PENGEMBANGAN APLIKASI *E-COMMERCE* MENGGUNAKAN *PAYMENT GATEWAY MIDTRANS*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Fariz Reynaldo Pratama
NIM: 135150201111052

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2020

PENGESAHAN

PENGEMBANGAN APLIKASI *E-COMMERCE*
MENGGUNAKAN *PAYMENT GATEWAY MIDTRANS*
SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Fariz Reynaldo Pratama
NIM: 135150201111052

Skripsi ini telah diuji dan dinyatakan lulus pada
29 April 2020
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing 2
Nurudin Santoso, S.T., M.T.	Lutfi Fanani, S.Kom., M.T., M.Sc.
NIP. 197409162000121001	NIK. 201607 890217 1 001

Mengetahui
Ketua Jurusan **Teknik Informatika**

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

PERNYATAAN ORISINALITAS

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

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

Malang, 12 Maret 2020

Fariz Reynaldo Pratama

NIM: 135150201111052

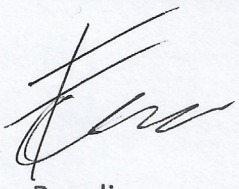
PRAKATA

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul "Pengembangan Aplikasi *E-commerce* Menggunakan *Payment Gateway Midtrans*". ini dapat terselesaikan.

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

1. Allah SWT yang telah memberikan kemudahan dan kelancaran dalam semua proses pengerjaan skripsi.
2. Dede Gunawan dan Almh. Susan Gardenia selaku orang tua serta keluarga saya yang selalu memberikan dukungan finansial, moral dan doa yang selalu diberikan kepada penulis untuk hingga skripsi ini selesai.
3. Bapak Nurudin Santoso, S.T., M.T. dan Lutfi Fanani, S.Kom., M.T., M.Sc. selaku dosen pembimbing yang senantiasa meluangkan waktu dan tenaganya untuk membimbing penulis dalam menyelesaikan penelitian ini.
4. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D dan Bapak Agus Wahyu Widodo, S.T, M.Cs selaku Ketua Jurusan Teknik Informatika dan Kepala Program Studi Teknik Informatika Fakultas Ilmu Komputer, Universitas Brawijaya.
5. Bapak Prasetyo Iskandar, S.T, Ibu Wiwin Lukitohadi, S.H, S.Psi, CHRM serta Nurudin Santoso, S.T., M.T. yang peduli serta membantu saya dan teman-teman pada semester kritis.
6. Hesti Harianti yang rutin membantu untuk membenahi format pada penulisan skripsi ini.

Malang, 12 Maret 2020



Penulis

Fariz.aldo@gmail.com

ABSTRAK

Fariz Reynaldo Pratama, Pengembangan Aplikasi *E-commerce* Menggunakan *Payment Gateway Midtrans* .

Pembimbing: Nurudin Santoso, S.T., M.T. dan Lutfi Fanani, S.Kom., M.T., M.Sc.

Anfield collection merupakan pengrajin sekaligus toko (UKM) yang menjual kerajinan. Saat menjalankan bisnisnya toko anfield akan mencatat stok dan memajang produk yang di jual pada tokonya sendiri dan pada toko mitra. Ada 25 macam produk yang dijual pada toko anfield hal ini dirasa sulit untuk mengingat variasi produk dan harga yang berbeda. Kemudian, saat terjadi transaksi petugas anfield akan memberikan kwitansi kepada pembeli lalu mencatat pada buku daftar transaksi. Pemasaran yang dilakukan belum terlalu luas hal ini dirasakan oleh pemilik toko alasanya karena belum cukup baik dalam memanfaatkan teknologi informasi. Banyaknya bisnis yang serupa juga membuat anfield harus bersaing dengan toko lainnya. Pemilik bisnis anfield menyebutkan bahwa perlu aplikasi *e-commerce* khusus untuk produsen anfield untuk mengatasi masalah dan meningkatkan kualitas proses bisnis yang berjalan dari sisi efektifitas dan efisiensi. Oleh karena itu dibuat aplikasi *e-commerce* pada platform *mobile*. Pada aplikasi terdapat fitur dasar dalam jual beli *online* yaitu kelola produk, pemesanan, transaksi pembayaran dan ekspedisi. Fitur yang menjadi pembeda adalah kategori produk kerajinan berdasarkan jenis, proses dan bahan utama. Untuk menangani transaksi pembayaran aplikasi harus dapat menangani berbagai macam jenis pembayaran. Aplikasi ini sudah diuji menggunakan metode *white-box* dengan pengujian unit dan *Usability Testing* dengan standar *Post Study System Usability Questionnaire (PSSUQ)*. Pengujian menunjukan 100% kebutuhan sudah terpenuhi dan secara umum rata-rata pengguna puas selama menggunakan aplikasi.

Kata kunci: *e-commerce*, *midtrans*, *react native*, kerajinan, laporan penjualan, pembayaran online, e-commerce kerajinan

ABSTRACT

Fariz Reynaldo Pratama, *Development of E-commerce Using Payment Gateway Midtrans.*

Supervisors: Nurudin Santoso, S.T., M.T. dan Lutfi Fanani, S.Kom., M.T., M.Sc.

Anfield collection is a handicraft factory and a shop that sells handicrafts. When running its business anfield store will record stock and display products in its own shop and in partner stores. There are 25 types of products on sale at anfield stores, it is difficult to remember the different product variations and prices. Then, when a transaction is made anfield officer will provide a receipt to the buyer and then record it in the transaction register book. The marketing that has been done is not too broad, this is felt by the shop owner, the reason being that it is not good enough in utilizing information technology. The number of similar businesses also makes Anfield have to compete with other stores. Anfield business owners mention that it needs special e-commerce applications for anfield producers to overcome problems and improve the quality of business processes that run in terms of effectiveness and efficiency. Therefore, e-commerce applications are made on the mobile platform. In the application there are basic features in buying and selling online, namely managing products, ordering, payment transactions and expeditions. The distinguishing feature is the category of craft products based on the type, process and main ingredients. To handle payment transactions applications must be able to handle various types of payments. This application has been tested using the white-box method with unit testing and Usability Testing with the standard Post Study System Usability Questionnaire (PSSUQ). Tests result show that requirement has passed 100% and generally average user is satisfied while using the application.

Keywords: *e-commerce, midtrans, react native, craft, sell report, payment gateway, craft e-commerce*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Kerajinan atau kria merupakan salah satu karya seni rupa terapan yang biasanya dihasilkan melalui proses kerja terampil para perajinnya (Sumanto, 2011). Produk yang dihasilkan dibuat dengan bahan dan alat yang alami juga diproses dengan teknik sederhana dan modern. Kerajinan dapat berfungsi sebagai barang ekonomi. Beberapa contoh kerajinan yang memiliki nilai ekonomi yaitu lukisan, tenunan, anyaman, batik, keramik, patung. Banyak jenis kerajinan sudah berkembang menjadi proses budaya dalam masyarakat. Perkembangan kerajinan juga telah menjadi tumpuan bagi sebagian masyarakat di Indonesia karena industri ini dapat menyerap tenaga kerja. Produksi kerajinan telah menyerap banyak tenaga kerja pada negara-negara berkembang, salah satunya Indonesia. Usaha Kecil Menengah (UKM) di bidang kerajinan terus bertambah karena daya tarik yang ditawarkan yaitu jam kerja yang fleksibel, dapat dikerjakan dirumah, modal awal yang kecil dan juga kebebasan dalam mengelola usaha (Detik, 2019).

Di Kota Blitar terdapat banyak pengrajin tas kulit seperti Toko Anfield *Collection*, toko batok koi, toko ibu surati, toko gemilang, toko sumber lancar, dan masih banyak lagi. Anfield collection merupakan pengrajin sekaligus Usaha Kecil dan Menengah (UKM) yang menjual kerajinan yang terletak pada desa ringin anom. Saat ini anfield memiliki 25 produk dengan 17 macam tas dan 8 souvenir yang dapat dikategorikan berdasarkan jenis barang, proses produksi dan bahan dasar. Hasil produksi yang dilakukan oleh anfield dijual pada toko milik sendiri dan pada toko lainnya yang bermitra seperti toko batok koi, toko ibu surati dan toko gemilang. Proses bisnis anfield secara umum yaitu produksi lalu mempromosikan kemudian transaksi pembelian dilakukan dan mengirim barang jika ada permintaan untuk pengiriman produk.

Saat menjalankan bisnisnya toko anfield akan mencatat stok dan memajang produk yang di jual pada tokonya sendiri dan pada toko mitra. Tidak ada daftar harga juga membuat calon pembeli harus bertanya kepada petugas untuk mengetahui harga dari barang yang menarik perhatian. Ketika melihat produk calon pembeli sering memperhatikan jenis produk, bahan baku dan proses pembuatan, tidak adanya kategorisasi membuat calon pembeli sering bertanya mengenai produk. Ada 25 macam produk yang dijual pada toko anfield hal ini dirasa sulit untuk mengingat variasi produk dan harga yang berbeda. Kemudian, saat terjadi transaksi petugas anfield akan memberikan kwitansi kepada pembeli lalu mencatat pada buku daftar transaksi. Seluruh temuan tersebut didapatkan pada wawancara yang dilakukan terhadap pemilik toko dan observasi yang dilakukan.

Dari wawancara dan observasi didapatkan informasi bahwa, dalam hal promosi saat ini anfield melakukanya dengan cara bermitra dengan toko lain, menitipkan gambar produk pada orang lain, mengirim pesan melalui *WhatsApp* kepada calon pembeli dan metode promosi dari mulut ke mulut. Pemasaran yang

dilakukan belum terlalu luas hal ini dirasakan oleh pemilik toko alasanya karena belum cukup baik dalam memanfaatkan teknologi informasi. Banyaknya bisnis yang serupa juga membuat anfield harus bersaing dengan toko lainnya, maka dari itu dirasa perlu strategi yang berbeda dari yang proses sudah dilakukan sebelumnya.

Secara garis besar didapatkan bahwa toko anfield *collection* memiliki 2 masalah yaitu pencatatan dan pemasaran. Pemilik bisnis anfield menyebutkan bahwa perlu aplikasi *e-commerce* khusus untuk produsen anfield untuk mengatasi masalah yang ada dan meningkatkan kualitas proses bisnis yang berjalan pada bisnis anfield dari sisi efektifitas dan efisiensi. Pada penelitian yang dilakukan oleh (Maulana, Susilo, & Riyadi, 2015) mendapatkan bahwa *e-commerce* dapat meningkatkan dan juga memenangkan persaingan usaha serta penjualan produk. Saat e-commerce sudah di implementasikan akan membuat proses pemasaran dan kegiatan jual beli menjadi lebih efisien terlihat pada beberapa sisi yaitu akselerasi proses transaksi, kemudahan dalam bertransaksi dan juga pengurangan biaya. Pada sisi kualitas data juga akan lebih baik dibandingkan dengan pencatatan manual dengan alat tulis dan kertas karena akan mengurangi resiko kesalahan dalam pencatatan, kualitas data akan juga akan lebih dinamis untuk diolah. Hal ini didukung teori adaptasi organisasi yang menegaskan bahwa bisnis akan terus berubah untuk mengikuti kondisi pasar (study, 2020). Transaksi *online* saat ini sangat besar dan terus meningkat dan data transaksi menunjukan bahwa 75% transaksi *online* di Indonesia terjadi pada perangkat *mobile* (Bisnis.Com, 2018).

Ditemukan setidaknya 2 *e-commerce* khusus untuk kerajinan yang pertama merupakan penelitian dengan topik pembuatan e-commerce untuk toko batik di pekalongan dengan platform web. Pada penelitian ini aplikasi yang dibuat memiliki beberapa kelemahan yang pertama adalah fitur yang dibuat tidak ada pembeda dari yang lain. Kedua tidak ada fitur pendaftaran alamat membuat proses pemesanan tidak efisien dengan asumsi ada pelanggan yang memesan lebih dari satu kali. Ketiga, tidak ada pemilihan layanan ekspedisi membuat ongkos pengiriman barang tidak terukur. Keempat, tidak dapat memesan lebih dari satu toko dengan asumsi ada pengguna yang memesan lebih dari satu toko secara bersamaan, hal tersebut merupakan kelemahan karena e-commerce pada umumnya dapat melakukan pemesanan lebih dari satu toko dengan satu kali pembayaran. Kelima metode pembayaran dilakukan dengan cara transfer manual yang tidak terintegrasi dengan aplikasi hal ini membuat efisiensi waktu yang buruk dan resiko kesalahan pada saat pengecekan karena tidak ada identifikasi pembayaran yang unik.

Aplikasi yang kedua merupakan aplikasi yang paling populer dalam skala global yaitu *Etsy*. Pada aplikasi ini secara umum sudah sangat baik dengan fitur yang lengkap namun memiliki setidaknya 2 kelemahan, yang pertama hasil observasi yang dilakukan banyak ditemukan produk Indonesia namun ternyata mayoritas produk tersebut dikirim dari luar negri utamanya Amerika Serikat & Kanada. Tidak diketahui apakah produk tersebut diproduksi di Indonesia atau tidak, jika produk tersebut merupakan produksi dari Indonesia maka potensi

keuntungan akan bertambah jika produk tersebut dijual oleh produsen langsung. Kelemahan yang kedua yaitu, aplikasi ini tidak mendukung mata uang rupiah sebagai alat pembayaran, Metode yang dapat digunakan hanya Paypal dan kartu kredit *visa* & *mastercard* dimana metode tersebut memiliki biaya tukar mata uang dan ditambah lagi penambahan dengan biaya transaksi. Belum lagi, di Indonesia pengguna kartu kredit masih sedikit yaitu 5% (Medcom, 2019).

