Android sebagai sebuah tumpukan *software*. Setiap lapisan pada tumpukan ini menghimpun beberapa program yang mendukung fungsi-fungsi spesifik dari sistem operasi.

1. Linux *Kernel*

   Linux *Kernel* adalah *layer* di mana inti dari *operating system* dari Android itu berada. Berisi *file-file* sistem yang mengatur *system processing, memory, resource, drivers,* dan sistem-sistem operasi Android lainnya.

2. Android *Runtime*

   Lapisan setelah Kernel Linux adalah Android *Runtime*. Android *Runtime* ini berisi *Core Libraries* dan *Dalvik Virtual Machine*. *Core Libraries* mencakup serangkaian inti *library Java*, artinya Android menyertakan satu set *library-library* dasar yang menyediakan sebagian besar fungsi-fungsi yang ada pada *library-library* dasar bahasa pemrograman *Java*.

3. Libraries

   Bertempat di level yang sama dengan Android *Runtime* adalah *Libraries*. Android menyertakan satu set *library-library* dalam bahasa C/C++ yang digunakan oleh berbagai komponen yang ada pada sistem Android.

4. Application Framework

   Lapisan selanjutnya adalah yang mencakup program untuk mengatur fungsi-fungsi dasar smartphone. *Application Framework* merupakan

serangkaian *tool* dasar seperti alokasi *resource smartphone*, aplikasi telepon, pergantian antar – proses atau program, dan pelacakan lokasi fisik telepon.

5. Application dan *Widgets*

Di lapisan teratas bertempat pada aplikasi itu sendiri. Di lapisan inilah anda menemukan fungsi-fungsi dasar *smartphone* seperti menelepon dan mengirim pesan singkat, menjalankan *web browser*, mengakses daftar kontak, dan lain-lain. Bagi rata-rata pengguna, lapisan inilah yang paling sering mereka akses. Mereka mengakses fungsi- fungsi dasar tersebut melalui user *interface*.

2.1.3. Tipe Aplikasi Android

Terdapat tiga kategori aplikasi pada Android :

1. Foregroud Activity

Aplikasi yang hanya dapat dijalankan jika tampil pada layar dan tetap efektif walaupun tidak terlihat. Aplikasi dengan tipe ini pasti memperhitungkan siklus hidup *activity*, sehingga perpindahan antar *activity* dapat berlangsung dengan lancar.

2. Background Service

Aplikasi yang memiliki interaksi terbatas dengan *user*, selain dari pengaturan konfigurrasi, semua dari prosesnya tidak tampak pada layar. Contohnya aplikasi penyaring panggilan atau sms *autorespon*.

3. Intermittent Activity

Aplikasi yang masih membutuhkan beberapa masukkan dari pengguna, namun sebagian sangat efektif jika dijalankan di *background* dan jika

diperlukan akan memberi tahu pengguna tentang kondisi tertentu. Contohnya pemutar musik.

Untuk aplikasi yang kompleks akan sulit untuk menentukan kategori aplikasi tersebut apalagi kategori aplikasi memiliki ciri-ciri dari semua kategori. Oleh karenanya perlu pertimbangan bagaimana aplikasi tersebut digunakan dan menentukan kategori aplikasi yang sesuai.

2.1.4. Versi Android

Berikut merupakan versi Android *OS* yang ada hingga sekarang :

1. Android *Donut* (Versi 1.6)
2. Android *Eclair* (Versi 2.1)
3. Android *Froyo* (Versi 2.2)
4. Android *GingerBread* (Versi 2.3)
5. Android *Honeycomb* (Versi 3.0)
6. Android *Ice Cream Sandwich* (Versi 4.0)
7. Android *Jelly Bean* (Versi 4.3)
8. Android *KitKat* (Versi 4.4)
9. Android *Lolipop* (Versi 5.0)
10. Android *Marshmallow* (Versi 6.0)
11. Android *Nougat* (Versi 7.0)
12. Android *Oreo* (Versi 8.0)
13. Android *Pie* (Versi 9)
14. Adroid Q / *Ten* (Versi 10)

2.1.5. Siklus Hidup Aplikasi Android

Siklus hidup aplikasi Android dikelola oleh sistem, berdasarkan kebutuhan pengguna, sumberdaya yang tersedia, dan sebagainya. Misalnya pengguna ingin menjalankan *browser web*, pada akhirnya sistem yang akan menentukan menjalankan aplikasi. Sistem sangat berperan dalam menentukan apakah aplikasi dijalankan, dihentikan sementara atau dihentikan sama sekali. Jika pengguna ketika itu sedang menjalankan sebuah *activity*, maka sistem akan memberikan prioritas utama untuk aplikasi tersebut. Sebaliknya, jika suatu *activity* tidak terlihat dan sistem membutuhkan sumber daya yang lebih, maka *activity* yang prioritas rendah akan ditutup.

Android menjalankan setiap aplikasi dalam proses secara terpisah, yang masing-masing memiliki mesin virtual pengolah sendiri, dengan ini melindungi penggunaan memori pada aplikasi. Selain itu juga Android dapat mengontrol aplikasi mana yang layak menjadi prioritas utama. Karenanya, Android sangat sensitif dengan siklus hidup aplikasi dan komponen-komponennya.

2.1.6. Kelebihan Android

Ada beberapa hal yang menjadi kelebihan android, diantaranya :

1. Keterbukaan, bebas pengembangan tanpa dikenakan biaya terhadap sistem karena berbasiskan *Linux* dan *Open Source*. Pembuat perangkat menyukai hal ini karena dapat membangun *platform* sesuai yang diinginkan tanpa

harus membayar *royality*. Sementara pengembang *software* menyukai karena Android dapat digunakan diperangkat manapun dan tanpa terikat oleh *vendor* manapun.

2. Arsitektur komponen dasar Android terinspirasi dari teknologi internet *Mashup*. Bagian dalam sebuah aplikasi dapat digunakan oleh aplikasi lainnya, bahkan dapat diganti dengan komponen lain yang sesuai dengan aplikasi yang dikembangkan.
3. Banyak dukungan *service*, kemudahan dalam menggunakan berbagai macam layanan pada aplikasi seperti penggunaan layanan pencarian lokasi, *database SQL, browser* dan penggunaan peta. Semua itu sudah tertanam pada Android sehingga memudahkan dalam pengembangan aplikasi.
4. Siklus hidup aplikasi diatur secara otomatis, setiap program terjaga antara satu sama lain oleh berbagai lapisan keamanan, sehingga sistem menjadi lebih stabil. Pengguna tak perlu khawatir dalam menggunakan aplikasi pada perangkat yang memorinya terbatas.
5. Dukungan grafis dan suara terbaik, dengan adanya dukungan 2D grafis dan animasi yang diilhami oleh *Flash* menyatu dalam 3D menggunakan *OpenGL* memungkinkan membuat aplikasi maupun *game* yang berbeda.
6. Portabilitas aplikasi, aplikasi dapat digunakan pada perangkat yang ada saat ini maupun akan datang. Semua program ditulis dengan menggunakan bahasa pemrograman java dan dieksekusi oleh mesin *Virtual Dalvik*, sehingga kode program portabel antara *ARM, X86*, dan arsitektur lainnya.

Sama halnya dengan dukungan masukan seperti penggunaan *keyboard*, layar sentuh, *trackball* dan resolusi layar dapat disesuaikan dengan program.

2.2. Konsep Dasar Sistem

Sistem berasal dari bahasa latin yaitu *systema* dan bahasa yunani yaitu *systema* adalah suatu kesatuan yang terdiri dari komponen atau elemen yang dihubungkan bersama untuk memudahkan aliran informasi, materi atau energi untuk mencapai suatu tujuan.

Jogianto mengemukakan bahwa sistem adalah kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu. sistem ini menggambarkan suatu kejadian-kejadian dan kesatuan yang nyata adalah suatu objek nyata, seperti tempat, benda, dan orang-orang yang betul-betul ada dan terjadi.

Dengan demikian sistem merupakan kumpulan dari beberapa bagaian yang memiliki keterkaitan dan saling bekerja sama serta membentuk suatu kesatuan untuk mencapai suatu tujuan dari sistem tersebut. maksud dari suatu sistem adalah untuk mencapai suatu tujuan dan sasaran dalam ruang lingkup yang sempit.

2.2.1. Karakteristik Sistem

Jogianto (2005: 3) mengemukakan sistem mempunyai karekteristik atau sifat-sifat tertentu, yakni:

1. Komponen

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan. komponen-komponen

sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem. setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu mempengaruhi proses sistem secara keseluruhan.

2. Batasan sistem.

Batasan sistem (*boundary*) merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. batasan suatu sistem menunjukan ruang lingkup dari sistem tersebut.

3. Lingkungan Luar Sistem.

Lingkungan luar (*evinronment*) dari suatu sistem adalah apapun diluar batas sistem yang mempengaruhi operasi. Lingkungan luar sistem dapat bersifat menguntungkan dana dapat juga bersifat menguntungkan sistem tersebut. Lingkungan luar yang menguntungkan berupa energi dari sistem dan dengan demikian harus tetap dijaga dan dipelihara. sedang lingkunagn luar yang merugikan harus ditahan dan dikendalikan, kalau tidak maka akan menggangu kalangsungan hidup dari sistem.

4. Penghubung Sistem

Penghubung (*interfance*) merupakan media penghubung antara satu subsistem dengan subsistem yang lainya. melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem yang lainya. Dengan penghubung satu subsistem dapat berintegrasi dengan subsistem yang lainya membentuk satu kesatuan.

Menurut Burch dan Grundnitski (dalam Jogiyanto 2005 :196) desain sistem dapat didefinisikan sebagai penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam suatu kesatuan yang utuh dan berfungsi.

Desain sistem menetukan bagaimana suatu sistem akan menyelesaikan tahap ini menyangkut mengkonfigurasi dari komponen-komponen perangkat lunak dan perangkat keras dari suatu sesistem sehingga setelah instalasi dari sistem akan benar-benar memuaskan rancang bangun yang telah ditetapkan pada akhir tahap analisis sistem (Jogiyanto ; 2005:196).

5. Masukan Sistem

Masukan (*input*) sistem adalah energi yang masukan kedalam sistem. masukan dapat berupa masukan perawatan (maintenance *input*), dan masukan sinyal (signal *input*). Maintenance input adalah energi yang dimasukan supaya dapat beroperasi. *Signal input* adalah energi yang diproses untuk didapatkan keluaran. sebagai contoh didalam komputernya dan data adalah *signal input* untuk diolah menjadi informasi.

6. Keluaran Sistem

Keluaran (*output*) sistem adalah hasil dari energi yang diolah dan diklafikasikan menjadi keluaran yang berguna dan sisa pembuangan. misalnya untuk sistem komputer, panas yang dihasilkan adalah keluaran yang tidak berguna dan merupakan hasil sisa pembuangan, sedang informasi adalah keluaran yang dibutuhkan.

7. Pengolahan Sistem

Suatu sistem dapat mempunyai suatu bagian pengolah yang akan merubah masukan menjadi keluaran. suatu sistem produksi akan mengolah masukan berupa bahan baku dan bahan-bahan yang lain menjadi keluaran berupa barang jadi.

8. Sasaran Sistem

Sebuah sistem sudah tentu mempunyai sasaran ataupun tujuan. Dengan adanya sasaran sistem, maka kita dapat menentukan masukan yang dibutuhkan sistem dan keluaran apa yang akan dihasilkan sistem tersebut dapat dikatakan berhasil apabila mencapai/mengenai sasaran atau pun tujuan.

Berdasarkan beberapa defenisi diatas maka desain sistem dapat diartikan sebagai berikut:

a. Tahap setelah analisis dari siklus pengembangan sistem.
b. Pendefenisisn dari kebutuhan-kebutuhan fungsional.
c. Persiapan untuk rancang bangun untuk implementasi.
d. Menggambarkan bagaimana suatu sistem dibentuk.
e. Yang dapat berupa penggambaran, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam suatu kesatuan yang utuh dan berfungsi.
f. Termasuk menyangkut mengkonfigurasi dari komponenen-komponen perangkat lunak dan perangkat keras dari suatu sistem.

2.2.2. Klasifikasi Sistem

Menurut (Jogiyanto:1999), sistem dapat diklasifikasikan menurut beberapa sudut pandangan, diantaranya adalah sebagai berikut:

1. Sistem diklasifikasikan sebagai sistem abstrak (*abstrack system*) dan sistem fisik (*physical system*). Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Sistem fisik merupakan sistem yang ada secara fisik.
2. Sistem diklasifikasikan sebagai sistem alamiah (*natural system*), dan sistem buatan manusia (*human made system*). Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat manusia. Sistem buatan manusia adalah sistem yang dirancang oleh manusia . Sistem buatan manusia melibatkan interaksi antara manusia dengan mesin disebut *human machine system.*
3. Sistem diklasifikasikan sebagai sistem tertentu (*deterministic system*) dan sistem tidak tentu (*probabilistic system*). Sistem tertentu beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi diantara bagian-bagiannya dapat dideteksi dengan pasti, sehingga keluaran dari sistem dapat diramalkan. Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas.
4. Sistem diklasifikasikan sebagai system tertutup (*closed system*) dan sistem terbuka (*open system*). Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Secara teoritis sistem tertutup ini ada, tetapi kenyataanya tidak ada sistem yang

benar-benar tertutup, yang ada hanyalah *relatively closed system* (secara relatif tertutup, tidak benar- benar tertutup). Sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. sistem ini menerima masukan dan menghasilkan keluaran untuk lingkungan luar atau sub sistem lainnya.

2.3. Informasi

Informasi merupakansebuah bahan penting bagi manajemen dan pengambilan keputusan. Sistem informasi ini di dalam suatu organisasi dibatasi oleh data yang diperoleh biaya untuk pengadaan pengolahan dan penyimpanan dan sebagainya.

Definisi informasi dalam pemakaian sistem informasi yaitu informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang (Davis, 1999).

Informasi adalah data yang diolah menjadi bentuk suatu yang lebih berguna atau berarti bagi yang menerimanya (Jogiyanto, 1989).

Dari definisi informasi dapat diambil kesimpulan informasi adalah:

1. Data yang telah diolah atau data yang memiliki arti.
2. Data yang menjadi bentuk yang lebih berguna dan lebih berarti bagi penerima.
3. Data yang menggambarkan suatu kejadian (*event*) dan kesatuan nyata bagi penerima.

Menurut (Kadir, 1999), informasi adalah analisis dan sintesis terhadap data atau informasi adalah data yang telah diorganisasikan ke dalam bentuk yang sesuai dengan kebutuhan seseorang, *manajer, staf,* atau orang lain di dalam suatu organisasi atau perusahaan.

2.3.1. Pengertian Sistem Infomasi Menurut Para Ahli

Sepuluh langkah pemrosesan atau operasi yang dilakukan untuk mengkonversi data hingga menjadi informasi (Prahasta 2001):

1. *Capturing*: perekaman data dari suatu peristiwa atau kejadian di dalam beberapa formulir seperti, slip penjualan, daftar isian pribadi, pesanan pelanggan, dan sebagainya.
2. *Verifying*: pemeriksaan atau validasi data untuk memastikan bahwa data tersebut telah direkam dengan benar.
3. *Classifying:* menempatkan elemen-elemen data ke dalam kategori-kategori tertentu yang memberikan pengertian pada penggunanya. Misalnya data penjualan dapat diklasifikasikan menjadi tipe, ukuran inventori, pelanggan, *salesperson*, dan sebagainya.
4. *Arranging (sorting):* menempatkan elemen-elemen data sesuai dengan tertentu. Misalnya *file* iventori dapat diurutkan menurut *file* kode, tingkat aktivitas, nilai atau olah atribut lain yang dikodekan didalam tabel yang bersangkutan.

5. *Summarizing:* mengkombinasikan atau mengumpulkan beberapa elemen data dalam salah satu cara. Pertama, mengakumulasikan data secara sistematis. Kedua, mereduksi data secara logis.
6. *Calculating*: pemanipulasian data secara aritmetik dan lojik. Contoh tagihan pelanggan.
7. *Storing:* menempatkan data pada media penyimpanan seperti kertas, disket, *harddisk* dan sebagainya.
8. *Retrieving*: memerlukan akses ke elemen-elemen data dari media penyimpanan.
9. *Reproducing*: menduplikasi data dari suatu media ke media lainnya atau ke medium yang sama.
10. *Communicating*: mentransfer data dari suatu tempat ke tempat lainnya.

