PERNYATAAN

Saya yang bertanda tangan dibawah ini, menyatakan bahwa skripsi yang saya buat ini adalah hasil karya saya sendiri, tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu perguruan tinggi atau karya yang pernah ditulis atau diterbitkan orang lain kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka. Karya ini adalah milik saya dan pertanggung jawaban sepenuhnya berada dipundak sayan.

Bandar Lampung,27 Agustus 2018

Pikri Pit Pratama
NPM 1411010086

HALAMAN PERSETUJUAN

Judul Skripsi : CUSTOMISABLE NOTIFICATION DALAM LAYANAN PELANGGAN BERBASIS ANDROID (STUDI KASUS : DEALER AUTO 2000)

Nama : Pikri Pit Pratama

NPM : 1411010086

Jurusan : S1 Teknik Informatika

Menyetujui :

Pembimbing	Ketua Jurusan Teknik Informatika
Hariyanto Wibowo, S.Kom., M.T.I	Yuni Arkhiansyah, M.Kom
NIK.00051297	NIK.00480802

HALAMAN PENGESAHAN

Telah di uji dan dipertahankan didepan tim penguji Skripsi Program Studi Teknik Informatika (TI) IIB Darmajaya Bandar Lampung dan dinyatakan diterima untuk memenuhi syarat guna memperoleh gelar Sarjana Komputer.

MENGESAHKAN

1. Tim Penguji **Tanda Tangan**

Ketua : Septilia Arfida, S.Kom., M.T.I .....

Anggota : Ketut Artaye, S.Kom., M.T.I .....

2. Dekan Fakultas Ilmu Komputer

Sriyanto,S.Kom.,MM

NIK.00210800

ABSTRACT

ANDROID-BASED CUSTOMISABLE NOTIFICATION FOR CUSTOMER SERVICES (A Case Study in AUTO 2000)

By

PIKRI PIT PRATAMA
1411010086

The 2000 Auto Dealer is one of the official Toyota dealers that provide the service cars, vehicle sales, and original spare parts sales. The problem statement of this research was that there was a downward trend in the operational activities viewed from the conventional processes that were still committed, the employees who did not implement the mobile-based technology used in the car services, and the customers who needed to come directly to the location. To solve this problem, a customisable notification for the customer services was needed to design. The software development method used in this research was the Prototype method with several stages i.e., gathering the needs, designing the system, and evaluating the prototypes. This application was used to create better service to the customers in Bandar Lampung. The Android-based customisable notification for the customer service was facilitated with several features i.e., notification features, booking services, catalogs, trading, and promotion.

Keywords: Notifications, Periodical Service, Android.

DAFTAR ISI

Halaman

BAB III METODOLOGI PENELITIAN

BAB IV HASIL PENELITIAN DAN PEMBAHASAN

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Jaman modern ini pertumbuhan dan tingkat konsumsi terhadap informasi semakin banyak dan cepat. Informasi seperti sudah menjadi keharusan setiap individu untuk mengetahui perkembangan pada saat ini. Sehingga untuk mendapatkan informasi secara cepat maka keinginan untuk mendapatkan informasi dengan instant jauh lebih tinggi. Akan tetapi setiap individu memiliki minat akan informasi yang berbeda beda.

Seiring dengan perkembangan zaman, kita telah memasuki era dimana mobile device merupakan kebutuhan yang wajib dimiliki oleh setiap orang untuk melakukan berbagai hal seperti komunikasi, bertukar informasi dan melakukan hal-hal lainnya. Salah satu sistem operasi perangkat seluler yang banyak digunakan dan perkembangannya sangat pesat saat ini adalah android. Hal ini dikarenakan teknologinya yang open source sehingga mendapatkan banyak dukungan dari berbagai teknologi lainnya.

Auto 2000 Way Halim adalah salah satu dealer Toyota yang berada di Lampung, Dealer toyota ini berlokasi di Jl. Lintas Timur Sumatera, Waydadi, Sukarame, Kota Bandar Lampung. Dealer Auto 2000 merupakan salah satu dealer resmi Toyota untuk melakukan servis kendaraan mobil dan dealer tersebut selain melayani servis kendaraan mobil juga menyediakan penjualan mobil dan pemasangan berbagai spare part asli kendaraan mobil yang dibutuhkan oleh pelanggan. Pemilik dealer, pegawai dealer yang ramah serta mekanik-mekanik yang handal dalam melayani pelanggan menjadi salah satu keunggulan dealer Auto 2000. Berdasarkan analisis yang dilakukan didapatkan beberapa kekurangan yang terdapat dalam kegiatan operasional perusahaan ini masih menggunakan proses konvensional dan belum menerapkan teknologi *mobile*.yaitu pada saat pelangggan ingin mengetahui kapan harus service berkala dilakukan, pelanggan harus datang langsung kelokasi perusahaan. Untuk mengetahui lebih jelas apa yang menjadi masalah dalam penelitian ini dilakukan wawancara dengan pegawai

dealer dan beberapa mekanik dealer, kebanyakan pelanggan yang umum nya sering menghiraukan service berkala dan sering terlambat atau tidak tepat waktu dalam melakukan servis berkala.

sebuah penerapan teknologi *Firebase cloud messaging* ini diharapkan dapat mempermudah pelanggan tidak menghiraukan atau tidak tepat waktu saat melakukan service berkala kendaraanya, karena dengan pelayanan yang baik akan memberikan keuntungan tersendiri bagi dealer Toyota. akan menciptakan kepuasan pelanggan yang dapat mendorong pada tingkat kepercayaan dan loyalitas pelanggan terhadap suatu produk secara tidak langsung kepada dealer Toyota.

1.2 Rumusan Masalah

Berdasarkan latar belakang terdapat masalah yang dihadapi yaitu bagaimana mengimplementasikan costumisable notification dalam layanan kepada pelanggan dengan memanfaatkan teknologi *firebase cloud messaging* untuk menyampaikan pelayanan kepada pelanggan.

1.3 Batasan Masalah

Pada customisable notifikasi dalam layanan kepada pelangan berbasis Android memiliki batasan masalah diantaranya :

1. Objek yang diteliti yaitu Dealer Toyota Auto 2000 Way Halim.
2. Batasan masalah hanya mencakup tentang notifikasi.
3. Batasan masalah mencakup tentang pelayanan service berkala kendaraan yang ternotifikasi dengan baik di Android.
4. Sistem yang akan dibangun menggunakan *firebase cloud messaging* dan Android Studio.

1.4 Tujuan Penelitian

Tujuan yang akan dicapai dengan melakukan penelitian ini adalah membuat sebuah notifikasi menggunakan *firebase cloud messaging* sebagai media penyampaian informasi mengenai pelayanan di dealer Toyota.

1.5 Manfaat Penelitian

Manfaat penelitian ini adalah agar terciptanya pelayanan yang baik kepada customer dealer Toyota Auto 2000 dengan memanfaatkan teknologi yang ada.

1.6 Sistematika Penulisan

Uraian singkat mengenai sistematika penulisan pada masing-masing bab adalah sebagai berikut :

BAB I : PENDAHULUAN

Pada bab ini berisi tentang latar belakang, perumusan masalah, ruang lingkup penelitian, tujuan dan manfaat penelitian, dan sistematika penulisan.

BAB II : LANDASAN TEORI

Pada bab ini berisi tentang teori-teori yang mendukung penelitian yang akan dilakukan oleh penulis/peneliti.

BAB III : METODELOGI PENELITIAN

Pada bab ini berisi tentang metode-metode pendekatan penyelesaian permasalahan yang dinyatakan dalam perumusan masalah pada penelitian yang dilakukan.

BAB IV : HASIL PENELITIAN DAN PEMBAHASAN

Pada bab ini berisi tentang pemaparan hasil analisa persoalan yang dibahas dengan berpedoman pada teori-teori yang dikemukakan pada Bab II.

BAB V : SIMPULAN DAN SARAN

Pada bab ini berisi tentang rangkuman dari pembahasan, yang terdiri dari jawaban atas perumusan masalah, tujuan penelitian, dan hipotesis. Selain itu berisi tentang saran bagi perusahaan/instansi (obyek penelitian) dan saran untuk penelitian selanjutnya, sebagai hasil pemikiran penelitian atas keterbatasan penelitian yang dilakukan.

BAB II
LANDASAN TEORI

2.1 Notifikasi

Menurut (Arif Winandar,2015) Notifikasi adalah pemberitahuan mengenai informasi atau pengumuman dari pihak tertentu kepada pihak yang dituju yang dilakukan melalui media seperti email, sms, maupun aplikasi chatting (line, whatsaap, bbm, wechat, dsb.) Notifikasi sangat penting bagi penerima informasi, karena dengan notifikasi, informasi tersebut dapat langsung diketahui oleh penerima. Notifikasi diperlukan ketika informasi yang akan disampaikan kepada pihak terkait tidak memungkinkan untuk diberitahukan secara langsung atau tatap muka.

Menurut Kamus Besar Bahasa Indonesia (2010 : 1008) notifikasi adalah pemberitahuan atau kabar tentang penawaran barang dan sebagainya, sedangkan Notifikasi pada android merupakan salah satu hal penting bagi pengguna android, notifikasi bisa datang dari aplikasi atau sistem dari android, biasanya jika dari sistem android datang karena ada pembaharuan baru dari software.

2.2 Aplikasi

Aplikasi disebut juga *software*, dimana merupakan salah satu dari komponen sistem informasi. Menurut Shelly, Cashman dan Vermaat aplikasi adalah seperangkat instruksi khusus dalam komputer yang dirancang agar kita dapat menyelesaikan tugas-tugas tertentu.

2.3 Aplikasi Mobile

Menurut (Ramadhan,2014) Aplikasi mobile dapat diartikan sebagai sebuah produk dari sistem komputasi mobile, yaitu sistem komputasi yang dapat dengan mudah dipindahkan secara fisik dan yang komputasi kemampuan dapat digunakan saat mereka sedang dipindahkan. Contohnya adalah personal digital assistant (PDA), smartphone dan ponsel (Reza B'Far, 2005:3).

Berdasarkan jenisnya, Brian Fling (2009:70) membagi aplikasi mobile menjadi beberapa kelompok yaitu:

a. Short Message Service (SMS) Merupakan aplikasi mobile paling sederhana, dirancang untuk berkirim pesan dan berguna ketika terintegrasi dengan jenis aplikasi mobile lainnya.

b. Mobile Websites (Situs WebMobile) Merupakan situs web yang dirancang khusus untuk perangkat mobile. Situs web mobile sering memiliki desain yang sederhana dan biasanya bersifat memberikan informasi.

c. Mobile Web Application (Aplikasi Web Mobile) Aplikasi web mobile merupakan aplikasi mobile yang tidak perlu diinstal atau dikompilasi pada perangkat target. Menggunakan XHTML, CSS, dan JavaScript, aplikasi ini mampu memberikan pengguna pengalaman layaknya aplikasi native/asli.

d. Native Application (Aplikasi Asli) Merupakan aplikasi mobile yang harus diinstal pada perangkat target. Aplikasi ini dapat disebut aplikasi platform, karena aplikasi ini harus dikembangkan dan disusun untuk setiap platform mobile secara khusus.

2.4 FCM (Firebase Cloud Messaging)

Menurut web resminya Firebase Cloud Messaging (FCM) adalah solusi perpesanan lintas-platform yang memungkinkan mengirimkan pesan dan pemberitahuan dengan terpercaya tanpa biaya. Dengan menggunakan FCM, Anda bisa memberi tahu aplikasi klien bahwa email baru atau data lainnya tersedia untuk disinkronkan. Mengirim pemberitahuan untuk mendorong pelibatan kembali dan retensi pengguna. Untuk kasus penggunaan seperti perpesanan instan, pesan dapat mentransfer payload hingga 4 KB ke aplikasi klien

Kemampuan utama dari firebase cloud messaging diantaranya :

1. Mengirim pesan notification atau pesan data
   Mengirim pesan notification yang ditampilkan kepada pengguna, atau mengirim pesan data dan menentukan sepenuhnya apa yang terjadi dalam kode aplikasi.

2. Penargetan pesan serbaguna

   Mendistribusikan pesan ke aplikasi klien dengan salah satu dari tiga cara-kesatu perangkat, ke grup perangkat, atau ke perangkat yang berlangganan topik.

3. Mengirim pesan dari aplikasi klien

   Mengirim notification, chat, dan pesan lain dari perangkat ke server melalui saluran koneksi FCM yang andal dan hemat baterai.

2.5 Sistem Operasi

Muhammad (2014: 01) menguraikan bahwa sistem operasi adalah program utama yang langsung berinteraksi dengan bahasa yang dikenal oleh mesin komputer (bahasa mesin) Sistem operasi (operating system) adalah perangkat lunak sistem yang mengatur sumber daya dari perangkat keras dan perangkat lunak, serta sebagai jurik (daemon) untuk program komputer. Tanpa sistem operasi, pengguna tidak dapat menjalankan program aplikasi pada komputer mereka, kecuali program booting. Berikut ini adalah jenis jenis sistem operasi :

a. DOS
b. WINDOWS
c. UNIX
d. Linux
e. Android

2.6 Android

Kasman (2013:2) menguraikan bahwa “Android” merupakan sebuah sistem operasi yang berbasis Linux untuk perangkat portable seperti smartphone dan komputer tablet”. Android menyediakan platform terbuka bagi programmer untuk mengembangkan aplikasi sendiri pada berbagai perangkat dengan sistem operasi Android.

Android merupakan sistem operasi untuk telepon seluler berbasis linux sebagai kernelnya. Android menyediakan platform terbuka (open source) bagi para pengembang untuk menciptakan aplikasi mereka sendiri. Awalnya, perusahaan search engine tebesar saat ini, yaitu Google Inc, membeli Android Inc, pendatang baru yang membuat perangkat lunak untuk ponsel. Android Inc. Didirikan oleh

Andy Rubin, Rich Milner, Nick Sears dan Chris White pada tahun 2003. Pada Agustus 2005 Google membeli Android Inc. Dimulai pada tahun 2005, Android Inc. dibawah naungan Google Inc. Berusaha membuat sebuah operating system mobile baru. Sejak saat itulah mulai beredar rumor bahwa Google akan melakukan ekspansi bisnis ke industri seluler. Akhirnya pada bulan September 2007 Google mengajukan hak paten atas produknya yang dinamai Nexus One.

Akhir tahun 2008, dibentuk sebuah tim kerja sama yang dinamai Open Handset Alliance (OHA). OHA ini terdiri dari beberapa produsen perangkat telekomunikasi ternama dunia, antara lain ASUS, Toshiba, Sony Ericsson (sekarang Sony), Garmin, Vodafone, dan Softbank. OHA bekerja sama untuk mengembangkan sebuah kernel Linux yang akan dijadkan sebuah program untuk perangkat seluler. Hingga akhirnya OHA berhasil dan mengumumkan produk operating system mobile yang diberi nama Android. Ponsel yang mendapat kehormatan untuk mencoba pertama kali sistem operasi Android adalah HTC Dream.

2.6.1 Perkembangan Android

Kasman(2013 :4) menguraikan bahwa Keunikan dari sistem operasi Android adalah dengan nama makanan hidangan penutup.Versi *Android* diawali dengan dirilisnya *Android beta* pada bulan November 2007. Versi komersial pertama, *Android 1.0*, dirilis pada September 2008. *Android* dikembangkan secara berkelanjutan oleh *Google* dan *Open Handset Alliance (OHA)*, yang telah merilis sejumlah pembaruan sistem operasi ini sejak dirilisnya versi awal.

Sejak April 2009, versi *Android* dikembangkan dengan nama kode yang dinamai berdasarkan makanan pencuci mulut dan penganan manis. Masing-masing versi dirilis sesuai urutan *alfabet*, yakni *Cupcake* (1.5), *Donut* (1.6), *Eclair* (2.0–2.1), *Froyo* (2.2–2.2.3), *Gingerbread* (2.3–2.3.7), *Honeycomb* (3.0–3.2.6), *Ice Cream Sandwich* (4.0–4.0.4), *Jelly Bean* (4.1–4.3), *KitKat* (4.4+), *Lollipop* (5.0+), *Marshmallow* (6.0+), hingga yang terbaru adalah *Nougat* (7.0+) dan selanjutnya versi *Android* terbaru yang ditunggu-tunggu

adalah *Android* O (8.0+), Berikut ini adalah table 2.1. yang menguraikan urutan versi *Android* dan *level API* pada *Android:*

Tabel 2.1. Urutan versi *Android*

Versi	**Nama kode**	**Tanggal rilis**	**Level API**	**Distribusi**
1.5	*Cupcake*	30 April 2009	3	
1.6	*Donut*	15 September 2009	4	
2.0–2.1	*Eclair*	26 Oktober 2009	7	
2.2	*Froyo*	20 Mei 2010	8	0,7%
2.3.3–2.3.7	*Gingerbread*	9 Februari 2011	10	11,7%
2.3–2.3.2	*Gingerbread*	6 Desember 2010	9	
3.1	*Honeycomb*	10 Mei 2011	12	
3.2	*Honeycomb*	15 Juli 2011	13	
4.0.3–4.0.4	*Ice Cream Sandwich*	16 Desember 2011	15	9,6%
4.1.x	*Jelly Bean*	9 Juli 2012	16	25,1%
4.2.x	*Jelly Bean*	13 November 2012	17	20,7%
4.3.x	*Jelly Bean*	24 Juli 2013	18	8%
4.4.x	*KitKat*	31 Oktober 2013	19	24,5%
5.x	*Lollipop*	15 Oktober 2014	21	
6.0	*Marshmallow*	19 Agustus 2015	23	
7.0	*Nougat*	22 Agustus 2016	24	Kurang dari 0.1%
8.0	*Oreo*	22 Agustus 2017	26/27	

2.7 Android Studio

Android Studio adalah Lingkungan Pengembangan Terpadu - Integrated Development Environment (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA . Selain merupakan editor kode IntelliJ dan alat pengembang yang berdaya guna, Android Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas Anda saat membuat aplikasi Android, misalnya:

1. Sistem versi berbasis Gradle yang fleksibel

2. Emulator yang cepat dan kaya fitur
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat Android
4. Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru
5. Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh
6. Alat pengujian dan kerangka kerja yang ekstensif
7. Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain
8. Dukungan C++ dan NDK
9. Dukungan bawaan untuk Google Cloud Platform, mempermudah pengintegrasian Google Cloud Messaging dan App Engine

2.8 Android SDK (*software development kit)*

Android SDK adalah tools API (*Application Programing Interface*) yang diperlukan untuk mengembangkan aplikasi pada platform android, Android-SDK merupakan tools bagi para programmer yang ingin mengembangkan aplikasi berbasis google android. Android SDK mencakup seperangkat alat pengembangan yang komprehensif. Android SDK terdiri dari debugger, libraries, handset emulator,dokumentasi, contoh kode, dan tutorial.

2.9 Json (JavaScript Object Notation)

adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (generate) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data.

2.10 Java

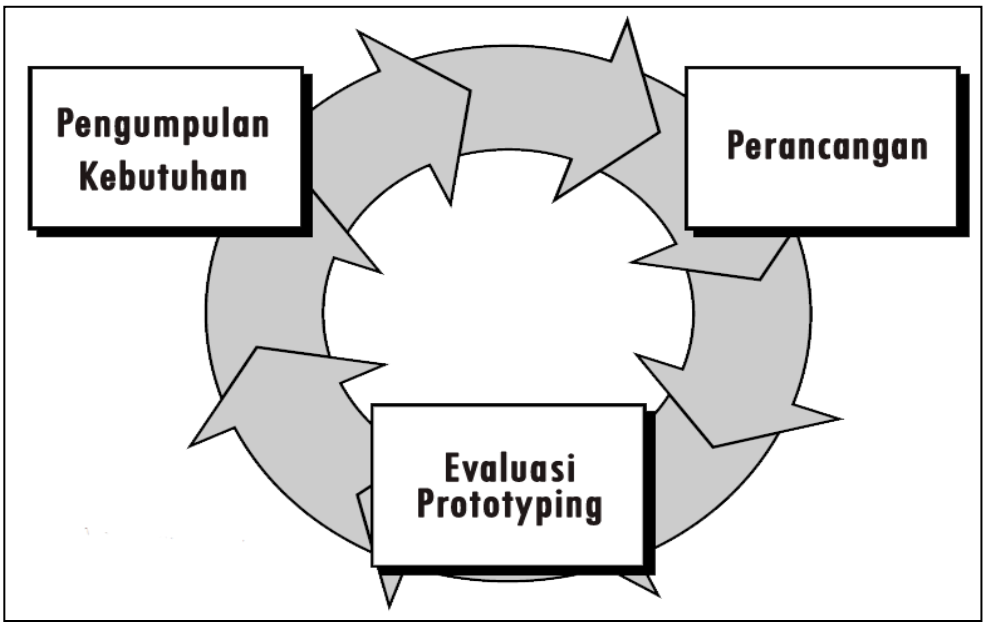
Rosa A. S dan M Shalahuddin (2016) menguraikan Java dikembangkan oleh perusahaan Sun Microsystem. Java menurut devinisi dari Sun Microsystem adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *standalone(berdiri sendiri),* ataupun pada lingkungan jaringan.Java 2 adalah generasi kedua dari java platform.

Java berdiri diatas sebuah mesin *interpreter* yang diberi nama *java virtual machine* (JVM). JVM inilah yang akan membaca *bytecode* dalam *file*. Class dari suatu program sebagai representasi langsung program yang berisi bahasa mesin. Oleh karena itu bahasa java disebut sebagai bahasa pemrograman yang portable karena dapat dijalankan pada berbagai sistem operasi, asalkan pada sistem operasi tersebut terdapat JVM.

Java merupakan bahsa pemrograman objek murni karena semua kode programnya dibungkus kelas. Saat ini Sun Microsystem sudah diakuisisi Oracle Corporation sehingga pengembangan java diteruskan oleh Oracle Corporation.

2.11 Metode Pengembangan Perangkat Lunak Model Prototype

Pressman (2012 : 40) menguraikan bahwa *Prototyping* paradigma dimulai dengan pengumpulan kebutuhan, pengembang bertemu dengan pengguna dan mengidentifikasikan objektif keseluruhan dari perangkat lunak, selanjutnya mengidentifikasi segala kebutuhan yang diketahui secara garis besar di mana definisi-definisi lebih jauh merupakan keharusan, kemudian dilakukan perancangan kilat, lalu diakhiri dengan evaluasi prototyping yang dapat dilihat pada gambar 2.1 berikut :



Gambar 2.1 *Model Prototype*

Tahap–tahap rekayasa *software* dalam *Prototype model* pada gambar 2.1 di atas adalah sebagai berikut :

1) Pengumpulan kebutuhan

   Developer dan klien bertemu untuk menentukan tujuan umum, kebutuhan yang diketahui dan gambaran bagian-bagian yang akan dibutuhkan berikutnya Detail kebutuhan mungkin tidak dibicarakan disini, pada awal pengumpulan kebutuhan. Selanjutnya peneliti akan melakukan analisis terhadap data apa saja yang dibutuhkan, seperti analisis terhadap sistem yang berjalan, analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis kebutuhan notofikasi service berkala.

2) Perancangan

   Perancangan dilakukan dengan cepat dan rancangan mewakili semua aspek software yang diketahui, dan rancangan ini menjadi dasar pembuatan *prototype*. Dalam tahap ini peneliti akan membangun sebuah

versi *prototype* yang dirancang kembali dimana masalah-masalah tersebut diselesaikan.

3) Evaluasi *prototype*

Pada tahap ini, calon pengguna mengevaluasi *prototype* yang dibuat dan digunakan untuk memperjelas kebutuhan *software*. *Software* yang sudah jadi dijalankan dan akan dilakukan perbaikan apabila kurang memuaskan. Perbaikan termasuk dalam memperbaiki kesalahan/ kerusakan yang tidak ditemukan pada langkah sebelumnya.

Kelebihan dari *Prototype Model* adalah sebagai berikut :

1) End user dapat berpartisipasi aktif.
2) Penentuan kebutuhan lebih mudah diwujudkan.
3) Mempersingkat waktu pengembangan *software*.

Kekurangan dari *Prototype Model* adalah sebagai berikut:

1) Proses analisis dan perancangan terlalu singkat.
2) Mengesampingkan alternatif pemecahan masalah.
3) Biasanya kurang fleksibel dalam menghadapi perubahan.
4) *Prototype* yang dihasilkan tidak selamanya mudah dirubah.
5) *Prototype* terlalu cepat selesai.