Berdasarkan beberapa permasalahan diatas maka solusi yang ditawarkan ialah mengembangkan aplikasi *e-commerce* pada Toko Anfield, yang menghubungkan antara pengelola toko dengan calon pembeli. Aplikasi akan dikembangkan dengan mengakomodasi karakteristik kerajian yang dapat di kategorikan berdasarkan jenis, proses pembuatan dan bahan utama (Sumanto, 2018). Implementasi metode pembayaran akan sangat dinamis karena dengan *Payment Gateway Midtrans* mampu mengakomodasi seluruh jenis pembayaran yang umum digunakan di Indonesia. Fitur lainnya adalah penyimpanan alamat dan pemilihan layanan ekspedisi agar ketika pemesanan dilakukan biaya ekspedisi dapat dihitung. Serta fitur lainya yang umumnya ada pada e-commerce pengelolaan produk, pemesanan. Aplikasi *e-commerce* ini dikembangkan dengan menggunakan metode *Waterfall Model* karena kebutuhan dari sistem sudah jelas diawal dan aplikasi ini akan dikembangkan untuk *platform android* akan tetapi memungkinkan untuk pengembangan lebih lanjut untuk *platform ios* dan *web* jika diperlukan, karena framework *react native* bersifat *hybrid*. Harapan dari penelitian ini ialah dengan adanya aplikasi *e-commerce* untuk toko Anfield *Collection* akan menyelesaikan permasalahan dari Anfield *Collection* meningkatkan efisiensi serta efektifitas pada proses bisnis anfield dan masalah pemasaran.

1.2 Rumusan Masalah

Berikut merupakan rumusan masalah yang didapatkan dari permasalahan yang didapatkan pada latar belakang:

1. Apa saja kebutuhan Pengembangan "Aplikasi *E-commerce* untuk *Payment Gateway Midtrans*"?
2. Bagaimana hasil perancangan "Aplikasi *E-commerce* Menggunakan *Payment Gateway Midtrans*"?
3. Bagaimana implementasi "Aplikasi *E-commerce* untuk Menggunakan *Payment Gateway Midtrans*"?
4. Bagaimana hasil pengujian dari "Aplikasi *E-commerce* untuk Menggunakan *Payment Gateway Midtrans*"?

1.3 Tujuan

Ada 4 Tujuan yang ingin dicapai dari dilakukannya penelitian ini yakni:

1. Memberikan informasi mengenai kebutuhan "Aplikasi *E-commerce* Menggunakan *Payment Gateway Midtrans*."
2. Mengetahui hasil perancangan dari "Aplikasi *E-commerce* Menggunakan *Payment Gateway Midtrans*".

3. Mengetahui hasil implementasi berdasarkan kebutuhan dan perancangan "Aplikasi *E-commerce* Menggunakan *Payment Gateway Midtrans*".
4. Memberikan informasi mengenai sistem yang dibuat sudah memenuhi proses kebutuhan.

1.4 Manfaat

Manfaat yang didapatkan adalah dapat membantu Toko Anfield *Collection* dalam memasarkan produk dan mempermudah pengelolaan stok barang. Berikut adalah manfaat dari penelitian ini:

1. Manfaat Praktis

   UKM *Anfield collection* akan mengetahui apakah dengan adanya aplikasi e-commerce akan menaikan efektifitas dan efisiensi dari aplikasi yang telah dibuat. Kemudian mengetahui apakah penjualan akan membaik atau tidak.

2. Manfaat Akademis

   Hasil dari penelitan yang dibuat dapat menjadi acuan untuk penelitian yang akan dilakukan. Hasil dari penelitian ini juga dapat dikembangkan untuk penelitian selanjutnya agar terus dapat berkembang.

1.5 Batasan Masalah

Ada batasan dalam melakukan penelitian ini agar masalah yang teliti lebih terfokus, berikut merupakan Batasan-batasannya :

1. Aplikasi dikembangkan dengan menggunakan *Framework React Native* sehingga dapat digunakan di Android.
2. Pembayaran dilakukan dengan dengan simulator pembayaran yang ada pada *midtrans*.
3. Tidak dilakukan proses pemantaun dan evaluasi terhadap penjualan pasca pembuatan aplikasi.

1.6 Sistematika Pembahasan

Berikut merupakan penyusunan hasil perancangan penelitian yang didokumentasikan dalam bentuk skripsi ini berdasarkan sistematika penulisan:

BAB 1 PENDAHULUAN

Bab ini merupakan permulaan dari penilitian yang menjabarkan latar belakang, rumusan masalah, Batasan masalah, tujuan penelitian, manfaat penelitian secara praktis dan akademis, sistematika pembahasan.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini membahas dan menguraikan referensi penelitian sebelumnya, dasar teori dan teknologi pendukung yang digunakan serta penelitian-penelitian yang menjadi dasar penelitian ini.

BAB 3 METODOLOGI PENELITIAN

Berisi penjelasan bagaimana tahapan-tahapan dalam melakukan peneleitian ini. Dari mulai studi literatur lalu tahapan berikutnya masuk ke metode pengembangan perangkat lunak yang digunakan dalm kemudian dan terakhir kesimpulan & saran dari hasi penelitian.

BAB 4 REKAYASA KEBUTUHAN

Dari latar belakang yang diuraikan pada bab 1 kemudian dilakukan elisitasi kebutuhan dari pengguna dan juga memahami proses dari aplikasi yang sudah ada. Setelah sumber didapatkan lalu dilakukan rekayasa kebutuhan. Melalui *Use Case Diagram* dan perangkat lainnya untuk memperjelas hasil dari rekayasa kebutuhan.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Bab 5 berisi pemodelan serta menjelaskan mengenai perancangan arsitektur sistem, *sequence diagram*, *class diagram*, kode program dan batasan sistem. Pada fase implementasi akan disertai hasil dari perncangan yang dilakukan.

BAB 6 PENGUJIAN

Pda bab ini dilakukan pengujian setelah program di implementasikan. Hasil dari pengujian akan dibahas pada bab ini.

BAB 7 PENUTUP

Bab 7 Menjelaskan hasil dari penelitian dengan mengambil kesimpulan lalu saran yang untuk pengembangan penelitian.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Terdapat 3 penelitian yang peneliti gunakan sebagai bahan dasar untuk menganalisis kajian penelitian. Untuk membandingkan beberapa penelitian yang pernah dilakukan sebelumnya dapat di lihat pada Tabel 2.1

Tabel 2.1 Kajian Pustaka

No.	Penelitian	Fokus Penelitian
1	Rancang Bangun Sistem Informasi Penjualan Online (*E-Commerce*) Pada Toko Batik Pekalongan Dengan Metode *Waterfall* (Nuraeni & Astuti, 2019)	Penelitian ini dilatarbelakangi oleh banyak pengrajin dan penjual yang masih melakukan proses pemasaran dengan cara-cara klasik dan sederhana. Kemudian masih banyaknya kendala dalam pendistribusian dan pemasaran batik pekalongan apabila diproduksi dalam jumlah yang besar, dikarenakan R3N batik masih menggunakan cara tradisional yaitu dengan melakukan pendistribusian produk secara langsung kepada konsumen. Fitur yang dibuat pada aplikasi tidak lengkpa dan tidak memiliki pembeda dari *e-commerce* pada umumnya.
2	Pemanfaatan *E-Commerce* untuk Peningkatan Strategi Promosi dan Penjualan UMKM Tas di Kabupaten Kudus (Azizah, Mahendra, & Lofian, 2019)	Penelitian ini latar belakangi oleh lemahnya promosi yang dilakukan untuk memasarkan produk karena barang yang dipasarkan setelah dari hasil produksi. Belum maksimalnya pemanfaatan teknologi informasi dalam menunjang kegiatan UMKM Tas Camelano ini, karena masih menggunakan metode konvensional dalam kegiatan promosi dan penjualannya seperti langsung ke pasar-pasar untuk mendistribusikan produk tersebut.
3	Penerapan *E-Commerce* untuk meningkatkan penjualan pada toko	Peneitian ini dilatarbelakangi oleh penjualan toko yang sedikit

	tas Batam No.1 (Mayangsari & Ariesta, 2019)	menurun dikarenakan banyaknya pesaingan penjual tas. Toko ini sedang menghadapi beberapa masalah seperti banyak pelanggan yang *complain*, kesalahan penotalan, promosi tidak menjangkau semua, pemilik sulit mengetahui hasil pendapatan, sulit mengetahui data pembelian produk ke *supplier*, sulit menentukan produk *best seller*, laporan rekap pesanan sulit dipahami dan sering keliru, sulit untuk menentukan pelanggan yang harus diberikan *reward* dan sulit mengetahui daftar pelanggan yang membeli produk dengan harga tertinggi.

2.2 Kerajinan

Kerajinan atau kria merupakan salah satu proses kerja kreatif dan terampil para perajin yang menghasilkan karya seni rupa terapan (Sumanto, 2011). Produk yang dihasilkan dibuat dengan bahan dan alat yang alami juga diproses dengan teknik sederhana dan modern.

Kerajinan dapat berfungsi sebagai barang ekonomi. Beberapa contoh kerajinan yang memiliki nilai ekonomi yaitu lukisan, tenunan, anyaman, batik, keramik, patung. Kerajinan dapat dikategorikan berdasarkan ragam jenis, Teknik pembuatan dan bahan utama (Sumanto, 2015) Banyak jenis kerajinan sudah berkembang menjadi proses budaya dalam masyarakat. Perkembangan kerajinan juga telah menjadi tumpuan bagi sebagian masyarakat di Indonesia karena industri ini dapat menyerap tenaga kerja. Produksi kerajinan telah menyerap banyak tenaga kerja pada negara-negara berkembang, salah satunya Indonesia.

Usaha Kecil Menengah (UKM) di bidang kerajinan terus bertambah karena daya tarik yang ditawarkan yaitu jam kerja yang fleksibel, dapat dikerjakan dirumah, modal awal yang kecil dan juga kebebasan dalam mengelola usaha (Detik, 2019).

2.3 Anfield *Collection*

Anfield *Collection* merupakan sebuah produsen yang memproduksi berbagai jenis tas kulit dan suvenir. Lokasinya terletak di Desa Ringin Anom kecamatan Kademangan, Jawa Timur. Anfield *Collection* memiliki lebih dari 50 karyawan yang setiap hari nya dapat memproduksi 100 tas lebih. Dalam kegiatan pemasarannya masih menggunakan cara tradisional yaitu memasarkannya langsung ke pasar-

pasar tradisional dan juga belum banyak orang yang mengetahui lokasi Toko Anfield *Collection*. Berikut adalah gambar yang diambil dari lokasi Anfield *Collection*.

Gambar 2.1 Lokasi Anfield *Collection*

2.4 E-Commerce

E-commerce adalah aktivitas jual atau beli produk secara elektronik pada layanan online atauoun jaringan internet. Terdapat 3 area pada ecommerce, yaitu retail online, pasar elektronik dan lelang online. E-commerce juga didukung oleh bisinis elektronik (Wienclaw, 2013). Dengan *e-commerce* sebuah usaha akan dapat memperluas pasar pada jangkauan yang lebih besar. Saat ini Perdagangan dengan *e-commerce* sangat diminati karena selain sangat mudah dan efektif. Sebagian besar masyarakat lebih dominan menggunakan *smartphone* untuk melakukan pembelian barang di sebuah toko onlineImplementasi e-commerce pada anfield collection akan memberikan banyak keuntungan seperti tidak perlunya konsumen untuk datang membeli barang secara langsung ke toko dan produsen pun dapat melaksanakan transaksi jual beli selama 24 jam. Kemudian konsumen dapat menghemat biaya yang dikeluarkan dan juga dapat menghemat biaya operasional produsen.

Memasarkan produk ke wilayah yang lebih luas dan biaya operasional yang lebih murah merupakan salah satu manfaat yang sangat besar pada penggunaan *e-commerce* ini. Dilansir dari situs *dailysocial.id* hasil survei yang di ungkapkan bahwa layanan *e-commerce* dianggap sangat di favoritkan karena harga yang diberikan lebih terjangkau (31%), adanya promo diskon (26%), banyaknya pilihan produk (19%), kemudian adanya program pengiriman gratis (15%) seperti pada Gambar 2.2.

Gambar 2.2 Gambar Survey E-commerce

(Sumber: dailysocial.id)

E-commerce memiliki karakteristik yaitu tidak terlepasnya dari keberadaan internet. Hal ini digunakan sebagai media dalam mengintegrasikan proses penjualan, pembelian, persediaan barang serta pemesanan. Apabila media internet tidak tersedia, maka proses tersebut hanya dapat dilakukan secara manual dan membutuhkan biasa operasional yang lebih besar.