2.4. Bahasa Pemograman *Java*

Java merupakan pemrograman yang sangat popular karena rentang aplikasi yang bisa di buat dengan bahasa ini sangatlah luas, mulai dari *computer* hingga *smartphone*. Bahasa pemrograman *Java* dikembangkan pertama kali oleh *Sun Microsystem* yang dimulai oleh James Gosling dan diliris pada tahun 1995. Saat ini *Sun Microsystem* telah diakuisisi oleh *Oracle Corporation*. Apabila, sudah terbisa dengan bahasa C dan C++, Anda bisa mempelajari *java* dengan cepat. *Java* bersifat *Write Once, Run Anywhere* (program yang ditulis satu kali dan dapat berjalan pada banyak *platform*). Dengan demikian tidak mengherankan apabila aplikasi yang

dibuat menggunakan java bisa ditemukan dilingkungan *computer* dan *smartphone* tanpa perbedaan yang berarti. Sama sepeti pemrograman pada umumnya, *java* merupakan bahasa pemrograman yang mampu berkerja dengan sebuah *database* [Kelompok Gramedia,2015].

2.4.1. Karakteristik Java

Pada tahun 2009 *Oracle* membeli *Sun Microsystem* yang berakibat secara tidak langsung *java* menjadi milik *oracle* secara penuh. *java* resmi diakuisi pada 27 Januari 2010. Berikut beberapa karakteristik *Java*:

1. Sederhana

   Sintaks java seperti sintaks pada C++ tetapi sintaks *java* tidak memerlukan *header file, pointer arithmatic, struktur union, operator overloading, class virtuall base*, dan yang lainnya. Jika mengenal C++ dengan baik, maka pengguna dapat berpindah ke sintaks *java* dengan mudah.

2. Berorientasi Objek (*Object Oriented*)

   Berorientasi objek merupakan suatu teknik yang memusatkan rancangan pada data (objek) dan *interface*. Fasilitas pemrograman berorientasi objek pada java pada dasarnya adalah sama dengan C++. perbedaan utama antara java dengan C++ terletak pada penurunan berganda (*multiple inheritance*). java tidak mengenal *multiple inheritance* seperti pada C/C++. *Multiple Inheritance* membingungkan dan berakibat pada sulitnya pembuatan aplikasi. Sebagai gantinya java menggunakan *interface.*

3. Portabel

Java dapat digunakan pada segala macam arsitekstur komputer dan perangkat karena sifatnya yang *portable*. Dapat dieksekusi di beragam *platform* tanpa harus melakukan perubahan kode secara menyeluruh. Sebagai contoh, aplikasi *java* yang dapat berjalan di *windows* maka juga dapat berjalan dengan baik di sistem operasi *linux* dan *mac*. Dengan hanya membutuhkan *java virtual machine* yang serupa tanpa melakukan perubahan pada kode aplikasi tersebut.

Platform java terdiri dari sekumpulan *library, compiler, debugger* dan alat lain yang dipaket dalam *java development kit (JDK).* Agar sebuah program *java* dapat dijalankan, maka *file* dengan ekstensi. *Java* harus dikompilasi menjadi file *bytecode*. Untuk menjalankan file *bytecode* tersebut dibutuhkan *JRE* (*java runtime environment*) yang memungkinkan pengguna untuk menjalankan program *java. JRE* terdiri dari *JVM* dan pustaka *java* yang digunakan.

2.5. *Android Software Development Kit (SDK)*

Android *SDK* adalah *tool API* (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman *Java*. Android Merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang *release* oleh *Google*. Saat ini disediakan oleh *Android SDK* sebagai alat bantu

dan *API* untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman *Java* (Safaat H 2011 : 5).

2.6. *Android Virtual Device (AVD)*

AVD yang merupakan *emulator* untuk menjalankan program aplikasi android yang kita buat. *AVD* ini nantinya yang akan dijadikan sebagai tempat mencoba dan menjalankan aplikasi Android yang dibuat. *AVD* berjalan di *Virtual Machine* (Safaat H 2011 : 19).

2.7. *Android Studio*

Android Studio adalah Lingkungan Pengembangan Terpadu (*Integrated Development Environment/IDE*) resmi untuk pengembangan aplikasi Android, yang didasarkan pada *IntelliJ IDEA*. Selain sebagai *editor* kode dan fitur *developer IntelliJ* yang andal, *Android Studio* menawarkan banyak fitur yang meningkatkan produktivitas Anda dalam membuat aplikasi Android, seperti:

1. Sistem *build* berbasis *Gradle* yang fleksibel.
2. *Emulator* yang cepat dan kaya fitur.
3. Lingkungan terpadu tempat Anda bisa mengembangkan aplikasi untuk semua perangkat Android.
4. Terapkan Perubahan untuk melakukan *push* pada perubahan kode dan *resource* ke aplikasi yang sedang berjalan tanpa memulai ulang aplikasi.
5. *Template* kode dan integrasi *GitHub* untuk membantu Anda membuat fitur aplikasi umum dan mengimpor kode sampel.

6. *Framework* dan fitur pengujian yang lengkap.
7. Fitur lint untuk merekam performa, kegunaan, kompatibilitas versi, dan masalah lainnya.
8. Dukungan C++ dan *NDK*.
9. Dukungan bawaan untuk *Google Cloud Platform*, yang memudahkan integrasi *Google Cloud Messaging* dan *App Engine.*

   (Sumber : https://developer.android.com/studio/intro?hl=id)

2.8. Basis Data *(Database)*

Database adalah kumpulan *file-file* yang mempunyai kaitan antara satu *file* dengan *file* yang lain sehingga membentuk satu bangunan data untuk menginformasikan satu perusahaan, instansi dalam batasan tertentu.

Bila terdapat *file* yang tidak dapat digunakan atau dihubngkan dengan *file* yang lainnya berarti *file* tersebut bukanlah kelompok dari satu *database*, ia akan dapat membentuk satu *database* sendiri (Kristanto 2004 : 3).

2.8.1. Jenjang Basis Data

Suatu bangunan basis data memiliki jenjang sebagai berikut:

1. **Karakter**, merupakan bagian data terkecil yang berupa angka, huruf, atau karakter khusus yang membentuk sebuah item data atau *field*. Contoh: A, B, Y, Z, 1, 2, 9, 0, =, <, >, dan sebagainya.
2. ***Field/item***, merupakan representasi suatu atribut dari *record* (rekaman/tupel) yang sejenis yang menunjukkkan suatu *item* dari data.

Contoh *field* nama (berisi data nama-nama pelanggan), *field* nomor meja (berisi data nomor meja pelanggan), *field* instansi (berisi data spesifikasi instansi peminjam), dan lain sebagainya.

3. ***Record*/rekaman/tupel**, kumpulan dari *field* membentuk suatu *record* atau rekaman. *Record* menggambarkan suatu unit data individu yang tertentu. Contoh: *file* pelanggan dimana tiap-tiap *recordnya* berisi kumpulan data nama dan nomor meja yang dapat mewakili tiap-tiap data.
4. ***File*/tabel**, merupakan kumpulan dari *record-record* yang menggambarkan satu kesatuan data yang sejenis. Contoh *file* pelanggan berisi data tentang semua yang berhubungan dengan pelanggan seperti nama pelanggan dan nomor meja pelanggan, dan sebagainya.
5. ***Database***, merupakan kumpulan dari *file* atau tabel yang membentuk suatu *database*.

2.9. *Database Management System (DBS)*

Database Management System adalah kumpulan *file* yang saling berkaitan bersama dengan program untuk pengelolanya disebut sebagai *DBMS. Database* adalah kumpulan datanya, sedangkan program pengelolanya berdiri sendiri dalam satu paket program yang komersial untuk membaca data, mengisi, menghapus data, melaporkan data dalam *database* (Kristanto 2004 : 3).

2.10. *FireBase Database*

Firebase Realtime Database adalah *database* yang di-*host* di *cloud*. Data disimpan sebagai *JSON* dan disinkronkan secara *realtime* ke setiap klien yang terhubung. Ketika Anda membuat aplikasi lintas-*platform* dengan *SDK Android*, *iOS*, dan *JavaScript*, semua klien akan berbagi sebuah *instance Realtime Database* dan menerima *update* data terbaru secara otomatis.

2.10.1. Kemampuan *FireBase*

Adapun beberapa kemampuan *Firebase*, diantaranya :

a. *Realtime*

Sebagai ganti permintaan *HTTP* biasa, *Firebase Realtime Database* menggunakan sinkronisasi data setiap kali data berubah, semua perangkat yang terhubung akan menerima *update* dalam waktu milidetik. Memberikan pengalaman yang kolaboratif dan imersif tanpa perlu memikirkan kode jaringan.

b. *Offline*

Aplikasi *Firebase* tetap responsif bahkan saat *offline* karena *SDK Firebase Realtime Database* menyimpan data ke *disk*. Setelah konektivitas pulih, perangkat klien akan menerima setiap perubahan yang terlewat dan melakukan sinkronisasi dengan status *server* saat ini.

c. Dapat Diakses dari Perangkat Klien

Firebase Realtime Database dapat diakses secara langsung dari perangkat seluler atau *browser web*, *server* aplikasi tidak diperlukan. Keamanan dan

validasi data dapat diakses melalui Aturan Keamanan *Firebase Realtime Database* yang merupakan kumpulan aturan berbasis ekspresi dan dijalankan ketika data dibaca atau ditulis.

d. Menskalakan di beberapa *database*

Dengan *Firebase Realtime Database* pada paket harga *Blaze*, Anda dapat mendukung kebutuhan data aplikasi Anda pada skala tertentu dengan membagi data Anda di beberapa *instance database* di *project Firebase* yang sama. Menyederhanakan autentikasi dengan *Firebase Authentication* pada project Anda dan mengautentikasi pengguna di *instance* database Anda. Mengontrol akses ke data di tiap *database* dengan aturan *Firebase Realtime Database* khusus untuk tiap *instance database.*

2.11. Unified Modeling *Languange (UML)*

Unified Modelling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. *UML* menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan *UML* dapat dibuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena *UML* juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka lebih cocok untuk penulisan piranti lunak dalam bahasa berorientasi objek seperti *C++, Java,* atau *VB. NET*.

2.11.1. Diagram UML

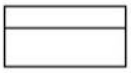
Setiap sistem yang kompleks seharusnya bisa dipandang dari sudut yang berbeda – beda sehingga bisa mendapatkan pemahaman secara menyeluruh. Untuk upaya tersebut *UML* menyediakan 9 jenis diagram yang dapat dikelompokkan berdasarkan sifatnya statis atau dinamis. Ke 9 diagram dalam *UML* itu adalah :

1. *Class Diagram*

   Class Diagram bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi serta relasi.

Tabel 2.1. Simbol dalam *Use Case Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.

3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

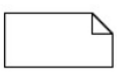
2. Diagram Objek

Diagram objek bersifat statis. Diagram ini memperlihatkan objek-objek serta relasi antar objek. Diagram objek memperlihatkan instansiasi statis dari segala sesuatu yang dijumpai pada diagram kelas.

3. Use case Diagram

Diagram ini bersifat statis. Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna.

Tabel 2.2. Simbol dalam *Use Case Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

4. *Sequence Diagram* (Diagram urutan)

Diagram ini bersifat dinamis. Diagram *sequence* merupakan diagram interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu.

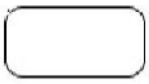
5. Collaboration Diagram

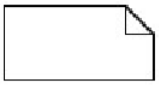
Diagram ini bersifat dinamis. Diagram kolaborasi adalah diagram interaksi yang menekankan organisasi struktural dari objek–objek yang menerima serta mengirim pesan (*message*).

6. Statechart Diagram

Diagram ini bersifat dinamis. Diagram ini memperlihatkan state – state pada sistem, memuat *state*, transisi, *event*, serta aktifitas. Diagram ini terutama penting untuk memperlihatkan sifat dinamis dari antarmuka, kelas, kolaborasi dan terutama penting pada pemodelan sistem – sistem yang reaktif.

Tabel 2.3. Simbol dalam *Statechart Diagram*

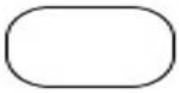
NO	GAMBAR	NAMA	KETERANGAN
1		*State*	Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		*Initial Pseudo State*	Bagaimana objek dibentuk atau diawali
3		*Final State*	Bagaimana objek dibentuk dan dihancurkan
4		*Transition*	Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

6		*Node*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

7. Activity Diagram

Diagram ini bersifat dinamis. Diagram ini adalah tipe khusus dari diagram state yang memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dari suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi – fungsi dalam suatu sistem dan memberi tekanan pada aliran kendali antar objek.

Tabel 2.4. Simbol dalam *Activity Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Activity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

8. Component Diagram

Diagram ini bersifat statis. Diagram ini memperlihatkan organisasi serta kebergantungan pada komponen–komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan ke dalam satu atau lebih kelas-kelas, antarmuka–antarmuka serta kolaborasi – kolaborasi.

9. Deployment Diagram

Diagram ini bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (saat *run time*). Dengan ini memuat simpul–simpul (*node*) beserta komponen–komponen yang ada di dalamnya. *Deployment* diagram berhubungan erat dengan diagram kompoen dimana *deployment* diagram memuat satu atau lebih komponen–komponen. Diagram ini sangat berguna saat aplikasi berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*).

BAB III

PERANCANGAN

3.1. Bahan dan Data

Bahan-bahan yang digunakan dalam perancangan aplikasi *Food Delivery Order Restaurant* meliputi:

1. Logo aplikasi.
2. Gambar menu makanan dan minuman.
3. Data pelanggan yang telah melakukan pesanan seperti lokasi, nomor *Handphone,* jenis makanan yang dipesan, jumlah total harga makanan yang dipesan pelanggan.
4. *Firebase database* digunakan untuk menampung data pelanggan dan data aplikasi yang mampu melakukan operasi *CRUD* (*Create, Read, Update, Delete*) secara *realtime*.
5. UML (*Unified Modeling Languange*) digunakan untuk membuat rancangan data aplikasi yang telah dikumpulkan.

3.2. Peralatan

Dalam proses penciptaan dan perancangan aplikasi *Food Delivery Order Restaurant* dijalankan oleh beberapa perangkat keras dan perangkat lunak yang digunakan yaitu :

1. Perangkat Keras

Perangkat keras yang digunakan sebagai media untuk menjalankan suatu perintah yang terdapat di dalam program *Food Delivery Order Restaurant* berbasis android yaitu:

a. Laptop *HP* 14-An004au *Amd* A8 dengan *VGA Radeon R5 Vram* 1GB
b. *Ram* 8GB
c. *Smartphone Xiaomi* Mi A1 android 10 (Q) *Snapdragon* 625 *Ram* 4GB

2. Perangkat Lunak

Adapun perangkat lunak yang digunakan sebagai media untuk menjalankan suatu perintah yang terdapat di dalam program *Food Delivery Order Restaurant* berbasis android yaitu:

a. *Windows* 10 *Pro*
b. *Microsoft Office* 365
c. *Android Studio*
d. *Corel Draw*
e. *Database Firebase*

3.3. Jenis dan Sumber Data

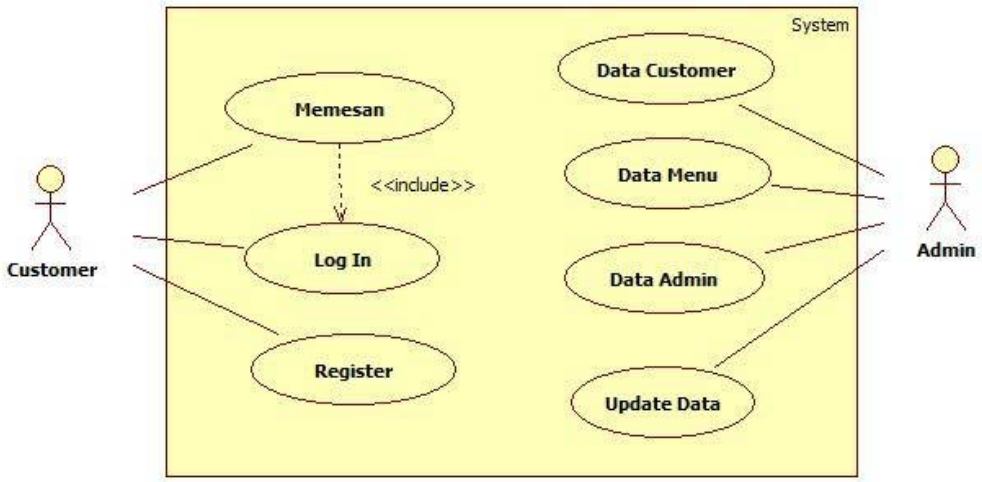
Jenis sumber data yang diperoleh untuk memperkuat argumentasi dari penelitian ini adalah data sekunder, yang diperoleh dari jurnal-jurnal, artikel, penelitian terdahulu, dan bahan-bahan lainnya yang berhubungan dengan penelitian.