2.12 Unified Modeling Language (UML)

Nugroho A (2010, 6) menguraikan bahwa UML merupakan bahasa untuk membangun dan mendokumentasikan *artifacts* (bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, *artifact* tersebut dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak. Selain itu UML adalah bahasa pemodelan yang menggunakan konsep orientasi object. UML dibuat oleh *Grady Booch, James Rumbaugh*, dan *Ivar Jacobson* di bawah bendera *Rational Software Crop*. UML menyediakan notasi-notasi yang

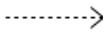
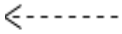
membantu memodelkan sistem dari berbagai perspektif. UML tidak hanya digunakan dalam pemodelan perangkat lunak, namu hampir dalam semua bidang yang membutuhkan pemodelan.

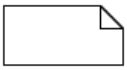
Tipe-tipe dari diagaram UML adalah sebagai berikut :

1. Use Case Diagram

Use case diagram adalah gambar dari beberapa atau seluruh aktor dan *use case* dengan tujuan yang mengenali interaksi mereka dalam suatu sistem. *Use case diagram* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* mepresentasikan sebuah interaksi antara actor dan sistem. Dalam *use case* diagram terdapat istilah seperti aktor, *use case* dan *case relationship.* Penjelasan simbol pada tabel 2.2

Tabel 2.2 Simbol *Use Case Diagram*

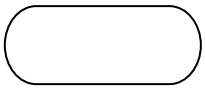
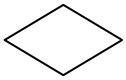
GAMBAR	NAMA	KETERANGAN
	*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
	Generalization	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	Include	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
	*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.

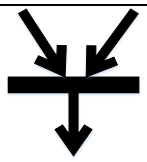
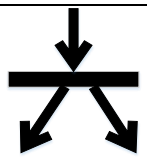
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.
	*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
	*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
	*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

2. Activity Diagram

Activity diagram menggambarkan rangkaian aliran dari aktifitas, digunakan untuk mendeskripsikan aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi. *Activity Diagram* berupa *flowchart* yang digunakan untuk memperlihatkan aliran kerja dari sistem. Notasi yang digunakan dalam *activity diagram* dapat dilihat pada Tabel 2.3.

Tabel 2.3 Simbol *Activity Diagram*

Simbol	Keterangan
	*Activity* : Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	*Initial Node* : Bagaimana objek dibentuk atau diawali
	*Actifity Final Node* : Bagaimana objek dibentuk dan diakhiri.
	*Decision* : Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu.

	*Swimlane* : Memisalkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi.
	*Join* : Digunakan untuk menunjukkan kegiatan yang digabungkan.
	*Fork* : Digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel

3. *Sequence Diagram*

Sequence diagram menggambarkan kolaborasi dinamis antara sejumalah dan untuk menunjukkan rangkaian pesan yang dikirim antar objek juga interaksi antar objek, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem. *Sequence* diagram menjelaskan interaksi objek yang disusun berdasarkan urutan waktu.

4. Class Diagram

Class diagram menggambarkan dtruktur data dan deskripsi *class*, *package*, dan objek beserta hubungan satu sama lain. *Class diagram* berfungsi untuk menjelaskan tipe dari objek sistem dan hubungannya dengan objek yang lain. *Class* memiliki 3 area pokok yaitu nama, atribut dan metode.

2.13 Penelitian Terkait

Penelitian terdahulu yang berkaitan dengan penelitian customisable notification dalam layanan kepada pelanggan dapat dilihat pada Tabel 2.4.

Tabel 2.4 Tabel Penelitian Terkait.

Nama	Judul	Terbit/Tahun	Keterangan

Taufik Ramadhan	Rancang bangun aplikasi mobile untuk notifikasi jadwal kuliah berbasis android	STMIK Provisi Semarang, 2014: Semarang	Dengan adanya sistem yang dapat memberikan rekomendasi sebagai bahan untuk mengimplemtasikan aplikasi mobile untuk notifikasi jadwal yang berbasis android.
Anggit Hernowo	Perancangan Dan Implementasi Sistem Penjadwalan Servis Berkala Kendaraan Bermotor Berbasis Website menggunakan Notifikasi SMS Gateway	Kristen Satya Wacana, 2016: Salatiga	Pembuatan sistem penjadwalan service berkala kendaraan bermotor untuk memberikan notifikasi berbasis SMS Gateway.
Muhammad Irsan	Rancang Bangun aplikasi mobile notifikasi berbasis android untuk mendukung kinerja di instansi pemerintahan	Universitas Tanjung Pura, 2015: Pontianak	Dengan penelitian membangun aplikasi mobile notifikasi untuk mendukung kinerja di instansi perusahaan.

Arif Winandar	Penerapan notifikasi android untuk membantu penyebaran informasi dan komunikasi sivitas Universitas Darma persada	Universitas Darma Persada, 2015:Jakarta	Penerapan Notifikas android untuk memudahkan penyebaran informasi dan komunikasi sivitas.
Yogiswara, dan Astriyanto	Penerapan Web Service dan firebase notification pada pengembangan aplikasi gerakan nasi bungkus jember berbasis android.	Politeknik Negeri Jember 2018.	Dengan menggunakan web service dan firebase notifikasi pada pengembangan aplikasi memudahkan pengguna untuk gerakan nasi bungkus
Asep saefullah, Diah Ariyani dan Andy Rienauld	Sistem notifikasi antrian berbasis android.	STMIK Raharja Tangerang, 2014.	Dengan menerapkan sistem notifikasi memudahkan pelanggan dalam antrian.
Bobby Ghani dan Nurfiana	Perancangan Sistem Informasi Promosi pada PT. Centerpoint Putra Sejahtera Berbasis SMS	Informatics and Business Institute Darmajaya, 2013.	Dengan menerapkan sostem informasi promosi berbasis SMSgateway Memudahkan pelanggan/pengguna untuk mendapatkan promosi yang terbaru

	Gateway		
Muhammad Zaky Faried, Anggraini Mulwinda,dan Yohanes Primadiyono	Pengembangan Aplikasi Android Bimbingan Skripsi Dengan fitur Notifikasi	Jurusan Teknik Elektro, Universitas Negeri Semarang, Gunungpati, Semarang, 50229, Indonesia ,Desember 2017	Aplikasi Android Bimbingan Skripsi dengan fitur notifikasi berfokus pada mengetahui pentingnya fitur notifikasi pada aplikasi,memvalidasi bimbingan skripsi, edit bimbingan skripsi, dan memberikan notifikasi pada bimbingan baru dan bimbingan yang telah divalidasi

BAB III

METODELOGI PENELITIAN

3.1 Metode Pengembangan Perangkat Lunak Model *Prototype*

3.1.1 Pengumpulan Kebutuhan

Tahapan ini adalah awal dari perancangan customisable notifikasi dalam layanan pelanggan berbasis *Android*. Di mana dengan mengumpulkan semua data yang dibutuhkan. Tahap ini berkaitan dengan penentuan kebutuhan pengguna dan perancang program. Peneliti akan menerapkan beberapa metode pengumpulan data serta melaksanakan analisis seperti analisis terhadap sistem yang berjalan, analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis kebutuhan informasi layanan darurat.

Metode pengumpulan data yang digunakan dalam penelitian ini untuk memperoleh data-data penelitian ini adalah sebagai berikut :

1. Observasi

   Metode pengumpulan data ini dilakukan dengan cara melakukan pengamatan langsung pada obyek penelitian. Obyek penelitian yang dilakukan antara lain dealer mobil Auto 2000 Wayhalim Bandar Lampung.

2. Wawancara

   Metode ini dilakukan dengan cara bertemu langsung dan melakukan tanya jawab/wawancara dengan pihak yang terkait.

3. Tinjauan Pustaka

   Tinjauan pustaka merupakan teknik pengumpulan data dengan cara membaca, mengutip, dan mengumpulkan teori-teori dari buku-buku, jurnal, internet serta mempelajari referensi dokumen dan catatan lain yang mendukung proses penelitian..

3.1.1.1 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang dikembangkan dalam membangun aplikasi *customisable ntification* adalah berbasis *firebase* untuk pengelola sistem dan

berbasis *mobile Android* bagi pelanggan. Perangkat lunak yang disarankan untuk menjalankan aplikasi ini adalah sebagai berikut :

1. Sistem Operasi yang digunakan adalah *Windows 10*
2. *Web Server* menggunakan Firebase
3. *Web Browser* Internet (*Mozilla Firefox/Google Chrome)*
4. *Software* yang digunakan adalah Android Studio
5. *Adobe Photoshop* digunakan untuk mendesain tampilan
6. *Provider* yang digunakan adalah yang mempunyai koneksi stabil.

3.1.1.2 Analisis Kebutuhan Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam perancangan sistem tersebut adalah satu unit Laptop Acer Aspire 4740 Series dengan spesifikasi sebagai berikut:

1. *Processor Core I3*
2. *Harddisk* 320 GB
3. RAM 5 GB
4. *Keyboard* dan *Mouse* standard
5. *Kabel USB*
6. *Printer* dengan jenis *Ink Jet* dan *Thermal Printer*

3.1.2 Perancangan

Perancangan merupakan tahapan yang dilakukan untuk memulai pembangunan sistem dimana disesuaikan dengan identifikasi pengumpulan kebutuhan yang telah dilakukan peneliti. Proses perancangan dimulai dari perancangan sistem yang telah disusulkan kemudian dilanjutkan dengan pembuatan perangkat lunak dimana berupa *Unified Modeling Language* (UML), dan Perancangan Antarmuka (*Intrface*) sistem.

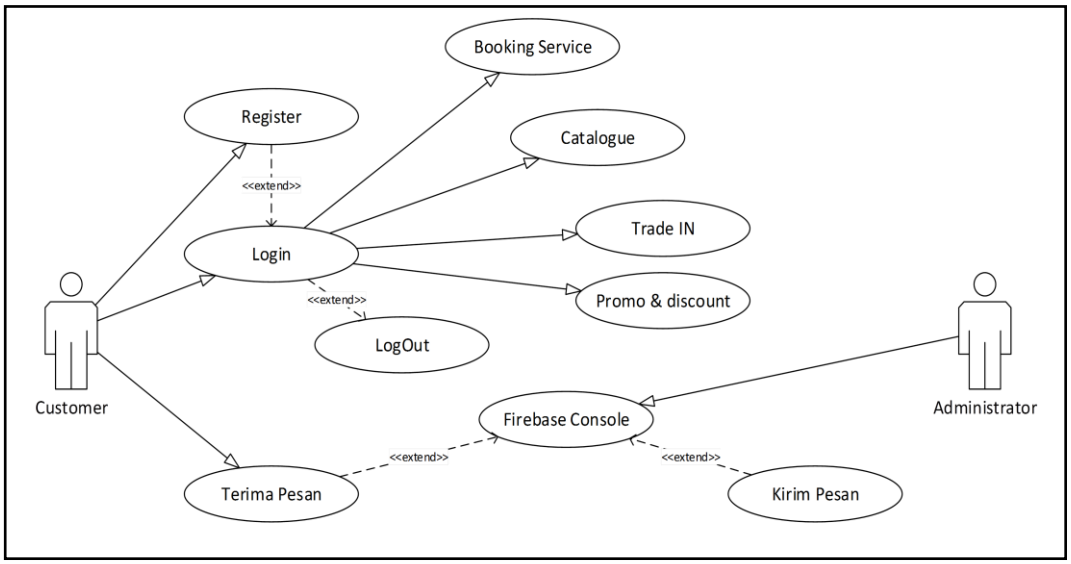
3.1.2.1 Unified Modeling Language (UML)

UML merupakan sebuah bahasa yang berdasarkan pada grafik atau gambar untuk memvisualisasikan dan mendokumentasikan informasi yang digunakan dalam proses pembuatan perangkat lunak. Informasi dalam pembuatan perangkat lunak berupa model dan atau deskripsi perangkat lunak. Terdapat berbagai macam jenis

diagram yang digunakan untuk memvisualisasikan perangkat lunak, dalam membangun *mobile delivery order* Simetri Digital Printing peneliti menggunakan diagram yaitu meliputi *Use Case Diagram, Class Diagram, Activity Diagram,* dan *Sequence Diagram.*

a. Use Case Diagram

Use case Diagram merupakan pemodelan sistem informasi yang digunakan untuk mendeskripsikan sebuah interaksi antara satu aktor atau lebih dengan aktor lainnya sesuai pada sistem yang ada. Dalam penelitian ini, aktor yang terlibat dalam aplikasi *customisable notification* adalah *customer* sebagai pihak pengguna layanan dan Admin sebagai pengelola sistem *customisable notification* tersebut yang dapat dilihat pada Gambar 3.1



Gambar 3.1 Use Case Diagram Aplikasi.

Berdasarkan Gambar 3.1 Use Case Diagram Sistem, skenario pendeskripsian dan pendefinisiannya dapat dilihat pada Tabel 3.1 dan Tabel 3.2

Tabel 3.1 Deskripsi Aktor.

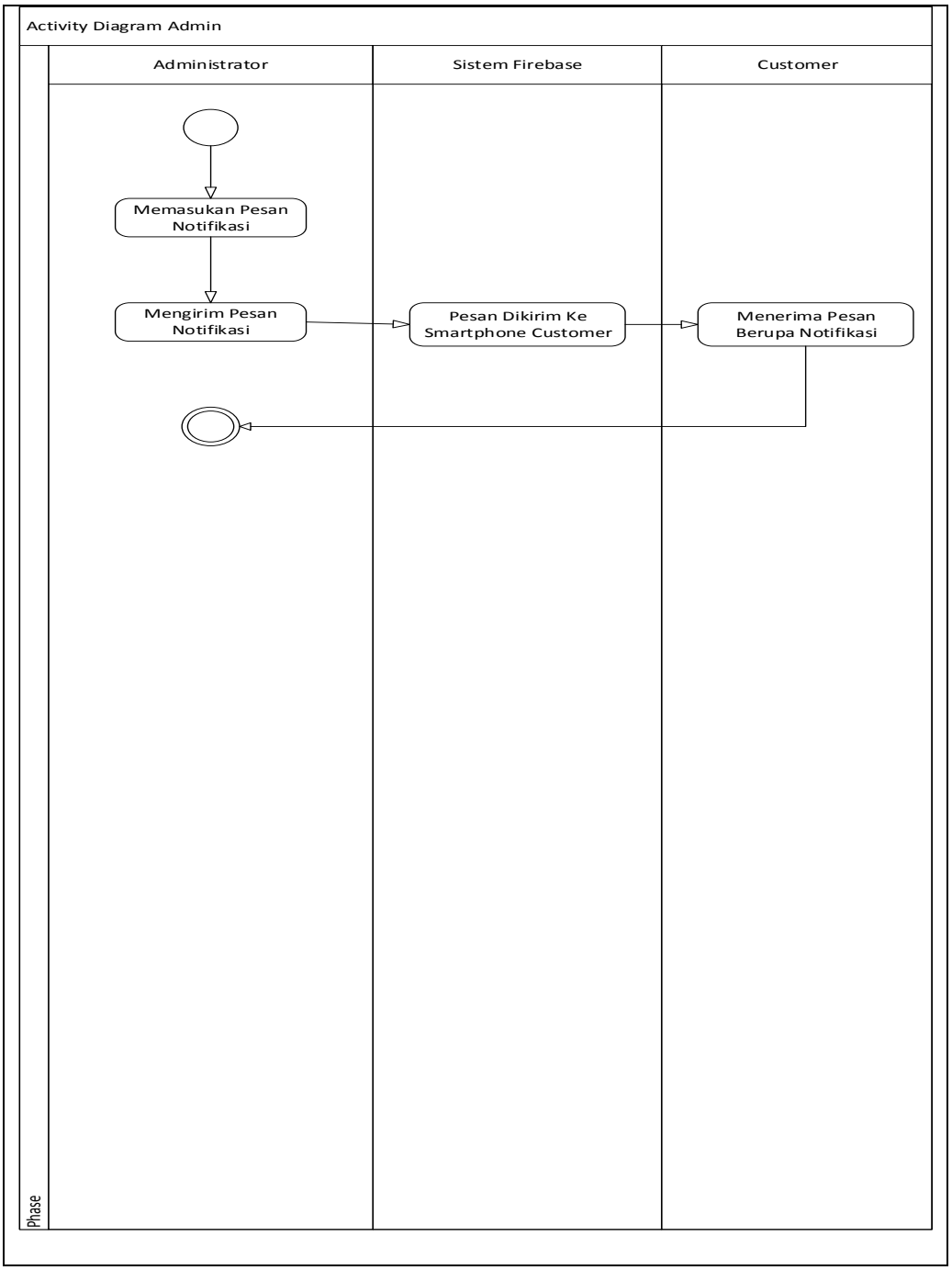
No	Aktor	Deskripsi
1	Customer	Orang yang ingin melakukan Service kendaraan dan melihat produk yang ada di dealer Auto 2000.
2	Admin	Orang yang mengelola seluruh prosedur yang ada pada sistem *Firebase console.*

Tabel 3.2 Skenario *Use Case Diagram* Aplikasi.

Aktor	No	Skenario	Sistem
Customer	1	Memasukkan *password* dan *username*	1. Jika belum memiliki *password* dan *username* diminta registrasi terlebih dahulu. 2. Masuk ke aplikasi *customisable notifikasi*.
	2	Booking Service	1. Menampilkan web untuk melakukan booking service yang akan dilakukan.
	3.	Catalogue	1. Menampilkan Produk- produk yang ada di dealer Auto 2000.
	4.	Trade IN	1. Menampilkan Layanan terbaru untuk memungkinkan menjual mobil lama.
	5.	Promo dan discount	1. Menampilkan promo yang terdpat pada dealer auto 2000.
	6.	Menerima notifikasi	2. Customer Menerima Notifikasi .
Admin	1	Memasukan *username* dan *password*	1. Memeriksa valid tidaknya data masukkan. 2. Masuk ke dalam Firebase Console.
	2	Mengelola Firebase Console	Admin dapat mengirim, dan mengedit pelayanan Service berkala yang ditentukan.

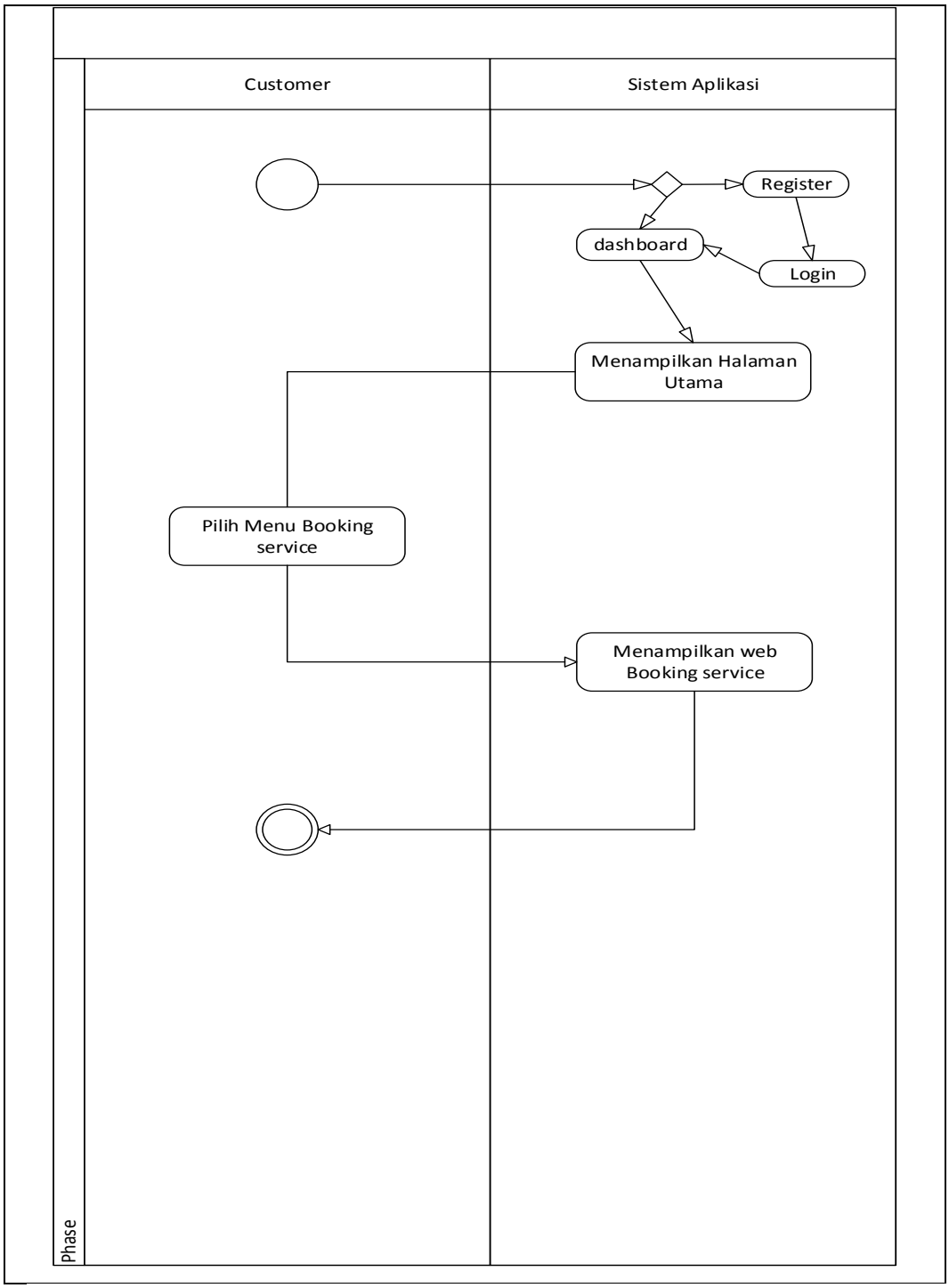
b. Activity Diagram

Activity Diagram merupakan aktivitas yang berfungsi untuk menggambarkan langkah-langkah aliran kerja dari sebuah sistem yang dirancang. Dalam membangun aplikasi *customisable notifikasi* aktivitas yang terjadi dalam sistem terdapat 3 bagian yaitu dapat dilihat pada Gambar 3.2 Gambar 3.3, Gambar 3.4 gambar 3.5 dan gambar 3.6.



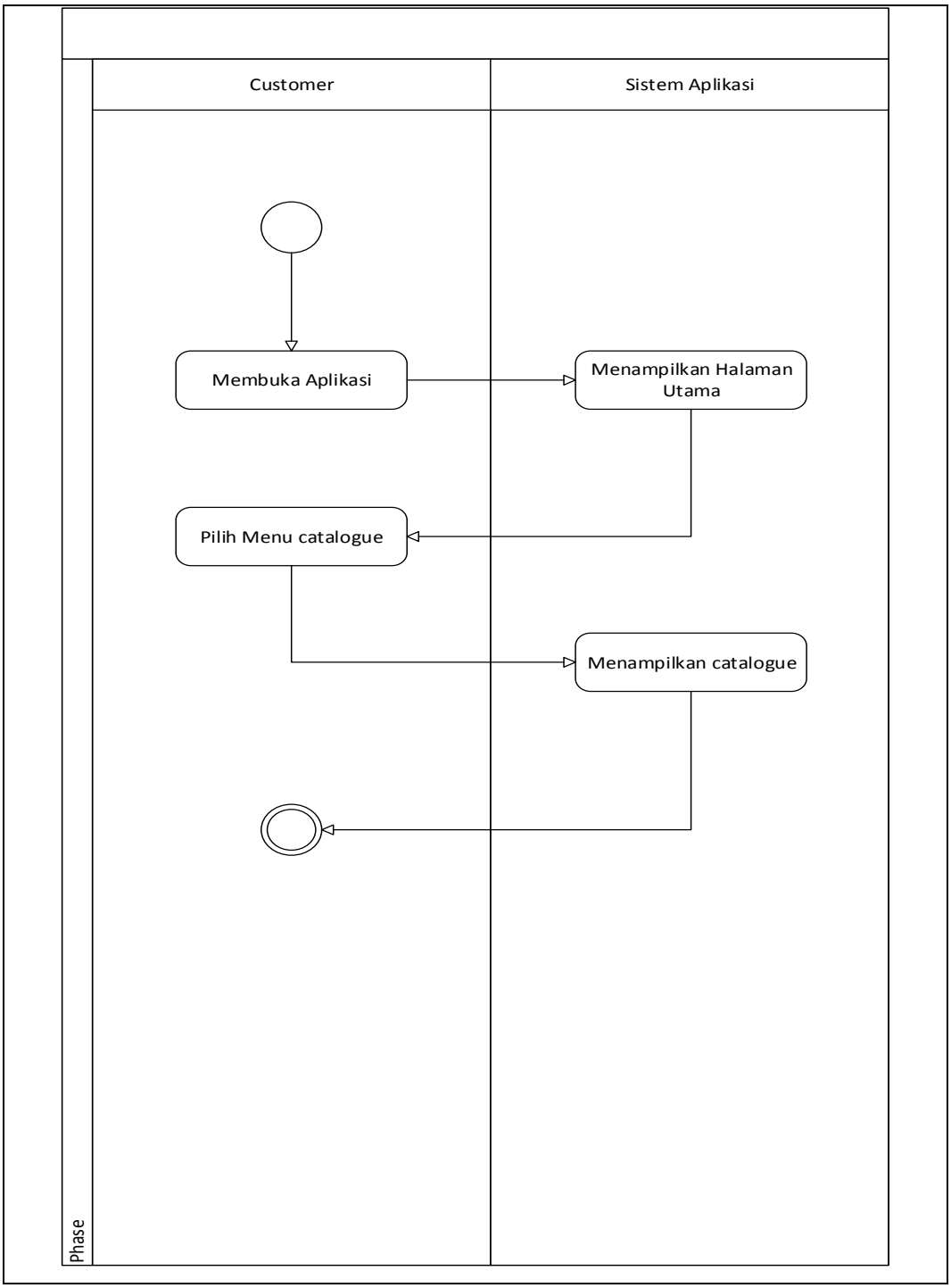
Gambar 3.2 *Activity Diagram Administrator* Dengan Sistem.