2.4.1 Manfaat menggunakan *E-Commerce*

Dalam menerapkan dan memanfaatan *e-commerce* pada bisnis terdapat beberapa kelebihanya yaitu:

1. Hasil pendapatan yang lebih besar dibandingkan dengan bertransaksi tradisional
2. Pangsa pasar menjadi meningkat
3. Biaya operasional dapat diturunkan
4. Melebarkan jangkuan penjualan
5. Meningkatkan rantai pendapatan sehingga *supplier management* pun akan meningkat
6. Adanya *customer royalty*

2.4.2 Kelemahan *E-Commerce*

Disamping manfaat dari *e-commerce* yang telah dijelaskan diatas, adapun kelemahan yang dimiliki *e-commerce* yaitu mudah sekali disalahgunakan oleh pihak-pihak yang tidak bertanggung jawab. Kemudian penyalahgunaan dan kegagalan pada sistem dapat terjadi apabila dilihat dari segi pandang bisnis. Hal tersebut diantaranya sebagai berikut:

1. Gangguan non-teknis seperti padamnya aliran listrik dapat menghilangkan kesempatan bisnis.
2. Kecurangan yang disengaja oleh penjual dengan adanya praktek bisnis yang tidak benar dan adanya kesalahan faktor sistem ataupun manusia yang dapat memungkinkan menyebabkan kerugian

3. Pencurian identitas ataupun membohongi kemungkinan dapat terjadi
4. Perlu pengembangan hukum dalam *e-commerce* ini.

2.5 Rekayasa Perangkat Lunak

Bidang profesi dan ilmu dalam teknologi informasi yang menjelaskan bagaimana teknik membangun sebuah perangkat lunak dengan memperhatikan seluruh aspek – aspek yang ada, dimulai dari tahapan awal yaitu *requirement engineering* sampai tahapan akhir yaitu pemeliharaan (Sommerville, 2011). Berikut adalah kriteria yang harus diperhatikan dalam melakukan rekayasa perangkat lunak:

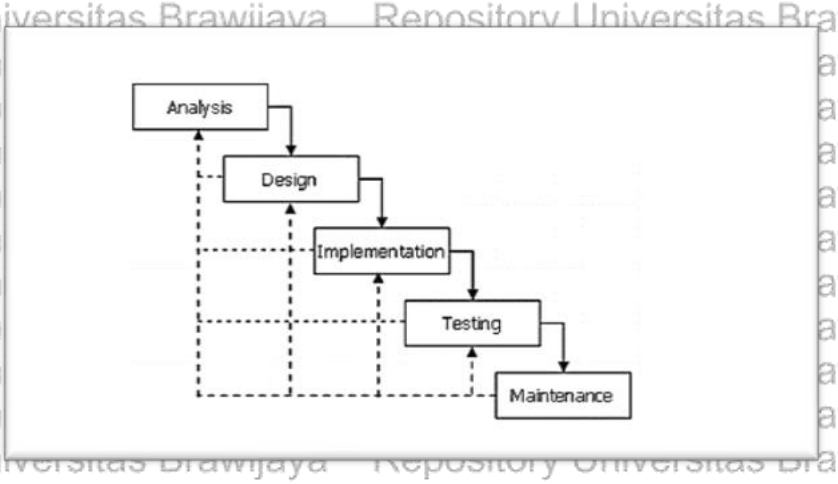
1. Tingkat pemeliharaan dan perawatan tinggi
2. Mengikuti perkembangan zaman
3. Mengikuti keinginan pengguna
4. Efektif dan efisien
5. memenuhi kebutuhan yang diinginkan

2.6 Software Development Life Cycle

Menurut (Bassil, 2012), banyak dikembangkan model atau tahapan-tahapan untuk membantu dalam mengembangkan perangkat lunak. *Software Development Life Cycle (SDLC)* umumnya didasarkan pada model proses sistem. SDLC merupakan proses pemahaman sebuah sistem informasi dalam mendukung kebutuhan bisnis dengan cara merancang suatu sistem. Dalam pengembangan SDLC terdapat banyak sekali model SDLC yang dapat digunakan oleh pengguna, contohnya adalah *waterfall model*, *prototype* dan lain lain. Setiap model SDLC memiliki kelebihan dan kekurangan, sehingga dalam penerapannya tergantung kebutuhan dari sistem yang ingin dikembangkan.

2.6.1 *Waterfall Model*

Model yang sering digunakan pada pengembangan perangkat lunak adalah *Waterfall*. Umumnya Proyek besar pemerintah dan perusahaan-perusahaan besar menggunakan *SDLC waterfall* sebagai model dalam pengembangan aplikasi. (Alshamrani & Bahattab, 2015). Dapat dijelaskan bahwa SDLC ini mempunyai 5 tahapan yaitu pendefinisian perencanaan sistem, kebutuhan, mengimplementasikan kode pemrograman, pengujian unit, pengujian sistem dan tahap pemeliharaan. Berikut merupakan gambaran visual tahap *SDLC waterfall*.



Gambar 2.3 Proses pengembangan sistem menggunakan *Waterfall Model*

Sumber : (Imam & Azhari, 2008)

2.6.2 Fase Pengembangan Perangkat Lunak

Fase-fase yang ada pada model *waterfall* dijelaskan secara mendetail seperti berikut ini:

1. *Requirements Definition*

Pada tahap awal dilakukan elisitasi kebutuhan, ada banyak macam cara untuk mendefinisikan kebutuhan pengguna, dapat dilakukan dengan observasi, wawancara ataupun dari dokumen yang berhubungan dengan subjek. Pemodelan kebutuhan dapat menggunakan *Use Case Diagram*, dijelaskan dengan *Use Case Scenario*.

2. *System and Software Design*

Desain teknis dilakukan berdasarkan kebutuhan fungsional yang telah di definisikan pada tahap elisitasi kebutuhan. Ada beberapa jenis yang dapat dirancang untuk sistem dimulai dari arsitektur, kode, *database model.* Pada perancangan berdasarkan pendekatan berorientasi objek dapat menggunakan *UML* seperti*, Class Diagram, Sequence Diagram, ERD* Pemodelan disesuaikan dengan kebutuhan pengembangan dan teknologi yang digunakan.

3. *Implementation*

Tahap Implementasi pada model waterfall dapat dimulai saat seluruh perancangan sistem dilakukan. Hasil dari perancangan diterjemahkan menjadi kode program, struktur data dan arsitektur.

4. *Testing*

Tujuan dari dilakukan pengujian adalah untuk memastikan perangkat lunak sudah layak di *deploy*. Ada banyak macam pengujian yang dapat dilakukan tergantung subjek atau objek yang ingin diuji. Berikut merupakan macam pengujian yang dapat dilakukan:

a. *White-box Testing*

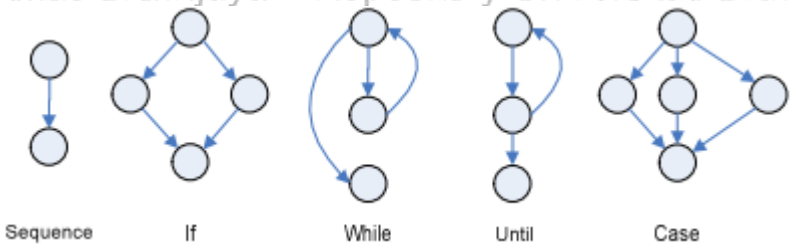
Tujuan dilakukanya *white-box testing* adalah untuk menguji struktur internal design dan kode. Pada jenis pengujian ini kode diperlihatkan kepada penguji (Pressman & Maxim, 2015). Salah satu Teknik pengujian yang dapat dilakukan pada *white-box testing* adalah *basis path testing*.

Pengujian jalur dasar dilakukan untuk mengukur tingkat kompleksitas dan hasil turunannya dapat untuk merancang kasus uji untuk menentukan jalur-jalur dasar. Pengujian tersebut untuk memastikan jalur kode akan dieksekusi setidaknya satu kali.

Tahapan yang dilakukan untuk melakukan *basis path testing* adalah sebagai berikut:

- Membuat bentuk visual dari algoritma / kode program berupa *flow graph*
- Menghitung kompleksitas
- Menentukan jalur dasar berdasarkan node
- Membuat kasus uji untuk tiap jalur

Flow Graph yang dapat digambarkan berdasarkan jenis control yang terdapat pada kode atau algoritma yang diuji. Berikut merupakan gambar dari *flow graph* berdasarkan jenis kontrol.



Gambar 2.4 *Flow Graph* berdasarkan jenis kontrol

Sumber : (Imam & Azhari, 2008)

b. *Black-box Testing*

Pengujian ini dilakukan dengan tujuan untuk memastikan input dan output dari kebutuhan yang telah didefinisikan pada tahap analisis kebutuhan diawal fase pengembangan perangkat lunak.

c. *Usability Testing*

Pengujian ini merupakan jenis pengujian untuk mengukur kepuasan *user* terhadap aplikasi serta nilai yang diambil merupakan subjektifitas pengguna (Bevan, 1995). Kuesioner dapat menjadi salah satu alat untuk mendapatkan nilai dari subjektifitas calon pengguna. Jenis pertanyaan yang digunakan pada kuesioner biasanya adalah skala ataupun pilihan ganda. *Post-study System Usability Questionnaire* merupakan salah satu metode untuk mengambil data

subjektif dari pengguna yang merupakan standar pertanyaan dan penilaian dari *IBM*.

Post-study system usability questionnaire merupakan kuesioner dengan menggunakan 3 kategori pertanyaan untuk mewakilkan 3 aspek yaitu kegunaan aplikasi, kualitas informasi dan kualitas antar muka (Sauro and Lewis, 2012). Kuesioner ini menggunakan pertanyaan dengan skala dari 1- 7 semakin rendah nilainya semakin tidak setuju.

5. *Operation and Maintenance*

Setelah tahap pengujian dilakukan dan perangkat lunak dinilai cukup layak, maka tahap *Depeloyment* dapat dilakukan. Apabila pengguna merasakan adanya *error* maka tahap *Maintenance* dapat digunakan untuk memperbaiki beberapa *error* pada sistem tersebut sehingga dapat membuat perangkat lunak mendapatkan perbaikan ataupun peningkatan versi yang lebih baik.

Namun pada penelitian ini tidak dilakukan tahap *operation and maintenance* karena diperlukan waktu dan resource yang besar. Tahap ini memungkinkan untuk dilakukan pada penelitian selanjutnya.

Semua tahapan yang telah disebutkan diatas akan terus mengalir satu sama lain. Tahapan selanjutnya dilakukan saat hasil sudah didapatkan pada fase sebelumnya.

2.7 Pendekatan *Object Oriented*

Sebuah sistem yang menggunakan pendekatan *object oriented* terdiri dari beberapa *object* yang saling berinteraksi dengan menyediakan fungsi – fungsi dan operasi bagi *object* itu sendiri. Proses desain sistem dengan pendekatan *object oriented* melibatkan desain kelas objek dan hubungan antar kelas. Untuk mengembangkan sebuah sistem dengan pendekatan *object oriented* memerlukan hal – hal sebagai berikut: (Sommerville, 2011)

1. Memahami dan mendefinisikan konteks dan interaksi eksternal dengan sistem.
2. Desain arsitektur sistem.
3. Identifikasi objek – objek utama dalam sistem.
4. Pengembangan model desain.
5. Antarmuka sistem.

2.8 Unified Modelling Language

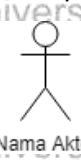
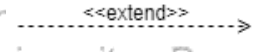
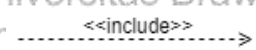
Apabila *Unified Modelling Language* (UML) dilihat dalam segi industriUML merupakan Bahasa ataupun alat untuk merancang, menggambarkan dan mendokumentasikan dalam fase pengembangan perangkat lunak. UML dapat digunakan untuk memodelkan pengembangan perangkat lunak dengan *Object Oriented* maupun *Structural* dan *Functional* juga dapat diterjemahkan ke Bahasa pemrograman apapun dan teknologi yang digunakan (Sulistyorini, 2009). Dalam pengembangan sistem perangkat lunak, UML sudah menjadi bahasa pemodelan baku. Yang paling utama dalam permodelan UML yaitu menjelaskan aspek

fungsionalitas sistem (Kurniawan T. A., 2018). Konsep yang diterapkan yaitu setiap informasi dari hasil pengolahan data akan dilakukan permodelan dalam *Use Case Diagram* dan setiap masing-masing proses digambarkan dalam *Activity Diagram, Sequence Diagram*, *Class Diagram, State Diagram dan Deployment Diagram*. Pada penelitian ini akan menggunakan *6 UML*.

2.8.1 Use Case Diagram

Diagram yang dapat digunaka untuk menggambarkan kebutuhan secara visual dan interaksi antara sistem dengan actor adalah *Use Case*. *Use case* digunakan untuk memahami seluruh aktifitas yang ada pada sistem serta menggambarkan seluruh aktor yang terlibat pada aktifitas tersebut (Sommerville, 2011). Spesifikasi perilaku (fungsionalitas) yang dibutuhkan actor dapat didefinisikan dan dijelaskan menggunakan *Use Case* (Kurniawan T. A., 2018). Simbol – simbol yang terdapat pada *Use Case Diagram* terdapat pada tabel dibawah ini.

Tabel 2.2 Simbol *Use Case Diagram*

No.	Simbol	Deskripsi
1.	*Use case* 	Unit fungsional yang berinteraksi dengan aktor
2.	Aktor/*Aktor* 	Memvisualisasikan bentuk dari orang, sistem lain atau proses dari luar yang berinteraksi dengan sistem.
3.	Asosiasi/*Association*	Menggambarkan relasi/ hubungan antara *actor dan use case*
4.	Ekstensi/*Extend* 	Relasi yang menghubungkan antara *use case* satu dan yang lainnya dimana *use case* tersebut dapat berdiri sendiri tanpa *use case* lainnya
5.	Generalisasi 	Relasi yang memiliki makna dari umum ke khusus
6.	*Include* 	Hubungan ini mirip dengan extend namun *use case* satu yang terhubung bergantung kepada *use case* lainnya

2.8.2 Use Case Scenario

Skenario dari use case yang menjelaskan prosedur, tujuan, alur dan kondisi sebelum ataupun sesudah *usecase* di jalankan. Pada Tabel 2.3 ini adalah format notasi *use case scenario*:

Tabel 2.3 Notasi *Use Case Scenario*

Nama *Use case*	Nama *Use case* yang menjadi pokok bahasan
Aktor	Aktor yang dapat menjalankan / terhubung dengan *use case*.
Objektif	Apa yang dilakukan dilakukan oleh aktor secara objektif.
Pre-condition	Kondisi yang menjadi syarat *use case* dijalankan.
Main-flow	Jalur utama dimulai dari kondisi awal hingga kondisi akhir.
Alternative-flow	Jalur alternatif yang menjelaskan alur yang berbeda dari alur utama, biasanya menjelaskan jika terjadi kesalahan dari sistem maupun pengguna.
Post-condition	Kondisi akhir ketika *Main_Flow* selesai dijalankan.

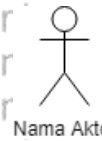
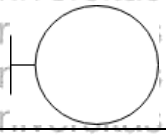
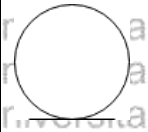
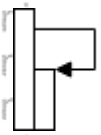
2.8.3 Activity Diagram

Activity diagram merupakan sebuah diagram yang menggambarkan aliran kerja yang didalamnya terdapat tindakan, aktivitas dan juga pilihan serta pengulangan (mheducation, 2020). Pada Unified Modeling Language (UML), diagram aktifitas dibuat untuk menggambarkan aktivitas komputer atau aktivitas pada aplikasi. Diagram aktivitas merupakan gambaran alur kontrol secara garis besar.

2.8.4 Sequence Diagram

Sequence diagram merupakan penjelasan tentang interaksi antar objek dan kelas yang ada dalam aplikasi berdasarkan garis waktu (*life line*). Perancangan *Sequence Diagram* mengacu kepada *Use Case Scenario*. Objek-objek *Sequence Diagram* dijelaskan seperti pada Tabel 2.4.