3.4. Perancangan

Perancangan sistem ini akan menjelaskan tentang bayangan dari sistem aplikasi *Food Delivery Order Restaurant* yang secara umum akan digambarkan melalui perancangan *UML* (*Unified Modeling Language*) dan gambaran *UI* (*User Interface*) yang akan dibangun.

3.4.1. *Use Case*

Use Case diagram dari aplikasi *Food Delivery Order Restaurant* ini menjelaskan proses- proses interaksi antara aktor dan sistem aplikasi *Food Delivery Order Restaurant.*



Gambar 3.1. *Use Case* Diagram aplikasi *Food Delivery Order Restaurant*

Customer harus malakukan *register* terlebih dahulu, kemudian *customer* dapat *login* kedalam aplikasi *Food Delivery Order Restaurant* dan melakukan pemesanan makanan.

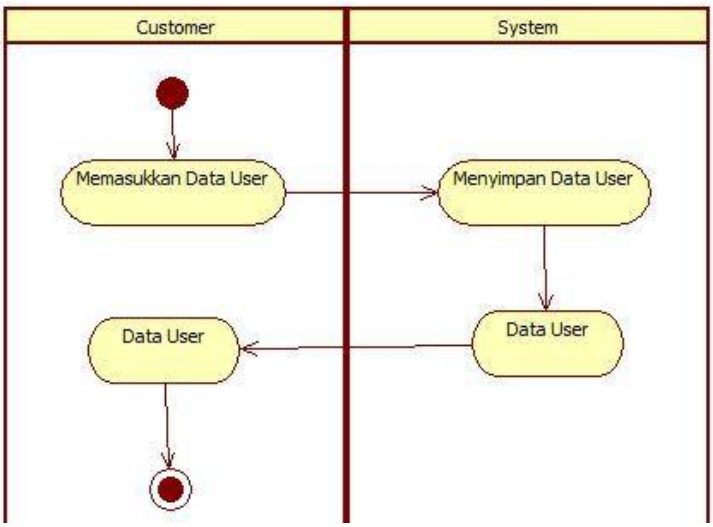
Admin dapat menggunakan hak aksesnya untuk input data *customer* baru, input data menu aplikasi *Food Delivery Order Restaurant*, *input* data admin baru, dan *update* data aplikasi *Food Delivery Order Restaurant*.

3.4.2. *Activity Diagram*

Activity diagram Food Delivery Order Restaurant menjelaskan aktivitas setiap proses rancangan aplikasi yang menggambarkan interaksi antara aktor dan sistem.

1. *Activity Diagram Register Customer*

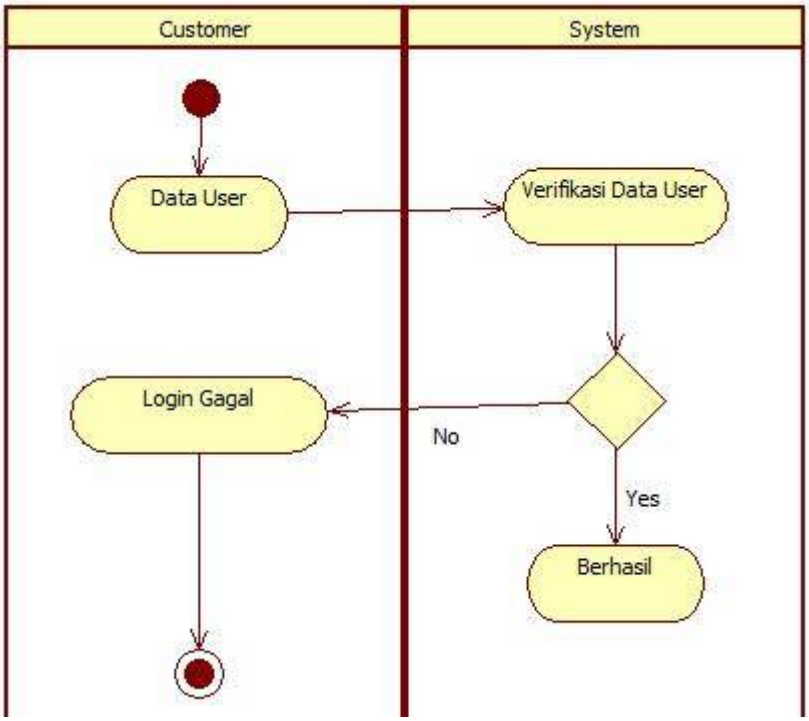
   Activity diagram ini menjelaskan bagaimana *customer* mendaftar pada aplikasi *Food Delivery Order.*



Gambar 3.2. *Activity Diagram Register Customer*

2. *Activity Diagram Login Customer*

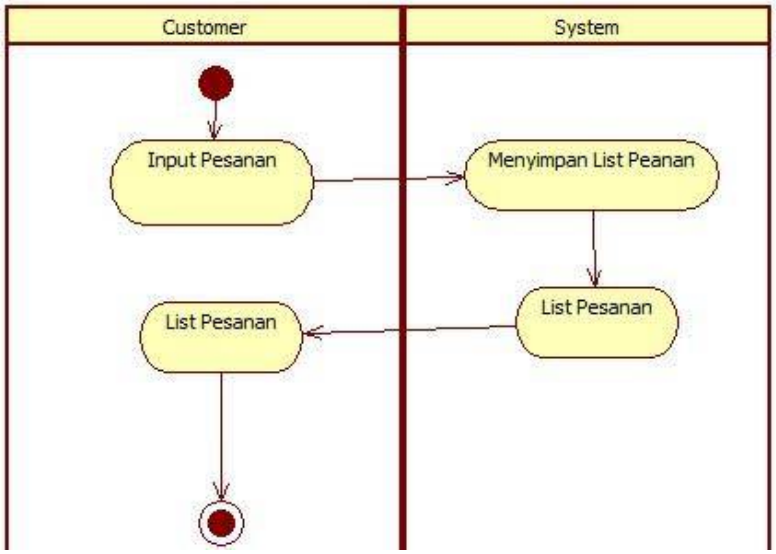
Activity diagram ini menjelaskan bagaimana *customer* melakukan proses *login* pada aplikasi *Food Delivery Order.*



Gambar 3.3. *Activity Diagram Login Customer*

3. *Activity Diagram Pesanan*

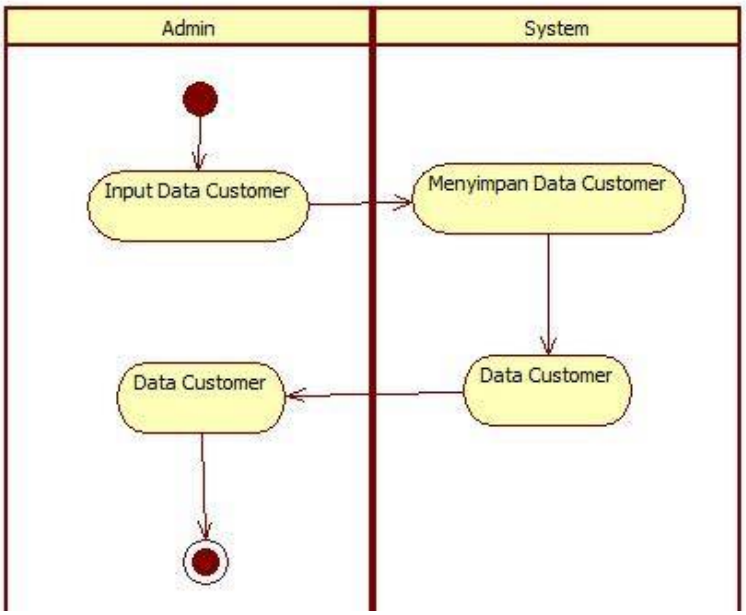
Activity diagram ini menjelaskan bagaimana *customer* melakukan proses pemesanan pada aplikasi *Food Delivery Order.*



Gambar 3.4. *Activity Diagram Pesanan*

4. *Activity Diagram Input Data Customer*

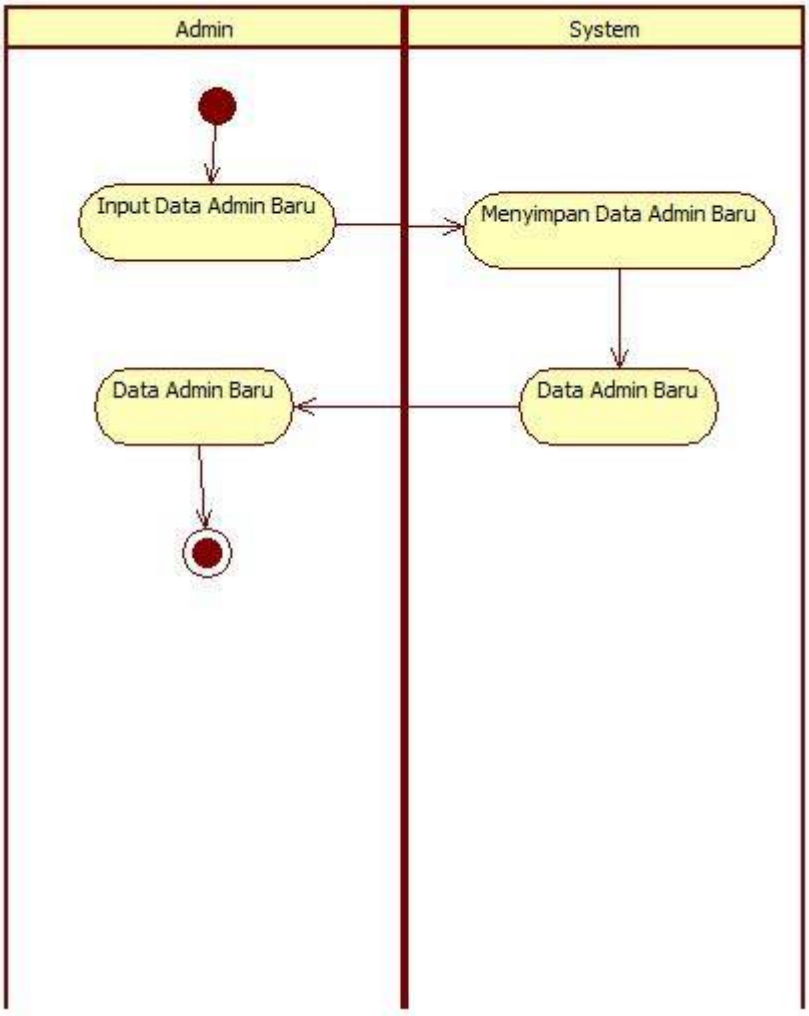
Activity diagram ini menjelaskan bagaimana *customer* melakukan pengisian data pada *aplikasi Food Delivery Order.*



Gambar 3.5. *Activity Diagram Input Data Customer*

5. *Activity Diagram Input Data* Admin

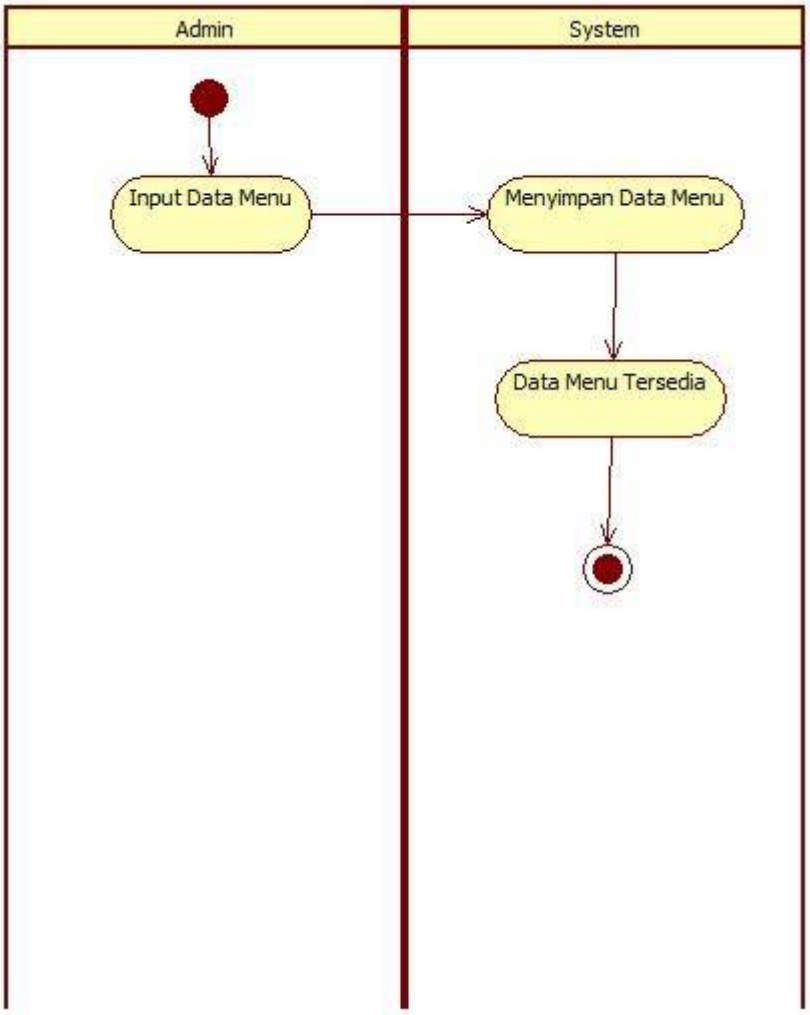
Activity diagram ini menjelaskan bagaimana admin melakukan proses pengisian data admin baru pada aplikasi *Food Delivery Order.*



Gambar 3.6. *Activity Diagram Input Data* Admin

6. *Activity Diagram Input Data Menu*

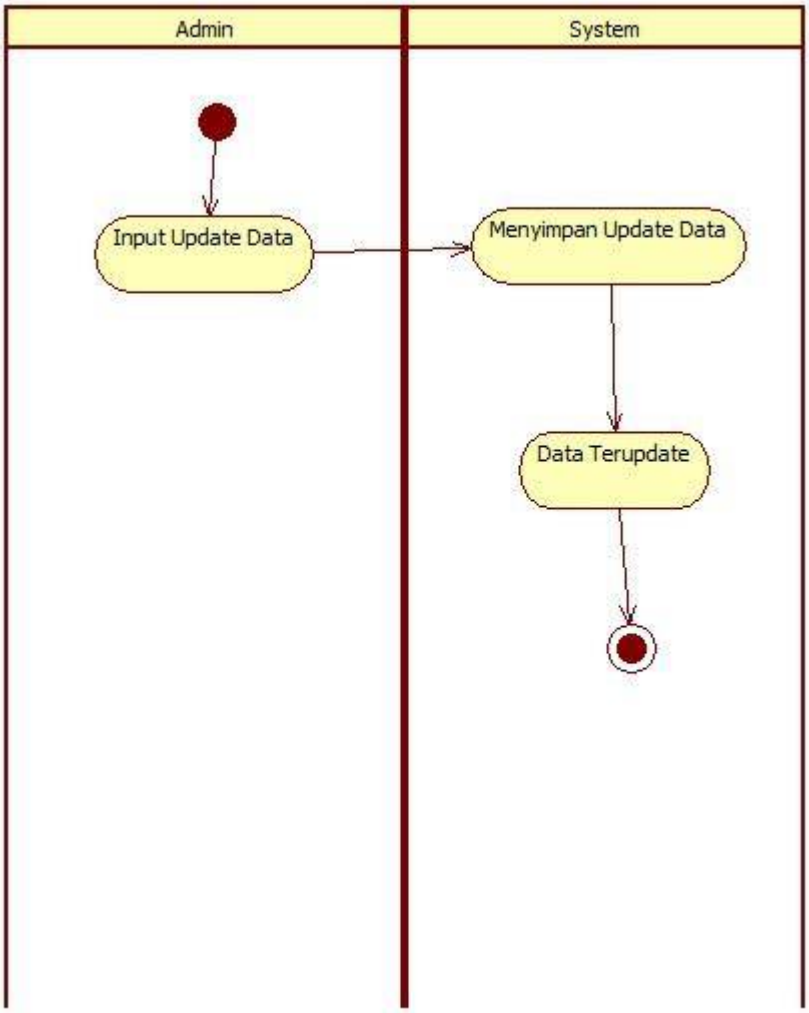
Activity diagram ini menjelaskan bagaimana admin melakukan proses pengisian data menu makanan pada aplikasi *Food Delivery Order*.