2. Activity Diagram Costumer dengan sistem menjelaskan bagaimana alur didalam menjalankan aplikasi seperti terlihat pada activity diagram pelanggan bisa melakukan registrasi dan login kemudian pelanggan dapat melihat halaman utama dan memilih menu booking service kemudian aplikasi akan menampikan booking service.



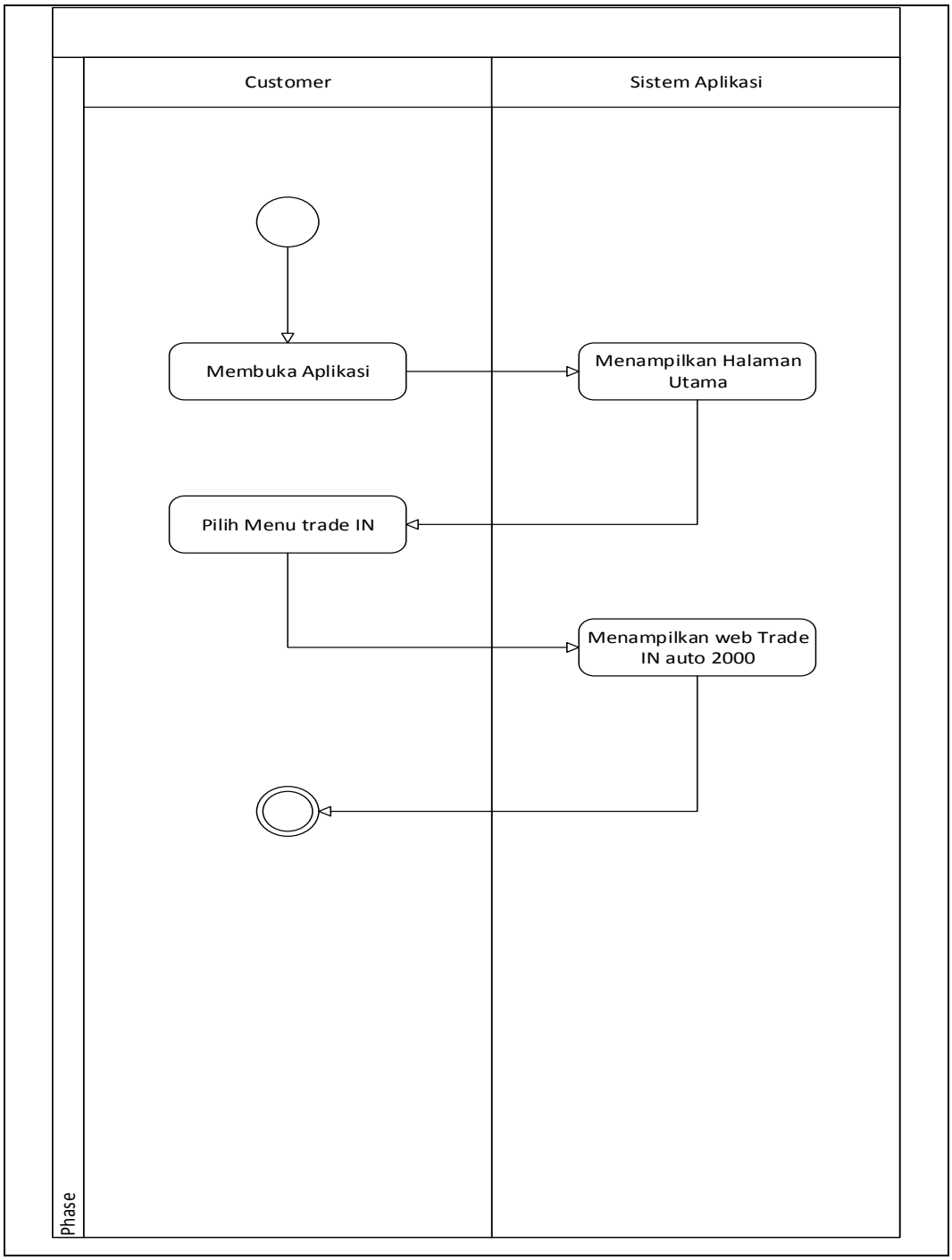
Gambar 3.3 *Activity Diagram* Costumer dengan sistem.

3. *Activity Diagram* menu catalog menjelaskan bagaimana alur didalam menjalankan aplikasi seperti terlihat pada activity diagram pelanggan bisa membuka aplikasi setelah login kemudian pelanggan dapat melihat halaman utama dan memilih menu catalog kemudian aplikasi akan menampilakn catalog.



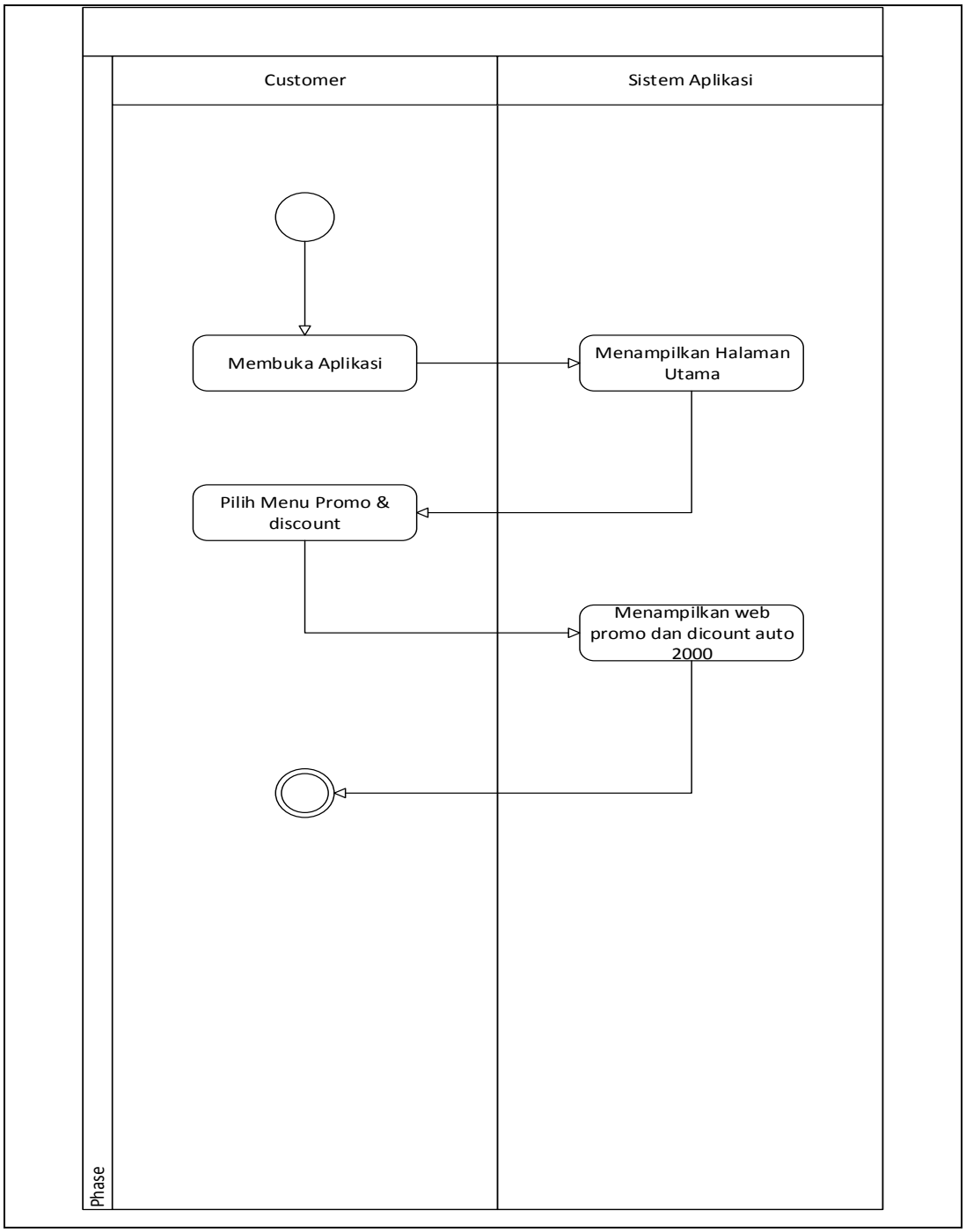
Gambar 3.4 *Activity Diagram* Menu catalogue.

4. *Activity Diagram* menu trade in menjelaskan bagaimana alur didalam menjalankan aplikasi seperti terlihat pada activity diagram pelanggan bisa membuka aplikasi setelah login kemudian pelanggan dapat melihat halaman utama dan memilih menu Trade in kemudian aplikasi akan menampilakn halaman Trade In.



Gambar 3.5 *Activity Diagram* menu tradein.

5. *Activity Diagram* menu promo menjelaskan bagaimana alur didalam menjalankan aplikasi seperti terlihat pada activity diagram pelanggan bisa membuka aplikasi setelah login kemudian pelanggan dapat melihat halaman utama dan memilih menu catalog kemudian aplikasi akan menampilakn halaman promo dan discount.



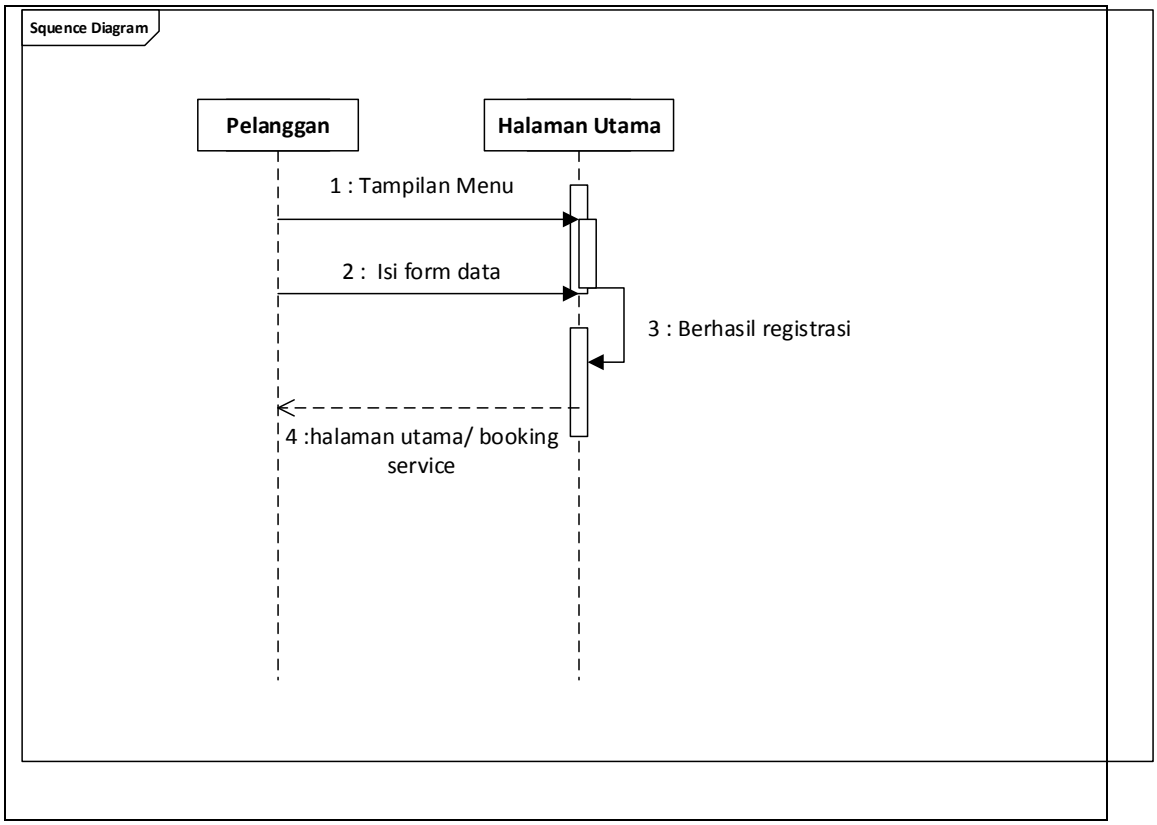
Gambar 3.6 *Activity Diagram* menu promo & dicount.

c. Sequence Diagram

Diagram sekuen menggambarkan interaksi antar objek di dalam sistem, dimana interaksi tersebut berupa *message* yang digambarkan terhadap waktu. Dalam menggambarkan interaksi objek harus diketahui objek-objek yang terlibat dalam sebuah *use case*, sehingga cocok untuk menggambarkan model deskripsi *use case* menjadi spesifikasi desain. Diagram sekuen yang digunakan dalam merancang aplikasi *customisable notifikasi* memiliki 2 bagian Aktor, yaitu *Customer* dan *Admin*.

1. *Sequence Diagram* (*Customer*)

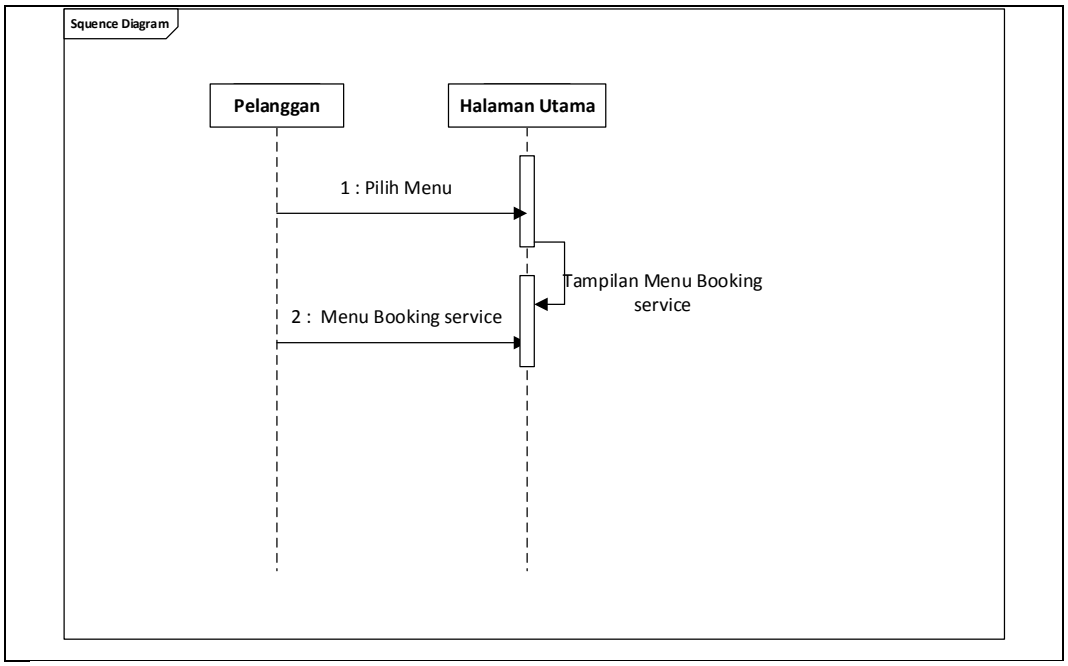
Dalam aplikasi *customisable notifikasi*, *customer* dapat melakukan registrasi atau pun langsung melakukan booking service kendaraan yang dapat dilihat pada Gambar 3.7.



Gambar 3.7 *Sequence Diagram* Registrasi.

a. Sequence Diagram (*Menu Booking service)*

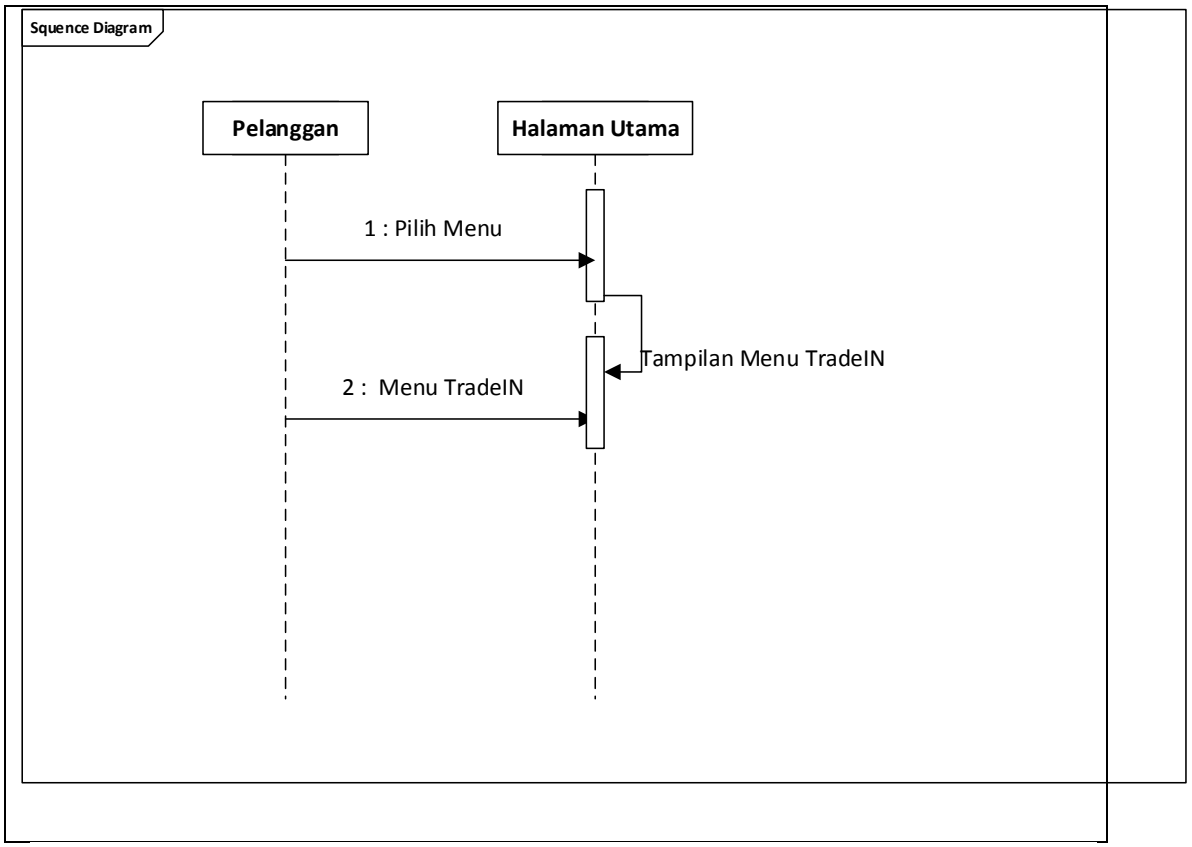
Merupakan urutan/tahap setelah login dimana Customer dapat mengakses fitur-fitur yang terdapat pada sistem aplikasi *customisable notifikasi* seperti menu booking service yang dapat dilihat pada Gambar 3.8



Gambar 3.8 *Sequence Diagram* Menu booking service.

b. Sequence Diagram (*Menu TradeIN)*

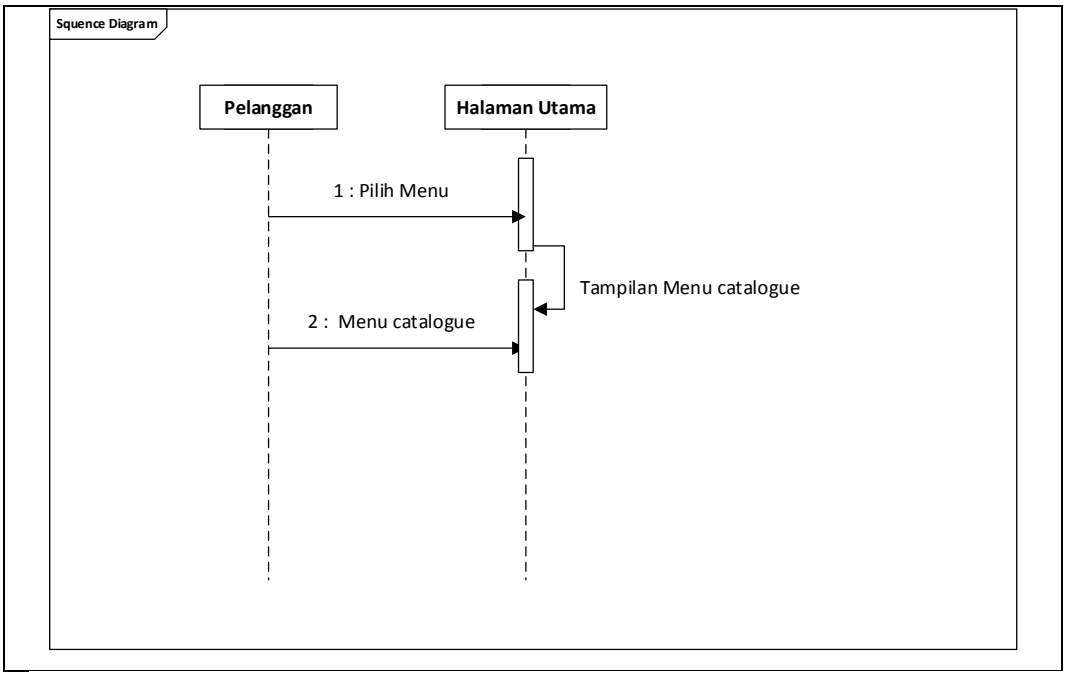
Merupakan urutan/tahap setelah login dimana Customer dapat mengakses fitur-fitur yang terdapat pada sistem aplikasi *customisable notifikasi* seperti menu Tradein yang dapat dilihat pada Gambar 3.9



Gambar 3.9 *Sequence Diagram* Menu TradeIN.

c. Sequence Diagram (*Menu catalogue)*

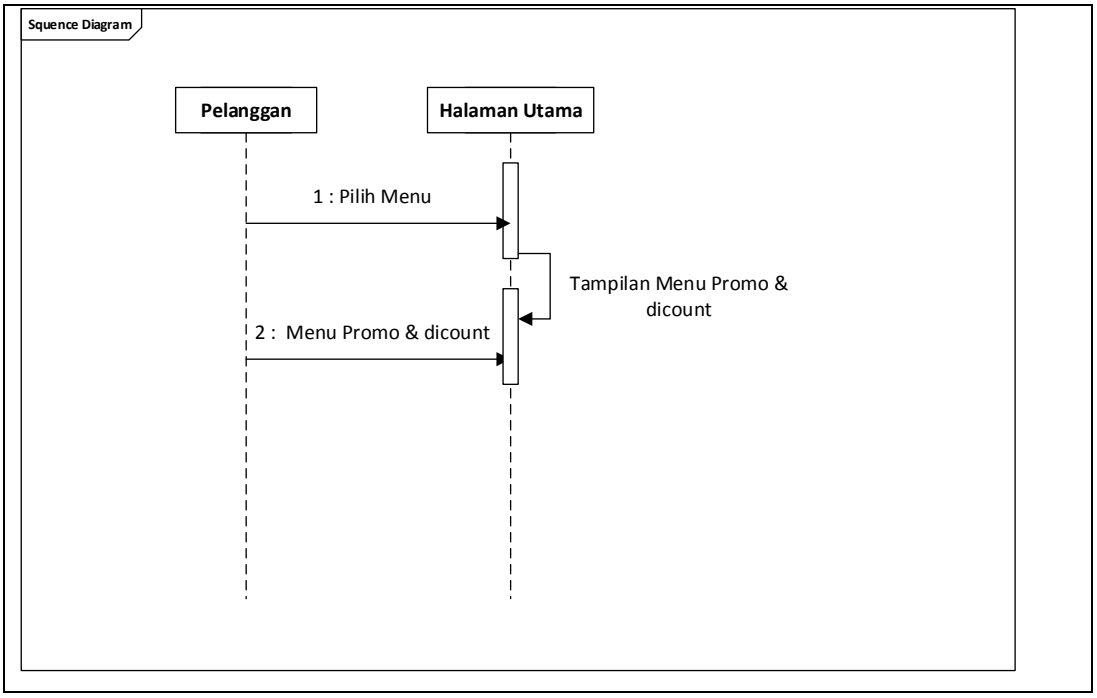
Merupakan urutan/tahap setelah login dimana Customer dapat mengakses fitur-fitur yang terdapat pada sistem aplikasi *customisable notifikasi* seperti menu catalogue yang dapat dilihat pada Gambar 3.10