Tabel 2.4 Simbol *Sequence Diagram*

No.	Simbol	Deskripsi
1.	Aktor/*Aktor* 	Entitas yang berada diluar sistem dan melakukan interaksi dengan sistem
2.	*Boundary Class* 	*Interface* / antarmuka berupa kelas yang berinteraksi dengan aktor.
3.	*Entity Class* 	Kumpulan kelas yang berinteraksi dengan basis data dan merupakan bagian dari sistem.
4.	*Control Class* 	Objek / kelas yang berisi proses kontrol / logika.
5.	*Message*	Arah pesan yang dikirim dan bentuk isinya antar entitas.
6.	*Recursive* 	Pesan yang dikirimkan pada dirinya sendiri.
7.	*Activation* 	Menggambarkan durasi aktivasi dari operasi.
8.	*Lifeline* 	Terdapat aktivasi di tiap lifeline, berada pada tiap entitas.

2.8.5 Class Diagram

Class Diagram digunakan untuk memvisualisasikan bagaimana abstrak atau detail dari struktur perangkat lunak dalam bentuk kelas secara komprehensif yang nantinya akan diimplementasikan pada perangkat lunak. *Class Diagram* akan menunjukkan interaksi antar kelas. Simbol generalisasi berbentuk panah dengan ujung bentuk segitiga menggambarkan relasi dari umum ke khusus (generalisisasi - spesialisasi) digunakan untuk menggambarkan sintaks extend pada kode program.

2.9 JavaScript

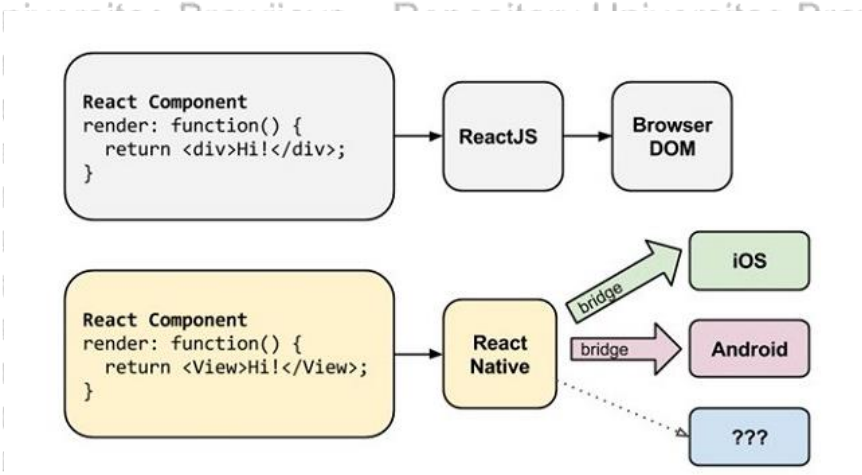
Javascript merupakan suatu bahasa pemrograman yang dinamis dan tingkat tinggi (Flanangan, 2011). Javascript dapat dikembangkan dengan paradigma *Functional Programming*, *Object Oriented Programming* dan *Structural*. Javascript awalnya hanya dapat di implementasikan pada *client side*, namun saat ini *engine javascript* dapat disisipkan pada perangkat lunak *web server & database*. *Javascript* juga dapat di gunakan untuk pengembangan perangkat *mobile* melalui *framework react-native*.

2.10 React

React merupakan *Library JavaScript* yang dikembangkan oleh facebook untuk membangun sebuah antar muka atau sering disebut *user interface* sebuah aplikasi (ReactJs.org). *React* menghasilkan tampilan yang interaktif dan secara efisien akan memperbarui dan membuat komponen yang tepat ketika data berubah. Selain itu *ReactjS* juga bersifat *composable user interface*, sehingga dapat membuat "*UI*" yang menarik dan dapat dibagi menjadi beberapa komponen.

2.10.1 React Native

React native merupakan bahasa pemrograman dimana fitur tersebut disediakan oleh *library react native* yang dikembangkan oleh *facebook (reactnative.dev, 2020). React native merupakan* sebuah *framework javascript hasil* pengembangan dari *facebook* yang ditujukan untuk mengembangkan serta membuat suatu aplikasi pada *mobile* IOS *atau android* dengan menggunakan teknologi *web*. Mekanisme kerja *React Native* yaitu dengan cara menanamkan rangkaian *file Javascript* didalam aplikasi dan menjalankannya secara lokal dari aplikasi yang telah dibuat. *React Native* juga mengunakan *Javascript* untuk *styling* pada *UI* dan *UX*. Berikut adalah mekanisme kerja *React Native* yang dapat dilihat pada Gambar 2.5

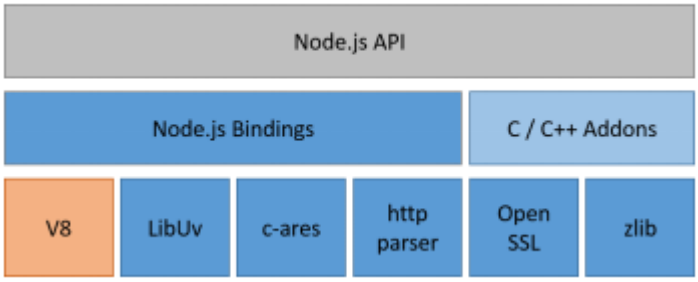


Gambar 2.5 Mekanisme *React Native*

(Sumber: medium.com)

2.11 NodeJS

NodeJS merupakan bahasa pemrograman yang ditulis dalam sintaks bahasa pemrograman *JavaScript* dan di desain untuk mengembangkan aplikasi berbasis *web*. Namun kini *NodeJs* berkembang menjadi *server* yang dapat berjalan sendiri tanpa memerlukan program server seperti *Apache*. Hal ini merepresentasikan sebuah paradigma *"javaScript everywhere" (gcumo, 2013)*. *NodeJs* juga merupakan *script library* dan runtime environment maka dari itu NodeJs dapat menjalankan fungsi dari Bahasa pemrograman dan juga memiliki kumpulan script yang dapat digunakan sewaktu waktu. Arsitektur *NodeJS* dapat dilihat pada Gambar 2.6.



Gambar 2.6 Arsitektur *NodeJS*

(Sumber: medium.com)

2.11.1 NPM (*Node Package Manager*)

NPM atau paket *manager* untuk *Node.js* adalah *command line* untuk mengembangkan serta mempublikasikan *module package* selain itu NPM juga dapat membantu pengembang *Javascript* dengan saling berbagi kode paket modul, hal ini ditemukan pada tahun 2009.

2.12 Firebase

Firebase merupakan layanan dari *google* yang menyediakan layanan *storage* dan berbagai fitur yang memungkinkan mempermudah *backend* dalam membangun sebuah aplikasi. Terdapat beberapa fitur *firebase* yaitu *firebase cloud* dan *firebase auth*.

2.12.1 Firebase Auth

Firebase Auth merupakan fitur dari *firebase* yang menyediakan layanan *backend* dengan SDK yang siap untuk digunakan untuk mengauthentikasi pengguna ke sebuah aplikasi yang ingin dikembangkan. Dengan fitur ini pengguna dapat melakukan authentikasi menggunakan gmail, facebook, twitter dan lainnya (firebase, 2020).

Firebase Cloud merupakan fitur dari *firebase* untuk para developer dengan tujuan untuk menyimpan file media yang di integrasikan dengan modul-modul *API*, dapat di integrasikan dengan *Java*, *PHP* dan *JavaScript* (firebase, 2020).

2.13 MongoDB

MongoDB adalah *database cloud* yang *full control* digunakan untuk menangani semua kompleksitas saat pengelolaan pada penyedia layanan *cloud* seperti *AWS, Azure*, dan *GCP.* Kemudian digunakan juga untuk menangani semua kompleksitas pada saat pendistribusian. *MongoDB* merupakan produk *database* yang menyimpan data dalam bentuk *JSON* dan merupakan *database NoSQL.* (MongoDB.com, 2020). *NoSQL* digunakan menampung data yang dinamis strukturnya secara efisien dalam skala besar (*big data/cloud*). *Grid Computing, Cloud,* atau *Big Data* merupakan aplikasi yang sering dipakai pada MongoDB. Kemudian dalam pengaksesan atau manipulasi datanya yaitu dapat menggunakan OOP (*Object Oriented Programming)*. *NoSQL* tidak mengenal skema tabel dan format data yang kaku. *NoSQL* sangat cocok untuk data yang bersifat dinamis, atau disebut *Dynamic Schema*.

MongoDB juga memiliki layanan cloud yang cakupanya global karena memiliki berbagai server otonom dan juga pada AWS, Azure dan GCP, Layanan ini disebut MongoDB Atlas.

2.14 Domainesia

Domainesia merupakan penyedia domain, hosting web dan VPS. Pada layanan hosting terdapat berbagai platform untuk server yaitu python, PHP dan NodeJs. Untuk DBMS domainesia menyediakan *PostgreSQL* dan *phpMyAdmin* (Domainesia, 2020).

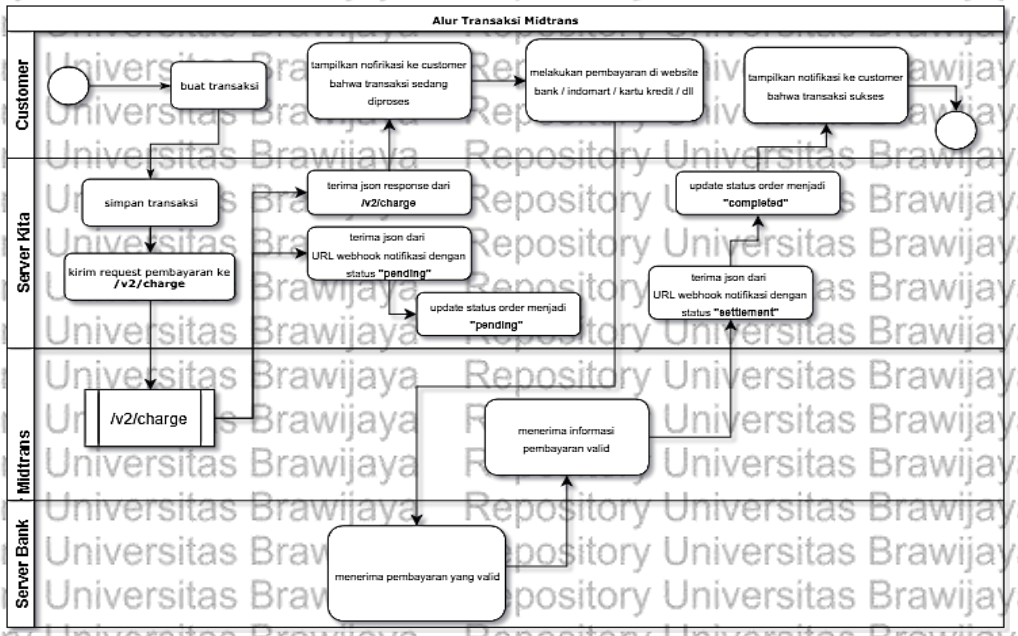
2.15 Midtrans

Midtrans adalah salah satu *payment gateway* yang menyediakan fasilitas berbagai cara pembayaran (Midtrans.com, 2020). *Card payment, bank transfer, direct debit, e-wallet dan over the counter* merupakan metode pembayaran yang

disediakan oleh *Midtrans.* Terdapat berbagai keuntungan menggunakan *midtrans* diantaranya:

1. Metode pembayaran terintegrasi dengan mudah.
2. Cara pembayaran di *Midtrans Payment* terdapat 20 metode pembayaran.
3. Mendukung berbagai platform.
4. *Anti-fraud* merupakan sistem yang lebih aman apabila digunakan.
5. Ribuan bisnis di Indonesia mempercayai *Midtrans Payments.*

Berdasarkan keuntungan diatas, pengembangan aplikasi *E-Commerce* ini menggunakan *Midtrans* sebagai *payment gateway* karena sangat mudah untuk di gunakan dan sudah banyak di terapkan pada *startup* besar seperti Bli-Bli, Gojek, Tokopedia dan lain-lain. Cara kerja *MidTrans* ketika di implementasikan sebagai pihak ketiga yang menangani pembayaran dapat dilihat pada Gambar 2.7.



Gambar 2.7 Cara Kerja *MidTrans*

(Medium.com, 2020)

Cara kerja *payment getaway MidTrans* Indonesia ini sama seperti *payment gateway* pada umumnya. *Midtrans* menghubungkan service pada bank ke aplikasi milik pengembang, lalu pengembang menghubungkan dengan pelanggan. Ada dua mode pada *midtrans*, yaitu mode *sandbox* dan mode *production*. Mode *Sandbox* memberikan seluruh *Api* yang ada pada mintrans untuk dicoba, namun transaksi yang dilakukan hanya sebatas simulasi. Pada mode *production Api* yang ada sama seperti pada mode *sandbox*, namun transaksi yang dilakukan akan diteruskan pada pihak utama yaitu bank atau pada penyedia metode pembayaran yang dipilih.

Untuk bisa menggunakan midtrans pertama, daftarkan toko *online* sendiri dengan cara terhubung ke *merchant* admin portal *MidTrans* didalam *mode testing*. Di *mode* tersebut dapat mencoba berbagai macam pengaturan yang telah tersedia tanpa perlu mengubah apapun dan dapat melakukan simulasi pembayaran. Akun tersebut dapat dimanfaatkan untuk keperluan bisnis, baik bersifat individu maupun kolektif seperti sebuah perusahaan. Terakhir dapat mengintegrasikan teknis yang sudah disediakan ke situs bisnis *online* atau *e-commerce* dengan tim *developer*. Ini dilakukan agar bisa memudahkan proses *check out* dari pelanggan.

2.16 Raja Ongkir

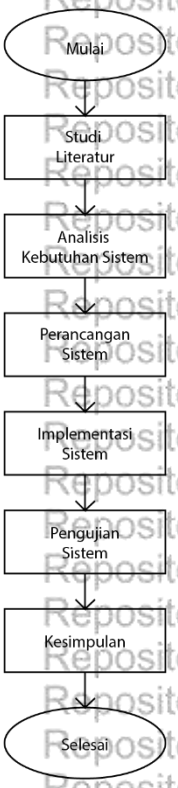
RajaOngkir merupakan sebuah *web service (API)* dan situs yang menyediakan data ongkos kirim dan juga macam layanan dari berbagai kurir di Indonesia seperti JNE, POS Indonesia, TIKI, ESL, PCP, dan RPX. RajaOngkir dapat di implementasikan pada Bahasa pemrograman *JavaScript* dan *PHP* (RajaOngkir, 2020).