Gambar 3.7. *Activity Diagram Input Data Menu*

7. *Activity Diagram Update Data*

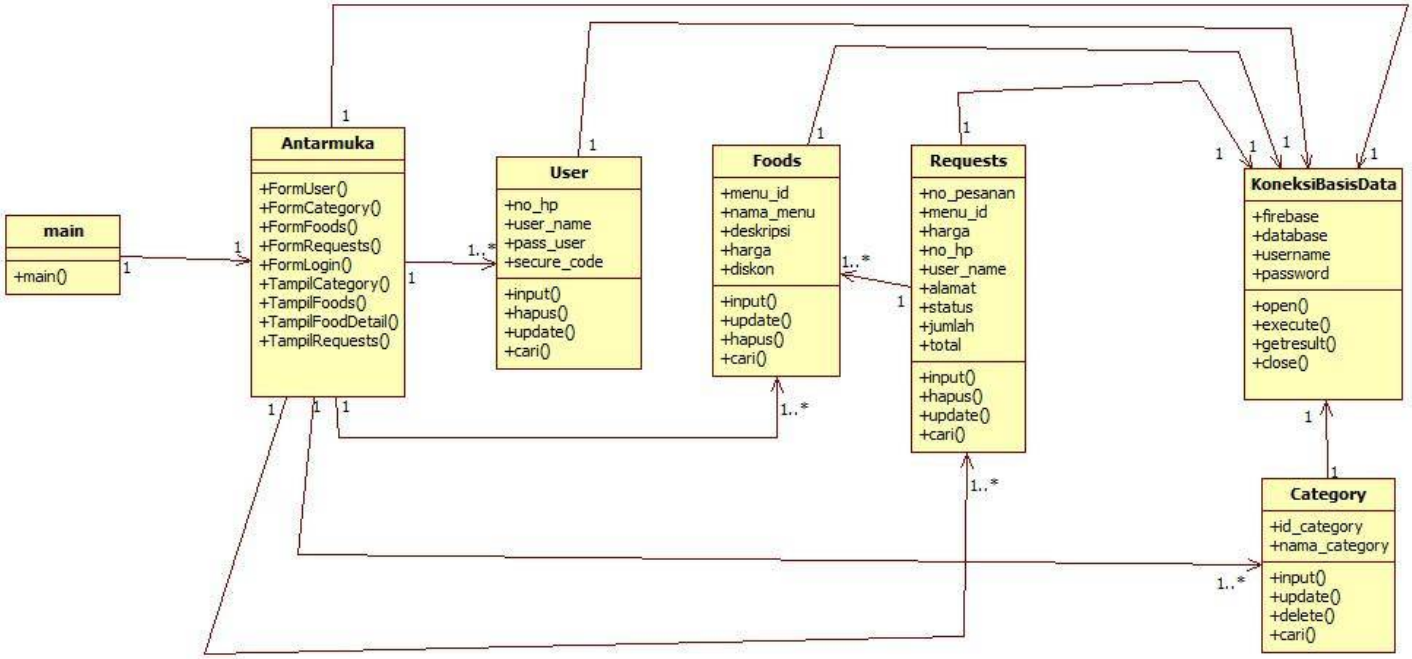
Activity diagram ini menjelaskan bagaimana admin melakukan proses *update* data menu makanan pada aplikasi *Food Delivery Order.*



Gambar 3.8. *Activity Diagram Update Data*

3.4.3. Class Diagram

Class diagram ini menjelaskan tentang hubungan antar kelas pada aplikasi *Food Delivery Order Restaurant*.



Gambar 3.9. Class Diagram Aplikasi Food Delivery Order Restaurant

3.5. Perancangan Tampilan

Tahap ini merupakan perancangan yang menampilkan *interface* (antar muka) aplikasi *Food Delivery Order Restaurant* yang akan dibangun dari awal jalannya proses aplikasi hingga menghasilkan output yang diinginkan. Adapun perancangan tampilannya sebagai berikut:

3.5.1. Tampilan Utama

Pada tampilan utama aplikasi saat dijalankan, pengguna akan diarahkan ke halaman menu makanan dan dapat melihat menu makanan yang tersedia pada aplikasi *Food Delivery Order Restaurant*. Pengguna diharuskan melakukan *Login* terlebih dahulu jika ingin memesan, dengan menekan tombol "masuk" pada informasi akun. Secara otomatis akan dialihkan ke halaman *Login*.

Gambar 3.10. Rancangan tampilan utama

3.5.2. Tampilan *Login*

Tampilan *login* merupakan aktivitas lanjutan dari halaman utama aplikasi *Food Delivery Order Restaurant*. Pada *form login* terdapat *textbox* nomor *handphone*, tombol masuk, dan tombol daftar. Pengguna diharuskan memasukkan data akun dan menekan tombol masuk untuk melanjutkan ke halaman berikutnya. Jika pengguna tidak memiliki akun, maka pengguna dapat menekan tombol daftar untuk mendaftarkan akun di halaman *SignUp*.

Gambar 3.11. Rancangan tampilan *login*

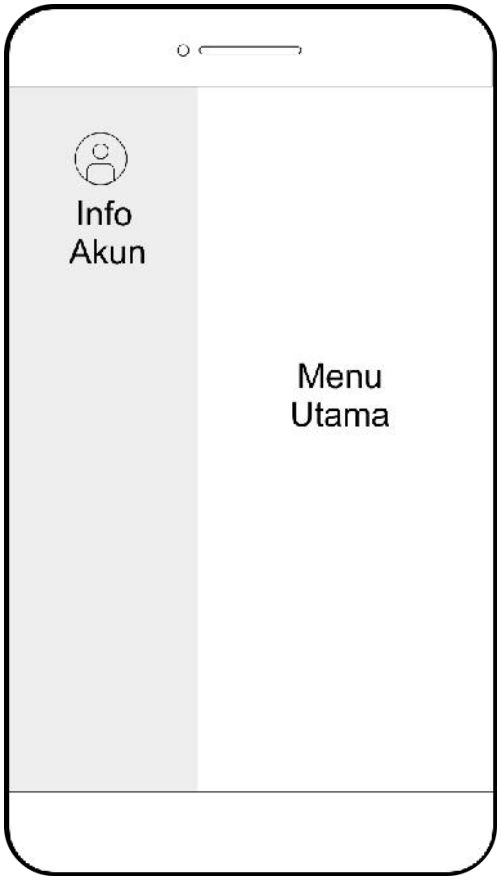
3.5.3. Tampilan *SignUp*

Tampilan *sign up* merupakan aktivitas lanjutan dari halaman *login* aplikasi *Food Delivery Order Restaurant.* Pada form *sign up* terdapat *textbox* nomor *handphone*, nama pengguna , kata sandi, dan tombol daftar. Setelah pengguna mendaftarkan akun, maka akun akan terdaftar di *database*, dan sudah dapat melakukan *login.*

Gambar 3.12. Rancangan tampilan *sign up*

3.5.4. Tampilan Akun

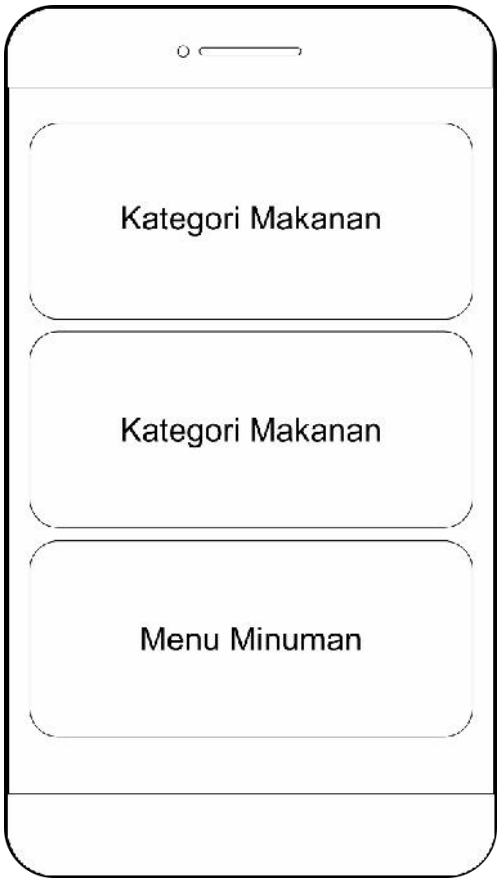
Setelah pengguna berhasil masuk kedalam aplikasi, maka informasi nama pengguna akan muncul di navigation drawer sebelah kiri aplikasi *Food Delivery Order Restaurant.*



Gambar 3.13. Rancangan tampilan *user*

3.5.5. Tampilan Menu

Pada tampilan menu akan ditampilkan gambar-gambar makanan dan nama kategori makanan yang terdapat di aplikasi *Food Delivery Order Restaurant*. Pengguna dapat menekan kategori menu yang akan dialihkan ke detail dan menuju pemesanan.



Gambar 3.14. Rancangan tampilan menu

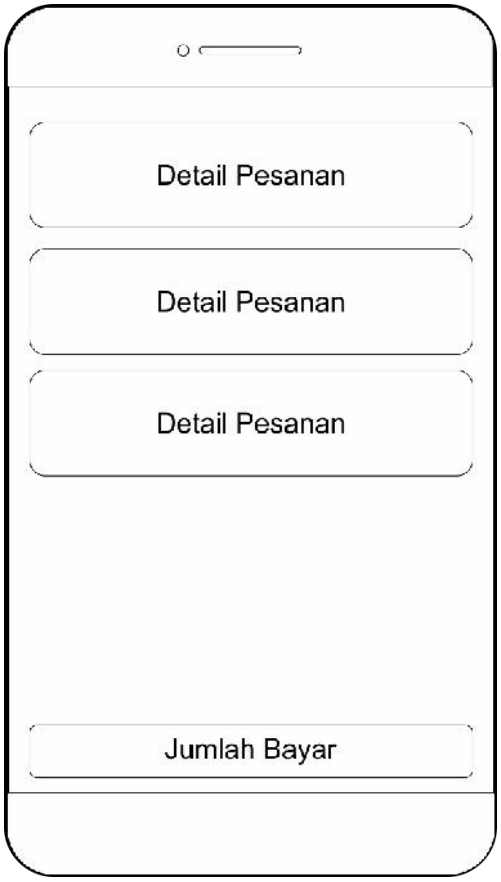
3.5.6. Tampilan Menu *Detail*

Setelah memilih makanan, maka pengguna diarahkan ke *form* detail makanan yang menampilkan deskripsi makanan dan tombol jumlah pesanan.

Gambar 3.15. Rancangan tampilan *detail* menu

3.5.7. Tampilan Pesanan

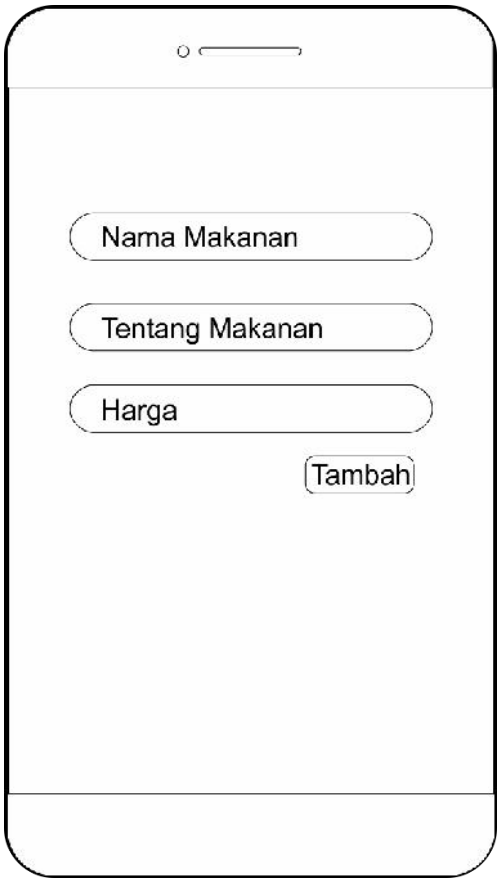
Pengguna dapat melihat jumlah total pesanan yang telah dipesan dengan jumlah yang harus dibayar. Pesanan akan masuk ke *database* sistem secara otomatis.



Gambar 3.16. Rancangan tampilan pesanan

3.5.8. Tampilan Tambah Data Menu

Admin dapat menambah data menu yang ada dengan masuk ke menu tambah kategori. *Form* tambah kategori pengisian data, admin harus menekan tombol tambah agar data menu dapat masuk kedalam *database*.

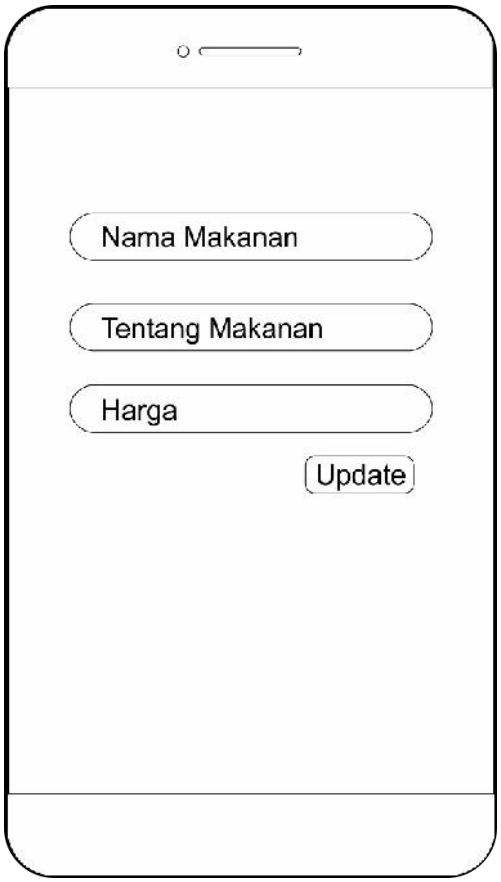


Gambar 3.17. Rancangan tampilan *form* tambah data menu

3.5.9. Tampilan Update Data Menu

Admin dapat mengubah data menu yang ada dengan masuk ke menu *update*. *Form update* menampilkan data-data menu yang ingin di-*update*. Setelah pengisian

data, admin harus menekan tombol update agar agar data menu dapat masuk kedalam *database* yang baru.