Gambar 3.10 *Sequence Diagram* Menu catalogue.

d. Sequence Diagram (*Menu promo & discount)*

Merupakan urutan/tahap setelah login dimana Customer dapat mengakses fitur-fitur yang terdapat pada sistem aplikasi *customisable notifikasi* seperti menu promo dan discount yang dapat dilihat pada Gambar 3.11



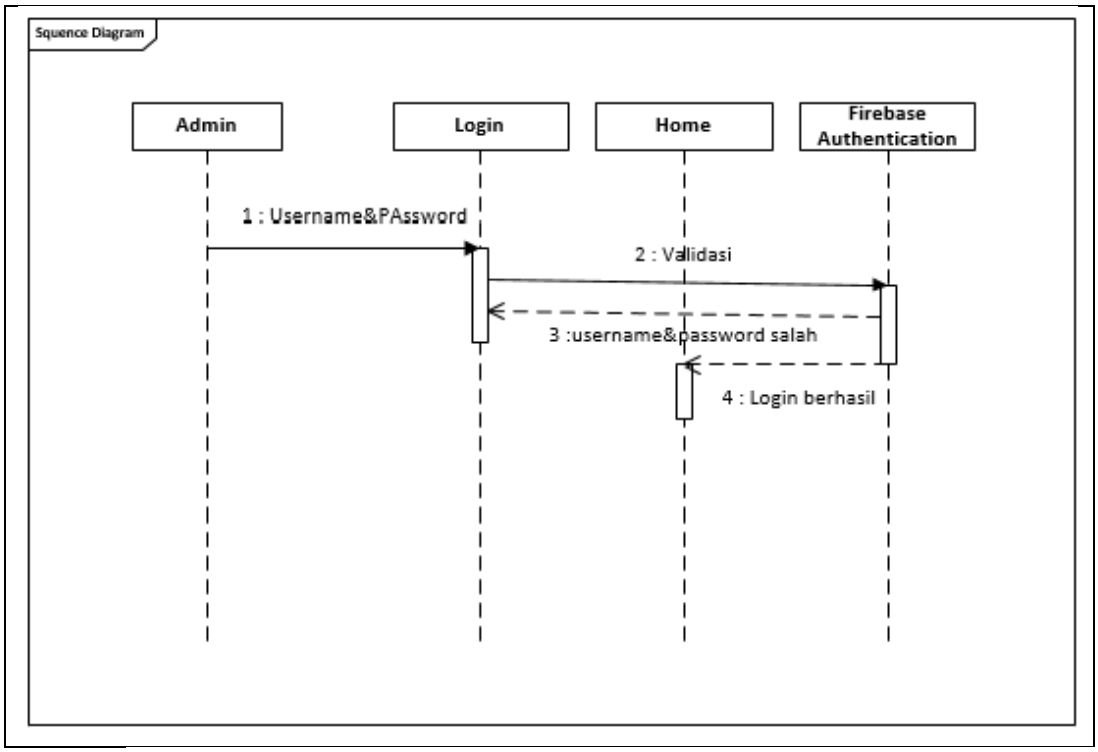
Gambar 3.11 *Sequence Diagram* Menu Promo & discount.

2. *Sequence Diagram* (Admin)

Dalam mengelola aplikasi *customisable notifikasi*, Admin memiliki peran dalam mengelola *firebas console*. Selain itu juga Admin harus mengelola proses-proses dari setiap objek-objek yang berkaitan dengan mengirim dan mengedit pesan yang akan dikirim kepada Customer. Sehingga dalam proses tersebut, dapat dilihat gambarannya sebagai berikut :

e. Sequence Diagram (*Login)*

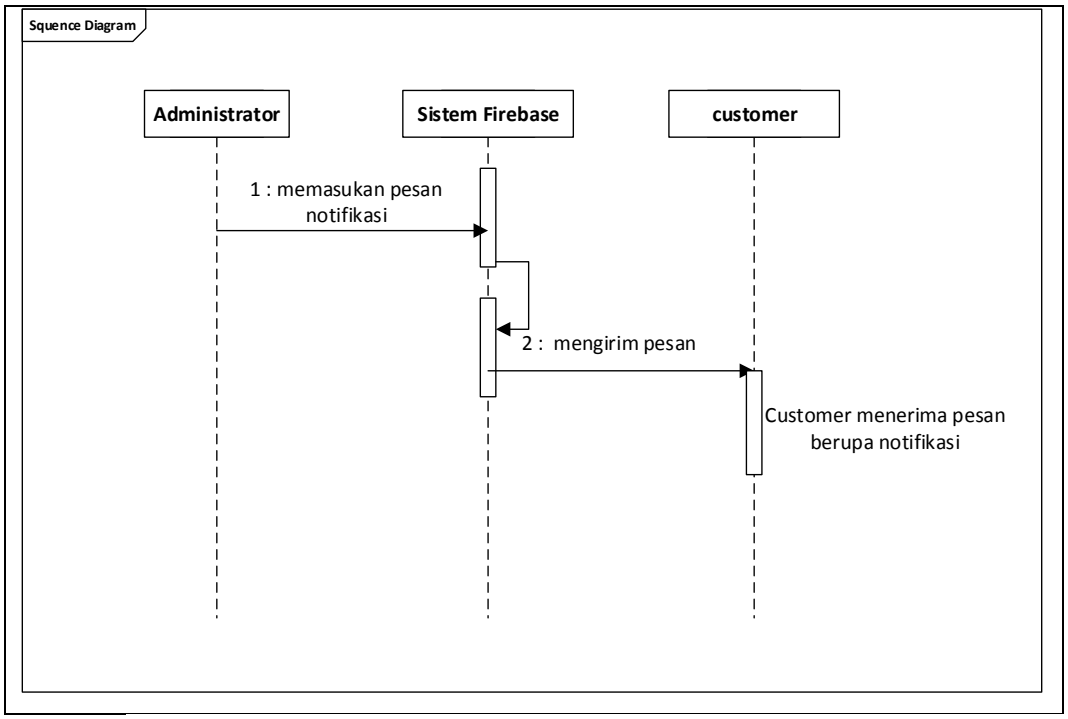
Merupakan urutan/tahap awal dimana Admin harus melakukan *login* untuk dapat mengakses fitur-fitur yang terdapat pada sistem *firebase console* yang dapat dilihat pada Gambar 3.12.



Gambar 3.12 *Sequence Diagram Login.*

f. Sequence Diagram (*mengirim notifikasi)*

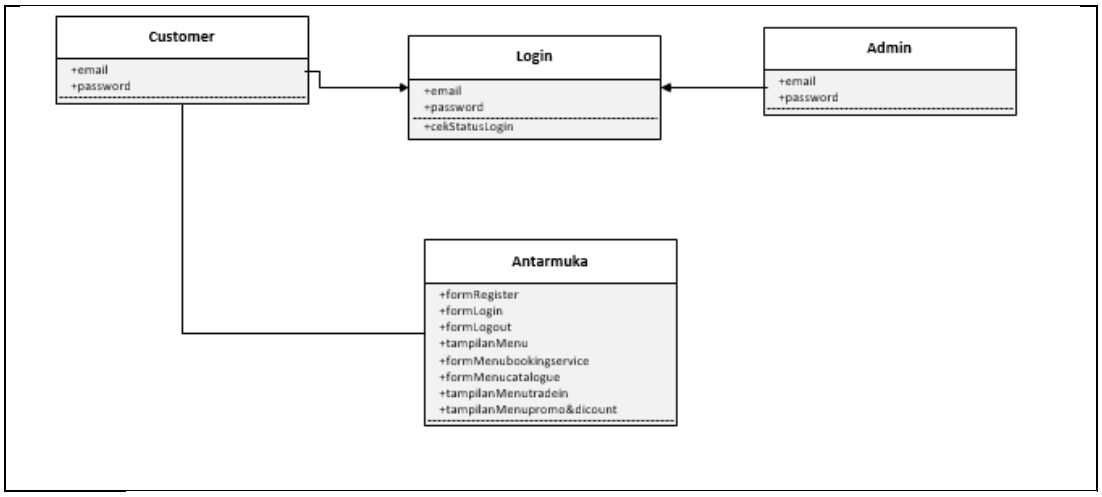
Merupakan urutan/tahap awal dimana Admin harus melakukan *login* untuk dapat mengakses fitur-fitur yang terdapat pada sistem *firebase console* yang dapat dilihat pada Gambar 3.13.



Gambar 3.13 *Sequence Diagram* mengirim notifikasi

d. Class Diagram

Class diagram merupakan diagram yang menggambarkan struktur dan kommponen-komponen yang terkait pada sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Selain itu *class diagram* berfungsi menjelaskan tipe dari objek sistem dan hubungannya dengan objek lain.Objek merupakan nilai tertentu yang memiliki atribut dan metode atau operasi. *Class diargram* pada perancangan aplikasi *customidable notification* dapat dilihat pada Gambar 3.14.



Gambar 3.14 *class Diagram* aplikasi customisable notifikasi.

Deskripsi berdasarkan Gambar 3.14 *Class Diagram* Aplikasi *customisable notifikasi*, dapat dilihat pada Tabel 3.3.

Tabel 3.3 Keterangan *Class Diagram.*

Nama Kelas	Keterangan
Antarmuka	Merupakan kelas yang menangani tampilan.
Login	Merupakan kelas proses yang diambil dari pendefinisian *use case login.*
Customer	Merupakan kelas data yang digunakan untuk menjalankan aplikasi customisable notifikasi.
Antarmuka	Merupakan kelas proses yang digunakan untuk mengelola proses from register, from login, tampilan menu.
Admin	Merupakan kelas data yang digunakan untuk memproses segala pengaksesan terhadap *firebase consloe.*

3.1.2.2 Desain Rancangan Aplikasi

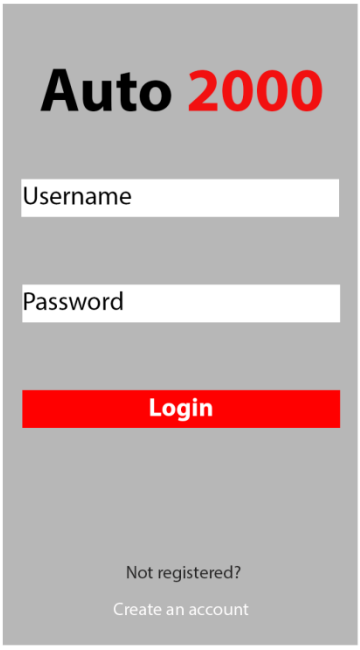
1. Rancang Halaman Mendaftar atau *(Sign UP)*

Rancangan halam ini dapat dilihat pada Gambar 3.15.

Gambar 3.15 Rancangan Halaman Mendaftar *(Sign Up).*

2. Rancangan Halaman LogIn

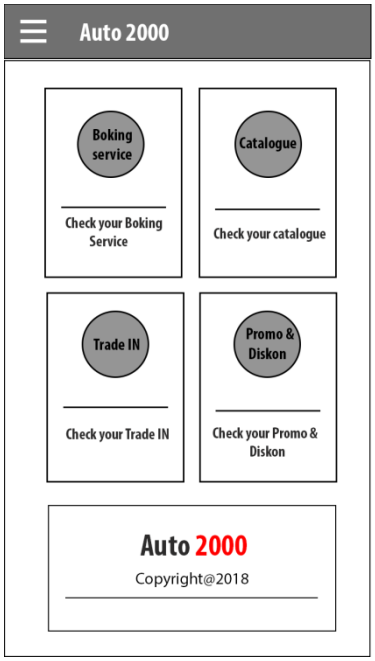
Rancangan halaman ini dapat dilihat pada Gambar 3.16



Gambar 3.16 Rancangan Halaman LogIn.

3. Rancangan Halaman Munu Home (*Dashboard*)

Rancangan halaman *Dashboard* ini dapat dilihat pada Gambar 3.17



Gambar 3.17 Rancangan Halaman Menu Home (*Dashboard).*

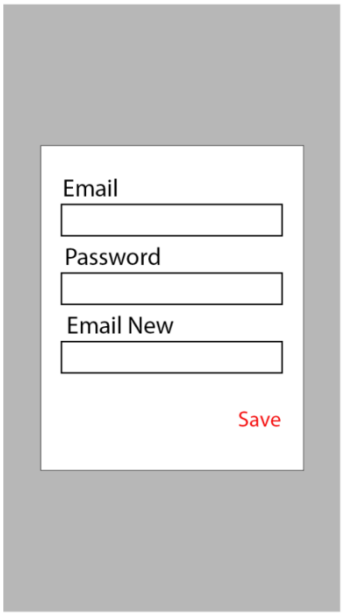
4. Rancangan Halaman Menu Dashboard

   Rancangan menu dashboard ini dapat dilihat pada Gambar 3.18

Gambar 3.18 Rancangan Halaman *Dashboard.*

5. Rancangan Halaman Penggantian Email Baru.

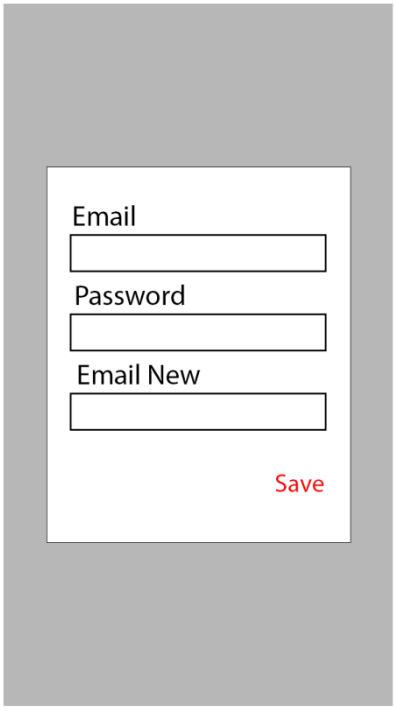
   Rancangan halaman penggantian email dapat dilihat pada gambar 3.19



Gambar 3.19 Rancangan Halaman *mengganti email.*

6. Rancangan Halaman Penggantian Password Baru

Rancangan halaman penggantian email dapat dilihat pada gambar 3.20



Gambar 3.20 Rancangan Halaman Penggantian Password Baru.

3.1.3 Evaluasi Prototype

Merupakan tahap dimana rancangan aplikasi yang telah selesai dibangun dilakukan evaluasi oleh *user*, dimana berkaitan dengan penelitian ini *user* atau pengguna dari aplikasi adalah *customer di* dealer Auto2000. Tahapan pengevaluasian ini dilakukan untuk memperjelas spesifikasi kebutuhan *customer* terhadap aplikasi sesuai yang diinginkan *customer*.

3.2 Proses Kerja Aplikasi *customisable notifikasi* Auto 2000

Aplikasi *customisable notifikasi* merupakan aplikasi *android mobile* yang merupakan tampilan atau fitur untuk menerima pesan berupa notifikasi yang dapat digunakan pelanggan atau pun masyarakat yang ingin melakukan service kendaraan berkala. Berikut ini adalah proses mengirim pesan notification :

1. Instal dan jalankan aplikasi pada perangkat target.
2. Pastikan aplikasi tersebut berjalan di latar belakang pada perangkat.
3. Buka Notifications composer lalu pilih Pesan Baru.
4. Masukkan teks pesan.
5. Pilih Satu Perangkat sebagai target pesan.

Dalam kolom yang berlabel Token Pendaftaran Firebase Cloud Messaging (FCM)masukkan token pendaftaran yang diperoleh di bagian sebelumnya pada panduan ini. Setelah mengklik Kirim Pesan, perangkat klien yang ditargetkan yang memiliki aplikasi di latar belakang menerima notification dalam baki notification sistem..

3.3 Source code Notifikasi

Terdapat 2 bagian coding yang digunakan untuk memunculkan notifikasi yaitu :

1. Myfirebase messaging service

```
package com.example.google.myauto2000;

/**
 * Created by Pikri on 25/07/2018.
 */


import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;
import android.support.v4.app.NotificationCompat;
import android.util.Log;
import com.google.firebase.messaging.FirebaseMessagingService;
import com.google.firebase.messaging.RemoteMessage;


public class MyFirebaseMessagingService extends FirebaseMessagingService {

    private static final String TAG = "MyFirebaseMsgService";

    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {

        // TODO(developer): Handle FCM messages here.
        Log.d(TAG, "From: " + remoteMessage.getFrom());
```

```
        // Check if message contains a data payload.
        if (remoteMessage.getData().size() > 0) {
            Log.d(TAG, "Message data payload: " + remoteMessage.getData());

        }


        // Check if message contains a notification payload.
        if (remoteMessage.getNotification() != null) {
            Log.d(TAG, "Message Notification Body: " +
remoteMessage.getNotification().getBody());
            sendNotification(remoteMessage.getNotification().getBody());
        }

    }

    private void sendNotification(String messageBody) {
        Intent intent = new Intent(this, MainActivity.class);
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent,
                PendingIntent.FLAG_ONE_SHOT);

        Uri defaultSoundUri=
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
        NotificationCompat.Builder notificationBuilder = new
NotificationCompat.Builder(this)
                .setSmallIcon(R.mipmap.icon1)
                .setContentTitle("FCM Message")
                .setContentText(messageBody)
                .setAutoCancel(true)
                .setSound(defaultSoundUri)
                .setContentIntent(pendingIntent);

        NotificationManager notificationManager =
                (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

        notificationManager.notify(0, notificationBuilder.build());
    }

}
```

2. Myfirebase instance ID service

```
package com.example.google.myauto2000;

/**
 * Created by Pikri on 25/07/2018.
 */
import android.content.Context;
import android.content.Intent;
import android.support.v4.content.LocalBroadcastManager;
```

```
import android.text.TextUtils;
import android.util.Log;



import com.google.firebase.messaging.FirebaseMessagingService;
import com.google.firebase.messaging.RemoteMessage;

import org.json.JSONException;
import org.json.JSONObject;

import com.example.google.myauto2000.MainActivity;
import com.example.google.myauto2000.Config;
import com.example.google.myauto2000.NotificationUtils;


public class MyFirebaseInstanceIDService extends FirebaseMessagingService {

   private static final String TAG =
MyFirebaseMessagingService.class.getSimpleName();

   private NotificationUtils notificationUtils;


   @Override
   public void onMessageReceived(RemoteMessage remoteMessage) {
      Log.e(TAG, "From: " + remoteMessage.getFrom());

      if (remoteMessage == null)
         return;

      // Check if message contains a notification payload.
      if (remoteMessage.getNotification() != null) {
         Log.e(TAG, "Notification Body: " +
remoteMessage.getNotification().getBody());
         handleNotification(remoteMessage.getNotification().getBody());
      }

      // Check if message contains a data payload.
      if (remoteMessage.getData().size() > 0) {
         Log.e(TAG, "Data Payload: " + remoteMessage.getData().toString());

         try {
            JSONObject json = new
JSONObject(remoteMessage.getData().toString());
            handleDataMessage(json);
         } catch (Exception e) {
            Log.e(TAG, "Exception: " + e.getMessage());
         }
      }
   }

   private void handleNotification(String message) {
      if (!NotificationUtils.isAppIsInBackground(getApplicationContext())) {
```

```
            // app is in foreground, broadcast the push message
            Intent pushNotification = new Intent(Config.PUSH_NOTIFICATION);
            pushNotification.putExtra("message", message);
            LocalBroadcastManager.getInstance(this).sendBroadcast(pushNotification);

            // play notification sound
            NotificationUtils notificationUtils = new
NotificationUtils(getApplicationContext());
            notificationUtils.playNotificationSound();
        }else{
            // If the app is in background, firebase itself handles the notification
        }
    }


    private void handleDataMessage(JSONObject json) {
        Log.e(TAG, "push json: " + json.toString());

        try {
            JSONObject data = json.getJSONObject("data");

            String title = data.getString("title");
            String message = data.getString("message");
            boolean isBackground = data.getBoolean("is_background");
            String imageUrl = data.getString("image");
            String timestamp = data.getString("timestamp");
            JSONObject payload = data.getJSONObject("payload");

            Log.e(TAG, "title: " + title);
            Log.e(TAG, "message: " + message);
            Log.e(TAG, "isBackground: " + isBackground);
            Log.e(TAG, "payload: " + payload.toString());
            Log.e(TAG, "imageUrl: " + imageUrl);
            Log.e(TAG, "timestamp: " + timestamp);


            if (!NotificationUtils.isAppIsInBackground(getApplicationContext())) {
                // app is in foreground, broadcast the push message
                Intent pushNotification = new Intent(Config.PUSH_NOTIFICATION);
                pushNotification.putExtra("message", message);

LocalBroadcastManager.getInstance(this).sendBroadcast(pushNotification);

                // play notification sound
                NotificationUtils notificationUtils = new
NotificationUtils(getApplicationContext());
                notificationUtils.playNotificationSound();
            } else {
                // app is in background, show the notification in notification tray
                Intent resultIntent = new Intent(getApplicationContext(),
MainActivity.class);
                resultIntent.putExtra("message", message);
```

```
                // check for image attachment
                if (TextUtils.isEmpty(imageUrl)) {
                    showNotificationMessage(getApplicationContext(), title, message,
timestamp, resultIntent);
                } else {
                    // image is present, show notification with image
                    showNotificationMessageWithBigImage(getApplicationContext(), title,
message, timestamp, resultIntent, imageUrl);
                }
            }
        } catch (JSONException e) {
            Log.e(TAG, "Json Exception: " + e.getMessage());
        } catch (Exception e) {
            Log.e(TAG, "Exception: " + e.getMessage());
        }
    }

    /**
     * Showing notification with text only
     */
    private void showNotificationMessage(Context context, String title, String
message, String timeStamp, Intent intent) {
        notificationUtils = new NotificationUtils(context);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
        notificationUtils.showNotificationMessage(title, message, timeStamp, intent);
    }

    /**
     * Showing notification with text and image
     */
    private void showNotificationMessageWithBigImage(Context context, String
title, String message, String timeStamp, Intent intent, String imageUrl) {
        notificationUtils = new NotificationUtils(context);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
        notificationUtils.showNotificationMessage(title, message, timeStamp, intent,
imageUrl);
    }
}
```

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil Penelitian

Berdasarkan hasil identifikasi masalah, pengumpulan kebutuhan dan perancangan aplikasi, maka dihasilkan sebuah aplikasi *customisable notifikasi* di Auto 2000

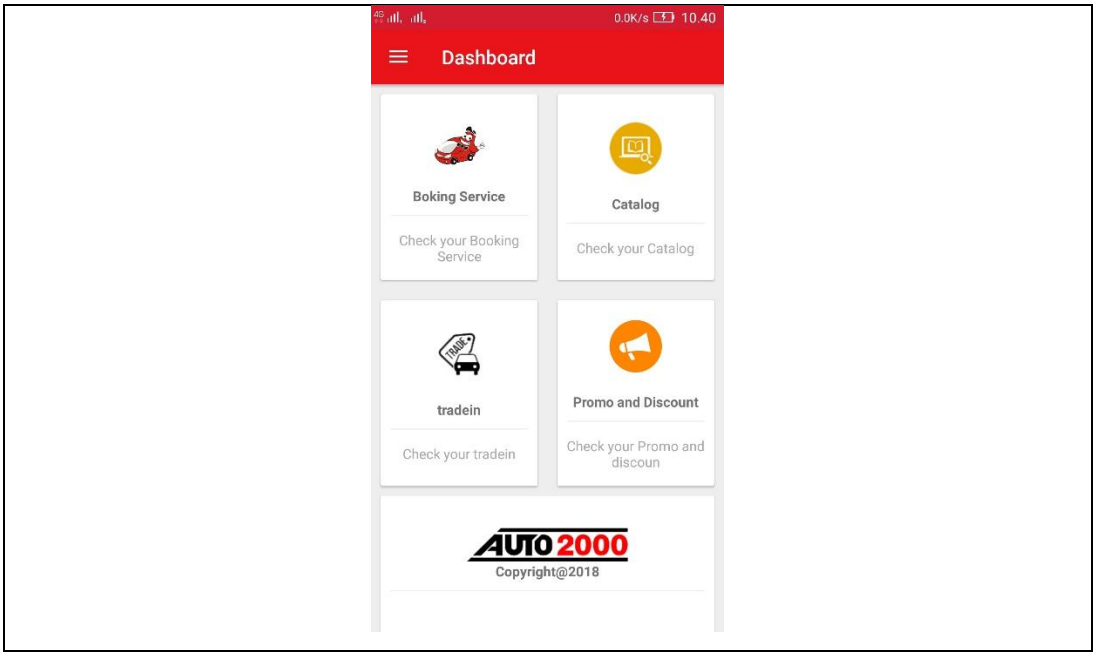
4.1.1 Implementasi Aplikasi *customisable notification* dalam layanan pelanggan

Tahap implementasi aplikasi adalah hasil rancangan perangkat lunak menjadi sebuah program aplikasi. Pada tahap ini menjelaskan tentang *output* dari aplikasi *mobile customisable notification* dalam layanan pelanggan di Auto 2000 beserta dengan fitur-fitur didalamnya.