2.17 Invoice-Generator

Invoice-Generator merupakan layanan yang menyediakan *template invoice* dan dapat membuat invoice secara otomatis dengan cara mengisi data pada *web official* ataupun dengan integrasi menggunakan *API* (Invoice-Generator,2020).

BAB 3 METODOLOGI

Pada bagian ini menjelaskan langkah–langkah untuk melakukan penelitian ini. Tahapan-tahapan yang dilakukan pada penelitian ini yang pertama melakukan studi literatur yang percabanganya ada landasan teori, referensi dan penelitian–penelitian sebelumnya yang berkaitan dengan penelitian ini. Kedua, analisis kebutuhan yang digunakan untuk menentukan kebutuhan apa saja yang diperlukan. Lalu yang ketiga, yaitu fase perancangan dimana fase ini menjadi dasar dan fondasi untuk melakukan implementasi. Selanjutnya adalah fase Implementasi yaitu pembuatan kode program dari semua hasil dari analisis kebutuhan dan desain perancangan. Hal kelima yang harus dilakukan dan sangat penting adalah adalah pengujian, karena ditahapan inilah suatu produk perangkat lunak atau aplikasi dilakukan *testing* dan untuk memastikan bahwa aplikasi yag dibuat sudah layak untuk di *deploy*. Hal keenam dan terakhir yang harus dilakukan yakni pengambilan kesimpulan dan saran untuk pengembangan penelitian selanjutnya. Kesimpulan dan saran menjadi sangat penting karena nantinya akan dijadikan sebagai referensi untuk penelitian selanjutnya. Berikut adalah metodologi penelitian yang digunakan, seperti pada Gambar 3.1.



Gambar 3.1 Diagram Alur Metodologi Penelitian

3.1 Studi Literatur

Pada tahap studi literatur terjadi proses referensi penelitian yang relevan dengan penelitian ini dan teori untuk menjadi pedoman untuk penyelesaian permasalahan yang telah di definisiakan (Permatasari, Studi, Informatika, Aset, & Mobile, n.d.). Tujuannya agar pengetahuan dasar untuk membangun sebuah sistem terpenuhi dengan cukup. Sumber teori pendukung dan pemahaman tentang penilitian ini diperoleh dari Jurnal, buku, *ebook* dan terdapat penelitian sebelumnya terkait dengan penelitian ini. Sebagai pendukung untuk penelitian ini maka berikut adalah daftar literatur yang akan digunakan sebagai dasar ataupun referensi:

1. Tinjauan Pustaka
2. Landasan Teori
    a. *Anfield Collection*
    b. Proses Bisnis Anfield *Collection*
    c. Pengembangan Perangkat Lunak
        1. *Waterfall Model*
        2. Pendekatan Pengembangan Perangkat Lunak
        3. Pemodelan OO
    d. Teknologi Pengembangan Perangkat Lunak
        1. *React Native*
        2. *NodeJs*
        3. *MongoDB*
        4. *Midtrans*
        5. *Firebase*
        6. Domainesia
        7. RajaOngkir
        8. *Invoice-Generator*

3.2 Analisis Kebutuhan

Pada tahap ini dilakukan elisitasi kebutuhan dari anfield collection, kemudian membandingkan e-commerce yang sudah ada, lalu mempelajari keunikan seni kerajinan setelah dapat dilakukan pendefinisian kebutuhan, berikut merupakan rincian tahap ini dengan pendekatan *object oriented analysis*:

1. Menganalisis dan mengelisitasi kebutuhan dan aplikasi *e-commerce* secara umum menjadi definisi dan spesifikasi kebutuhan dimana hal tersebut telah diidentifikasi melalui suatu penomoran dengan melihat kebutuhan utama Anfield *collection*.
2. Pada tahap ini dilakukan pendefinisian kebutuhan dengan kategori kebutuhan fungsional dan non fungsional berdasarkan hasil dari analisis dan elisitasi kebutuhan.
3. Pada masing – masing kebutuhan yang terdapat pada kebutuhan fungsional perlu dilakukan permodelan kebutuhan *use case diagram*
4. Penjelasan *Use Case* yang terdapat pada *Use Case Diagram, dijelaskan lebih detail pada Use Case Scenario. Dari* pemetaan kebutuhan fungsional, pemodelan kebutuhan *Use Case* Diagram dan *Use-Case Scenario* akan digunakan dalam melakukan perancangan sistem selanjutnya.

3.3 Perancangan Sistem

Pada tahapan perancangan sistem ini diharapkan dapat mempermudah dalam implementasi. Hasil dari tahapan ini digunakan sebagai pedoman dalam melakukan perancangan, dalam hal ini tahapan yang digunakan adalah pendekatan OO. Berikut merupakan pendekatan OO yang digunakan pada tahap perancangan sistem:

1. Hasil dari pemodelan kebutuhan *Use-Case Scenario dilakukan* pemodelan 5 *Sequence Diagram* sebagai sampel.
2. Hasil dari pemodelan *Use-Case Scenario* dibuat *activity diagram* untuk menggambarkan 5 aktivitas ataupun alur dari fitur utama secara garis besar ataupun detail.
3. Dari masing-masing *Sequence Diagram* dilakukan pengelompokan menjadi beberapa klas pada pemodelan *Class Diagram.* Akan tetapi hanya 5 kelas yang jabarkan variable & fungsinya.
4. Membuat algoritma dari masing-masing fungsi pada setiap klas. Hasil dari permodelan *Class Diagram* tersebut menjadi acuan dalam implementasi, dan hanya terdapat tiga sampel algoritma yang dijelaskan.
5. Dalam perancangan basis data, dilakukan pembuatan model *Entity Relationship Diagram*.
6. Kemudian yang dilakukan selanjutnya adalah membuat rancangan antarmuka dan hanya beberapa yang diambil untuk dijadikan sampel dalam rancangan antarmuka tersebut. Hasil rancangan antarmuka digunakan pada saat melakukan implementasi *Front-End system* sedangkan permodelan *Class Diagram* dan hasil perancangan basis data digunakan pada saat melakukan implementasi pada sisi *Back-End*.

3.4 Implementasi

Hasil rancangan yang telah dilakukan dan akan diimplementasikan disebut dengan implementasi sistem. Adapun proses yang dilakukan pada saat implementasi yaitu sebagai berikut:

1. Proses mengimplementasikan seluruh kebutuhan yang telah didefinisikan pada tahap perancangan atau disebut dengan Implementasi kebutuhan sistem.
2. Proses implementasi dari perancangan basis data atau disebut dengan Implementasi basis data digambarkan melalui *Entity Relation Diagram, mongoDB* adalah jenis *DBMS* yang digunakan. *MongoDB* merupakan jenis basis data *nosql* dimana dalam pengaplikasian nya akan sangat dinamis maka implementasi yang di dokumentasikan merupakan tangkapan gambar dari struktur dokumen (*JSON*) dari sistem yang sudah dijalankan.
3. Proses mengimplementasikan *pseudocode* yang telah dibuat pada fase perancangan menjadi sebuah kode dalam Bahasa pemrograman *Javascript* atau disebut dengan Implementasi kode program
4. Proses implementasi dari hasil rancangan antarmuka atau disebut dengan Implementasi antarmuka. Hal ini digunakan untuk mengimplementasikan antarmuka sistem menggunakan *framework react native* atau format *JSX.*

3.5 Pengujian

Pada proses ini, dilakukannya evaluasi sistem terhadap hasil implementasi sistem sesuai dibutuhkan atau telah definisikan pada tahap rekayasa kebutuhan. Tujuan dari dilakukanya pengujian pada aplikasi yaitu untuk mengurangi peluang terjadinya kesalahan dan mengetahui apakah terdapat kesalahan dalam sistem. Terdapat beberapa aspek yang diuji dalam penelitian ini dengan maksud untuk menilai kualitas dari sistem, sebagai berikut:

1. *Unit Testing*

   Pengujian digunakan untuk memastikan kelayakan pada tiap-tiap komponen dapat disebut dengan pengujian unit / *unit testing*. Komponen yang ada didalam nya berupa atribut (objek), *class* dan fungsi. Pada penelitian ini, pengujian dilakukan dengan mengambil 3 buah fungsi yang dijadikan kasus uji. Pada penelitian ini, metode *white-box* digunakan sebagai metode pengujian unit.

2. *Validation Testing*

   Untuk memastikan setiap fungsi yang diimplementasikan sudah sesuai dengan apa yang didefinisikan sebelumnya maka dilakukan pengujian validasi. *Metode black*-box merupakan metode *Validation Testing* yang dilakukan pada penelitian ini.

3. *Usability Testing*

Untuk menilai dari sisi penggunaan aplikasi maka dilakukan *Usability Testing* pada calon pengguna agar dapat diambil kesimpulan berdasarkan subjektifitas pengguna. Faktor-faktor yang akan dinilai pada pengujian ini adalah sebagai berikut: kualitas informasi yang terdapat dalam sistem, kualitas antarmuka sistem, dan juga kualitas dari sistem tersebut. Metode yang diambil untuk mendapatkan respon dari calon pengguna yaitu dengan menggunakan metode *poststudy system usability questionnaire* pada aktifitas jual dan beli. Metode ini merupakan metode yang berisikan kuesioner yang didalam nya terdapat 15 pernyataan mewakili 3 aspek yaitu penggunaan, antarmuka & informasi.

4. *Compability Testing*

Pengujian kompabilitas dilakukan untuk memastikan aplikasi dapat dijalankan sesuai dengan aspek kompabilitas pada kebutuhan non fungsional. Metode pengujian akan dijelaskan pada sub bab compability testing. Pengujian ini hanya dilakukan jika di deklarasikan pada kebutuhan non fungsional.

3.6 Kesimpulan & Saran

Kesimpulan yang diambil dari penelitian ini merupakan hasil yang di dapatkan dari proses awal hingga akhir, dari penelitian yang berjudul "Pembangunan aplikasi *E-commerce* Menggunakan *Payment Gateway Midtrans*". Kesimpulan yang dapat diperoleh secara valid berdasarkan bukti dari pengujian yang sudah dilakukan. Penarikan kesimpulan ditujukan untuk menjawab perumusan permasalahan yang sudah dirumuskan sebelumnya.

Saran juga dituangkan pada akhir penulisan, penulis perlu melakukan evaluasi dalam penelitian ini mulai dari kekurangan ataupun kesalahan yang terjadi. Saran dilakukan dengan tujuan dapat diinisiasi untuk pengembangan ataupun perbaikan jika penelitian selanjutnya dilakukan. Saran dapat diambil dari berbagai aspek mulai dari pola perancangan, metode pengembangan, fitur tambahan dan lain sebagainya. Setelah penelitian selesai diharapkan ada penelitian selanjutnya yang ditujukan untuk mengembangkan penelitian ini dengan mengambil saran yang dilakukan oleh penulis.

BAB 4 REKAYASA KEBUTUHAN

Rekayasa kebutuhan yang dilakukan pada pengembangan aplikasi *e-commerce* ini dilakukan dengan cara wawancara kepada pemilik Anfield *Collection*. Setelah melakukan wawancara dengan pemilik, dilakukan observasi untuk melihat proses bisnis yang sedang berjalan. Lalu setelah kebutuhan didapat maka tahap selanjutnya dibuat daftar kebutuhan fungsional dan non fungsional. Pemodelan *Use Case* diagram dilakukan setelah kebutuhan didapatkan. Tahap terakhir adalah dengan membuat *Use-Case Scenario* untuk menjelaskan tahapan, *objective*, alur utama dan alur alternatif dari *use case*.

4.1 Elisitasi

Berdasarkan wawancara yang telah dilaksanakan terhadap pengelola toko Anfield Collection yaitu Ibu Hepy. Diperoleh informasi bahwa pengelola Anfield Collection memiliki proses bisnis seperti toko pada umumnya yaitu dimulai dengan membuat produk, lalu memasarkan produk setelah itu melakukan transaksi pembayaran dengan pembeli dan mengirim barang jika pembeli berasal dari daerah lain. Akan tetapi produk yang dijual memiliki jenis yang dapat dikategorikan berdasarkan bahan, proses pembuatan dan jenis produk. Dari penjabaran proses diatas penulis menawarkan solusi dengan membuatkan aplikasi seperti aplikasi e-commerce yang umum seperti tokopedia, bukalapak dan elevenia namun dengan. Pengelola setuju dengan solusi yang ditawarkan ditambah dengan spesifikasi fitur tambahan yakni pendefinisian produk berdasarkan 3 parameter yaitu jenis produk, proses pembuatan, bahan dasar. Berikut merupakan sampel 7 dari 25 macam produk yang didapatkan dari Anfield *Collection*:

Tabel 4.1 Data produk mitra Anfield *Collection* (Toko Wijaya)

Kategori	Nama Produk	Deskripsi	Ket.	Bahan
Toko Wijaya, JL. Pasar PIIP Kota Blitar Blok IIIB Email: wijayatoko12@gmail.com				
Tas	Tas mendong	Tas mendong adalah tas yang didesain untuk orang yang banyak barang	Dianyam	Kulit kayu
	Tas passport songket	Tas ini didesain untuk menyimpan semua kartu penting yang dimiliki	Di rajut	Benang khusus

Souvenir	Dompet	Dompet yang cocok untuk dimiliki karena modelnya yang simple dan bervariatif	Dijahit	Kulit imitasi
	Tempat tissue	Sebuah souvenir unik yang digunakan sebagai tempat tissue di ruang keluarga	Dijahit	Kulit khhusus
	Tempat pensil	Suvenir yang cocok untuk menyimpan alat-alat tulis karena bentuknya yang unik	Dijahit	Kulit imitasi

Proses bisnis pada layanan aplikasi didapatkan dari wawancara dan observesi yang dilakukan pada toko anfield *collection*. Di dapatkan bahwa proses bisnis yang sedang berlangsung yaitu dimulai dengan membuat produk, lalu memasarkan produk setelah itu melakukan transaksi pembayaran dengan pembeli dan mengirim barang jika pembeli berasal dari daerah lain. Proses bisnis tersebut merupakan proses jual beli pada umumnya. Berdasarkan observasi dan wawancara pemilik ingin dibuatkan aplikasi *e-commerce* spesifik untuk penjualan produk kerajinan. Fitur yang menjadi pembeda adalah definisi produk yang diberikan parameter berupa jenis produk, bahan dasar dan proses pembuatan.