Gambar 3.18. Rancangan tampilan *update* data menu

BAB IV

IMPLEMENTASI DAN PEMBAHASAN

4.1. Implementasi

Pada bab ini akan dijelaskan mengenai aplikasi yang telah dirancang ke-dalam bahasa pemograman dan tampilan antar muka. Dalam tahapan implementasi, terdapat beberapa spesifikasi *hardware* dan *software* yang harus dipenuhi untuk menginstal aplikasi *Food Delivery Order Restaurant.* Bahasa pemograman yang digunakan adalah bahasa *Java* dengan bantuan *Firebase database*. Adapun spesifikasi yang dibutuhkan untuk menjalankan aplikasi *Food Delivery Order Restaurant* sebagai berikut:

1. Android OS
2. Versi android minimal *Lollipop* (5.0)
3. *Processor 1.5 GHZ*
4. *RAM memory 1GB*
5. *Storage memory 100 MB*

4.2. Implementasi *Interface*

Implementasi *interface* aplikasi yang telah dirancang dari perencanaan rancangan sebelumnya merupakan wujud desain yang ditampilkan pada aplikasi *Food Delivery Order Restaurant.* Adapun tampilan *interface* sebagai berikut :

4.2.1. Splash Screen Interface

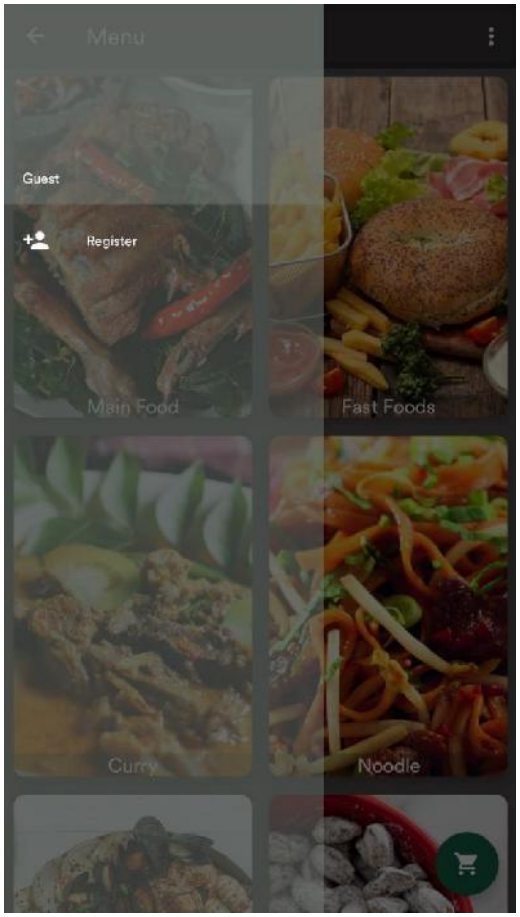
Splash screen merupakan tampilan awal yang berupa antarmuka yang hanya muncul selama beberapa detik untuk memberikan kesan yang baik terhadap aplikasi. Splash screen akan muncul ketika *customer* membuka aplikasi pertama kali.

Gambar 4.1. *Splash Screen* aplikasi *Food Order Delivery Restaurant*

4.2.2. Tampilan *Home Interface*

Halaman *home* merupakan halaman dimana pelanggan dapat melihat-lihat menu yang tersedia di dalam aplikasi sebelum melakukan pemesanan, namun menu bersifat *read only* atau tidak bisa melakukan pemesanan. Pemesanan dapat dilakukan setelah melakukan *Sign In.*

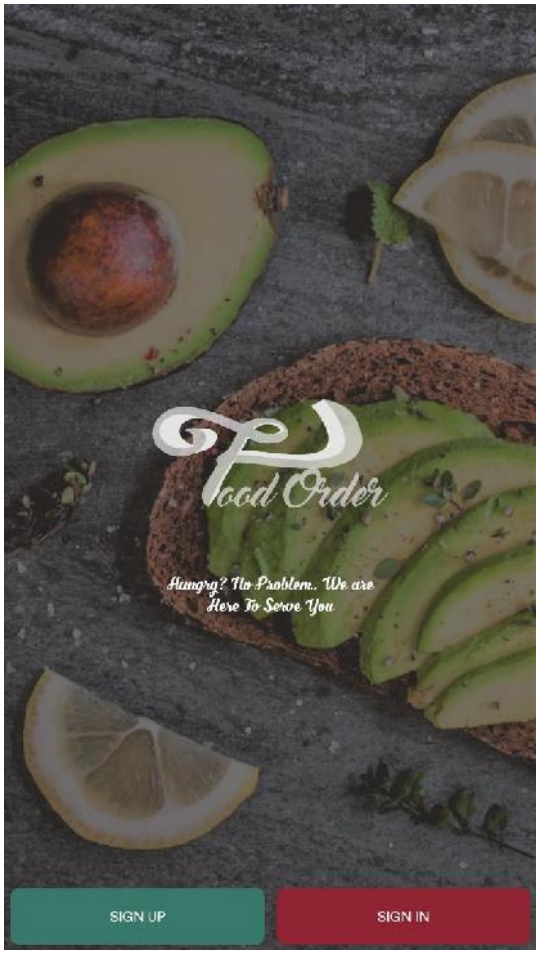
Gambar 4.2. Tampilan *home* aplikasi Food Delivery Order Restaurant



Gambar 4.3. Tombol *Register* sebelum *Sign In*

4.2.3. Tampilan Halaman *Sign In* dan *Register*

Pada halaman ini *customer* dihadapkan pada dua pilihan tombol *sign in* bagi *customer* yang telah memiliki akun, atau tombol *sign up* bagi *customer* yang belum memiliki akun agar dapat membuat akun baru.



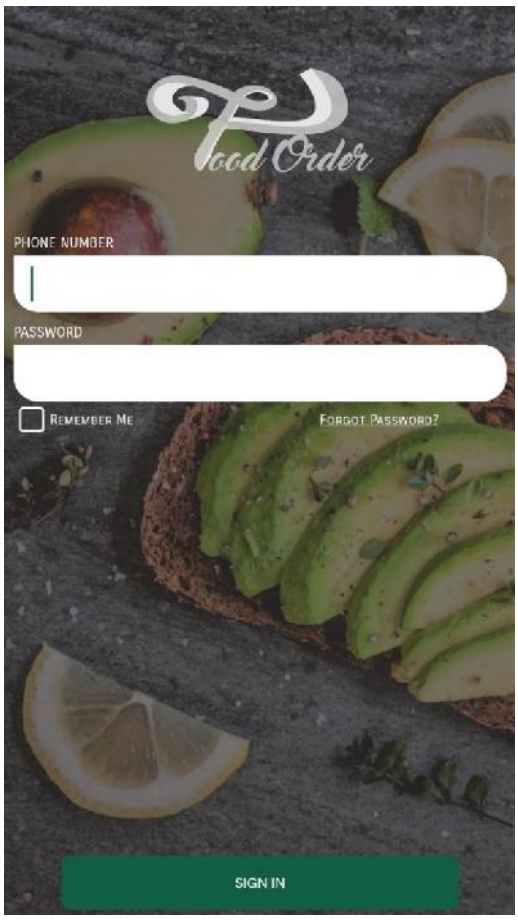
Gambar 4.4. Halaman *Sign In* dan *Register*

1. *Form Sign In*

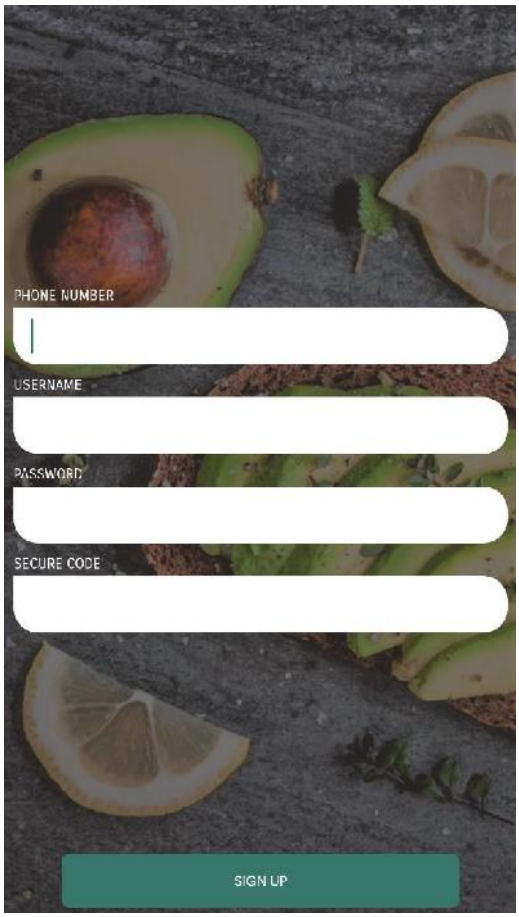
Customer dapat memasukkan data pada form *Sign In* berupa nomor handphone dan password dengan centang remember me untuk autentifikasi.

2. *Form Sign Up*

Form Sign Up merupakan interface dari proses pengisian data pada saat *customer* ingin mendaftarkan akun baru. *Customer* diharuskan mengisi nomor *handphone, username, password,* dan *security code. Security code* digunakan ketika customer ingin mengubah *password.*



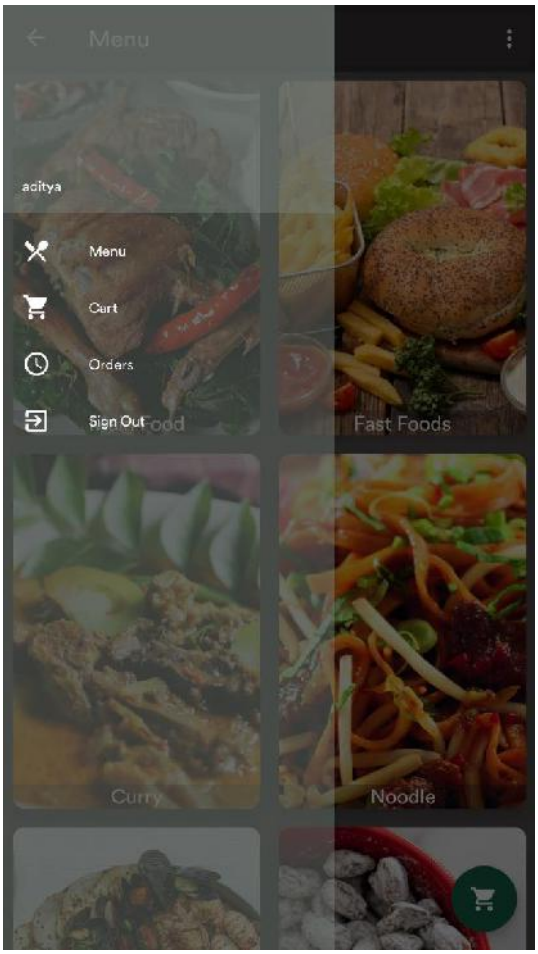
Gambar 4.5. *Form Sign In* aplikasi *Food Delivery Order Restaurant.*



Gambar 4.6. *Form Sign Up* aplikasi *Food Delivery Order Restaurant.*

4.2.4. Tampilan Informasi Akun dan *Menu Bar*

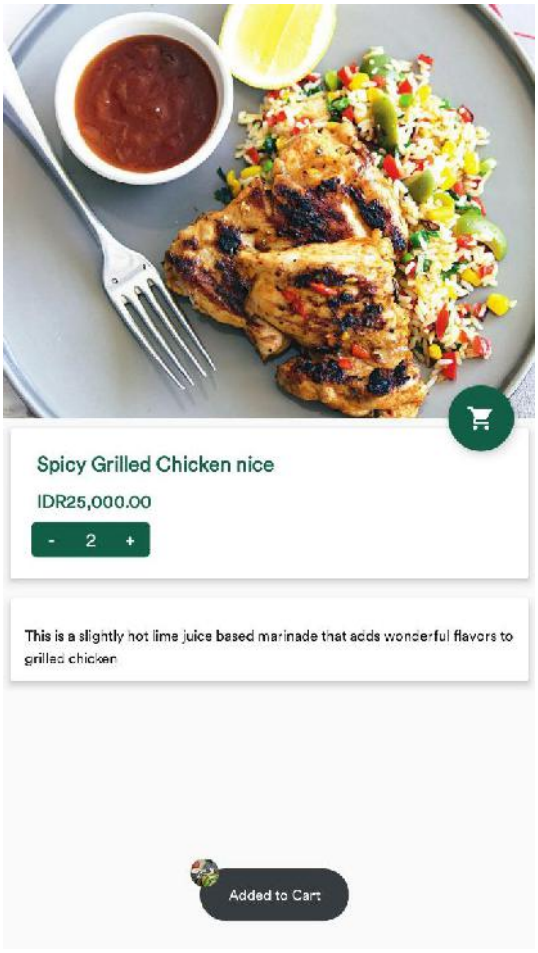
Setelah *customer* berhasil melakukan *login*, maka akan muncul *navigation drawer* yang memuat *username customer* dan menu lainnya. Sebagai penunjuk bahwa *customer* telah dapat melakukan pemesanan.



Gambar 4.7. Tampilan informasi akun dan *navigation drawer* menu

4.2.5. Tampilan *Detail* Menu

Pada tampilan *detail* menu akan ditampilkan gambar menu, tombol untuk menentukan jumlah pesanan, dan deskripsi menu sehingga *customer* dapat menentukan seleranya untuk memilih.



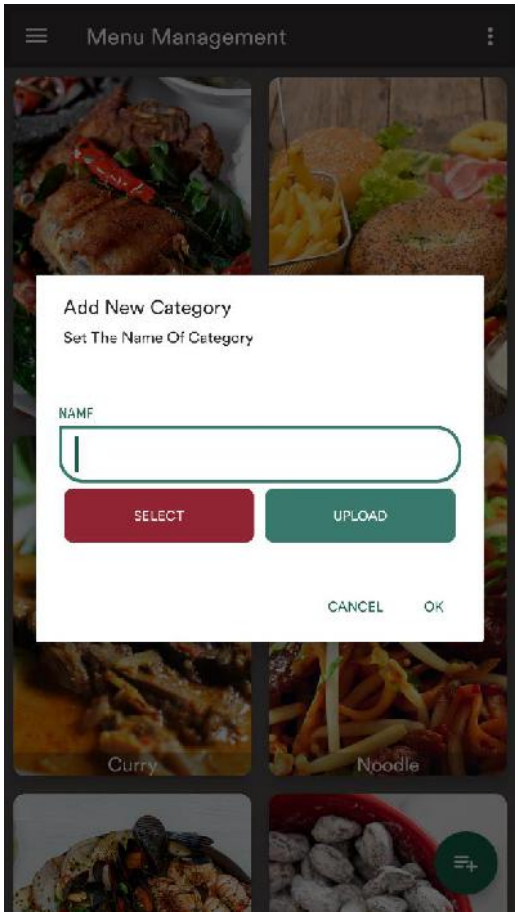
Gambar 4.8. Tampilan *detail* menu

4.2.6. Tampilan Awal dan Tambah Kategori Menu pada Aplikasi *Server*

Fitur tambah kategori menu hanya dapat dilakukan oleh admin yang masuk melalui aplikasi server dan hak akses dari admin. Berikut tampilan untuk aplikasi server dan *form* tambah kategori menu.

Gambar 4.9. Tampilan *splash screen* aplikasi *server*

Gambar 4.10. Tampilan *Sign In* aplikasi *server*

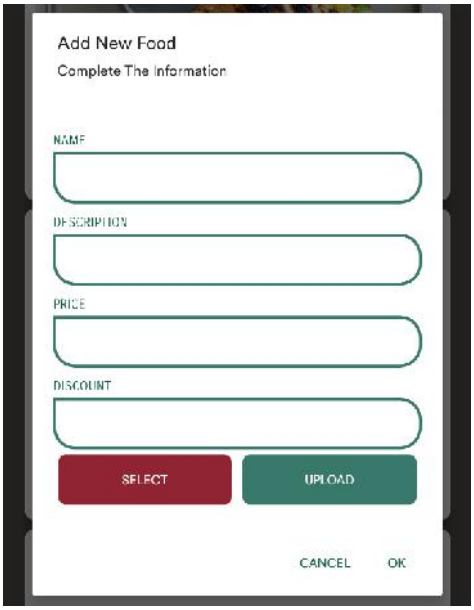


Gambar 4.11. Tampilan *form* tambah kategori menu aplikasi *server*

4.2.7. Tampilan Tambah Menu Makanan

Admin dapat menambah menu makanan dengan elemen-elemen yang dibutuhkan seperti gambar makanan yang dapat di-*upload* dari direktori lokal, menambah harga, nama menu, dan deskripsi menu makanan.

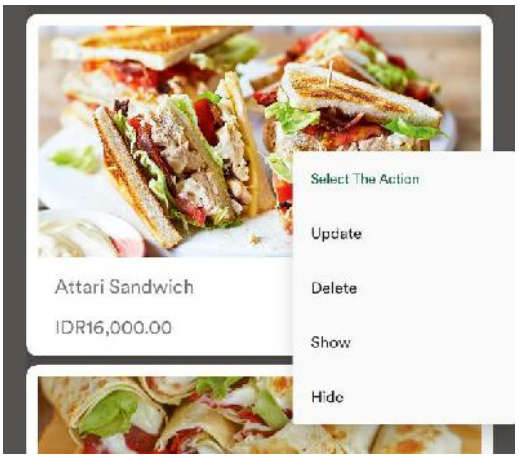
Ada dua tombol yang dapat menjalankan aksi pada form ini, yaitu tombol *select* yang berfungsi sebagai pengambil gambar dan tombol *upload* yang menjalankan aksi upload kedalan database setelah data dimasukkan.