Fitur-fitur yang terdapat di dalam aplikasi ini yaitu :

4.1.1.1 Menu Utama Aplikasi

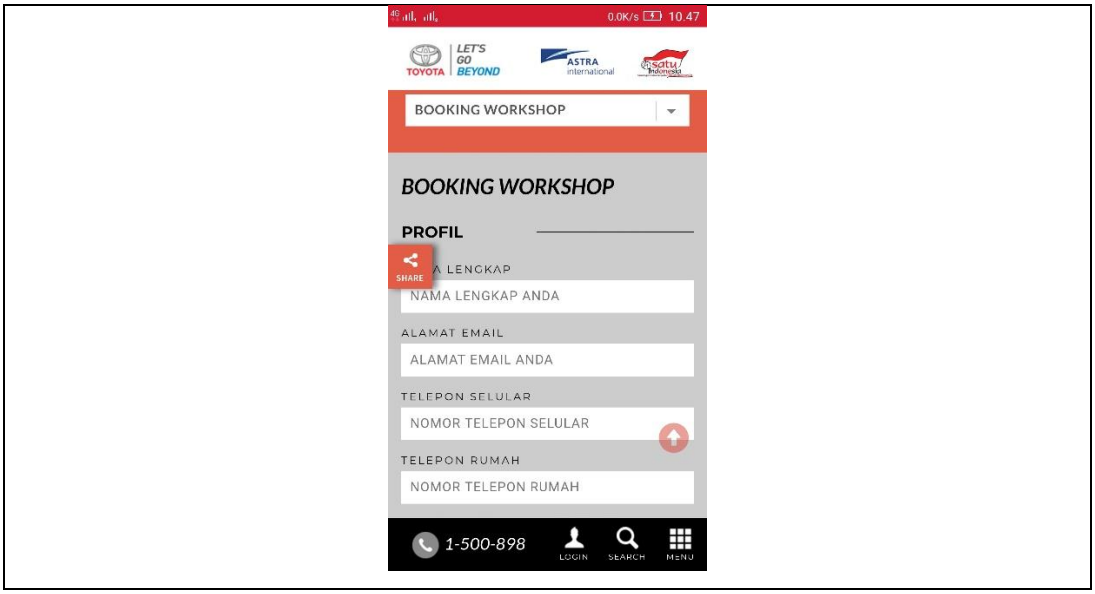
Aplikasi *mobile customisable notification* memiliki menu utama yang dapat dilihat pada Gambar 4.1



Gambar 4.1 Tampilan Menu Utama Aplikasi.

4.1.1.2 Menu Booking Service

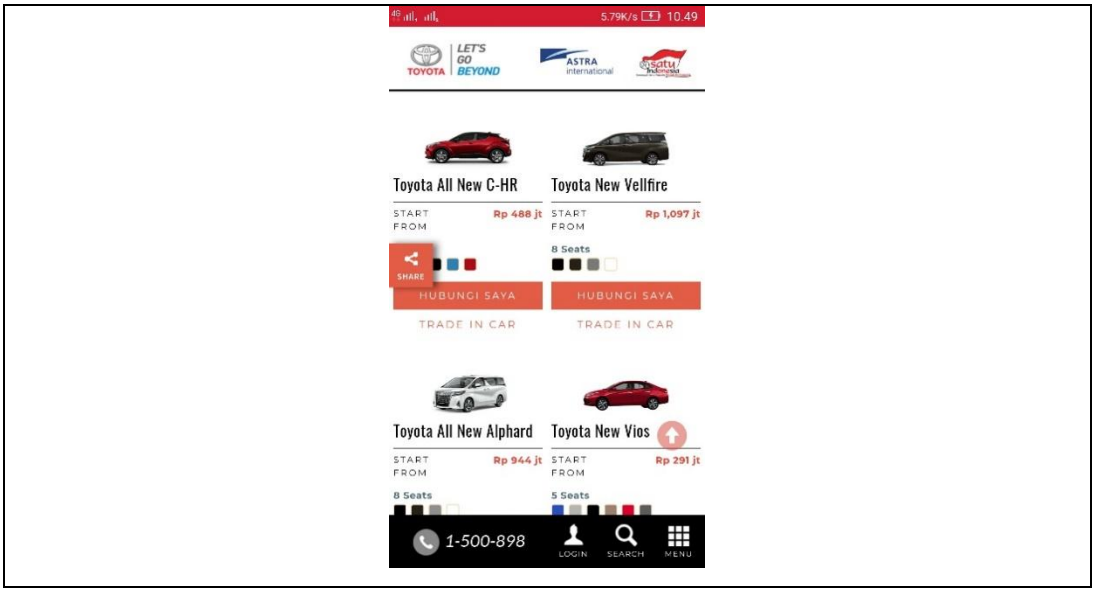
Halaman ini merupakan halaman yang berisi mengenai informasi booking service jika ingin melakukan booking service. *Interface* dari halaman ini dapat dilihat pada Gambar 4.2.



Gambar 4.2 Tampilan Menu Booking service.

4.1.1.3 Menu Catalog

Halaman ini merupakan halaman yang berisi mengenai informasi kendaraan-kendaraan terbaru yang ada di catalog tersebut. *Interface* dari halaman ini dapat dilihat pada Gambar 4.3.



Gambar 4.3 Tampilan Menu catalog.

4.1.1.4 Menu Trade IN

Halaman ini merupakan halaman yang berisi mengenai informasi langkah-langkah untuk melakukan Tradein . *Interface* dari halaman ini dapat dilihat pada Gambar 4.4.

Gambar 4.4 Tampilan Menu Trade IN.

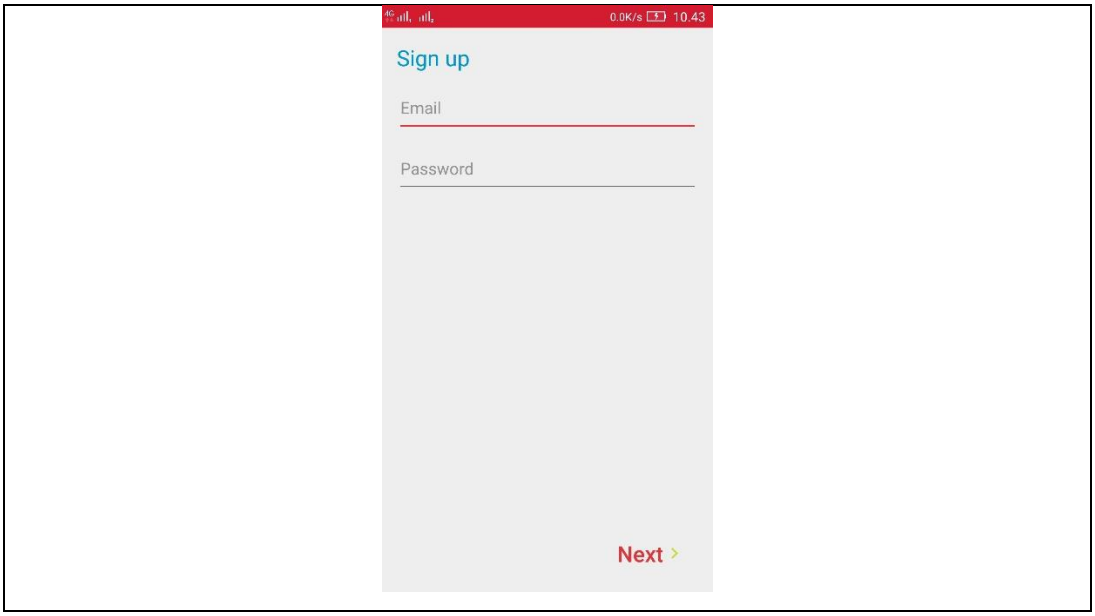
4.1.1.5 Menu Promo

Halaman ini merupakan halaman yang berisi mengenai informasi promo dan diskon yang terdapat didealer Auto2000. *Interface* dari halaman ini dapat dilihat pada Gambar 4.5.

Gambar 4.5 Tampilan Menu Promo.

4.1.1.6 Menu Registrasi

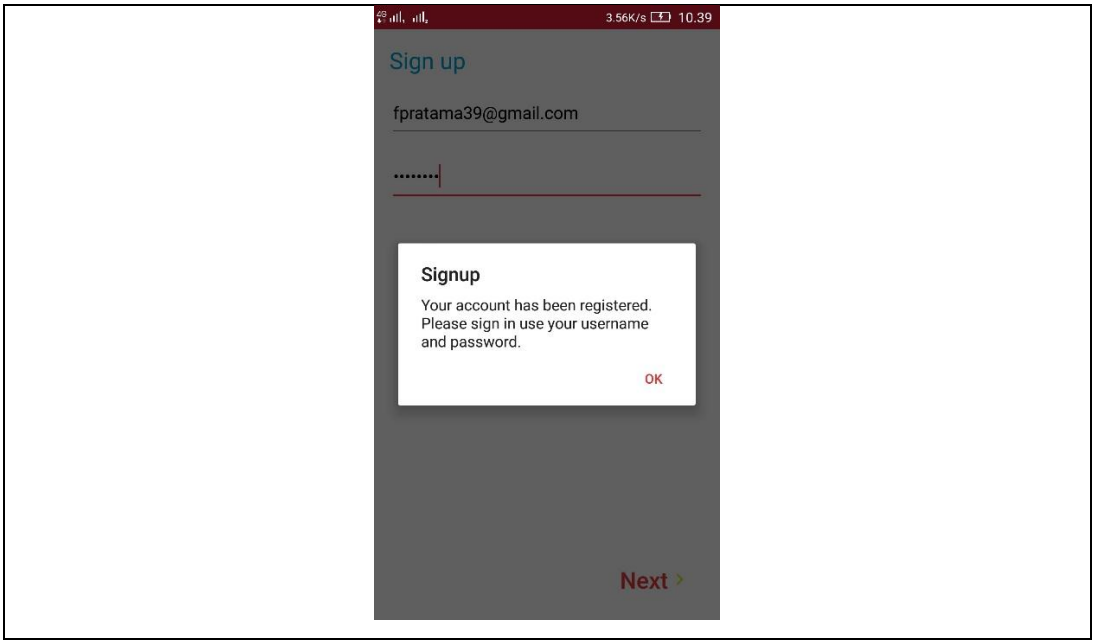
Halaman registrasi ini digunakan apabila pengguna ingin menjadi pelanggan dari *aplikasi custumisable notifikasi* di Auto2000. *Interface* halaman registrasi dapat dilihat pada Gambar 4.6.



Gambar 4.6 Tampilan Menu registrasi.

4.1.1.7 Menu registrasi berhasil

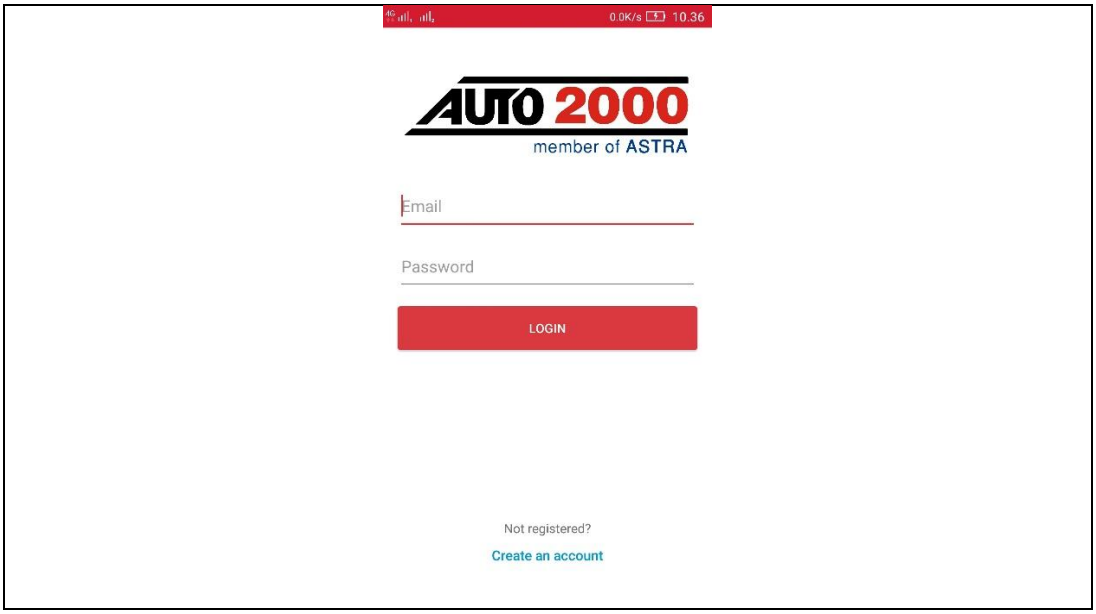
Halaman ini merupakan halaman yang berisii mengenai informasi jika regstrasi berhasil dilakukan. *Interface* dari halaman ini dapat dilihat pada Gambar 4.7.



Gambar 4.7 Tampilan Registrasi berhasil.

4.1.1.8 Menu login

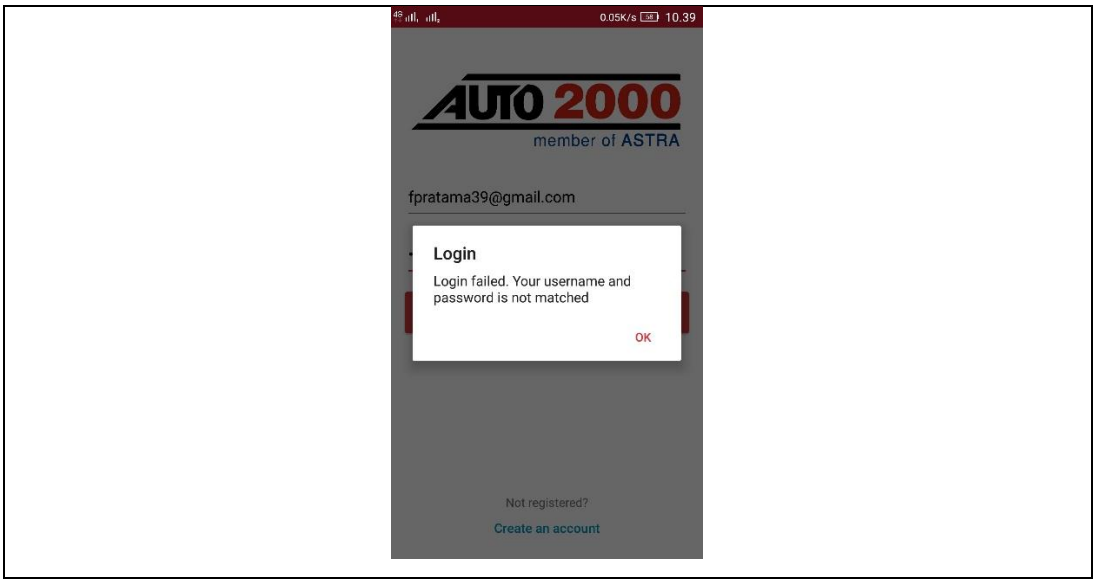
Halaman ini merupakan halaman yang berisi informasi mengenai langkah-langkah dalam melakukan Login. *Interface* dari halaman ini dapat dilihat pada Gambar 4.8.



Gambar 4.8 Tampilan Form login.

4.1.1.9 Menu From login (failed)

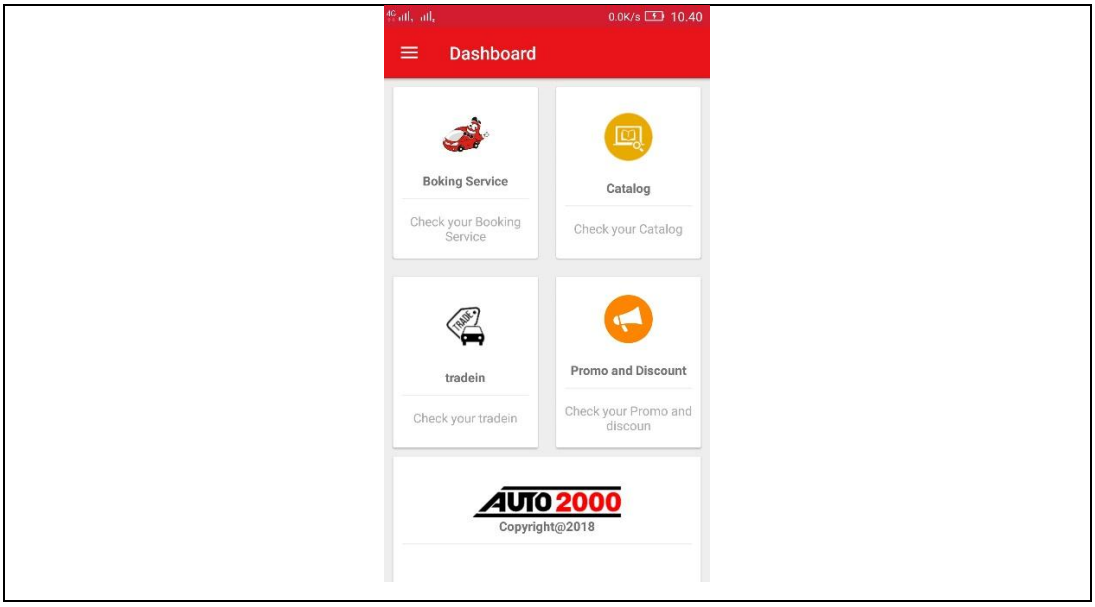
Halaman ini merupakan halaman yang berisi informasi mengenai langkah-langkah dalam melakukan Login jika melakukan login salah terdapat pemberitahuan seperti digambar. *Interface* dari halaman ini dapat dilihat pada Gambar 4.9.



Gambar 4.9 Tampilan Form login jika failed.

4.1.1.10 Menu From login (berhasil)

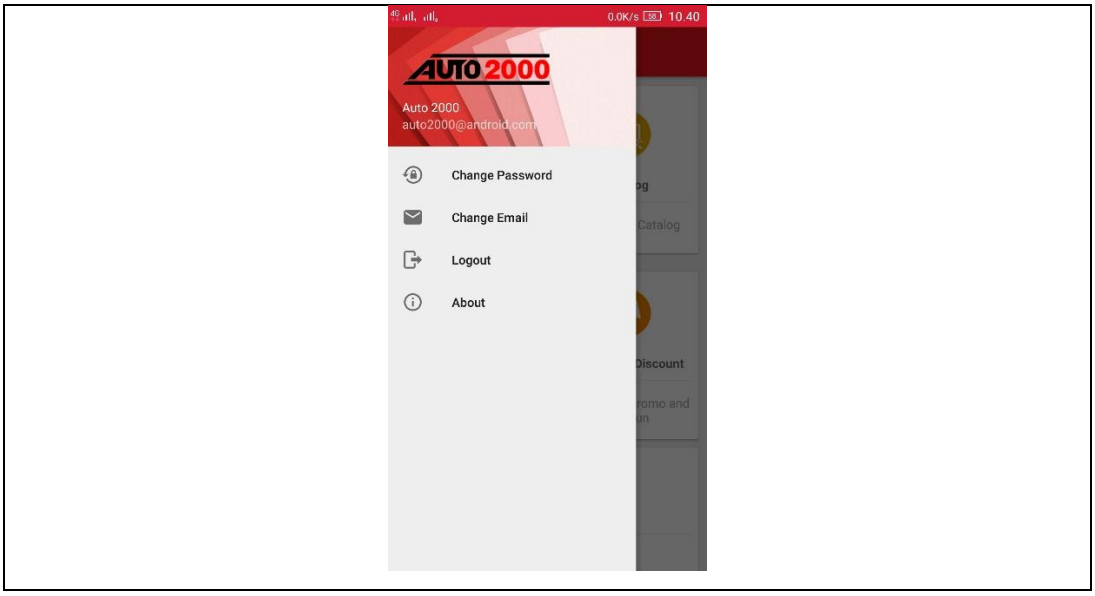
Halaman ini merupakan halaman yang berisi informasi mengenai langkah-langkah dalam melakukan Login, jika login berhasil maka akan langsung masuk ke menu utama. *Interface* dari halaman ini dapat dilihat pada Gambar 4.10.



Gambar 4.10 Tampilan Form login jika berhasil.

4.1.1.11 Menu Drawer (dashboard)

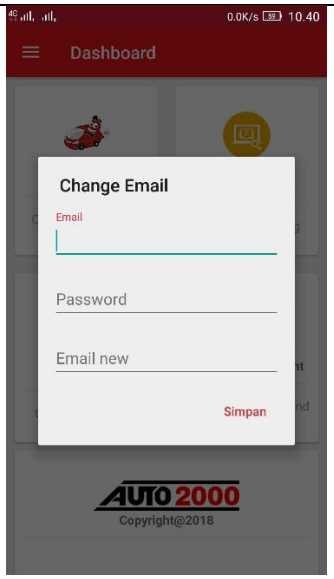
Halaman ini merupakan halaman yang berisi informasi mengenai langkah-langkah dalam melakukan pergantian password, pergantian email, about dan logut. *Interface* dari halaman ini dapat dilihat pada Gambar 4.11.



Gambar 4.11 Tampilan Menu dashboard.

4.1.1.12 Menu pergantian email

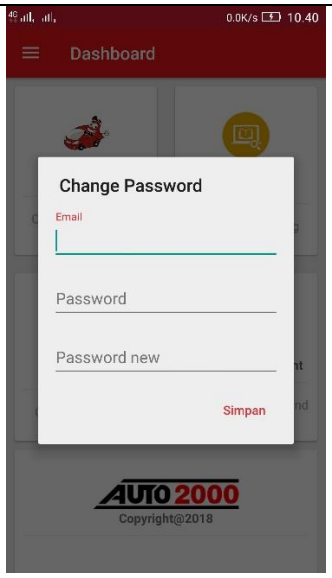
Halaman ini merupakan halaman yang berisi informasi mengenai langkah-langkah dalam melakukan pergantian email. *Interface* dari halaman ini dapat dilihat pada Gambar 4.12.



Gambar 4.12 Tampilan Menu pergantian email.

4.1.1.13 Menu pergantian password

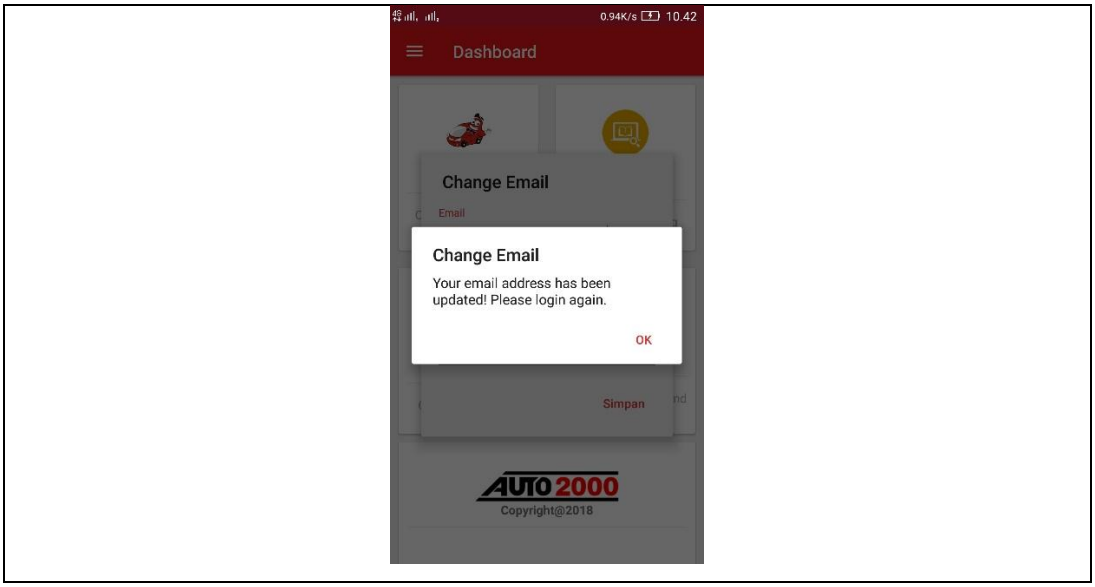
Halaman ini merupakan halaman yang berisi informasi mengenai langkah-langkah dalam melakukan pergantian password. *Interface* dari halaman ini dapat dilihat pada Gambar 4.13.