Proses bisnis pada ecommerce besar di Indonesia di observasi dan didapatkan secara umum proses bisnis seperti dibawah ini.

Tabel 4.2 Proses pemesanan dan pembayaran *e-commerce*

Pemesanan dan Pembayaran Barang	
Tokopedia	*Customer* mendaftar sebagai *member* Tokopedia untuk melakukan pemesanan barang. Sebelum melakukan proses pemesanan, *customer* perlu melakukan pengisian data secara lengkap. Setelah itu keterangan rincian pemesanan dan pemberitahuan untuk ke tahap pembayaran akan dikirimkan kepada *customer* melalui *e-mail*. *Customer* dapat membayar pesanan tersebut dengan memilih berbagai macam cara pembayaran yang tersedia. Pemesanan akan otomatis dibatalkan oleh Tokopedia jika tidak adanya pembayaran dalam waktu 3x24 jam. Kemudian Tokopedia akan memberikan notifikasi melalui e-mail apabila pembayaran telah berhasil.
Bukalapak	*Customer* dapat melakukan pemilihan barang yang apabila telah dipilih, barang tersebut akan masuk kedalam keranjang belanja. Kemudian pembeli

	melakukan pengisian data pengiriman serta memastikan kembali produk yang akan dibeli. Selanjutnya dilakukan proses pembayaran, pada proses pembayaran ini *customer* dapat membayar dengan berbagai jenis pembayaran yang tersedia. uang akan masuk ke dalam saldo *customer* apabila dalam waktu 2x24 jam barang tidak dikirimkan.
Elevenia	Pembeli perlu melakukan *login* sebelum melakukan pemesanan barang. Kemudian pembeli dapat melakukan pemilihan barang yang akan dibeli dan melakukan pengisian data pengiriman barang. Selanjutnya pembeli dapat melakukan pembayaran dengan memilih metode pembayaran yang tersedia sebelum melakukan proses *check out* pemesanan.

Setelah itu pendefinisian alur proses bisnis utama pada layanan aplikasi dapat dijabarkan seperti berikut:

1. Dimulai dari *login*, user yang sudah melakukan registrasi sebelumnya diminta untuk meng-*input*kan *username* dan *password* yang sudah terdaftar.
2. Apabila user belum terdaftar, user diminta untuk melakukan registrasi dengan mengisi formulir pada halaman registrasi.
3. *Member* (user yang sudah melakukan registrasi) dapat merubah informasi profilnya pada halaman edit profil.
4. *Member* penjual harus meng-upload barang yang ingin dijualnya dengan meng-input foto barang, nama barang, harga barang, kategori barang dan deskripsi barang apabila ingin menjual barang bekas miliknya.
5. *Member* penjual dapat merubah informasi barang pada halaman edit barang.
6. *Member* pembeli dapat memilih barang yang ingin dibelinya pada halaman home atau dapat mencari dengan meng-inputkan kata kunci pada kotak form yang terdapat pada navbar. Pembeli dapat memilih maksimal 15 barang dari 3 toko yang berbeda.
7. Semua barang yang sudah dipilih oleh akan ditampung pada *cart*. Pada setiap barang akan terkelompok sendiri berdasarkan pemiliknya. Apabila *member* pembeli sudah selesai memilih barang, maka *member* pembeli diminta untuk meng-inputkan harga tawaran untuk memulai proses pembayaran.
8. Setelah pembayaran dilakukan maka penjual dapat mengirim barang.
9. Ketika barang diterima pembeli dapat merubah status menjadi diterima.

10. Penjual dapat melihat laporan penjualan sesuai dengan rentang waktu yang ditentukan.

Kesimpulan yang di dapat dari proses elisitasi kebutuhan adalah dengan membuat fitur yang memiliki kategori produk khususnya untuk produk kerajinan tanpa mengurangi fitur dasar dari e-commerce.

4.2 Gambaran Umum

Aplikasi yang akan dibuat dengan tujuan untuk menaikan kualitas proses bisnis dari entitas yang menjual utamanya produk kerajinan namun juga dapat digunakan untuk produk yang lainnya agar dinamis. Memudahkan untuk mengelola toko dan produk yang dalam jumlah cukup banyak, Selain itu dapat memperluas pasar agar memaksimalkan pendapatan sebuah bisnis usaha. Selain itu para pemilik toko juga dapat menambahkan memperbarui informasi data apabila terjadi penambahan data apapun. Sistem ini juga terdapat fitur untuk pencatatan jumlah penjualan produk sehingga pemilik toko dapat mengetahui produk apa yang paling banyak diminati.

Secara umum aplikasi yang dibuat memiliki fitur yang mirip dengan aplikasi e-commerce yang umum seperti tokopedia, bukalapak dan elevenia. Namun ada beberapa fitur spesifik untuk aplikasi *e-commerce* ini, karena produk yang jual dapat di kategorikan berdasarkan bahan, jenis produk dan proses pembuatan sesuai dengan karakteristik produk kerajian tradisional.

4.3 Identifikasi Aktor

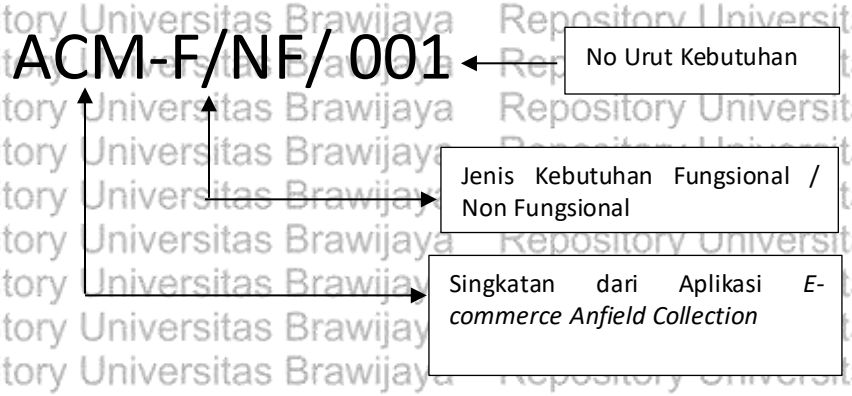
Aktor dapat berupa individu ataupun sistem dari luar yang terhubung ke aplikasi. Namun pada aplikasi ini didapatkan aktor hanya individu. Dapat dijelaskan bahwa terdapat beberapa Aktor yang terlibat seperti Tabel dibawah ini.

Tabel 4.3 Karakteristik Aktor

Aktor	Deskripsi
Admin	*Admin* merupakan pemilik otoritas yang tinggi, Fungsinya dapat mengatur member, barang & kategori yang ada pada aplikasi.
Member	Aktor yang sudah berhasil melakukan pendaftaran dan login pada aplikasi. Memiliki otoritas yang dibawahi oleh admin. *Member* dapat melakukan aktivitas penjualan dan pembelian di dalam aplikasi.
Guest	Entitas yang menggunakan sistem namun belum melakukan login pada aplikasi.

4.4 Kebutuhan Sistem

Kebutuhan sistem berisikan fitur yang harus tersedia untuk menyelesaikan masalah yang ada. Kebutuhan sistem dibagi menjadi 2, Kebutuhan fungsional dan non fungsional. Dalam kasus ini kebutuhan fungsional ditulis menggunakan kode unik yaitu ACM, yang diambil dari nama Aplikasi Anfield Collection berbasis Mobile.Huruf F merepresentasikan jenis kebutuhan fungsional sedangkan untuk kebutuhan non-fungsional di simbolnya dengan huruf NF. Terakhir 001 adalah no urut kebutuhan. Lebih jelasnya dapat lihat pada Gambar 4.6.



Gambar 4.1 Penomoran Kebutuhan Sistem

4.5 Kebutuhan Fungsional

Kebutuhan Fungsional mendeskripsikan fitur-fitur yang harus ada pada Aplikasi *E-commerce* berdasarkan kebutuhan. Selain itu terdapat keterangan yang ditujukan untuk menjelaskan tiap-tiap kebutuhan yang di definisikan. Berikut adalah daftar kebutuhan fungsional untuk aplikasi *E-Commerce* untuk penjualan produk kerajinan.

Tabel 4.4 Kebutuhan Fungsional

No	Kode Kebutuhan	Nama Kebutuhan	Definisi & Spesifikasi
1	ACM-F-01	*Register*	Aplikasi harus memiliki fitur untuk pengguna agar dapat melakukan registrasi.
2	ACM-F-02	Verifikasi Email	Pengguna yang telah mendaftar harus melakukan verifikasi pada *e-mail* yang di daftarkan.
3	ACM-F-03	*Reset* kata sandi	Harus terdapat fitur *reset* kata sandi yang dikirim melalui *email* bagi pengguna yang lupa *password*.

4	ACM-F-04	*Login*	Aplikasi harus memiliki fitur login untuk pengguna sesuai dengan hak akses.
5	ACM-F-05	*Logout*	Pada aplikasi harus terdapat fitur *logout* untuk pengguna yang sudah masuk ke sistem.
6	ACM-F-06	Melihat Profil	Harus terdapat fitur untuk *member* agar dapat melihat profil akunnya.
7	ACM-F-07	Mengubah profil member	Aplikasi harus memiliki fitur yang dapat mengubah profil menber, fitur tersebut dapat dilakukan oleh *member* yang memiliki akunnya atau pun admin.
8	ACM-F-08	Melihat *list member*	Harus terdapat fitur Melihat list *member* untuk admin.
9	ACM-F-09	Menghapus *member*	Pada aplikasi harus ada fasilitas untuk admin dapat menghapus *member*.
10	ACM-F-10	Edit status *member*	Aplikasi harus memiliki fasilitas agar dapat mengubah status member. Fasilitas tersebut harus terdapat pada otoritas admin dan member.
11	ACM-F-11	Tambah Alamat	Pada aplikasi harus terdapat fitur untuk *member* dapat menambahkan alamat.
12	ACM-F-12	Lihat Alamat	Pada aplikasi harus terdapat fitur untuk *member* dapat melihat daftar alamat.
13	ACM-F-13	Edit Alamat	Pada aplikasi harus terdapat fitur untuk *member* dapat edit alamat yang dipilih.
14	ACM-F-14	Hapus Alamat	Pada aplikasi harus terdapat fitur untuk *member* dapat menghapus alamat yang dipilih.
15	ACM-F-15	Tambah kategori	Aplikasi harus memiliki fitur agar admin dapat menambah kategori produk berdasarkan jenis, proses dan bahan utama.
16	ACM-F-16	Melihat *List* Kategori	Aplikasi harus memiliki fitur agar admin dapat melihat *list* kategori

			produk berdasarkan jenis, proses dan bahan utama.
17	ACM-F-17	*Edit* kategori	Aplikasi harus memiliki fitur agar admin dapat mengubah kategori produk berdasarkan jenis, proses dan bahan utama.
18	ACM-F-18	Hapus kategori	Aplikasi harus memiliki fitur agar admin dapat mengubah kategori produk berdasarkan jenis, proses dan bahan utama.
19	ACM-F-19	Tambah Produk	Pada aplikasi harus terdapat fasilitas untuk *member* agar dapat menambah produk.
20	ACM-F-20	*Manage product image*	Pada aplikasi harus terdapat fasilitas untuk *member* & admin agar dapat mengelola gambar produk untuk tiap produk memiliki minimal 1 gambar & maksimal 5 gambar.
21	ACM-F-21	Tambah Gambar	Pada aplikasi harus terdapat fasilitas untuk *member* & admin agar dapat menambah gambar produk hingga 5.
22	ACM-F-22	Hapus Gambar	Pada aplikasi harus terdapat fasilitas untuk *member* & admin agar dapat menghapus gambar produk minimal tersisa 1.
23	ACM-F-23	Melihat List Produk	Pada aplikasi admin & *member* harus dapat melihat *list* produk.
24	ACM-F-24	Mengubah informasi Produk	Pada aplikasi admin & *member* harus dapat melihat informasi produk.
25	ACM-F-25	Menghapus produk	Pada aplikasi admin & *member* harus dapat menghapus produk yang aktif maupun tidak aktif.
26	ACM-F-26	Melihat detail Produk	Pada aplikasi admin & *member* harus dapat melihat produk yang aktif maupun tidak aktif.
27	ACM-F-27	*Edit* Status Produk	Pada aplikasi, admin & *member* harus dapat mengubah status produk.

28	ACM-F-28	Cari produk	Aplikasi harus memiliki fitur untuk *member* dapat melakukan pencarian produk berdasarkan kata kunci.
29	ACM-F-29	*Filter* Kategori	Aplikasi harus memiliki fitur untuk *member* dapat melakukan pencarian produk berdasarkan kategori.
30	ACM-F-30	*Filter* harga	Aplikasi harus memiliki fitur untuk *member* dapat melakukan pencarian produk berdasarkan harga.
31	ACM-F-31	Menambahkan barang ke keranjang	Pada aplikasi harus terdapat fitur untuk *member* menambahkan barang ke keranjang belanja Maksimal 3 toko & 15 produk.
32	ACM-F-32	Melihat Keranjang	Pada aplikasi harus terdapat fitur untuk *member* melihat isi keranjang.
33	ACM-F-33	Mengubah Jumlah Barang pada Keranjang	Pada aplikasi harus terdapat fitur untuk *member* mengubah kuantitas barang pada keranjang.
34	ACM-F-34	Menghapus barang pada keranjang	Pada aplikasi harus terdapat fitur untuk *member* menghapus barang pada keranjang.
35	ACM-F-35	*Order*	Pada aplikasi harus terdapat fasilitas untuk *member* dapat melakukan pesmeanan berdasarkan barang yang ada di dalam keranjang. Maksimal 3 toko & 15 produk.
36	ACM-F-36	Bayar	Aplikasi harus mampu menyediakan fitur *Payment* agar pengguna dapat membayar pesanan yang sudah dibuat
37	ACM-F-37	Melihat Daftar Belanja	Harus terdapat fitur untuk *member* melihat daftar belanja.
38	ACM-F-38	Melihat Daftar *order*	Pada aplikasi harus terdapat fitur untuk admin & *member* melihat daftar *order* yang telah dibayar oleh pembeli.
39	ACM-F-39	Melihat detail *order*	Harus terdapat fitur untuk *member* & admin melihat detail *order*.