Gambar 4.12. Tampilan *form* tambah menu aplikasi *server*

4.2.8. Tampilan *Action Box* pada Menu

Action Box merupakan panel yang berisikan sekumpulan aksi yang dapat dijalankan admin terhadap menu makanan yang terdapat dalam aplikasi *Food Delivery Order Restaurant*.

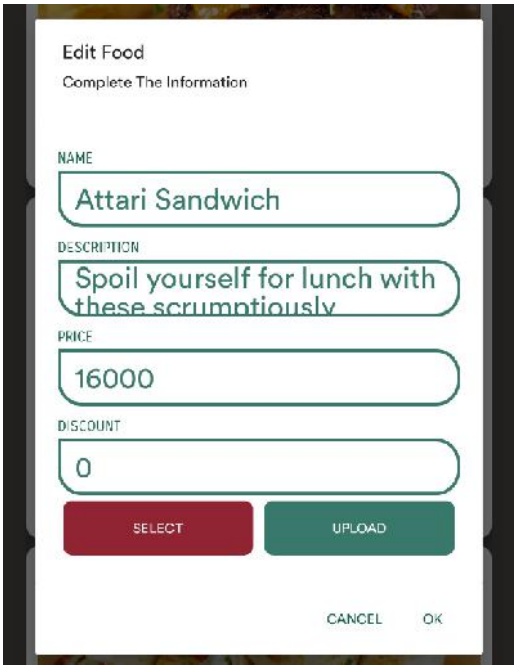


Gambar 4.13. Tampilan *Action box*

Adapun aksi yang dapat dijalankan admin adalah sebagai berikut:

1. *Update*

Admin dapat melakukan *update* data jika ada data menu yang perlu untuk diperbaharui. Ada dua tombol yang dapat menjalankan aksi pada *form* ini, yaitu tombol *select* yang berfungsi sebagai pengambil gambar dan tombol *update* yang menjalankan aksi pembaharuan data kedalan *database* setelah data dimasukkan.



Gambar 4.14. Tampilan *form update*

2. *Delete*

Fungsi *delete* pada menu makanan dapat dilakukan admin jika merasa menu makanan tersebut sudah tidak diperlukan lagi, dan akan langsung terhapus dari *database.*

3. *Show*

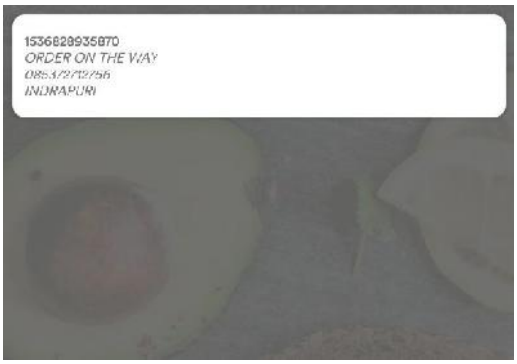
Fungsi *show* pada menu makana berfungsi sebagai aksi apabila makanan tersedia maka makanan akan ditampilkan pada list menu makanan.

4. *Hide*

Fungsi *hide* pada menu makanan berfungsi sebagai aksi untuk menyembunyikan makanan pada aplikasi customer, hal ini dilakukan apabila stok makanan tidak tersedia.

4.2.9. Tampilan *Order* Pelanggan dan *Status Action*

Setiap pesanan yang di-pesan pelanggan akan muncul pada halaman *order*, dan *Admin* dapat menentukan status pesanan yang ada, secara *realtime* akan di update ke pelanggan. Pelanggan dapat memantau pesanan melalui status dari pesanan tersebut.



Gambar 4.15. Tampilan *list order*

Gambar 4.16**.** Tampilan *action status*

4.3. Pemograman

Pada tahap pemograman aplikasi *Food Delivery Order Restaurant* menggunakan bahasa pemograman *Java* yang membantu dalam proses penciptaan program, sedangkan aplikasin pemograman yang digunakan yaitu *Android Studio* yang dapat menjalankan program berbasis bahasa *Java* dan *Kotlin*. Aplikasi *Android Studio* dibantu oleh *JDK (Java Development Kit)* yang berfungsi sebagai penghubung bahasa pemograman *Java*. *Database* yang di integrasikan menggunakan *Firebase* berbasis *online database* yang membutuhkan koneksi jaringan untuk dapat terhubung.

Emulator Android yang digunakan yaitu *Handphone Xiaomi Mi A1* berbasis android 10 sebagai media yang menjalankan aplikasi *Food Delivery Order Restaurant.* Sedangkan untuk desain *UI (User Interface)* menggunakan aplikasi *Corel Draw* 2018 sebagai aplikasi desain grafis yang dibutuhkan oleh aplikasi *Food Delivery Order Restaurant.*

4.4. Pengujian Sistem

Setiap program menjalani pengujian secara pribadi untuk memastikan bahwa program yang telah kita buat bisa bebas dari kesalahan (*bug*), walaupun tidak menutup kemungkinan masih terjadi sedikit *bug* atau tidak 100% bebas dari *bug*, namun pengujian ini setidaknya bisa meminimalisasi kesalahan yang akan terjadi. Pada tahap ini, penulis menggunakan metode pengujian unit dengan pendekatan *black-box testing*.

Pengujian unit yaitu pengujian secara individual terhadap semua program untuk memastikan bahwa program bebas dari kesalahan. Jika terjadi kesalahan, pemakai akan berusaha mencari penyebabnya dan proses untuk melakukan pencarian kesalahan ini dikenal dengan *debugging*. (Abdul Kadir, 2002:410)

Pengujian dilakukan untuk memastikan apakah aplikasi berjalan sesuai dengan yang diharapkan atau masih ada kekurangan yang harus diperbaiki. Hal-hal yang diperhatikan dalam pengujian yaitu :

a. Fungsi-fungsi yang tidak benar, baik *Input* mauput *Output.*
b. Kesalahan *Interface.*
c. Kesalahan dalam struktur data atau akses *Database.*

Dibawah ini merupakan tabel hasil pengujian aplikasi *Food Delivery Order Restaurant* menggunakan metode *black-box testing* :

Tabel 4.1. Hasil Pengujian dengan Pendekatan *Black-Box Testing*

No	Rancangan Proses	Hasil Yang Diharapkan	Hasil	Keterangan
1	Klik tombol *Cart* sebelum *Sign In*	Tidak masuk ke pemesanan dan muncul *toast "Sign In First"*	Sesuai	
2	Klik tombol *Register* pada *Navigation Drawer*	Masuk ke halaman tombol *Sign In* dan *Register*	Sesuai	
3	Mengisi *form Sign In* dan klik tombol *Sign In*	Masuk ke halaman Utama/ *Home*	Sesuai	
4	Mengisi *form Sign Up* dan klik tombol *Sign Up*	Akun terdafdar di *database*	Sesuai	
5	Klik *Forgot Password*	Masuk ke halaman *Security Code*	Sesuai	
6	Klik tombol menu pada *Navigtion Drawer*	Masuk ke halaman kategori menu	Sesuai	
7	Klik tombol *Cart*	Masuk ke halaman keranjang	Sesuai	
8	Klik tombol *Orders*	Masuk ke halaman pesanan	Sesuai	
9	Klik kategori makanan	Masuk ke halaman *list* makanan	Sesuai	
10	Klik list makanan	Masuk ke halaman detail makanan	Sesuai	
11	Klik tombol jumlah pesanan dan tombol *Cart*	Jumlah pesanan tertambah kedalam keranjang pesanan	Sesuai	

12	Klik tombol *Cart* setelah memesan	Jumlah total bayar dan jumlah pesanan muncul	Sesuai	
13	Klik tombol *Sign Out*	Keluar akun dan masuk ke halaman *login*	Sesuai	
14	Memasukkan data dan klik tombol *login admin*	Masuk ke halaman utama sebagai *admin*	Sesuai	
15	Klik tombol tambah kategori menu	Menampilkan *form Add New Category*	Sesuai	
16	Klik tombol Select pada *form Add New Category*	Masuk ke direktori lokal *Handphone*	Sesuai	
17	Klik tombol *Upload* pada form *Add New Category*	Menambahkan masukan data kategori baru ke *database*	Sesuai	
18	Klik tombol *OK* pada *form Add New Category*	Submit perubahan pada *form Add New Category*	Sesuai	
19	Klik tombol *Cancel* pada *form Add New Category*	Membatalkan perubahan dan menutup *form Add New Category*	Sesuai	
20	Klik tombol tambah menu	Menampilkan *form Add New Food*	Sesuai	
21	Klik tombol *Select* pada *form Add New Food*	Masuk ke direktori lokal *Handphone*	Sesuai	
22	Klik tombol *Upload* pada *form Add New Food*	Menambahkan masukan data menu baru ke *database*	Sesuai	
23	Klik tombol *OK* pada form *Add New Food*	Submit perubahan pada *form Add New Food*	Sesuai	
24	Klik tombol *Cancel* pada *form Add New Food*	Membatalkan perubahan dan menutup *form Add New Food*	Sesuai	

25	Klik tahan menu makanan	Menampilkan *Action Box*	Sesuai	
26	Klik *Action Update* pada *Action Box*	Menampilkan *form Edit Food*	Sesuai	
27	Klik tombol *Select* pada *form Edit Food*	Masuk ke direktori lokal *Handphone*	Sesuai	
28	Klik tombol *Upload* pada *form Edit Food*	Memperbarui data menu ke *database*	Sesuai	
29	Klik tombol *OK* pada *form Edit Food*	*Submit* perubahan pada *form Edit Food*	Sesuai	
30	Klik tombol *Cancel* pada *form Edit Food*	Membatalkan perubahan dan menutup *form Edit Food*	Sesuai	
31	Klik *Action Delete* pada *Action Box*	Menghapus data menu dari tampilan aplikasi dan *database*	Sesuai	
32	Klik *Action Hide* pada *Action Box*	Menyembunyikan data menu dari tampilan aplikasi	Sesuai	
33	Klik *Action Show* pada *Action Box*	Menampilkan kembali data menu yang telah disembunyikan dari aplikasi	Sesuai	
34	Klik tombol *Orders* pada *Navigation Drawer* admin	Menampilkan *list* pesanan pelanggan	Sesuai	
35	Klik tahan *list* pesanan pelanggan	Menampilkan *action box Status*	Sesuai	
36	Klik *Status* pada *Action Box list* pesanan pelanggan	Mengubah *Status list* pesanan pelanggan	Sesuai	

BAB IV

IMPLEMENTASI DAN PEMBAHASAN

4.1. Implementasi

Pada bab ini akan dijelaskan mengenai aplikasi yang telah dirancang ke-dalam bahasa pemograman dan tampilan antar muka. Dalam tahapan implementasi, terdapat beberapa spesifikasi *hardware* dan *software* yang harus dipenuhi untuk menginstal aplikasi *Food Delivery Order Restaurant.* Bahasa pemograman yang digunakan adalah bahasa *Java* dengan bantuan *Firebase database*. Adapun spesifikasi yang dibutuhkan untuk menjalankan aplikasi *Food Delivery Order Restaurant* sebagai berikut:

1. Android OS
2. Versi android minimal *Lollipop* (5.0)
3. *Processor 1.5 GHZ*
4. *RAM memory 1GB*
5. *Storage memory 100 MB*

4.2. Implementasi *Interface*

Implementasi *interface* aplikasi yang telah dirancang dari perencanaan rancangan sebelumnya merupakan wujud desain yang ditampilkan pada aplikasi *Food Delivery Order Restaurant.* Adapun tampilan *interface* sebagai berikut :

4.2.1. Splash Screen Interface

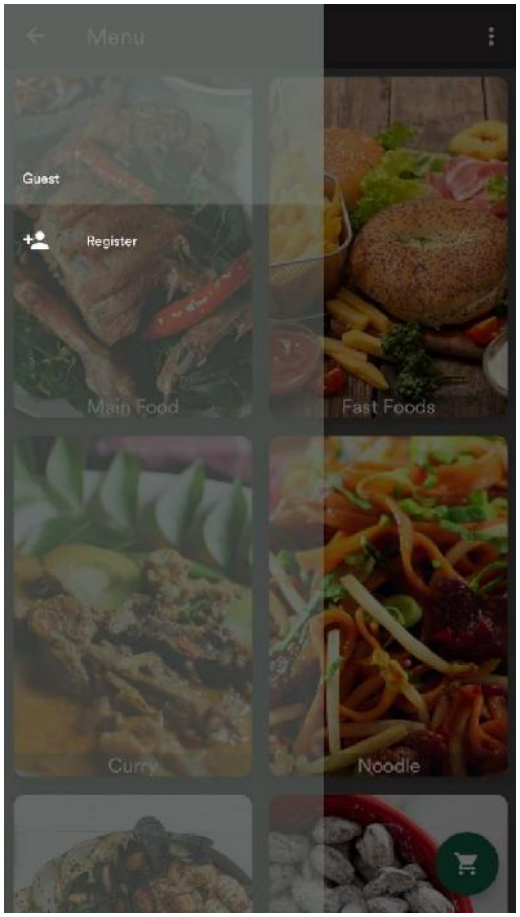
Splash screen merupakan tampilan awal yang berupa antarmuka yang hanya muncul selama beberapa detik untuk memberikan kesan yang baik terhadap aplikasi. Splash screen akan muncul ketika *customer* membuka aplikasi pertama kali.

Gambar 4.1. *Splash Screen* aplikasi *Food Order Delivery Restaurant*

4.2.2. Tampilan *Home Interface*

Halaman *home* merupakan halaman dimana pelanggan dapat melihat-lihat menu yang tersedia di dalam aplikasi sebelum melakukan pemesanan, namun menu bersifat *read only* atau tidak bisa melakukan pemesanan. Pemesanan dapat dilakukan setelah melakukan *Sign In.*

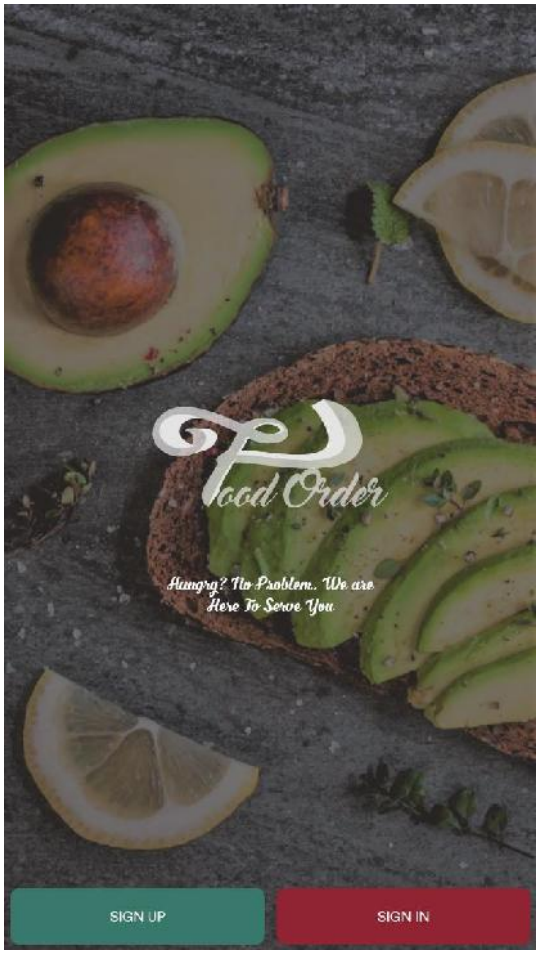
Gambar 4.2. Tampilan *home* aplikasi Food Delivery Order Restaurant



Gambar 4.3. Tombol *Register* sebelum *Sign In*

4.2.3. Tampilan Halaman *Sign In* dan *Register*

Pada halaman ini *customer* dihadapkan pada dua pilihan tombol *sign in* bagi *customer* yang telah memiliki akun, atau tombol *sign up* bagi *customer* yang belum memiliki akun agar dapat membuat akun baru.