Gambar 4.13 Tampilan Menu pergantian password.

4.1.1.14 Menu berhasil mengganti email.

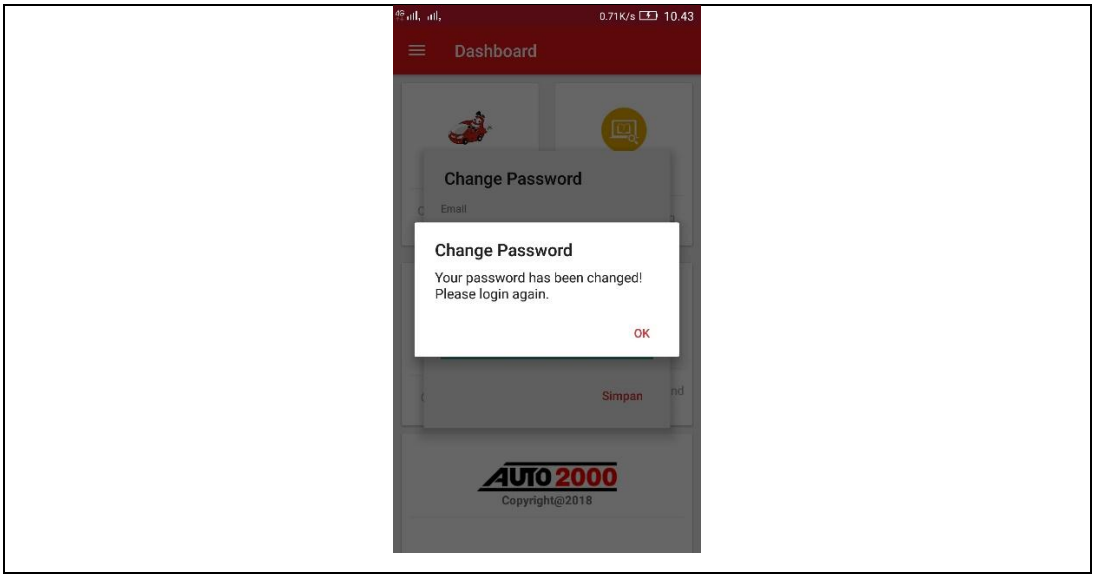
Halaman ini merupakan halaman yang berisi informasi mengenai keberhasilan mengganti email baru. *Interface* dari halaman ini dapat dilihat pada Gambar 4.14



Gambar 4.14 Tampilan berhasil mengganti email baru.

4.1.1.15 Menu berhasil meganti password

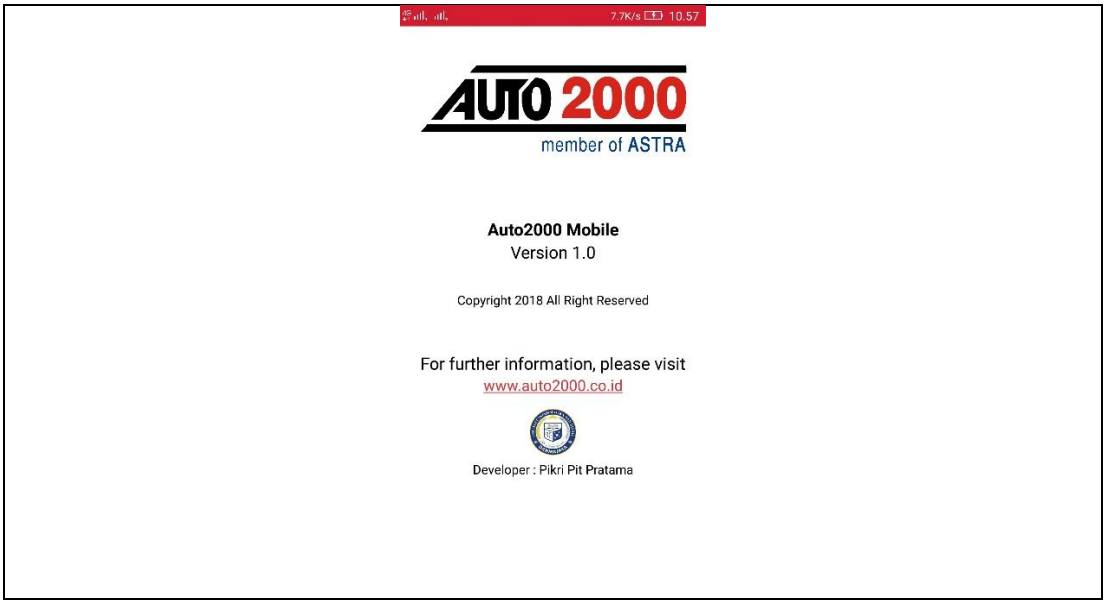
Halaman ini merupakan halaman yang berisi informasi mengenai keberhasilan mengganti password baru. *Interface* dari halaman ini dapat dilihat pada Gambar 4.15.



Gambar 4.15 Tampilan berhasil mengganti password baru.

4.1.1.16 Menu About

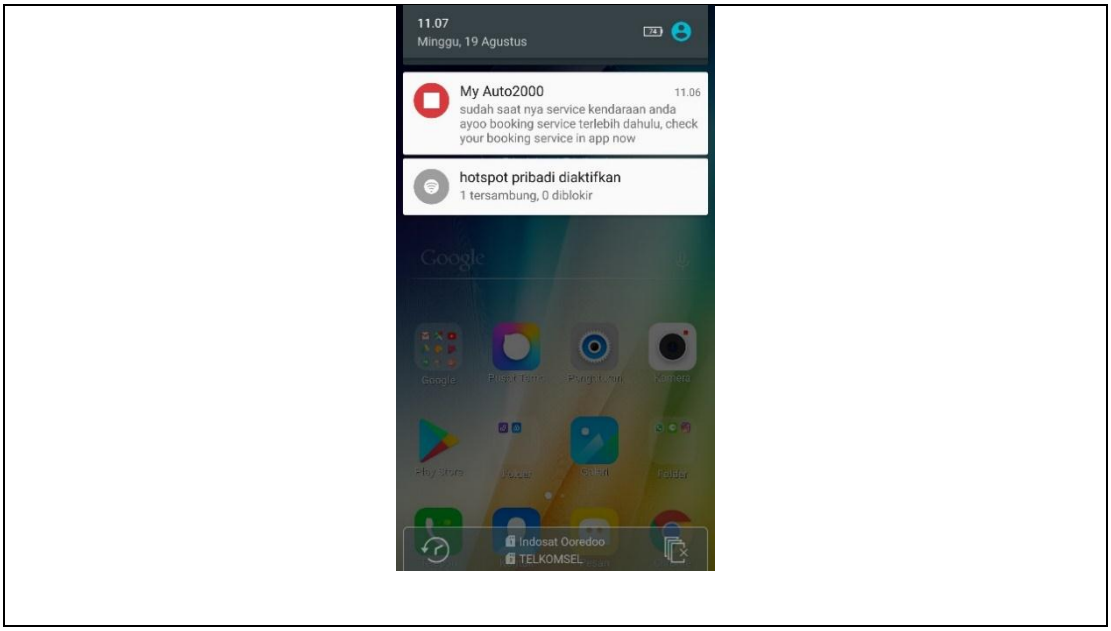
Halaman ini merupakan halaman yang berisi informasi tentang aplikasi yang dibuat. *Interface* dari halaman ini dapat dilihat pada Gambar 4.16.



Gambar 4.16 Tampilan menu about.

4.1.1.17 Tampilan Pesan Notifikasi Diterima

Halaman ini adalah berisi pesan notifikasi yang sudah diterima ke pengguna aplikasi. *Interface* halaman ini dapat dilihat pada gambar 4.17.



Gambar 4.17 Tampilan pesan notifikasi berhasil diterima pengguna.

4.2 Pembahasan

4.2.1 Pengujian *System*

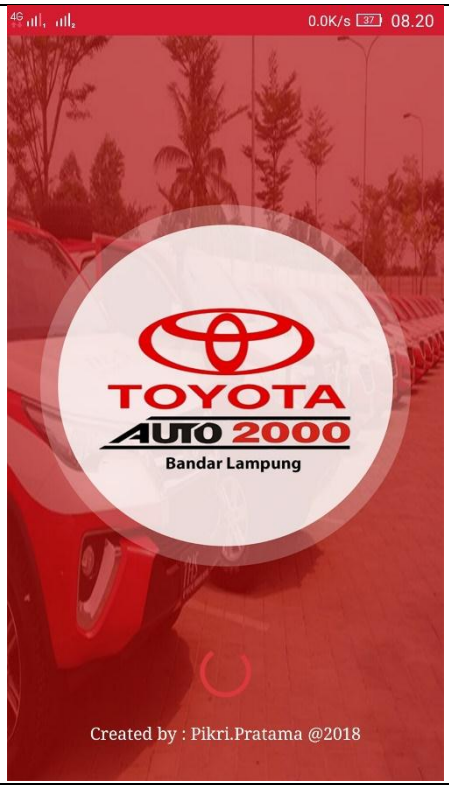
Proses pengujian sistem dilakukan dengan menggunakan metode *black box* testing. Pada saat melakukan pengujian dengan menggunakan *smartphone* Lenovo vibe shot, *ram* 3gb, *rom* 32gb, *Android API 27 Lollipop 5.1* proses berjalan dengan baik, semua *menu* aplikasi dapat diakses. Aplikasi ini bisa digunakan di *smartphone* apa saja yang sudah berbasis *android* dengan minimum versi *5.1 API 19 (Lollipop)* atau yang terbaru.

4.2.2 Pengujian Program

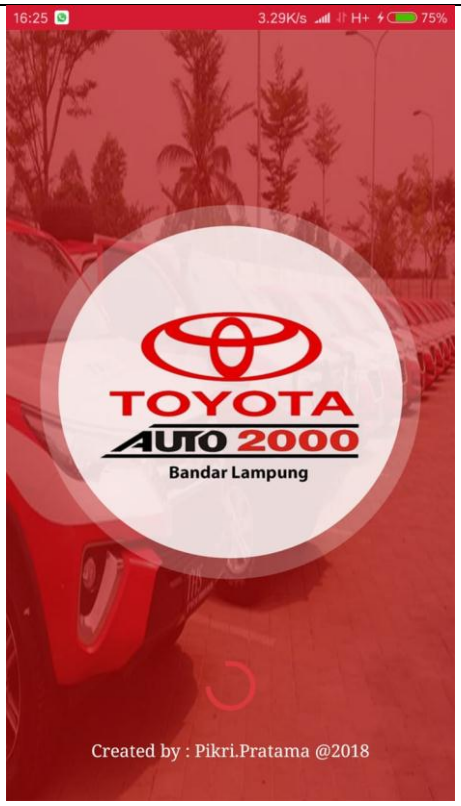
4.2.2.1 Pengujian Pada *Splash Screen*

Hasil uji aplikasi pada *splash screen* terhadap beberapa merk *smartphone* berbasis *android* akan dijelaskan pada tabel 4.1 halaman berikut:

Tabel 4.1. Pengujian *Splash Screen*

No	Item Uji	Tipe Item	Gambar	Keterangan
1	*Processor*	Qualcomm MSM8939 (8 Core 1,7GHz)		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	3 GB		
	Merk	Lenovo		
	Android	Lollipop		
	Layar	5.0 Inch		

Tabel 4.1. (Lanjutan) Pengujian *Splash Screen*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Hexa-core Max 1,8GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	2 GB		
	Merk	Xiaomi mi 4c		
	Android	nouget		
	Layar	5.0 Inch		

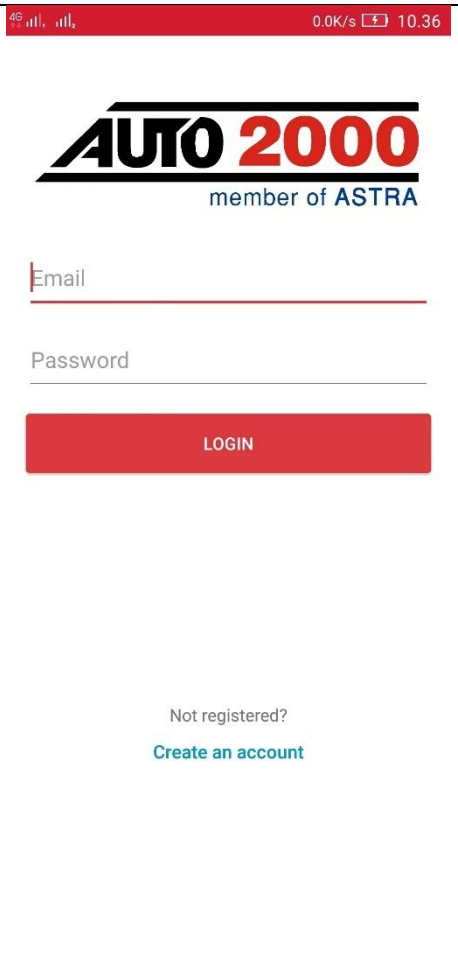
Tabel 4.1. (Lanjutan) Pengujian *Splash Screen*

No	Item Uji	Tipe Item	Gambar	Keterangan
3	*Processor*	Qualcomm MSM8998 (Octa-core 1.9 GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	4 GB		
	Merk	Samsung Galaxy S8		
	Android	Oreo		
	Layar	5.8 Inch		

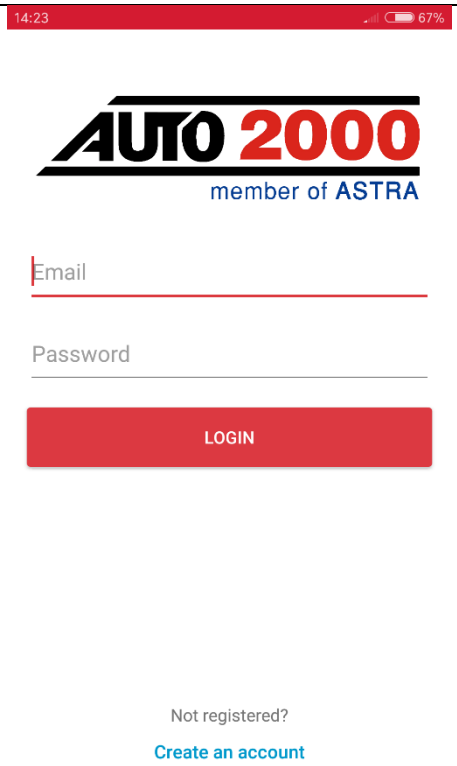
4.2.2.2 Pengujian Pada *Menu* Login

Hasil uji aplikasi pada *menu* Login terhadap beberapa merk *smartphone* berbasis *android* akan dijelaskan pada tabel 4.2 di bawah ini:

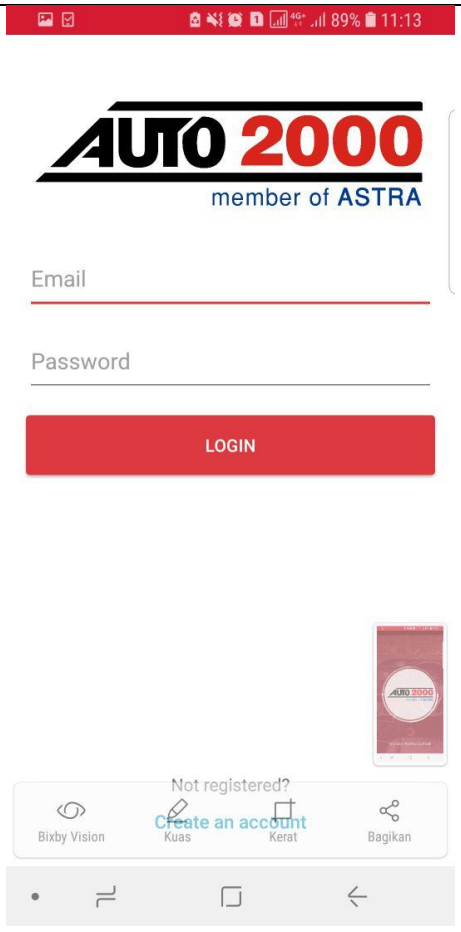
Tabel 4.2. Pengujian *Menu Login*

No	Item Uji	Tipe Item	Gambar	Keterangan
1	*Processor*	Qualcomm MSM8939 (8 Core 1,7GHz)		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	3 GB		
	Merk	Lenovo		
	Android	Lollipop		
	Layar	5.0 Inch		

Tabel 4.2. (Lanjutan) Pengujian *Menu login*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Hexa-core Max 1,8GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	2 GB		
	Merk	Xiaomi mi 4c		
	Android	nouget		
	Layar	5.0 Inch		

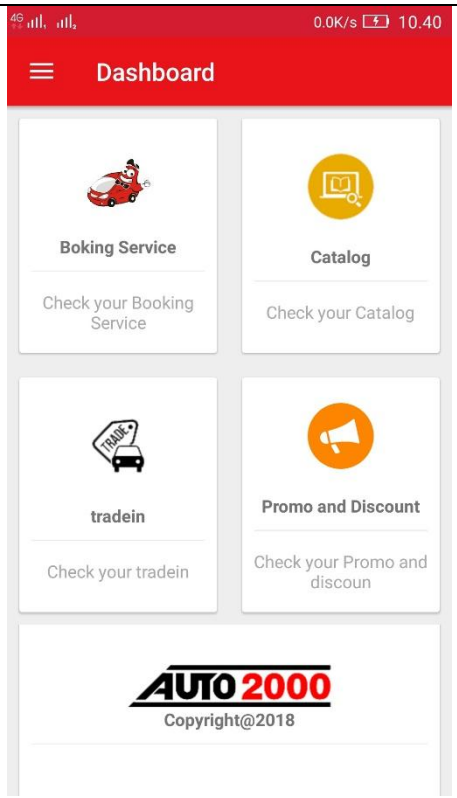
Tabel 4.2. (Lanjutan) Pengujian *Menu Login*

No	Item Uji	Tipe Item	Gambar	Keterangan
3	*Processor*	Qualcomm MSM8998 (Octa-core 1.9 GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	4 GB		
	Merk	Samsung Galaxy S8		
	Android	Oreo		
	Layar	5.8 Inch		

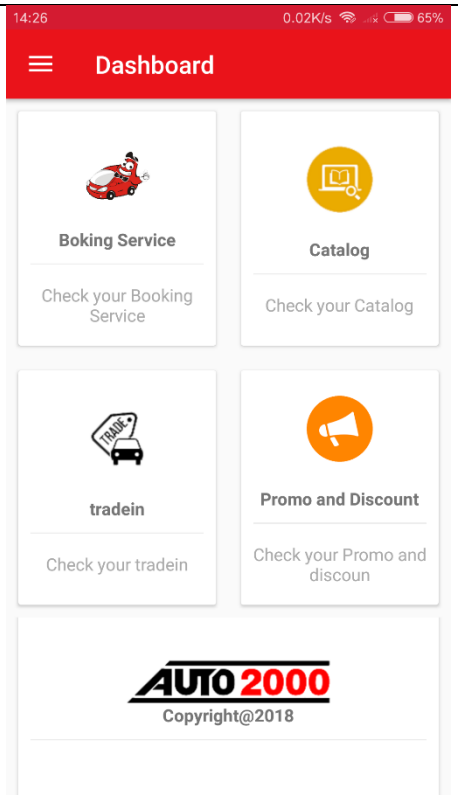
4.2.2.3 Pengujian Pada *Menu* Utama

Hasil uji aplikasi pada *menu* utama terhadap beberapa merk *smartphone* berbasis *android* akan dijelaskan pada tabel 4.3 di bawah ini:

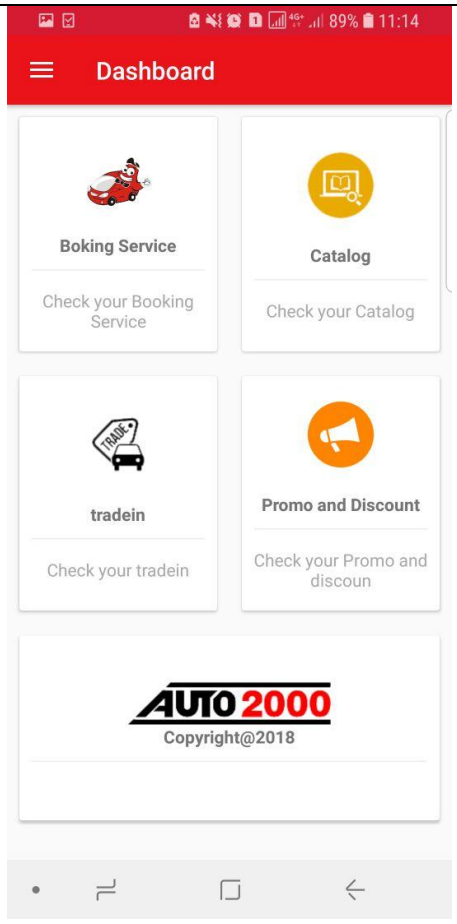
Tabel 4.3. Pengujian *Menu Utama*

No	Item Uji	Tipe Item	Gambar	Keterangan
1	*Processor*	Qualcomm MSM8939 (8 Core 1,7GHz)		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	3 GB		
	Merk	Lenovo		
	Android	Lollipop		
	Layar	5.0 Inch		

Tabel 4.3. (Lanjutan) Pengujian *Menu Utama*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Hexa-core Max 1,8GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	2 Gb		
	Merk	Xiamoi mi 4c		
	Android	Nouget		
	Layar	5.0 Inch		

Tabel 4.3. (Lanjutan) Pengujian *Menu Utama*

No	Item Uji	Tipe Item	Gambar	Keterangan
3	*Processor*	Qualcomm MSM8998 (Octa-core 1.9 GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	4 GB		
	Merk	Samsung Galaxy S8		
	Android	Oreo		
	Layar	5.8 Inch		

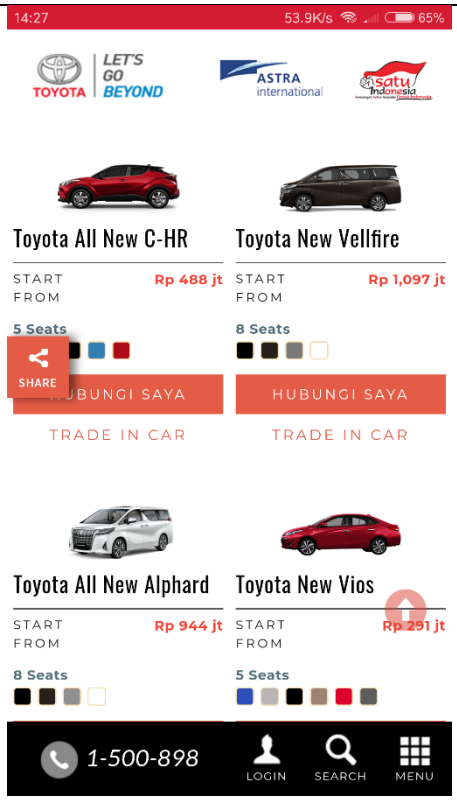
4.2.2.4 Pengujian Pada Beberapa *Menu*

Hasil uji aplikasi pada beberapa *menu* terhadap beberapa merk *smartphone* berbasis *android* akan dijelaskan pada tabel 4.4 di halaman berikut:

Tabel 4.4. Pengujian *Menu Booking Service*

No	Item Uji	Tipe Item	Gambar	Keterangan
1	*Processor*	Qualcomm MSM8939 (8 Core 1,7GHz)		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	3 GB		
	Merk	Lenovo		
	Android	Lollipop		
	Layar	5.0 Inch		

Tabel 4.4. (Lanjutan) Pengujian *Menu Catlog*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Hexa-core Max 1,8GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	2 GB		
	Merk	Xiaomi Mi 4c		
	Android	Nouget		
	Layar	5.0 Inch		