40	ACM-F-40	*Input* Resi	Aplikasi harus mampu menyediakan fitur yang memberikan *member* agar dapat mengisi resi ketika order sudah dibayar
41	ACM-F-41	Verifikasi Pengiriman	Aplikasi harus mampu menyediakan opsi verifikasi pengiriman ketika *member* yang menyediakan produk sudah menginput resi
42	ACM-F-42	Unduh *Invoice*	Aplikasi harus dapat menyediakan pilihan pada *member* untuk mengunduh invoice ketika barang pada pesanan sudah diterima (verifikasi pengiriman).
43	ACM-F-43	Melihat *sell report*	Aplikasi harus dapat menyediakan fitur untuk pennggguna melihat laporam hasil penjualan pada rentang waktu yang ditentukan berdasarkan waktu.

4.6 Kebutuhan Non Fungsional

Kebutuhan Non Fungsional didapatkan dari target os pada aplikasi tokopedia, bukalapak & elevenia yang terdapat pada *Requirement Google Playstore*. Dapat disimpulkan Kebutuhan Non-Fungsional yang ada pada Aplikasi E-commerce yang dapat dijelaskan pada Tabel dibawah ini.

Tabel 4.5 Kebtuhan Non Fungsional

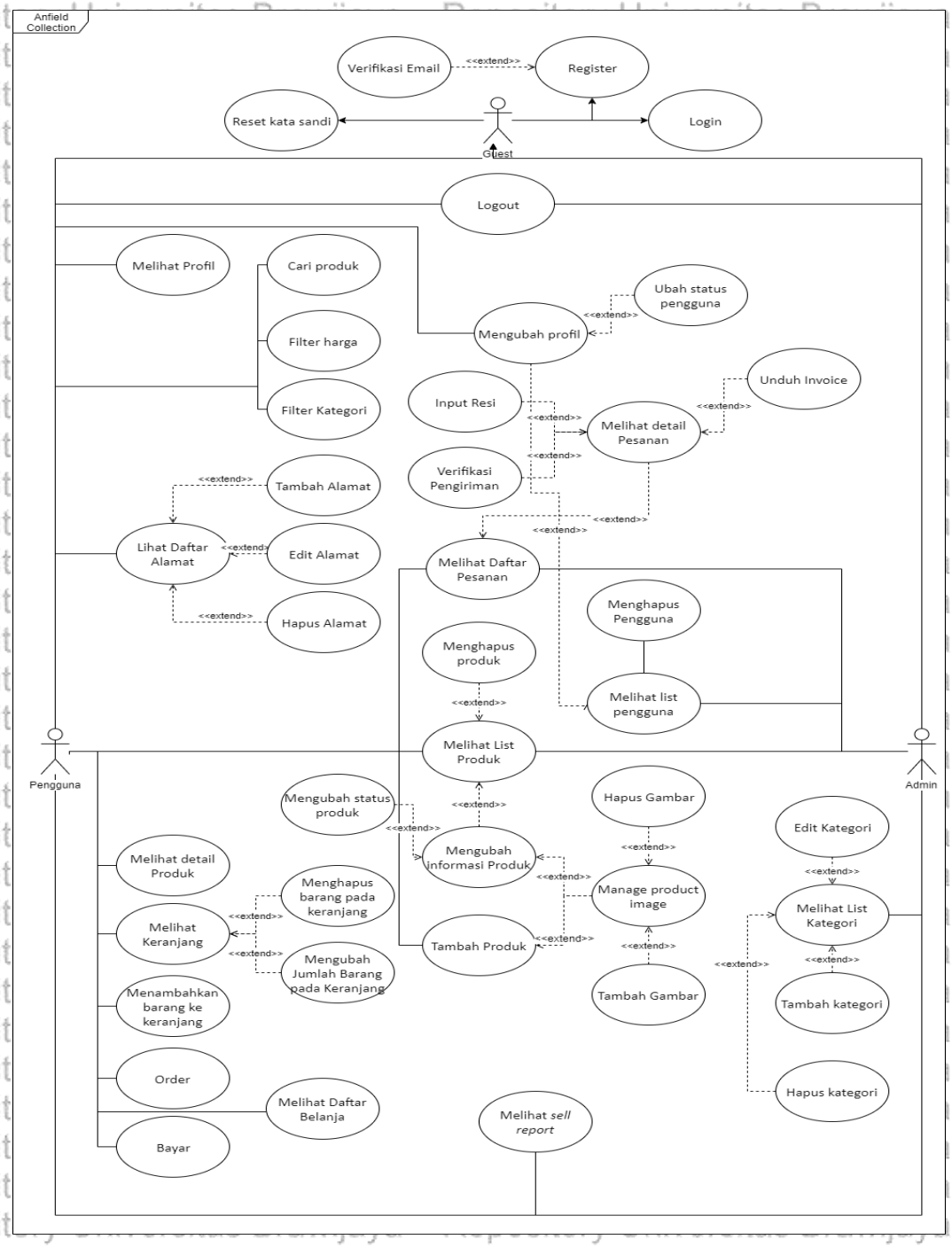
No	Kode Kebutuhan	Nama Kebutuhan	Deskripsi
1	ACM-NF-001	*Compatibility*	Aplikasi dapat digunakan pada *Android* 6.0 *Marshmallow* sampai yang terbaru.

4.7 Pemodelan Kebutuhan

Pada pemodelan kebutuhan, hasil analisis kebutuhan yang telah dilakukan akan ditranslasikan menjadi pemodelan berupa *Use Case* Diagram dan *Use Case Scenario*. *Use Case* Diagram berfungsi untuk menjelaskan berbagai aktivitas yang dapat dilakukan oleh pengguna pada sistem. Penjelasan lebih lanjut mengenai aktivitas pengguna pada *Use Case* Diagram dapat dilihat pada *Use Case Scenario.*

4.7.1 *Use Case Diagram*

DIbawah ini merupakah hasil pemodelan *Use Case Diagram Aplikasi E-Commerce* untuk produk kerajinan yang digambarkan menggunakan konsep pemodelan UML dapat dilihat pada Gambar 4.7.



Gambar 4.2 *Use Case Diagram*

Dapat dijelaskan bahwa terdapat interaksi aktor terhadap sistem. Terdapat 43 *use case* dan 3 aktor yang saling berinteraksi. Masing–masing aktor dapat melakukan interaksi dengan sistem yang sesuai dengan karakteristik yang dijabarkan pada gambar diatas.

4.7.2 Use Case Scenario

Use-Case Scenario pada *section* ini menjelaskan proses dalam perancangan perangkat lunak yang menjelaskan tentang alur seorang aktor terhadap sistem untuk memenuhi aksi yang dijelaskan pada suatu kebutuhan dalam bentuk tabel.

4.7.2.1 *Use-Case Scenario Register*

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Register*:

Tabel 4.6 Skenario *Use-Case Register*

ACM-F-01	
Objective	Aktor harus mampu daftarkan dirinya ke sistem.
Aktor	*Guest*
Pre_Condition	-
Main_Flow	1. Aktor membuka halaman awal sistem lalu memilih menu "*Sign Up*" 2. Pada aplikasi tampil Halaman *Sign-Up* 3. Pada aplikasi tampil *form* register 4. Aktor mengisi form register dengan data yang sesuai dengan format 5. Tombol "*Sign-Up Now*" ditekan oleh aktor 6. Pada aplikasi tampil dialog sukses
Alternative_Flow	6.1 apabila satu atau lebih field pada *register* dikosongkan maka Pada aplikasi tampil pesan "Harap isi bidang ini" pada field yang dikosongkan. 6.2 pesan "*email not valid*" muncul jika format email tidak sesuai
Post_Condition	Pada aplikasi tampil dialog *register* sukses dan aktor terdaftar sebagai member

4.7.2.2 *Use-Case Scenario* Verifikasi email

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Verifikasi *email* :

Tabel 4.7 Skenario *Use-Case Verifikasi email*

ACM-F-02	
Objective	Sistem harus dapat mengirimkan url verifikasi ke email aktor
Aktor	*Guest*
Pre_Condition	Pada layar perangkat tampil halaman login
Main_Flow	1. Aktor mendapatkan email dari sistem pada alamat email yang terdaftar untuk melakukan verifikasi setelah melakukan register 2. Link yang dikirim ditekan oleh aktor
Alternative_Flow	-
Post_Condition	Verifikasi email berhasil, akun menjadi terdaftar & aktif

4.7.2.3 *Use-Case Scenario* Reset kata sandi

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Reset kata sandi:

Tabel 4.8 Skenario *Use-Case* Reset Kata Sandi

ACM-F-03	
Objective	Aktor harus mampu melakukan Reset kata sandi
Aktor	*Guest*
Pre_Condition	Pada layar perangkat tampil halaman login
Main_Flow	1. Pada halaman login, tombol "*Forgotten Password*" Ditekan oleh aktor 2. Pada aplikasi tampil halaman Reset Kata Sandi 3. Kata sandi yang baru ditekan oleh aktor 4. Tombol "Reset kata Sandi" ditekan oleh aktor 5. Dialog sukses tampil pada layar
Alternative_Flow	-

Post_Condition	Pada aplikasi tampil dialog sukses dan kata sandi berhasil di *update*

4.7.2.4 *Use-Case Scenario* Register

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Register*:

Tabel 4.9 Skenario *Use-Case Login*

ACM-F-04	
Objective	Aktor mendapatkan otorisasi sesuai dengan *role*
Aktor	*Guest*
Pre_Condition	User sedang berada di halaman *register*
Main_Flow	1. Pada aplikasi tampil halaman login 2. *email* dan *password* dimasukan oleh aktor 3. Tombol login ditekan 4. Pada aplikasi tampil halaman utama
Alternative_Flow	3.1 Pada aplikasi tampil pesan eror saat login
Post_Condition	Pada aplikasi tampil halaman utama yang berarti aktor sudah masuk ke sistem

4.7.2.5 *Use-Case Scenario Logout*

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Logout*:

Tabel 4.10 Skenario *Use-Case Logout*

ACM-F-05	
Objective	Aktor harus mampu mengeluarkan informasi akun dari aplikasi
Aktor	*Member* & Admin
Pre_Condition	Halaman utama tampil pada layar perangkat pengguna

Main_Flow	1. Menu akun dipilih oleh aktor 2. Pada aplikasi tampil halaman akun 3. Menu Logout dipilih oleh aktor 4. Dialog *prompt*muncul 5. Aktor menekan tombo "yes"
Alternative_Flow	
Post_Condition	Pada aplikasi tampil login yang berarti aktor sudah keluar aplikasi

4.7.2.6 *Use-Case Scenario* Melihat Profil

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat Profil:

Tabel 4.11 Skenario *Use-Case* Melihat Profil

ACM-F-06	
Objective	Aktor harus mampu melihat halaman profil
Aktor	*Member*
Pre_Condition	Aktor berada dalam halaman utama
Main_Flow	1. Menu gambar *account* di pojok kanan bawah dipilih oleh aktor 2. Pada aplikasi tampil halaman profil akun aktor
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil halaman profil

4.7.2.7 *Use-Case Scenario* Mengubah Informasi Member

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Informasi Pengguna untuk admin:

Tabel 4.12 Skenario *Use-Case* Mengubah Informasi *Member* (Admin)

ACM-F-07	
Objective	Aktor harus mampu mengubah infromasi pada akun
Aktor	Admin
Pre_Condition	Halaman daftar *member* tampil pada layar perangkat aktor

Main_Flow	1. Aktor menekan tombol edit profil pada salah satu member 2. Pada aplikasi tampil halaman profil 3. Tombol edit profil yang berada pada halaman profil *member* ditekan oleh aktor 4. Pada aplikasi tampil *form* yang berada pada halaman *edit* profil 5. Informasi baru yang valid dimasukan oleh aktor 6. Tombol save ditekan oleh aktor 7. Pada aplikasi tampil *dialog* sukses mengganti perubahan
Alternative_Flow	-
Post_Condition	DIalog sukses akan tampil dan perubahan informasi akun berhasil diubah

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Informasi Pengguna untuk member:

Tabel 4.13 Skenario *Use-Case* Mengubah Informasi *Member* (Member)

ACM-F-07	
Objective	Aktor harus mampu mengubah infromasi pada akun
Aktor	*Member*
Pre_Condition	Halaman utama tampil pada layar perangkat aktor
Main_Flow	8. Menu gambar account di pojok kanan bawah dipilih oleh aktor 9. Pada aplikasi tampil halaman profil 10. Tombol edit profil yang berada pada halaman profil *member* ditekan oleh aktor 11. Pada aplikasi tampil *form* yang berada pada halaman *edit* profil 12. Informasi baru yang valid dimasukan oleh aktor 13. Tombol save ditekan oleh aktor

	14. Pada aplikasi tampil *dialog* sukses mengganti perubahan
Alternative_Flow	-
Post_Condition	DIalog sukses akan tampil dan perubahan informasi akun berhasil diubah

4.7.2.8 *Use-Case Scenario List Member*

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case List* Pengguna:

Tabel 4.14 Skenario *Use-Case* Melihat *List Member*

ACM-F-08	
Objective	Aktor harus mampu melihat list *member* yang terdaftar
Aktor	Admin
Pre_Condition	Halaman utama admin tampil pada layar perangkat aktor
Main_Flow	1. Menu *list* pengguna dipilih oleh aktor 2. Halaman *list member* tampil pada layar
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil *list* member

4.7.2.9 *Use-Case Scenario* Menghapus Member

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Menghapus* Member:

Tabel 4.15 Skenario *Use-Case* Menghapus Member

ACM-F-09	
Objective	Aktor harus mampu menghapus akun *member* yang terdaftar
Aktor	Admin

Pre_Condition	Halaman *list member* tampil pada layar
Main_Flow	1. Salah satu *member* dipilih oleh aktor 2. Aktor menekan tombol hapus 3. Dialog "apakah anda ingin menghapus akun ini?" tampil pada layar aktor 4. Aktor menekan tombol ya 5. Pada aplikasi tampil list *member* terbaru
Alternative_Flow	3.1 Tombol batal ditekan oleh aktor
Post_Condition	*Member* yang dipilih terhapus & Pada aplikasi tampil list *member* terbaru