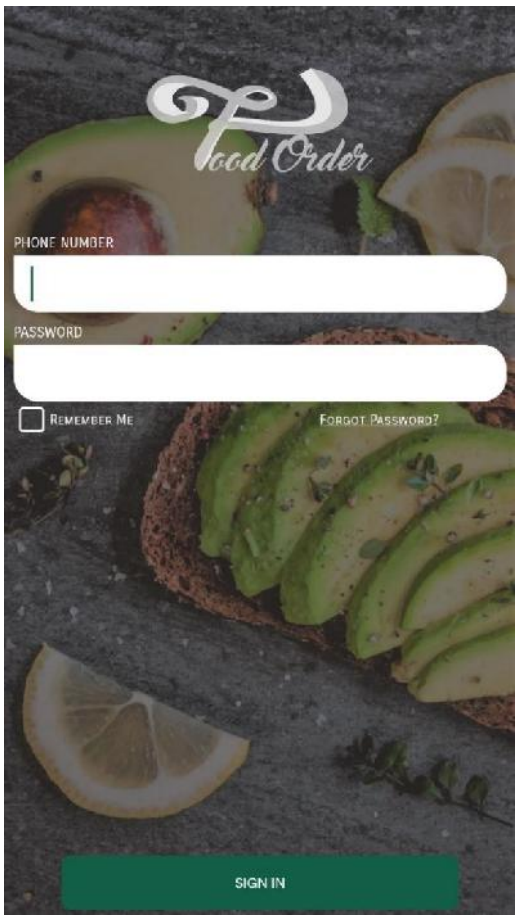
Gambar 4.4. Halaman *Sign In* dan *Register*

1. *Form Sign In*

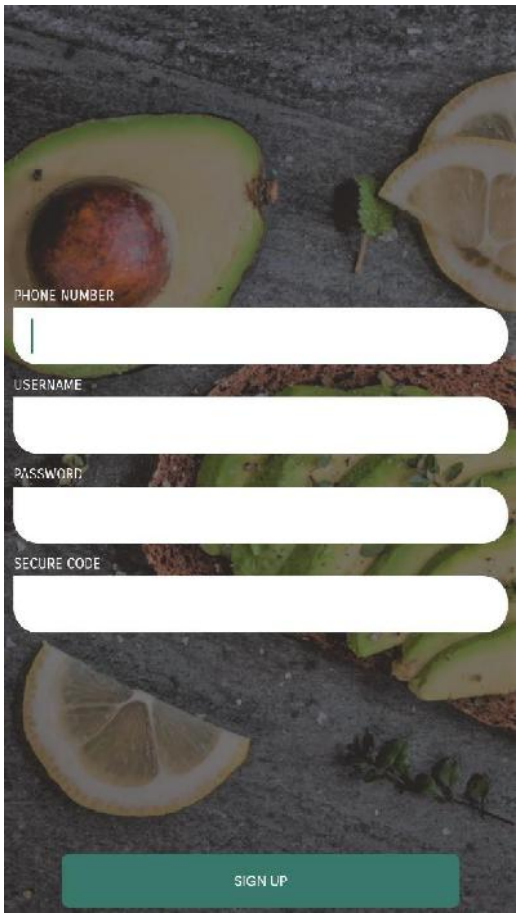
Customer dapat memasukkan data pada form *Sign In* berupa nomor handphone dan password dengan centang remember me untuk autentifikasi.

2. *Form Sign Up*

Form Sign Up merupakan interface dari proses pengisian data pada saat *customer* ingin mendaftarkan akun baru. *Customer* diharuskan mengisi nomor *handphone, username, password,* dan *security code. Security code* digunakan ketika customer ingin mengubah *password.*



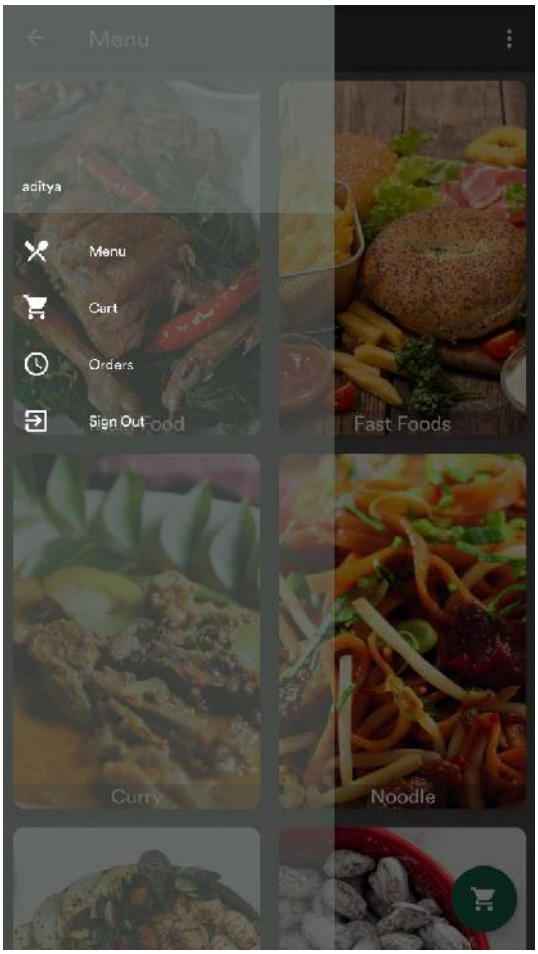
Gambar 4.5. *Form Sign In* aplikasi *Food Delivery Order Restaurant.*



Gambar 4.6. *Form Sign Up* aplikasi *Food Delivery Order Restaurant.*

4.2.4. Tampilan Informasi Akun dan *Menu Bar*

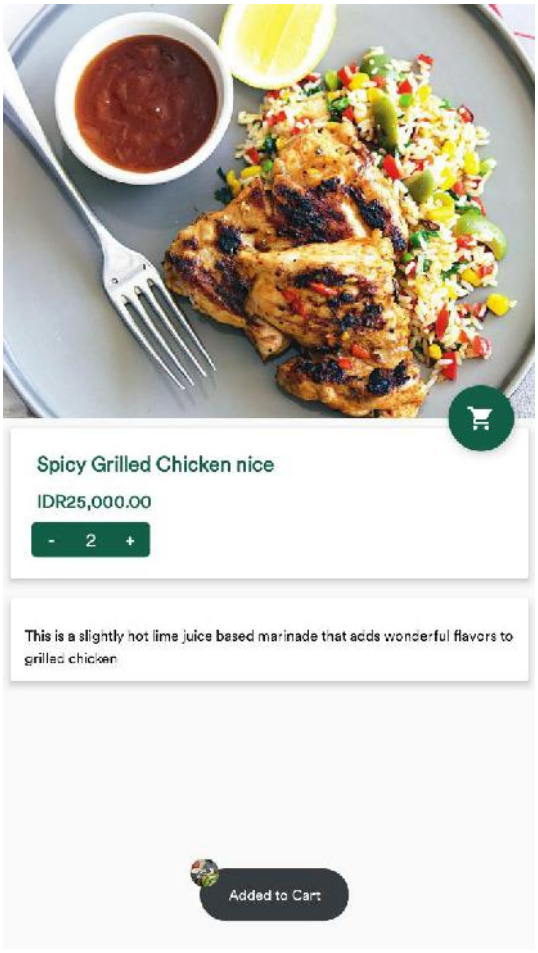
Setelah *customer* berhasil melakukan *login*, maka akan muncul *navigation drawer* yang memuat *username customer* dan menu lainnya. Sebagai penunjuk bahwa *customer* telah dapat melakukan pemesanan.



Gambar 4.7. Tampilan informasi akun dan *navigation drawer* menu

4.2.5. Tampilan *Detail* Menu

Pada tampilan *detail* menu akan ditampilkan gambar menu, tombol untuk menentukan jumlah pesanan, dan deskripsi menu sehingga *customer* dapat menentukan seleranya untuk memilih.



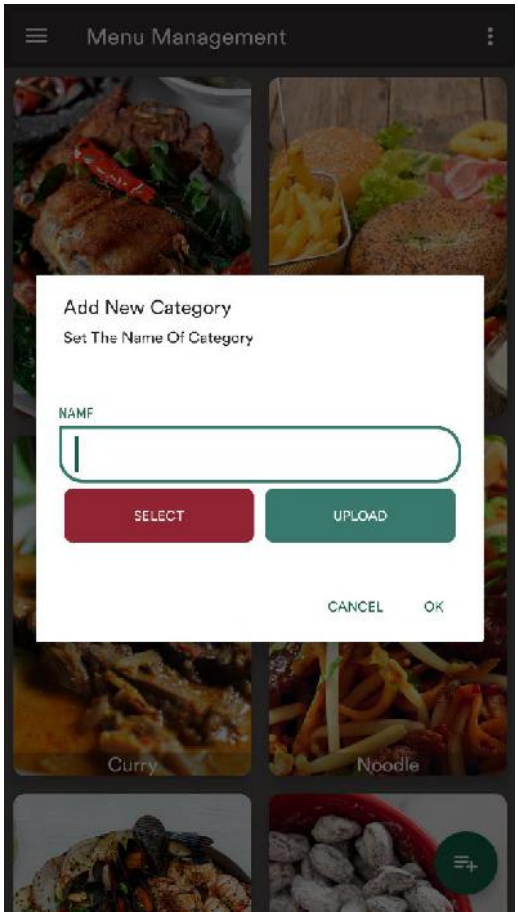
Gambar 4.8. Tampilan *detail* menu

4.2.6. Tampilan Awal dan Tambah Kategori Menu pada Aplikasi *Server*

Fitur tambah kategori menu hanya dapat dilakukan oleh admin yang masuk melalui aplikasi server dan hak akses dari admin. Berikut tampilan untuk aplikasi server dan *form* tambah kategori menu.

Gambar 4.9. Tampilan *splash screen* aplikasi *server*

Gambar 4.10. Tampilan *Sign In* aplikasi *server*

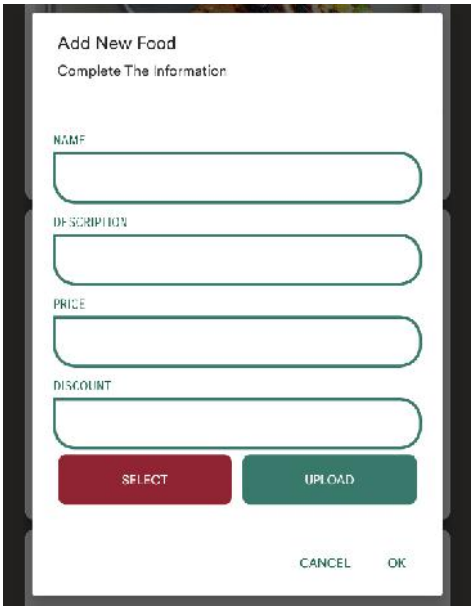


Gambar 4.11. Tampilan *form* tambah kategori menu aplikasi *server*

4.2.7. Tampilan Tambah Menu Makanan

Admin dapat menambah menu makanan dengan elemen-elemen yang dibutuhkan seperti gambar makanan yang dapat di-*upload* dari direktori lokal, menambah harga, nama menu, dan deskripsi menu makanan.

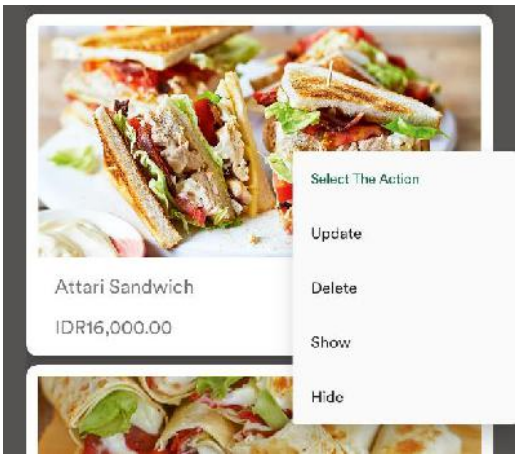
Ada dua tombol yang dapat menjalankan aksi pada form ini, yaitu tombol *select* yang berfungsi sebagai pengambil gambar dan tombol *upload* yang menjalankan aksi upload kedalan database setelah data dimasukkan.



Gambar 4.12. Tampilan *form* tambah menu aplikasi *server*

4.2.8. Tampilan *Action Box* pada Menu

Action Box merupakan panel yang berisikan sekumpulan aksi yang dapat dijalankan admin terhadap menu makanan yang terdapat dalam aplikasi *Food Delivery Order Restaurant*.

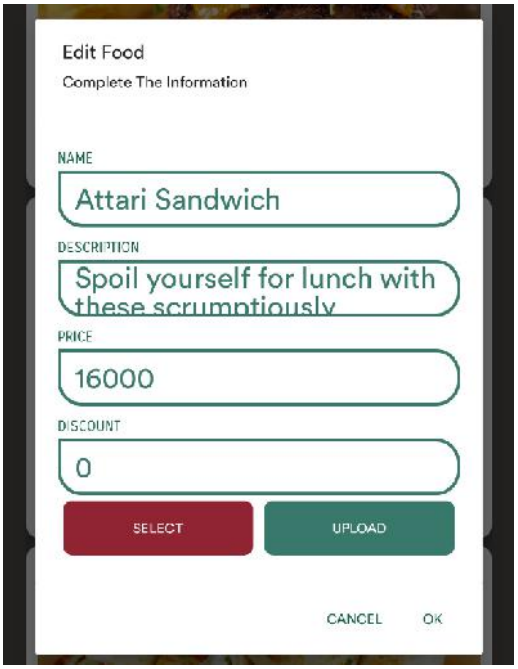


Gambar 4.13. Tampilan *Action box*

Adapun aksi yang dapat dijalankan admin adalah sebagai berikut:

1. *Update*

Admin dapat melakukan *update* data jika ada data menu yang perlu untuk diperbaharui. Ada dua tombol yang dapat menjalankan aksi pada *form* ini, yaitu tombol *select* yang berfungsi sebagai pengambil gambar dan tombol *update* yang menjalankan aksi pembaharuan data kedalan *database* setelah data dimasukkan.



Gambar 4.14. Tampilan *form update*

2. *Delete*

Fungsi *delete* pada menu makanan dapat dilakukan admin jika merasa menu makanan tersebut sudah tidak diperlukan lagi, dan akan langsung terhapus dari *database.*

3. *Show*

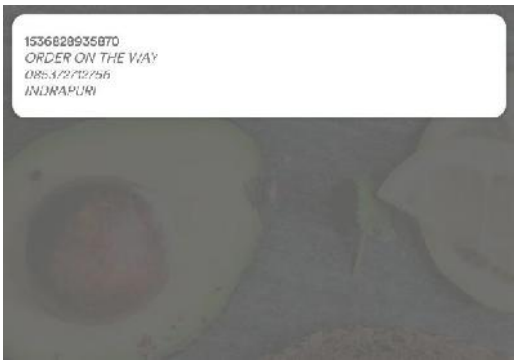
Fungsi *show* pada menu makana berfungsi sebagai aksi apabila makanan tersedia maka makanan akan ditampilkan pada list menu makanan.

4. *Hide*

Fungsi *hide* pada menu makanan berfungsi sebagai aksi untuk menyembunyikan makanan pada aplikasi customer, hal ini dilakukan apabila stok makanan tidak tersedia.

4.2.9. Tampilan *Order* Pelanggan dan *Status Action*

Setiap pesanan yang di-pesan pelanggan akan muncul pada halaman *order*, dan *Admin* dapat menentukan status pesanan yang ada, secara *realtime* akan di update ke pelanggan. Pelanggan dapat memantau pesanan melalui status dari pesanan tersebut.



Gambar 4.15. Tampilan *list order*

Gambar 4.16**.** Tampilan *action status*

4.3. Pemograman

Pada tahap pemograman aplikasi *Food Delivery Order Restaurant* menggunakan bahasa pemograman *Java* yang membantu dalam proses penciptaan program, sedangkan aplikasin pemograman yang digunakan yaitu *Android Studio* yang dapat menjalankan program berbasis bahasa *Java* dan *Kotlin*. Aplikasi *Android Studio* dibantu oleh *JDK (Java Development Kit)* yang berfungsi sebagai penghubung bahasa pemograman *Java*. *Database* yang di integrasikan menggunakan *Firebase* berbasis *online database* yang membutuhkan koneksi jaringan untuk dapat terhubung.