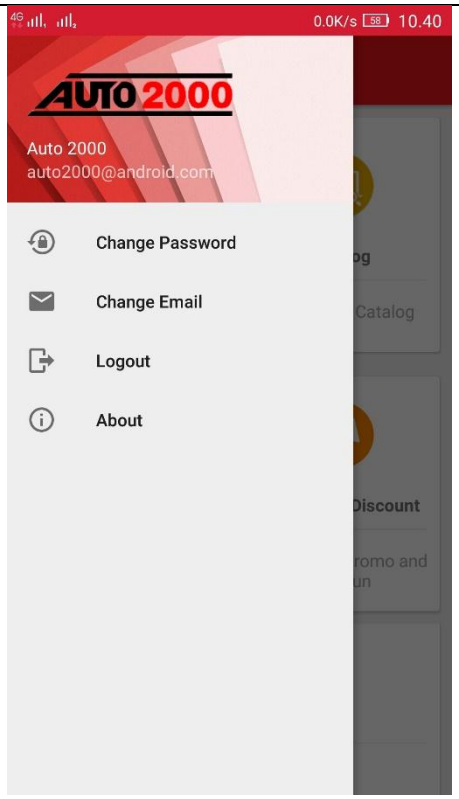
Tabel 4.4. (Lanjutan) Pengujian *Menu Trade IN*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Qualcomm MSM8998 (Octa-core 1.9 GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	4 GB		
	Merk	Samsung Galaxy S8		
	Android	Oreo		
	Layar	5.8 Inch		

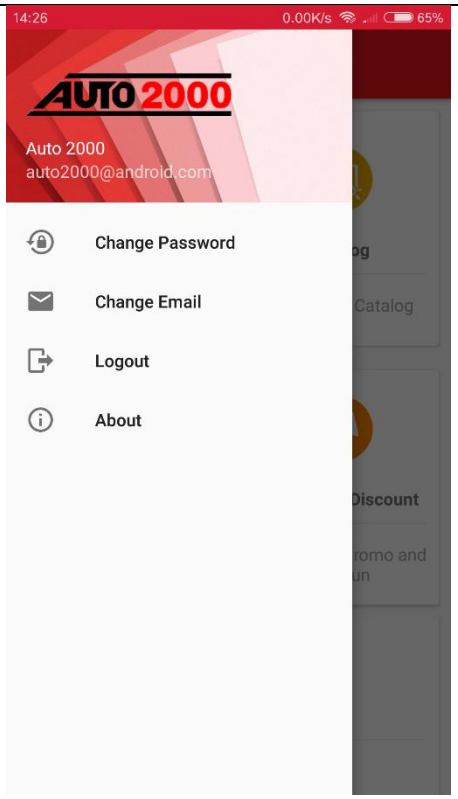
4.2.2.5 Pengujian Pada Menu Dashboard (*drawer*)

Hasil uji aplikasi pada *menu dashboard* terhadap beberapa merk *smartphone* berbasis *android* akan dijelaskan pada tabel 4.5 di halaman berikut:

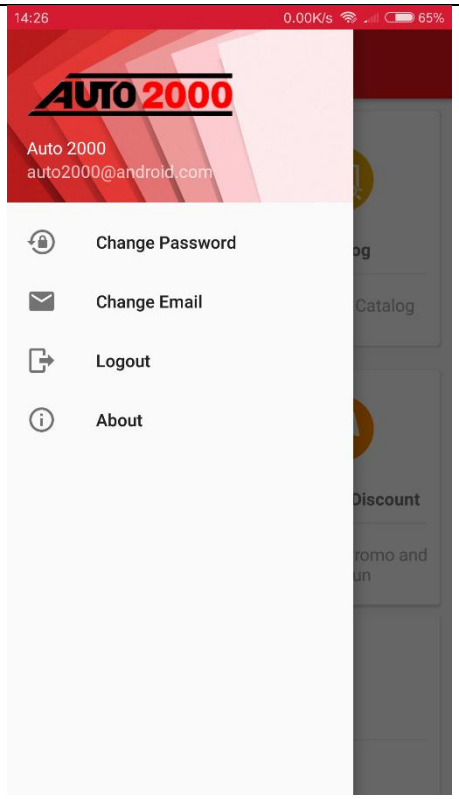
Tabel 4.5. Pengujian *Dashboard*

No	Item Uji	Tipe Item	Gambar	Keterangan
1	*Processor*	Qualcomm MSM8939 (8 Core 1,7GHz)		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	3 GB		
	Merk	Lenovo		
	Android	Lollipop		
	Layar	5.0 Inch		

Tabel 4.5. (Lanjutan) Pengujian *Menu Dashboard*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Hexa-core Max 1,8GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	2 GB		
	Merk	Xiaomi Mi 4c		
	Android	Nouget		
	Layar	5.0 inch		

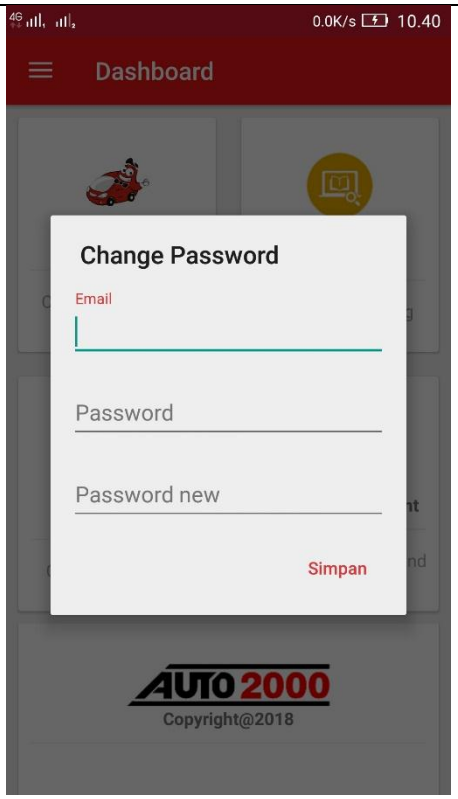
Tabel 4.5. (Lanjutan) Pengujian *Menu Dashboard*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Qualcomm MSM8998 (Octa-core 1.9 GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	4 GB		
	Merk	Samsung Galaxy S8		
	Android	Oreo		
	Layar	5.8 Inch		

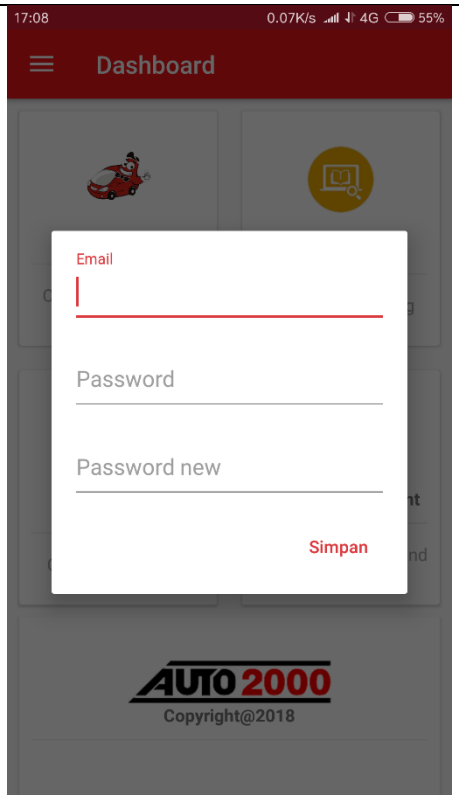
4.2.2.6 Pengujian Pada Beberapa Menu Dashboard

Hasil uji aplikasi pada beberapa *menu Dashboard* terhadap beberapa merk *smartphone* berbasis *android* akan dijelaskan pada tabel 4.6 di halaman berikut:

Tabel 4.6. Pengujian *Menu change password*

No	Item Uji	Tipe Item	Gambar	Keterangan
1	*Processor*	Qualcomm MSM8939 (8 Core 1,7GHz)		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	3 GB		
	Merk	Lenovo		
	Android	Lollipop		
	Layar	5.0 Inch		

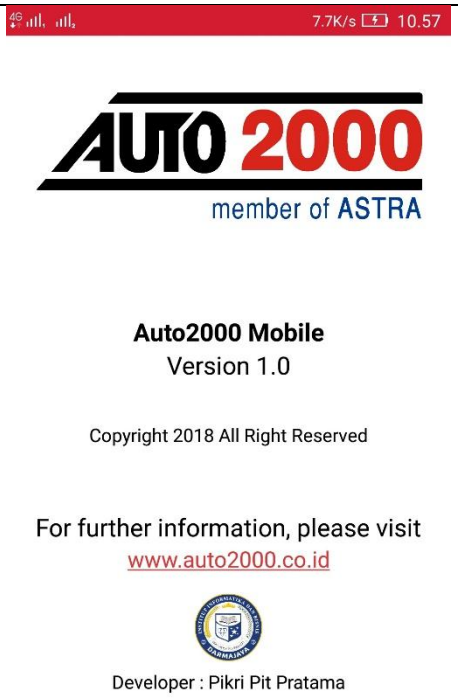
Tabel 4.6. (Lanjutan)Pengujian *change Email*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Hexa-core Max 1,8GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	2 GB		
	Merk	Xiaomi Mi4c		
	Android	Nouget		
	Layar	5.0 Inch		

4.2.2.7 Pengujian Pada Menu about

Hasil uji aplikasi pada *menu about* terhadap beberapa merk *smartphone* berbasis *android* akan dijelaskan pada tabel 4.7 di halaman berikut:

Tabel 4.7. Pengujian *Menu about*

No	Item Uji	Tipe Item	Gambar	Keterangan
1	*Processor*	Qualcomm MSM8939 (8 Core 1,7GHz)		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	3 GB		
	Merk	Lenovo		
	Android	Lollipop		
	Layar	5.0 Inch		

Tabel 4.7. (Lanjutan)Pengujian *about*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Hexa-core Max 1,8GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	2 GB		
	Merk	Xiaomi Mi4c		
	Android	Nouget		
	Layar	5.0 Inch		

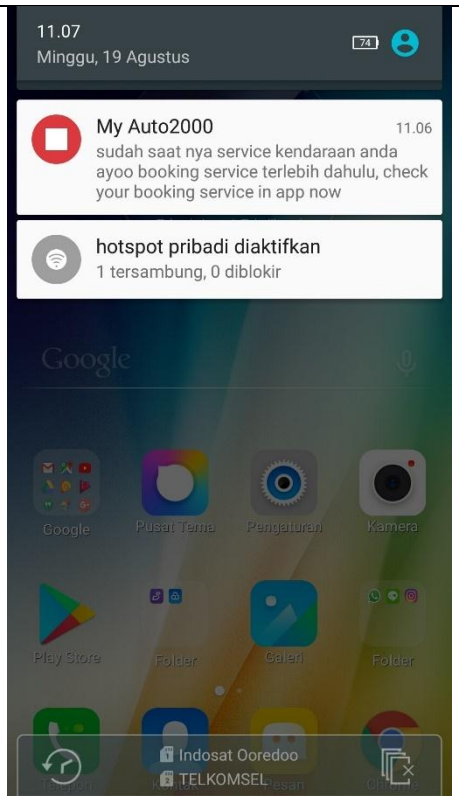
Tabel 4.7. (Lanjutan)Pengujian *about*

No	Item Uji	Tipe Item	Gambar	Keterangan
3	*Processor*	Qualcomm MSM8998 (Octa-core 1.9 GHz		Aplikasi dapat di buka dengan lancar dan dapat berfungsi dengan baik.
	RAM	4 GB		
	Merk	Samsung Galaxy S8		
	Android	Oreo		
	Layar	5.8 Inch		

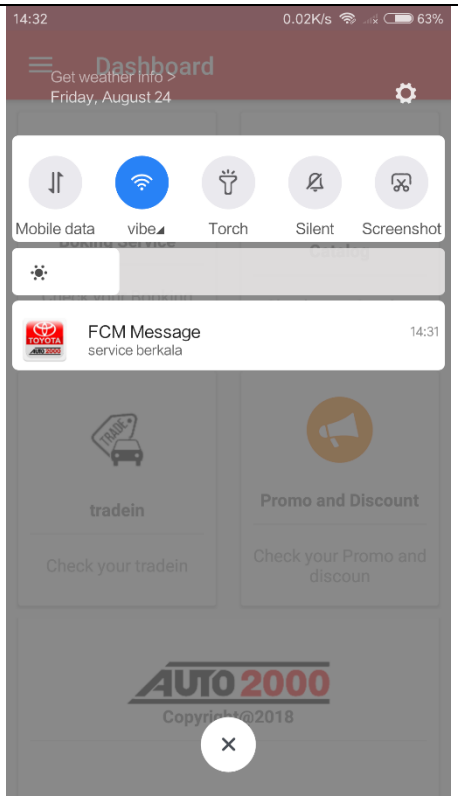
4.2.2.8 Pengujian Pada *Notifikasi*

Hasil uji aplikasi pada *notifikasi* terhadap beberapa merk *smartphone* berbasis *android* akan dijelaskan pada tabel 4.8 di halaman berikut:

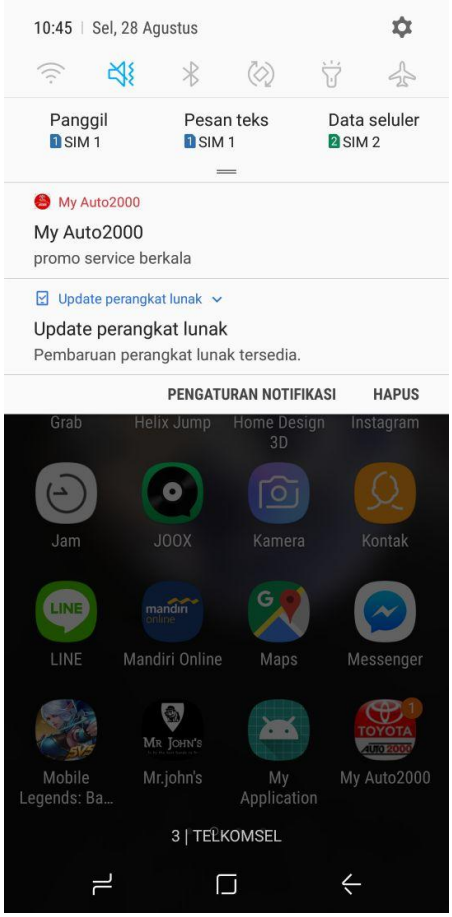
Tabel 4.8. Pengujian *Menu promo*

No	Item Uji	Tipe Item	Gambar	Keterangan
1	*Processor*	Qualcomm MSM8939 (8 Core 1,7GHz)		Notifikasi dapat di terima dengan lancar dan dapat berfungsi dengan baik.
	RAM	3 GB		
	Merk	Lenovo		
	Android	Lollipop		
	Layar	5.0 Inch		

Tabel 4.8. (Lanjutan) Pengujian *Notifikasi*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Hexa-core Max 1,8GHz		Notifikasi dapat di terima dengan lancar dan dapat berfungsi dengan baik.
	RAM	2 GB		
	Merk	Xiaomi Mi 4c		
	Android	Nouget		
	Layar	5.0 Inch		

Tabel 4.8. (Lanjutan) Pengujian *Notifikasi*

No	Item Uji	Tipe Item	Gambar	Keterangan
2	*Processor*	Qualcomm MSM8998 (Octa-core 1.9 GHz		Notifikasi dapat di terima dengan lancar dan dapat berfungsi dengan baik.
	RAM	4 GB		
	Merk	Samsung Galaxy S8		
	Android	Oreo		
	Layar	5.8 Inch		

4.2.3 Kelebihan Aplikasi

Terdapat beberapa kelebihan yang terdapat pada aplikasi *customisable notifikasi dalam pelayanan pelanggan*, yaitu :

a) Aplikasi yang dibangun berbasis *mobile android* sehingga bisa diakses dimanapun dan kapanpun menggunakan *smartphone android* yang sudah menjamur dikalangan masyarakat umum
b) Memudahkan pengguna (*customer*) dalam mendapat informasi kapan harus service berkala jika pelanggan tidak ingat kapan harus service kendaraan secara berkala.
c) Memudahkan pengguna (*customer*) dalam melakukan booking service tanpa harus datang ke tempat dengan menggunakan webview yang sudah ada dan di implementasikan ke aplikasi mobile.

4.2.4 Kekurangan Aplikasi

Dalam membangun aplikasi ini pun masih terdapat banyak kekurangan yang perlu dikembangkan kedepannnya, diantaranya adalah :

a) *View* yang ditampilkan pada aplikasi masih sangat sederhana.
b) Objek dalam penelitian hanya dilakukan di dealer Auto2000 wayhalim saja.
c) Membutuhkan koneksi yang stabil saat membuka webview yang ada pada aplikasi.
d) Aplikasi yang dibangun masih sederhana, hanya menyampaikan notifikasi service berkala, dan terdapat fitur tambahan yang ada pada web auto2000 yg dimasukan pada aplikasi seperti booking service, catalog, tradein, dan promo saja.

BAB V
SIMPULAN DAN SARAN

5.1 Simpulan

Kesimpulan dari hasil penelitian yang telah dilakukan adalah sebagai berikut :

1. Aplikasi *customisable notification dalam layanan pelanggan* dapat mengoptimalkan dan memberikan perhatian lebih kepada pelanggan untuk informasi kapan harus service kendaraan berkala.
2. Mengunakan metode pengembangan sistem model *prototype* membantu perusahaan dalam mengetahui perubahahan-perubahan spesifikasi kebutuhan *pelanggan* yang terjadi dikemudian hari.

5.2 Saran

Aplikasi yang dibangun masih memiliki banyak kelemahan, untuk itu masih perlu diperlukan perbaikan maupun pengembangan untuk penelitian berikutnya.

Saran yang diperlukan untuk pengembangan aplikasi ini adalah :

1. Aplikasi customisable notification ini dapat dikembangkan dengan menambahkan *database*, penggunaan *database* untuk merekam hasil history notifikasi service berkala.
2. Pengembangan aplikasi ini dapat dilakukan dengan menambahkan objek service kendaraan yang lain nya tidak hanya service berkala saja dan menambahkan fitur – fitur menarik lainnya.
3. Menggunakan koneksi yang stabil agar mudah membuka webview.
4. Aplikasi ini dapat di kembangkan dan dapat dijalankan di *smartphone* bersistem operasi *mobile* lainnya seperti *IOS* atau *windows mobile* serta sistem operasi *android versi 4.4 kebawah*.

DAFTAR PUSTAKA

Akhmad D Kasman.2013. Kolaborasi Dahsyat Android dengan PHP & MySQL. Yogyakarta.

Anggit Hernowo. 2016. Perancangan Dan Implementasi Sistem Penjadwalan Servis Berkala Kendaraan Bermotor Berbasis Website menggunakan Notifikasi SMS Gateway.

Arif Winandar. 2015. Penerapan notifikasi android untuk membantu penyebaran informasi dan komunikasi sivitas Universitas Darma persada

Asep saefullah, Diah Ariyani dan Andy Rienauld.2014. Sistem notifikasi antrian berbasis android.

Bobby Ghani dan Nurfiana.2013. Perancangan Sistem Informasi Promosi pada PT. Centerpoint Putra Sejahtera Berbasis SMS Gateway.

Hariyanto wibowo dan Eri Setyani.2017. Rancang Bangun *Student Payment* Dalam Mewujudkan *Customer Relationship management* (CRM) Di Perguruan Tinggi.

Kamus Besar Bahasa Indonesia.2010.Jakarta

Muhammad Irsan. 2015. Rancang Bangun aplikasi mobile notifikasi berbasis android untuk mendukung kinerja di instansi pemerintahan.

Pressman, R.S. 2012. Rekayasa Perangkat Lunak. Yogyakarta.

Ramadhan, dan Utomo. 2014. Rancang bangun aplikasi mobile untuk notifikasi jadwal kuliah berbasis android : Studi kasus STMIK Provisi Semarang. Jurnal Teknologi Informasi dan Komunikasi, ISSN : 2087 - 0868, Volume 5 Nomor 2 Agustus 2014.

Rosa A. S dan M Shalahuddin.2016.Rekayasa Perangkat Lunak(terstruktur dan berorientasi objek).Bandung: Informatika Bandung.

Thomas Suselo. 2010. Pembangunan Aplikasi SMS untuk Pengingat Perawatan Berkala Kendaraan.

Yogiswara, dan Astriyanto.2018. Penerapan Web Service dan firebase notification pada pengembangan aplikasi gerakan nasi bungkus jember berbasis android.

https://firebase.google.com/docs/cloud-messaging/?hl=id

(diakses tanggal 20 November 2017, jam 19:03 wib)

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

(diakses tanggal 20 November 2017, jam 19:05 wib)

https://www.codepolitan.com/mengenal-uml-diagram-use-case

(diakses tanggal 20 November 2017, jam 19:10 wib)

https://www.codepolitan.com/mengenal-uml-contoh-uml-diagram-model-activity-diagram

(diakses tanggal 20 November 2017, jam 19:23 wib)

https://www.codepolitan.com/belajar-uml-sequence-diagram-57fdb1a5ba777-17044

(diakses tanggal 20 November 2017 jam 19:30 wib)

https://www.json.org/json-id.html

(diakses tanggal 22 November 2017 jam 20:00 wib)

https://haidibarasa.wordpress.com/2013/07/06/pengertian-android-sdk-software-development-kit/

(diakses tanggal 22 November 2017 jam 20:32 wib)

Manifest.XML

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.google.myauto2000">

    <uses-permission android:name="android.permission.INTERNET" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/iconapp"
        android:hardwareAccelerated="true"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/iconapp"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".flash">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
        <activity android:name=".Main2Activity" />

        <meta-data

android:name="com.google.firebase.messaging.default_notification_icon"
            android:resource="@mipmap/icon1" />
        <meta-data

android:name="com.google.firebase.messaging.default_notification_color"
            android:resource="@color/colorAccent" />

        <service android:name=".MyFirebaseMessagingService">
            <intent-filter>
                <action android:name="com.google.firebase.MESSAGING_EVENT"
/>
            </intent-filter>
        </service>
        <service android:name=".MyFirebaseInstanceIDService">
            <intent-filter>
                <action
android:name="com.google.firebase.INSTANCE_ID_EVENT" />
            </intent-filter>
        </service>
        <activity
            android:name=".DashboardActivity"
            android:label="@string/title_activity_dashboard"
            android:theme="@style/AppTheme.NoActionBar" />
        <activity android:name=".SignupActivity" />
        <activity android:name=".MainActivity" />
        <activity android:name=".dialog_fragment_about" />
        <activity android:name=".booking_services" />
        <activity android:name=".catalog" />
        <activity android:name=".tradein" />
        <activity android:name=".promo"></activity>
    </application>
</manifest>
```

Main 2 Activity(Main Login).Java

```
package com.example.google.myauto2000;

import android.content.DialogInterface;
import android.content.Intent;
import android.support.annotation.NonNull;
import android.support.design.widget.Snackbar;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RelativeLayout;


import com.example.google.myauto2000.DashboardActivity;
import com.example.google.myauto2000.R;
import com.example.google.myauto2000.SignupActivity;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

import butterknife.BindView;
import butterknife.ButterKnife;
import butterknife.OnClick;

public class Main2Activity extends AppCompatActivity {

    @BindView(R.id.relative_layout_progress_activity_main)
    RelativeLayout relativeLayoutProgress;
    @BindView(R.id.edit_text_username_activity_main)
    EditText editTextUsername;
    @BindView(R.id.edit_text_password_activity_main)
    EditText editTextPassword;

    private FirebaseAuth firebaseAuth;
    private boolean loggedIn;

    @Override
    public void onBackPressed(){
        new android.app.AlertDialog.Builder(this)
                .setMessage("Apakah anda yakin ingin keluar ?")
                .setCancelable(false)
                .setPositiveButton("Ya", new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {


                        Main2Activity.this.finish();
                    }
                })
                .setNegativeButton("Tidak", null).show();
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);
        initFirebase();
        ButterKnife.bind(this);
        loggedIn = isLoggedIn();
        if (loggedIn) {
            //  go to dashboard
            goToDashboard();
        }
    }


    private void initFirebase() {
        firebaseAuth = FirebaseAuth.getInstance();
    }

    private void goToDashboard() {
        Intent intent = new Intent(this, DashboardActivity.class);
        intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
        startActivity(intent);
    }

    @OnClick({R.id.button_login_activity_main,
R.id.button_sign_up_activity_main})
    public void onClick(Button button) {
        switch (button.getId()) {
            case R.id.button_login_activity_main:
                String username =
editTextUsername.getText().toString().trim();
                String password =
editTextPassword.getText().toString().trim();
                login(username, password);
                break;
            case R.id.button_sign_up_activity_main:
                //  go to form pendaftaran
                startActivity(new Intent(this, SignupActivity.class));
                break;
        }
    }

    private void login(final String username, final String password) {
        if (TextUtils.isEmpty(username)) {
            Snackbar.make(findViewById(android.R.id.content),
R.string.error_message_username_empty, Snackbar.LENGTH_LONG)
                    .show();
        } else if (TextUtils.isEmpty(password)) {
            Snackbar.make(findViewById(android.R.id.content),
R.string.error_message_password_empty, Snackbar.LENGTH_LONG)
                    .show();
        } else {
            //  do login
            showProgress();
            firebaseAuth.signInWithEmailAndPassword(username, password)
                    .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
                        @Override
                        public void onComplete(@NonNull Task<AuthResult>
task) {
                            hideProgress();
```

```
                            if (task.isSuccessful()) {
                                //  login sucess
                                //  go to dashboard
                                goToDashboard();
                            } else {
                                //  login failed
                                showMessageBox("Login failed. Your
username and password is not matched");
                            }
                        }
                    });

        }
    }

    private void hideProgress() {
        relativeLayoutProgress.setVisibility(View.GONE);
        editTextUsername.setEnabled(true);
        editTextPassword.setEnabled(true);
    }

    private void showProgress() {
        relativeLayoutProgress.setVisibility(View.VISIBLE);
        editTextUsername.setEnabled(false);
        editTextPassword.setEnabled(false);
    }

    private void showMessageBox(String message) {
        AlertDialog.Builder alertDialogBuilder = new
AlertDialog.Builder(this);
        alertDialogBuilder.setTitle("Login");
        alertDialogBuilder.setMessage(message);
        alertDialogBuilder.setCancelable(false);
        alertDialogBuilder.setPositiveButton("OK", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                dialogInterface.dismiss();
            }
        });
        alertDialogBuilder.show();
    }

    public boolean isLoggedIn() {
        //  user logged in
        return firebaseAuth.getCurrentUser() != null;
    }
}
```

Home Fragment.Java

```
package com.example.google.myauto2000;

import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.RectF;
import android.net.Uri;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.Snackbar;
import android.support.v4.app.Fragment;
import android.support.v7.widget.DividerItemDecoration;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.support.v7.widget.helper.ItemTouchHelper;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ProgressBar;

import com.example.google.myauto2000.EditDataDialogFragment;
import com.example.google.myauto2000.NewDataFragment;
import com.example.google.myauto2000.R;
import com.example.google.myauto2000.AdapterData;
import com.example.google.myauto2000.Data;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.ChildEventListener;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import org.greenrobot.eventbus.EventBus;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import butterknife.BindView;
import butterknife.ButterKnife;
import butterknife.OnClick;

public class HomeFragment extends Fragment {

    private static final String TAG = "HomeFragmentTAG";
    private View view;

    private FirebaseDatabase firebaseDatabase;
    private FirebaseAuth firebaseAuth;
    private List<Data> listData;
    private List<String> listKey;
```

```
    private AdapterData adapterData;
    private Paint paint = new Paint();
    private int editPosition;

    public HomeFragment() {
        // Required empty public constructor
    }

    private void initFirebaseListener() {
        final List<Data> listDataNew = new ArrayList<>();
        firebaseDatabase.getReference()
                .child("member")

.child(firebaseAuth.getCurrentUser().getEmail().replaceAll("\\.", "_"))
.child("data")
.addChildEventListener(new ChildEventListener() {

@Override
public void onChildAdded(DataSnapshot dataSnapshot, String s) {
Log.d(TAG, "onChildAdded");
                    }

@Override
         public void onChildChanged(DataSnapshot dataSnapshot, String s) {
                 //  do something
                Log.d(TAG, "onChildChanged");
                Map<String, String> mapDataChanged = new HashMap<>();
                for (DataSnapshot dataItem : dataSnapshot.getChildren()) {
                mapDataChanged.put(dataItem.getKey(),
dataItem.getValue().toString());
                    }
                    adapterData.refreshItem(mapDataChanged, editPosition);
                    }

                    @Override
                    public void onChildRemoved(DataSnapshot dataSnapshot)
{
                        Log.d(TAG, "onChildRemoved");
                    }

                    @Override
           public void onChildMoved(DataSnapshot dataSnapshot, String s) {
                        Log.d(TAG, "onChildMoved");
                    }

                    @Override
                    public void onCancelled(DatabaseError databaseError) {
                        Log.d(TAG, "onCancelled");
                    }
                });
    }

    private void initSwipe() {
ItemTouchHelper.SimpleCallback simpleCallback = new
ItemTouchHelper.SimpleCallback(0, ItemTouchHelper.LEFT |
ItemTouchHelper.RIGHT) {
            @Override
            public boolean onMove(RecyclerView recyclerView,
RecyclerView.ViewHolder viewHolder, RecyclerView.ViewHolder target) {
            return false;
```

```
            }

 @Override
 public void onSwiped(RecyclerView.ViewHolder viewHolder, int direction) {
                final int position = viewHolder.getAdapterPosition();
                String key = listKey.get(position);


                if (direction == ItemTouchHelper.LEFT) {
                    //  delete data
                    firebaseDatabase.getReference()
                            .child("member")

.child(firebaseAuth.getCurrentUser().getEmail().replaceAll("\\.", "_"))
                            .child("data")
                            .child(key)
                            .removeValue()
                            .addOnCompleteListener(new
OnCompleteListener<Void>() {
                        @Override
                        public void onComplete(@NonNull Task<Void> task) {
                            if (task.isSuccessful()) {
                            adapterData.removeItem(position);
                            showSnackbarMsg("Data has been deleted!");
                            } else {
                            showSnackbarMsg("Data fail to deleted!");
                                    }
                                }
                            });
                } else {
                    //  update data
                    editPosition = position;
                    String username =
firebaseAuth.getCurrentUser().getEmail().replaceAll("\\.", "_");
                    Data data = listData.get(position);
                    Map<String, String> mapData = new HashMap<>();
                    mapData.put("title", data.getTitle());
                    mapData.put("content", data.getContent());
                    mapData.put("key", key);
                    mapData.put("username", username);
                    EditDataDialogFragment editDataDialogFragment = new
EditDataDialogFragment();
                    EventBus.getDefault().postSticky(mapData);
                    editDataDialogFragment.show(getFragmentManager(),
null);
                }
            }

            @Override
            public void onChildDraw(Canvas c, RecyclerView recyclerView,
RecyclerView.ViewHolder viewHolder, float dX, float dY, int actionState,
boolean isCurrentlyActive) {
                Bitmap icon;
                if (actionState == ItemTouchHelper.ACTION_STATE_SWIPE) {
                    View itemView = viewHolder.itemView;
                    float height = (float) itemView.getBottom() - (float)
itemView.getTop();
                    float width = height / 3;

                    if (dX > 0) {
```

```
                        //  effect for edit

paint.setColor(getResources().getColor(R.color.colorEdit));
                        RectF background = new RectF((float)
itemView.getLeft(), (float) itemView.getTop(), dX, (float)
itemView.getBottom());

                        c.drawRect(background, paint);
                        icon =
BitmapFactory.decodeResource(getResources(), R.drawable.ic_edit_white);
                        RectF iconDest = new RectF((float)
itemView.getLeft() + width, (float) itemView.getTop() + width, (float)
itemView.getLeft() + 2 * width, (float) itemView.getBottom() - width);
                        c.drawBitmap(icon, null, iconDest, paint);
                    } else {
                        //  effect for delete

paint.setColor(getResources().getColor(R.color.colorDelete));
                        RectF background = new RectF((float)
itemView.getRight() + dX, (float) itemView.getTop(), (float)
itemView.getRight(), (float) itemView.getBottom());
                        c.drawRect(background, paint);
                        icon =
BitmapFactory.decodeResource(getResources(),
R.drawable.ic_delete_forever_white);
                        RectF iconDest = new RectF((float)
itemView.getRight() - 2 * width, (float) itemView.getTop() + width,
(float) itemView.getRight() - width, (float) itemView.getBottom() -
width);
                        c.drawBitmap(icon, null, iconDest, paint);
                    }
                }
                super.onChildDraw(c, recyclerView, viewHolder, dX, dY,
actionState, isCurrentlyActive);
            }
        };
        ItemTouchHelper itemTouchHelper = new
ItemTouchHelper(simpleCallback);

    }

    private void showSnackbarMsg(String message) {
        Snackbar.make(getActivity().findViewById(android.R.id.content),
message, Snackbar.LENGTH_LONG)
                .show();
    }

    private void removeView() {
        if (view.getParent() != null) {
            ((ViewGroup) view.getParent()).removeView(view);
        }
    }
}
```

Fragment Home.XML

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
    tools:context="com.example.google.myauto2000.HomeFragment"
android:orientation="vertical"
android:padding="10dp"
android:background="#fcfcfc"
android:gravity="center"
android:id="@+id/ll">
    <LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context="com.example.google.myauto2000.Main3Activity"
        android:orientation="vertical"
        android:padding="10dp"
        android:background="#fcfcfc"
        android:gravity="center">


        <LinearLayout
            android:clipToPadding="false"
            android:gravity="center"
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <android.support.v7.widget.CardView
                android:id="@+id/booking_card"

android:foreground="?android:attr/selectableItemBackground"
                android:layout_width="160dp"
                android:layout_height="190dp"
                android:layout_margin="10dp">
                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:orientation="vertical"
                    android:gravity="center">

                    <ImageView
                        android:layout_width="74dp"
                        android:layout_height="74dp"
                        android:background="@drawable/cercleshape"
                        android:padding="10dp"
                        android:src="@mipmap/boking" />
                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:textStyle="bold"
                        android:layout_marginTop="10dp"
                        android:text="Boking Service"/>
                    <View
```

```
                        android:layout_width="match_parent"
                        android:layout_height="1dp"
                        android:background="@color/lightgray"
                        android:layout_margin="10dp"/>
                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:gravity="center"
                        android:text="Check your Booking Service"
                        android:padding="5dp"
                        android:textColor="@android:color/darker_gray"/>

                </LinearLayout>
            </android.support.v7.widget.CardView>

            <android.support.v7.widget.CardView
                android:id="@+id/catalog_card"

android:foreground="?android:attr/selectableItemBackground"
                android:layout_width="160dp"
                android:layout_height="190dp"
                android:layout_margin="10dp">
                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:orientation="vertical"
                    android:gravity="center">
                    <ImageView
                        android:layout_width="74dp"
                        android:layout_height="74dp"
                        android:background="@drawable/cercleshape"
                        android:src="@mipmap/catalok"
                        android:padding="10dp"/>
                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:textStyle="bold"
                        android:layout_marginTop="10dp"
                        android:text="Catalog"/>
                    <View
                        android:layout_width="match_parent"
                        android:layout_height="1dp"
                        android:background="@color/lightgray"
                        android:layout_margin="10dp"/>
                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:gravity="center"
                        android:text="Check your Catalog"
                        android:padding="5dp"
                        android:textColor="@android:color/darker_gray"/>

                </LinearLayout>
            </android.support.v7.widget.CardView>

        </LinearLayout>
        <LinearLayout
            android:clipToPadding="false"
            android:gravity="center"
            android:orientation="horizontal"
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <android.support.v7.widget.CardView
                android:id="@+id/tradein_card"

android:foreground="?android:attr/selectableItemBackground"
                android:layout_width="160dp"
                android:layout_height="190dp"
                android:layout_margin="10dp">
                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:orientation="vertical"
                    android:gravity="center">
                    <ImageView
                        android:layout_width="74dp"
                        android:layout_height="74dp"
                        android:background="@drawable/circleshape"
                        android:src="@mipmap/tradein"
                        android:padding="10dp"/>
                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:textStyle="bold"
                        android:layout_marginTop="10dp"
                        android:text="tradein"/>
                    <View
                        android:layout_width="match_parent"
                        android:layout_height="1dp"
                        android:background="@color/lightgray"
                        android:layout_margin="10dp"/>
                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:gravity="center"
                        android:text="Check your tradein"
                        android:padding="5dp"
                        android:textColor="@android:color/darker_gray"/>

                </LinearLayout>
            </android.support.v7.widget.CardView>

            <android.support.v7.widget.CardView
                android:id="@+id/promo_card"

android:foreground="?android:attr/selectableItemBackground"
                android:layout_width="160dp"
                android:layout_height="190dp"
                android:layout_margin="10dp">

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:gravity="center"
                    android:orientation="vertical">

                    <ImageView
                        android:layout_width="74dp"
                        android:layout_height="74dp"
```

```
                        android:background="@drawable/cercleshape"
                        android:padding="10dp"
                        android:src="@mipmap/promo" />

                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:layout_marginTop="10dp"
                        android:text="Promo and Discount"
                        android:textStyle="bold" />

                    <View
                        android:layout_width="match_parent"
                        android:layout_height="1dp"
                        android:layout_margin="10dp"
                        android:background="@color/lightgray" />

                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:gravity="center"
                        android:padding="5dp"
                        android:text="Check your Promo and discoun"
                        android:textColor="@android:color/darker_gray" />

                </LinearLayout>
            </android.support.v7.widget.CardView>

        </LinearLayout>
        <LinearLayout
            android:clipToPadding="false"
            android:gravity="center"
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">
            <android.support.v7.widget.CardView

android:foreground="?android:attr/selectableItemBackground"
                android:layout_width="340dp"
                android:layout_height="150dp"
                android:layout_margin="10dp">
                <LinearLayout

                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:orientation="vertical"
                    android:gravity="center">

                    <ImageView
                        android:layout_width="164dp"
                        android:layout_height="wrap_content"
                        android:src="@mipmap/logo" />
                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:textStyle="bold"
                        android:text="Copyright@2018"/>
                    <View
                        android:layout_width="match_parent"
                        android:layout_height="1dp"
```

```
                        android:background="@color/lightgray"
                        android:layout_margin="10dp"/>
                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:gravity="center"
                        android:padding="5dp"
                        android:textColor="@android:color/darker_gray"/>


                </LinearLayout>
            </android.support.v7.widget.CardView>


        </LinearLayout>
    </LinearLayout></LinearLayout>
```

Dashboard Activity.Java

```
package com.example.google.myauto2000;

import android.content.Intent;
import android.os.Bundle;
import android.support.design.widget.NavigationView;
import android.support.v4.view.GravityCompat;
import android.support.v4.widget.DrawerLayout;
import android.support.v7.app.ActionBarDrawerToggle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.CardView;
import android.support.v7.widget.Toolbar;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;

import com.example.google.myauto2000.ChangePasswordDialogFragment;
import com.example.google.myauto2000.ChangeUsernameDialogFragment;
import com.example.google.myauto2000.HomeFragment;
import com.example.google.myauto2000.R;
import com.example.google.myauto2000.Main2Activity;
import com.google.firebase.auth.FirebaseAuth;

import static android.icu.lang.UCharacter.GraphemeClusterBreak.V;

public class DashboardActivity extends AppCompatActivity implements
NavigationView.OnNavigationItemSelectedListener {

    private CardView bookingcard,catalogcard, tradeincard, promocard;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_dashboard);
        Toolbar toolbar = findViewById(R.id.toolbar_dashboard);
        setSupportActionBar(toolbar);
```

```
        DrawerLayout drawer = findViewById(R.id.drawer_layout);
        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
                this, drawer, toolbar, R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
        drawer.setDrawerListener(toggle);
        toggle.syncState();

        NavigationView navigationView = findViewById(R.id.nav_view);
        navigationView.setNavigationItemSelectedListener(this);

        bookingcard = (CardView) findViewById(R.id.booking_card);
        catalogcard = (CardView) findViewById(R.id.catalog_card);
        tradeincard = (CardView) findViewById(R.id.tradein_card);
        promocard = (CardView) findViewById(R.id.promo_card);

        bookingcard.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(DashboardActivity.this,
booking_services.class);
                startActivity(i);
            }
        });

        catalogcard.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(DashboardActivity.this,
catalog.class);
                startActivity(i);
            }
        });

        tradeincard.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(DashboardActivity.this,
tradein.class);
                startActivity(i);
            }
        });

        promocard.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(DashboardActivity.this,
promo.class);
                startActivity(i);
            }
        });
    }

    @Override
    public void onBackPressed() {
        DrawerLayout drawer = findViewById(R.id.drawer_layout);
        if (drawer.isDrawerOpen(GravityCompat.START)) {
            drawer.closeDrawer(GravityCompat.START);
        } else {
            super.onBackPressed();
        }
```

```
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        /*getMenuInflater().inflate(R.menu.dashboard, menu);*/
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_sort_by_title_note:
                //  do something
                return true;
            case R.id.action_sort_by_date_time_note:
                //  do something
                return true;
        }

        return super.onOptionsItemSelected(item);
    }

    @SuppressWarnings("StatementWithEmptyBody")
    @Override
    public boolean onNavigationItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.nav_change_password:
                ChangePasswordDialogFragment changePasswordDialogFragment
= new ChangePasswordDialogFragment();

changePasswordDialogFragment.show(getSupportFragmentManager(), null);
                break;
            case R.id.nav_change_email:
                ChangeUsernameDialogFragment changeUsernameDialogFragment
= new ChangeUsernameDialogFragment();

changeUsernameDialogFragment.show(getSupportFragmentManager(), null);
                break;
            case R.id.nav_logout:
                FirebaseAuth firebaseAuth = FirebaseAuth.getInstance();
                firebaseAuth.signOut();
                Intent intent = new Intent(this, Main2Activity.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK |
Intent.FLAG_ACTIVITY_NEW_TASK);
                startActivity(intent);
                break;
            case R.id.nav_about:
                    startActivity(new Intent(this,
dialog_fragment_about.class));
                break;
        }

        DrawerLayout drawer = findViewById(R.id.drawer_layout);
        drawer.closeDrawer(GravityCompat.START);
        return true;
    }
}
```

Dashboard Activity.XML

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:openDrawer="start">

    <include
        layout="@layout/app_bar_dashboard"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <android.support.design.widget.NavigationView
        android:id="@+id/nav_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/nav_header_dashboard"
        app:menu="@menu/activity_dashboard_drawer" />

</android.support.v4.widget.DrawerLayout>
```

Dashboard Activity.XML

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/nav_change_password"
        android:icon="@drawable/ic_reset_password"
        android:title="Change Password" />
    <item
        android:id="@+id/nav_change_email"
        android:icon="@drawable/ic_email_black"
        android:title="Change Email" />
    <item
        android:id="@+id/nav_logout"
        android:icon="@drawable/ic_logout"
        android:title="Logout" />
    <item
        android:id="@+id/nav_about"
        android:icon="@drawable/ic_about_24dp"
        android:title="About" />

</menu>
```

My Firebase Instance ID Service.Java

```
package com.example.google.myauto2000;

/**
 * Created by Pikri on 25/07/2018.
 */
import android.content.Context;
import android.content.Intent;
import android.support.v4.content.LocalBroadcastManager;
import android.text.TextUtils;
import android.util.Log;

import com.google.firebase.messaging.FirebaseMessagingService;
import com.google.firebase.messaging.RemoteMessage;

import org.json.JSONException;
import org.json.JSONObject;

import com.example.google.myauto2000.MainActivity;
import com.example.google.myauto2000.Config;
import com.example.google.myauto2000.NotificationUtils;


public class MyFirebaseInstanceIDService extends FirebaseMessagingService
{

    private static final String TAG =
MyFirebaseMessagingService.class.getSimpleName();

    private NotificationUtils notificationUtils;

    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {
        Log.e(TAG, "From: " + remoteMessage.getFrom());

        if (remoteMessage == null)
            return;

        // Check if message contains a notification payload.
        if (remoteMessage.getNotification() != null) {
            Log.e(TAG, "Notification Body: " +
remoteMessage.getNotification().getBody());
            handleNotification(remoteMessage.getNotification().getBody());
        }

        // Check if message contains a data payload.
        if (remoteMessage.getData().size() > 0) {
            Log.e(TAG, "Data Payload: " +
remoteMessage.getData().toString());

            try {
                JSONObject json = new
JSONObject(remoteMessage.getData().toString());
                handleDataMessage(json);
            } catch (Exception e) {
                Log.e(TAG, "Exception: " + e.getMessage());
            }
        }
    }
```

```
    private void handleNotification(String message) {
        if
(!NotificationUtils.isAppIsInBackground(getApplicationContext())) {
            // app is in foreground, broadcast the push message
            Intent pushNotification = new
Intent(Config.PUSH_NOTIFICATION);
            pushNotification.putExtra("message", message);

LocalBroadcastManager.getInstance(this).sendBroadcast(pushNotification);

            // play notification sound
            NotificationUtils notificationUtils = new
NotificationUtils(getApplicationContext());
            notificationUtils.playNotificationSound();
        }else{
            // If the app is in background, firebase itself handles the
notification
        }
    }

    private void handleDataMessage(JSONObject json) {
        Log.e(TAG, "push json: " + json.toString());

        try {
            JSONObject data = json.getJSONObject("data");

            String title = data.getString("title");
            String message = data.getString("message");
            boolean isBackground = data.getBoolean("is_background");
            String imageUrl = data.getString("image");
            String timestamp = data.getString("timestamp");
            JSONObject payload = data.getJSONObject("payload");

            Log.e(TAG, "title: " + title);
            Log.e(TAG, "message: " + message);
            Log.e(TAG, "isBackground: " + isBackground);
            Log.e(TAG, "payload: " + payload.toString());
            Log.e(TAG, "imageUrl: " + imageUrl);
            Log.e(TAG, "timestamp: " + timestamp);


            if
(!NotificationUtils.isAppIsInBackground(getApplicationContext())) {
                // app is in foreground, broadcast the push message
                Intent pushNotification = new
Intent(Config.PUSH_NOTIFICATION);
                pushNotification.putExtra("message", message);

LocalBroadcastManager.getInstance(this).sendBroadcast(pushNotification);

                // play notification sound
                NotificationUtils notificationUtils = new
NotificationUtils(getApplicationContext());
                notificationUtils.playNotificationSound();
            } else {
                // app is in background, show the notification in
notification tray
                Intent resultIntent = new Intent(getApplicationContext(),
MainActivity.class);
```

```
                resultIntent.putExtra("message", message);

                // check for image attachment
                if (TextUtils.isEmpty(imageUrl)) {
                    showNotificationMessage(getApplicationContext(),
title, message, timestamp, resultIntent);
                } else {
                    // image is present, show notification with image

showNotificationMessageWithBigImage(getApplicationContext(), title,
message, timestamp, resultIntent, imageUrl);
                }
            }
        } catch (JSONException e) {
            Log.e(TAG, "Json Exception: " + e.getMessage());
        } catch (Exception e) {
            Log.e(TAG, "Exception: " + e.getMessage());
        }
    }

    /**
     * Showing notification with text only
     */
    private void showNotificationMessage(Context context, String title,
String message, String timeStamp, Intent intent) {
        notificationUtils = new NotificationUtils(context);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
        notificationUtils.showNotificationMessage(title, message,
timeStamp, intent);
    }

    /**
     * Showing notification with text and image
     */
    private void showNotificationMessageWithBigImage(Context context,
String title, String message, String timeStamp, Intent intent, String
imageUrl) {
        notificationUtils = new NotificationUtils(context);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
        notificationUtils.showNotificationMessage(title, message,
timeStamp, intent, imageUrl);
    }
}
```

My Firebase Messaging Service.Java

```
package com.example.google.myauto2000;

import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;
import android.support.v4.app.NotificationCompat;
```

```
import android.util.Log;
import com.google.firebase.messaging.FirebaseMessagingService;
import com.google.firebase.messaging.RemoteMessage;


public class MyFirebaseMessagingService extends FirebaseMessagingService {

    private static final String TAG = "MyFirebaseMsgService";

    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {

        // TODO(developer): Handle FCM messages here.
        Log.d(TAG, "From: " + remoteMessage.getFrom());

        // Check if message contains a data payload.
        if (remoteMessage.getData().size() > 0) {
            Log.d(TAG, "Message data payload: " +
remoteMessage.getData());

        }

        // Check if message contains a notification payload.
        if (remoteMessage.getNotification() != null) {
            Log.d(TAG, "Message Notification Body: " +
remoteMessage.getNotification().getBody());
            sendNotification(remoteMessage.getNotification().getBody());
        }

    }

    private void sendNotification(String messageBody) {
        Intent intent = new Intent(this, MainActivity.class);
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0,
intent,
                PendingIntent.FLAG_ONE_SHOT);

        Uri defaultSoundUri=
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
        NotificationCompat.Builder notificationBuilder = new
NotificationCompat.Builder(this)
                .setSmallIcon(R.mipmap.icon1)
                .setContentTitle("FCM Message")
                .setContentText(messageBody)
                .setAutoCancel(true)
                .setSound(defaultSoundUri)
                .setContentIntent(pendingIntent);

        NotificationManager notificationManager =
                (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

        notificationManager.notify(0, notificationBuilder.build());
    }

}
```