Emulator Android yang digunakan yaitu *Handphone Xiaomi Mi A1* berbasis android 10 sebagai media yang menjalankan aplikasi *Food Delivery Order Restaurant.* Sedangkan untuk desain *UI (User Interface)* menggunakan aplikasi *Corel Draw* 2018 sebagai aplikasi desain grafis yang dibutuhkan oleh aplikasi *Food Delivery Order Restaurant.*

4.4. Pengujian Sistem

Setiap program menjalani pengujian secara pribadi untuk memastikan bahwa program yang telah kita buat bisa bebas dari kesalahan (*bug*), walaupun tidak menutup kemungkinan masih terjadi sedikit *bug* atau tidak 100% bebas dari *bug*, namun pengujian ini setidaknya bisa meminimalisasi kesalahan yang akan terjadi. Pada tahap ini, penulis menggunakan metode pengujian unit dengan pendekatan *black-box testing*.

Pengujian unit yaitu pengujian secara individual terhadap semua program untuk memastikan bahwa program bebas dari kesalahan. Jika terjadi kesalahan, pemakai akan berusaha mencari penyebabnya dan proses untuk melakukan pencarian kesalahan ini dikenal dengan *debugging*. (Abdul Kadir, 2002:410)

Pengujian dilakukan untuk memastikan apakah aplikasi berjalan sesuai dengan yang diharapkan atau masih ada kekurangan yang harus diperbaiki. Hal-hal yang diperhatikan dalam pengujian yaitu :

a. Fungsi-fungsi yang tidak benar, baik *Input* mauput *Output.*
b. Kesalahan *Interface.*
c. Kesalahan dalam struktur data atau akses *Database.*

Dibawah ini merupakan tabel hasil pengujian aplikasi *Food Delivery Order Restaurant* menggunakan metode *black-box testing* :

Tabel 4.1. Hasil Pengujian dengan Pendekatan *Black-Box Testing*

No	Rancangan Proses	Hasil Yang Diharapkan	Hasil	Keterangan
1	Klik tombol *Cart* sebelum *Sign In*	Tidak masuk ke pemesanan dan muncul *toast "Sign In First"*	Sesuai	
2	Klik tombol *Register* pada *Navigation Drawer*	Masuk ke halaman tombol *Sign In* dan *Register*	Sesuai	
3	Mengisi *form Sign In* dan klik tombol *Sign In*	Masuk ke halaman Utama/ *Home*	Sesuai	
4	Mengisi *form Sign Up* dan klik tombol *Sign Up*	Akun terdafdar di *database*	Sesuai	
5	Klik *Forgot Password*	Masuk ke halaman *Security Code*	Sesuai	
6	Klik tombol menu pada *Navigtion Drawer*	Masuk ke halaman kategori menu	Sesuai	
7	Klik tombol *Cart*	Masuk ke halaman keranjang	Sesuai	
8	Klik tombol *Orders*	Masuk ke halaman pesanan	Sesuai	
9	Klik kategori makanan	Masuk ke halaman *list* makanan	Sesuai	
10	Klik list makanan	Masuk ke halaman detail makanan	Sesuai	
11	Klik tombol jumlah pesanan dan tombol *Cart*	Jumlah pesanan tertambah kedalam keranjang pesanan	Sesuai	

12	Klik tombol *Cart* setelah memesan	Jumlah total bayar dan jumlah pesanan muncul	Sesuai	
13	Klik tombol *Sign Out*	Keluar akun dan masuk ke halaman *login*	Sesuai	
14	Memasukkan data dan klik tombol *login admin*	Masuk ke halaman utama sebagai *admin*	Sesuai	
15	Klik tombol tambah kategori menu	Menampilkan *form Add New Category*	Sesuai	
16	Klik tombol Select pada *form Add New Category*	Masuk ke direktori lokal *Handphone*	Sesuai	
17	Klik tombol *Upload* pada form *Add New Category*	Menambahkan masukan data kategori baru ke *database*	Sesuai	
18	Klik tombol *OK* pada *form Add New Category*	Submit perubahan pada *form Add New Category*	Sesuai	
19	Klik tombol *Cancel* pada *form Add New Category*	Membatalkan perubahan dan menutup *form Add New Category*	Sesuai	
20	Klik tombol tambah menu	Menampilkan *form Add New Food*	Sesuai	
21	Klik tombol *Select* pada *form Add New Food*	Masuk ke direktori lokal *Handphone*	Sesuai	
22	Klik tombol *Upload* pada *form Add New Food*	Menambahkan masukan data menu baru ke *database*	Sesuai	
23	Klik tombol *OK* pada form *Add New Food*	Submit perubahan pada *form Add New Food*	Sesuai	
24	Klik tombol *Cancel* pada *form Add New Food*	Membatalkan perubahan dan menutup *form Add New Food*	Sesuai	

25	Klik tahan menu makanan	Menampilkan *Action Box*	Sesuai	
26	Klik *Action Update* pada *Action Box*	Menampilkan *form Edit Food*	Sesuai	
27	Klik tombol *Select* pada *form Edit Food*	Masuk ke direktori lokal *Handphone*	Sesuai	
28	Klik tombol *Upload* pada *form Edit Food*	Memperbarui data menu ke *database*	Sesuai	
29	Klik tombol *OK* pada *form Edit Food*	*Submit* perubahan pada *form Edit Food*	Sesuai	
30	Klik tombol *Cancel* pada *form Edit Food*	Membatalkan perubahan dan menutup *form Edit Food*	Sesuai	
31	Klik *Action Delete* pada *Action Box*	Menghapus data menu dari tampilan aplikasi dan *database*	Sesuai	
32	Klik *Action Hide* pada *Action Box*	Menyembunyikan data menu dari tampilan aplikasi	Sesuai	
33	Klik *Action Show* pada *Action Box*	Menampilkan kembali data menu yang telah disembunyikan dari aplikasi	Sesuai	
34	Klik tombol *Orders* pada *Navigation Drawer* admin	Menampilkan *list* pesanan pelanggan	Sesuai	
35	Klik tahan *list* pesanan pelanggan	Menampilkan *action box Status*	Sesuai	
36	Klik *Status* pada *Action Box list* pesanan pelanggan	Mengubah *Status list* pesanan pelanggan	Sesuai	

BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan pembahasan dan kajian yang dilakukan terhadap aplikasi *Food Delivery Order Restaurant*, dapat disimpulkan bahwa dengan adanya aplikasi *Food Delivery Order Restaurant* akan mempermudah pemesanan yang mulanya dilakukan secara manual menjadi pemesanan yang lebih efektif.

Kesimpulan yang dapat kita ambil dari aplikasi *Food Delivery Order Restaurant* adalah sebagai berikut:

1. Aplikasi *Food Delivery Order Restaurant* ini dapat dijalankan dan di implementasikan pada *smartphone* dengan operasi sistem android 5.0 ke atas.
2. Kapasitas yang dibutuhkan dalam penggunaan aplikasi *Food Delivery Order Restaurant* adalah sebesar 56 MB.
3. Aplikasi *Food Delivery Order Restaurant* membutuhkan koneksi internet untuk dapat terhubung ke *Firebase database.*
4. Aplikasi dapat mengatasi pemesanan makanan dan minuman dengan *platform* android berbasis *client server*.
5. Aplikasi mobile android ini bisa memberitahu pesanan pelanggan kebagian admin *restaurant*.
6. Aplikasi pemesanan makanan yang dibuat ini juga dapat mempermudah pelayan dalam melakukan penginputan pemesanan.

7. Hasil aplikasi masih dalam bentuk apk dimana proses upgrade/pembaruan masih digunakan secara manual belum bisa di gunakan secara online.

5.2. Saran

Setelah selesainya perancangan dan implementasi dari aplikasi *Food Delivery Order Restaurant* dan berdasarkan simpulan dan analisis yang telah dilakukan, maka ada beberapa saran yang dapat penulis sampaikan berkaitan dengan aplikasi *Food Delivery Order Restaurant* yaitu:

1. Aplikasi dapat dikembangkan dengan pengembangan fitur pembayaran *E-money* dan pembayaran menggunakan kerjasama dengan *Bank*.
2. Aplikasi *server* diharapkan dapat dikembangkan ke aplikasi desktop agar dapat memudahkan admin dalam mengelola aplikasi.
3. Menambah hak akses pada struktur database untuk bagian dapur.
4. Pencatatan keuangan pada aplikasi dapat tambahkan laporan keuangan dan rekap data aplikasi.
5. Disarankan agar aplikasi dapat terintegrasi dengan *website* menggunakan *API.*

DAFTAR PUSTAKA

Arifianto, Teguh. (2011). *Membuat Interface Aplikasi Android Lebih Keren dengan LWUIT*.Yogyakarta: Andi Publisher.

Davis, Gordon. (1999). *Kerangka Dasar Sistem Informasi Managemen Bagian I***.** Jakarta: PT Ikrar Mandiri Abadi.

Friesen, Jeff. (2011). *Learn Java for Android Development*. United States of America: Apress.

Hariman, A. S. (2002). *Visual Modelling Menggunakan UML dan Rational Rose*, Bandung: Penerbit Informatika.

Hartono, Jogiyanto. (2005). *Sistem Teknologi Informasi Edisi 3*. Yogyakarta: Penerbit Andi.

Hermawan, S. Stephanus. (2011). *Mudah Membuat Aplikasi Android*. Yogyakarta: Andi Offset.

Indrajit, E. R. (2001). *Manajemen Sistem Informasi Dan Teknologi Informasi*. Jakarta: Elex Media Komputindo.

Jogiyanto, H. M. (1989). *Analisa dan Desain Sistem Informasi*. Yogyakarta: Penerbit Andi Offset.

Jogiyanto, H. M. (1990). *Pengenalan Komputer*. Yogyakarta: Penerbit Andi Offset.

Kadir, Abdul. (1999). *Konsep Dan Tuntunan Praktis Basis Data*. Yogyakarta: Penerbit Andi.

Kadir, Abdul.. (2003). *Pengenalan Sistem Informasi*. Yogyakarta: Penerbit Andi.

Kelompok Gramedia, & Anggota IKAPI. (2015). *Kumpulan Aplikasi Java*. Jakarta: PT Elexmedia Komputindo.

Kirstanto, Harianto. (2004). *Konsep dan Perancangan Database*. Yogyakarta: Penerbit Andi.

Safaat, Nazruddin. (2011). *Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android*. Bandung: Informatika.

JURNAL :

Aryza, S., Irwanto, M., Lubis, Z., Siahaan, A. P. U., Rahim, R., & Furqan, M. (2018). A Novelty Design Of Minimization Of Electrical Losses In A Vector Controlled Induction Machine Drive. In IOP Conference Series: Materials Science And Engineering (Vol. 300, No. 1, P. 012067). IOP Publishing.

Aziz, Abdul. (2015). *Perancangan Aplikasi Knowledge Base System untuk Instruksi Kerja Berbasis Android di PT. Kyowa Indonesia.* Teknik. Jakarta: Universitas Darma Persada.

Batubara, Supina. "Analisis perbandingan metode fuzzy mamdani dan fuzzy sugeno untuk penentuan kualitas cor beton instan." IT Journal Research and Development 2.1 (2017): 1-11.

Busran, dan Wina Anggraini. (2016). *Perancangan Aplikasi Pemesanan Makanan dan Minuman Berbasis Sistem Operasi Android pada Pecel Lele Lela.* Teknologi Industri. Padang: Institut Teknologi Padang.

Fitriani, W., Rahim, R., Oktaviana, B., & Siahaan, A. P. U. (2017). Vernam Encypted Text in End of File Hiding Steganography Technique. Int. J. Recent Trends Eng. Res, 3(7), 214-219.

Hamdani, H., Tharo, Z., & Anisah, S. (2019, May). Perbandingan Performansi Pembangkit Listrik Tenaga Surya Antara Daerah Pegunungan Dengan Daerah Pesisir. In Seminar Nasional Teknik (Semnastek) Uisu (Vol. 2, No. 1, Pp. 190-195).

Hariyanto, E., Lubis, S. A., & Sitorus, Z. (2017). Perancangan prototipe helm pengukur kualitas udara. KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer), 1(1).

Iqbal, M., Siahaan, A. P. U., Purba, N. E., & Purwanto, D. (2017). Prim's Algorithm for Optimizing Fiber Optic Trajectory Planning. Int. J. Sci. Res. Sci. Technol, 3(6), 504-509.

Laila, Nur, dan Sri Wahyuni. (2011). *Sistem Informasi Pengolahan Data Inventory Pada Toko Buku Studi Cv. Aneka Ilmu Semarang.* Teknik Elektro. Semarang: Universitas Negeri Semarang (UNNES).

Muttaqin, Muhammad. "Analisa Pemanfaatan Sistem Informasi E-Office Pada Universitas Pembangunan Panca Budi Medan Dengan Menggunakan Metode Utaut." Jurnal Teknik dan Informatika 5.1 (2018): 40-43.

Putra, Rizky Maulana. (2015). *Perancangan Aplikasi Kamus Bahasa Minang Berbasis Android*. Manajemen Informatika. Batusangkar: Sekolah Tinggi Agama Islam Negeri (STAIN).

Putra, Yudi Ismail. (2017). *Rancang Bangun Sistem Informasi Akademik pada Yayasan Perguruan SMP Abadi Bangsa Binjai*. D3 Teknik Komputer. Medan: Universitas Pembangunan Pancabudi.

Rahim, R., Aryza, S., Wibowo, P., Harahap, A. K. Z., Suleman, A. R., Sihombing, E. E., ... & Agustina, I. (2018). Prototype File Transfer Protocol Application For LAN And Wi-Fi Communication. Int. J. Eng. Technol., 7(2.13), 345-347.

Rahmaniar, R. (2019). Model flash-nr Pada Analisis Sistem Tenaga Listrik (Doctoral Dissertation, Universitas Negeri Padang).

Rossanty, Y., Aryza, S., Nasution, M. D. T. P., & Siahaan, A. P. U. (2018). Design Service Of QFC And SPC Methods In The Process Performance Potential Gain And Customers Value In A Company. Int. J. Civ. Eng. Technol, 9(6), 820-829.

Siagian, P., & Fahreza, F. (2020, February). Rekayasa Penanggulangan Fluktuasi Daya Pembangkit Listrik Tenaga Angin Dengan Vehicle To Grid (V2G). In Seminar Nasional Teknologi Komputer & Sains (SAINTEKS) (Vol. 1, No. 1, Pp. 356-361).

Siagian, P., Syafruddin, H. S., & Tharo, Z. (2020, September). Pengaruh Tekanan Terhadap Inception Partial Discharge Pada Bahan Dielektrik Komposit Dan Non-Komposit. In Seminar Nasional Teknik (SEMNASTEK) UISU (Vol. 3, No. 1, Pp. 134-141).

Siahaan, A. P. U., Ikhwan, A., & Aryza, S. (2018). A Novelty Of Data Mining For Promoting Education Based On FP-Growth Algorithm

Sulistyorini, Prastuti. (2009). *Pemodelan Visual dengan Menggunakan UML dan Rational Rose*. Teknik Informatika. Pekalongan: STMIK Widya Pratama.

Tarigan, A. D., & Pulungan, R. (2018). Pengaruh Pemakaian Beban Tidak Seimbang Terhadap Umur Peralatan Listrik. RELE (Rekayasa Elektrikal Dan Energi): Jurnal Teknik Elektro, 1(1), 10-15.

Wibowo, P., Lubis, S. A., & Hamdani, Z. T. (2017). Smart Home Security System Design Sensor Based On Pir And Microcontroller. International Journal Of Global Sustainability, 1(1), 67-73.

INTERNET :

Android.com. (2020, 9 Februari). *Sejarah Android*. Diakses pada 10 Februari 2020, dari https://www.android.com/intl/id_id/history/

Developer.Android.Com. (2020, 10 Februari). *Mengenal Android Studio.* Diakses pada 11 Februari 2020, dari https://developer.android.com/studio/intro?hl=id

Firebase.Google.Com. (2020, 10 Februari). *Firebase Realtime Database.* Diakses pada 11 Februari 2020, dari https://firebase.google.com/docs/database?hl=id