4.7.2.10 *Use-Case Scenario* Mengubah Status Member

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Mengubah Status Member:

Tabel 4.16 Skenario *Use-Case* Mengubah Status Member

ACM-F-10	
Objective	Aktor dapat mengubah status *member*
Aktor	*Member* & Admin
Pre_Condition	Aktor dalam halaman edit *member*
Main_Flow	1. Pilihan *member*: Aktif /Nonaktif dipilih oleh aktor 2. Aktor memilih tombol simpan 3. Pada aplikasi tampil informasi *member* terbaru beserta statusnya
Alternative_Flow	-
Post_Condition	Status *member* berhasil disimpan & Pada aplikasi tampil informasi *member* terbaru beserta statusnya

4.7.2.11 *Use-Case Scenario* Tambah Alamat

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Tambah Alamat:

Tabel 4.17 Skenario *Use-Case* Tambah Alamat

ACM-F-11	
Objective	Alamat dapat ditambahkan
Aktor	Member
Pre_Condition	Pada layar tampil halaman menu "Akun Pembeli"
Main_Flow	1. Aktor memilih menu kelola alamat 2. Pada aplikasi tampil list alamat yang terdaftar 3. Tombol tambah alamat ditekan oleh aktor 4. Form tambah alamat tampil 5. Form tambah alamat tampil 6. Tombol simpan ditekan oleh aktor 7. Pada aplikasi tampil daftar alamat yang telah ditambah
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil daftar alamat terbaru

4.7.2.12 *Use-Case Scenario* Lihat Alamat

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Lihat Alamat:

Tabel 4.18 Skenario *Use-Case* Lihat Alamat

ACM-F-12	
Objective	Alamat *member* yang terdaftar dapat dilihat
Aktor	*Member*
Pre_Condition	Aktor berada pada menu "Akun Pembeli"
Main_Flow	1. Menu kelola alamat dipilih oleh aktor 2. Pada aplikasi tampil list alamat yang terdaftar pada member
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil daftar alamat member

4.7.2.13 *Use-Case Scenario* Edit Alamat

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Edit Alamat:

Tabel 4.19 Skenario *Use-Case* Edit Alamat

ACM-F-13	
Objective	Aktor dapat mengubah alamat yang terdaftar
Aktor	Member
Pre_Condition	Halaman menu "Akun Pembeli" tampil
Main_Flow	1. Menu kelola alamat dipilih oleh aktor 2. Pada aplikasi tampil *list* alamat yang terdaftar 3. Aktor menekan tombol edit 4. Form edit alamat tampil 5. Form edit alamat diisi leh aktor 6. Tombol simpan ditekan oleh aktor 7. Pada aplikasi tampil daftar alamat yang sudah *update*
Alternative_Flow	-
Post_Condition	Alamat terubah & Pada aplikasi tampil daftar alamat terbaru

4.7.2.14 *Use-Case Scenario* Hapus Alamat

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Hapus Alamat:

Tabel 4.20 Skenario *Use-Case* Hapus Alamat

ACM-F-14	
Objective	Aktor harus mampu menghapus alamat yang terdaftar
Aktor	*Member*
Pre_Condition	Halaman menu "Akun Pembeli" tampil
Main_Flow	1. Aktor memilih menu kelola alamat 2. Pada aplikasi tampil list alamat yang terdaftar pada *member* 3. Menu hapus alamat ditekan oleh aktor 4. Pada aplikasi tampil dialog "apakah anda ingin menghapus alamat ini?"

	5. Tombol "ya" ditekan oleh aktor 6. Pada aplikasi tampil daftar alamat terbaru
Alternative_Flow	5.1 Aktor menekan tombol batal
Post_Condition	Pada aplikasi tampil *dialog* sukses dan *member* berhasil diblacklist

4.7.2.15 *Use-Case Scenario* Tambah Kategori

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Tambah Kategori:

Tabel 4.21 Skenario *Use-Case* Tambah Kategori

ACM-F-15	
Objective	Aktor harus mampu menambah kategori produk
Aktor	Admin
Pre_Condition	Halaman kelola kategori tampil
Main_Flow	1. Salah satu jenis kategori dipilih aktor 2. Pada aplikasi tampil list kategori yang dipilih 3. Tombol tambah kategori ditekan 4. Field kategori baru tampil 5. Field baru diisi oleh aktor 6. Tombol simpan ditekan oleh aktor 7. Pada aplikasi tampil dialog "kategori berhasil disimpan"
Alternative_Flow	-
Post_Condition	Kategori bertambah & Pada aplikasi tampil dialog "kategori berhasil disimpan"

4.7.2.16 *Use-Case Scenario* Lihat Kategori

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Lihat Kategori:

Tabel 4.22 Skenario *Use-Case* Lihat Kategori

ACM-F-16	
Objective	Seluruh kategori dapat dilihat
Aktor	Admin

Pre_Condition	Aktor berada pada halaman utama
Main_Flow	1. Menu kelola kategori ditekan oleh aktor 2. Pada aplikasi tampil jenis kategori: bahan, proses, jenis 3. Aktor memilih jenis kategori 4. Pada aplikasi tampil list kategori yang dipilih
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil *list* kategori

4.7.2.17 *Use-Case Scenario* Edit Kategori

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Edit Kategori:

Tabel 4.23 Skenario *Use-Case* Edit Kategori

ACM-F-17	
Objective	Aktor dapat mengubah kategori
Aktor	Admin
Pre_Condition	Aktor berada pada halaman kelola kategori
Main_Flow	1. Salah satu jenis kategori dipilih aktor 2. Pada aplikasi tampil list kategori yang dipilih 3. Aktor mengganti list kategori 4. Tombol simpan ditekan oleh aktor 5. Dialog "kategori berhasil disimpan" tampil
Alternative_Flow	-
Post_Condition	Kategori terubah & Pada aplikasi tampil dialog "kategori berhasil disimpan"

4.7.2.18 *Use-Case Scenario* Hapus Kategori

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Hapus Kategori:

Tabel 4.24 Skenario *Use-Case* Hapus Kategori

ACM-F-18	

Objective	Aktor harus mampu menghapus kategori yang dipilih
Aktor	Admin
Pre_Condition	Halaman kelola kategori tampil
Main_Flow	1. Salah satu jenis kategori dipilih oleh aktor 2. Pada aplikasi tampil list kategori yang dipilih 3. Tombol hapus pada kategori yang ingin dihapus ditekan oleh aktor 4. Aplikasi menghilangkan kategori yang dihapus 5. Tombol simpan ditekan oleh aktor 6. Dialog "kategori berhasil disimpan" tampil
Alternative_Flow	-
Post_Condition	Kategori terhapus & Pada aplikasi tampil dialog "kategori berhasil disimpan"

4.7.2.19 *Use-Case Scenario* Tambah Produk

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Tambah Produk:

Tabel 4.25 Skenario *Use-Case* Tambah Produk

ACM-F-19	
Objective	Aktor harus dapat menambah produk beserta kategorinya
Aktor	*Member*
Pre_Condition	Aktor berada dalam halaman "Akun merchant"
Main_Flow	1. Tombol "tambah produk" ditekan 2. Pada aplikasi tampil halaman kelola gambar produk 3. Gambar produk ditambahkan oleh aktor 4. Aktor menekan tombol simpan gambar 5. Pada aplikasi tampil form tambah produk 6. Aktor mengisi form tambah produk 7. Aktor menekan tombol "tambahkan produk" 8. *Dialog* sukses tampil
Alternative_Flow	-
Post_Condition	*Dialog* sukses tampil pada layar dan *member* berhasil menambahkan produk baru

4.7.2.20 *Use-Case Scenario Manage Product Image*

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Manage Product Image*:

Tabel 4.26 Skenario *Use-Case Manage Product Image*

ACM-F-20	
Objective	Aktor dapat mengelola gambar produknya
Aktor	*Member* & Admin
Pre_Condition	Aktor menekan tombol tambah prduk atau tombol edit gambar pada halaman gambar edit produk
Main_Flow	1. Pada aplikasi tampil halaman kelola gambar produk 2. Tombol tambah gambar ditekan oleh aktor 3. Satu gambar dari galeri dipilih aktor 4. Tombol simpan ditekan oleh aktor
Alternative_Flow	2.1 Tombol hapus gambar ditekan oleh aktor 4.1 Aktor kembali menekan tombol tambah gambar
Post_Condition	Pada aplikasi tampil *dialog* sukses dan *member* berhasil menyimpan gambar produk yang telah dikelola

4.7.2.21 *Use-Case Scenario* Tambah Gambar

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Tambah Gambar:

Tabel 4.27 Skenario *Use-Case* Tambah Gambar

ACM-F-21	
Objective	Aktor dapat menambah gambar yang diambil dari galeri ataupun ambil langsung dari kamera
Aktor	*Member* & Admin
Pre_Condition	Aktor berada pada halaman kelola gambar produk
Main_Flow	1. Aktor menekan tombol tambah gambar 2. Aktor memilih gambar dari galeri 3. Pada aplikasi tampil daftar gambar beserta gambar yang baru saja ditambahkan

Alternative_Flow	2.1 Aktor mengambil gambar menggunakan kamera
Post_Condition	*member* berhasil menambahkan gambar

4.7.2.22 *Use-Case Scenario* Hapus Gambar

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Hapus Gambar:

Tabel 4.28 Skenario *Use-Case* Hapus Gambar

ACM-F-22	
Objective	Aktor dapat menghapus gambar yang ditambahkan
Aktor	*Member* & Admin
Pre_Condition	Halaman kelola gambar produk tampil
Main_Flow	1. Gambar yang akan dihapus dipilih oleh aktor 2. Pada aplikasi tampil dialog "Apakah anda ingin menghapus gambar ini ?" 3. Tombol ya ditekan oleh aktor
Alternative_Flow	3.1 Tombol batal ditekan oleh aktor
Post_Condition	foto yang dipilih aktor berhasil terhpus

4.7.2.23 *Use-Case Scenario* Melihat *List* Produk

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat *List* Produk untuk admin:

Tabel 4.29 Skenario *Use-Case* Melihat *List* Produk (Admin)

ACM-F-23	
Objective	Aktor mampu melihat daftar produk yang tersedia
Aktor	Admin
Pre_Condition	Berada pada menu utama
Main_Flow	1. Admin memilih menu kelola produk 2. Pada aplikasi tampil list produk
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil *list* produk seluruh pengguna

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat *List* Produk untuk member:

Tabel 4.30 Skenario *Use-Case* Melihat *List* Produk (Member)

ACM-F-23	
Objective	Aktor mampu melihat daftar produk yang tersedia
Aktor	*Member*
Pre_Condition	*Member* berada dalam halaman "Akun *Merchant*" atau Admin berada pada menu utama
Main_Flow	1. *Member* memilih menu kelola produk 2. Pada aplikasi tampil list produk
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil *list* produk pengguna

4.7.2.24 *Use-Case Scenario* Mengubah Informasi Produk

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Mengubah Informasi Produk:

Tabel 4.31 Skenario *Use-Case* Mengubah Informasi Produk

ACM-F-24	
Objective	Aktor dapat mengubah seluruh / sebagian informasi produk
Aktor	*Member* & Admin
Pre_Condition	Halaman List Produk tampil
Main_Flow	1. Tombol "Edit" dipilih oleh aktor 2. Pada aplikasi tampil form edit produk 3. Form edit produk di isi oleh aktor 4. Tombol simpan ditekan oleh aktor 5. Pada aplikasi tampil *list* produk terbaru
Alternative_Flow	3.1 Tombol kelola gambar ditekan oleh aktor maka aplikasi akan mengarahkan ke halaman kelola gambar produk
Post_Condition	Pada aplikasi tampil *dialog* sukses dan *member* berhasil mengubah informasi produk

4.7.2.25 *Use-Case Scenario* Menghapus produk

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Menghapus produk:

Tabel 4.32 Skenario *Use-Case* Menghapus produk

ACM-F-25	
Objective	Aktor dapat menghapus produk yang dipilih
Aktor	*Member* & Admin
Pre_Condition	Halaman List Produk tampil pada layar
Main_Flow	1. Tombol "Hapus" dipilih oleh aktor 2. Dialog "Apakah anda ingin menghapus produk" tampil pada layar 3. Tombol ya ditekan oleh aktor 4. Pada aplikasi tampil *list* produk terbaru tanpa produk yang sudah dihapus
Alternative_Flow	3.1 Tombol batal ditekan oleh aktor maka aplikasi tidak mengeksekusi hapus produk
Post_Condition	Pada aplikasi tampil *dialog* sukses dan produk berhasil dihapus

4.7.2.26 *Use-Case Scenario* Melihat Detail Produk

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat Detail Produk:

Tabel 4.33 Skenario *Use-Case* Melihat Detail Produk

ACM-F-26	
Objective	Detail informasi dari koleksi produk dapat dilihat oleh aktor
Aktor	*Member*
Pre_Condition	Halaman utama tampil pada layar
Main_Flow	1. Aktor menekan salah satu gambar produk yang berada di halaman utama 2. Pada aplikasi tampil halaman detail produk sesuai dengan gambar yang dipilih
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil halaman detail produk

4.7.2.27 *Use-Case Scenario* Mengubah status Produk

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Mengubah status Produk:

Tabel 4.34 Skenario *Use-Case* Mengubah Status Produk

ACM-F-27	
Objective	Aktor dapat mengubah status produk yang dipilih, sehingga dapat tidak dapat dipesan oleh *member* sampai produk diaktifkan kembali, Jika status tidak aktif
Aktor	*Member* & Admin
Pre_Condition	Halaman edit produk tampil pada layar
Main_Flow	1. Sistem menampikan form edit produk 2. Aktor memilih status produk 3. Aktor menekan tombol simpan
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil *dialog* sukses dan status produk terubah

4.7.2.28 *Use-Case Scenario* Cari Produk

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Cari Produk:

Tabel 4.35 Skenario *Use-Case* Cari Produk

ACM-F-28	
Objective	Aktor harus mampu melakukan pencarian produk berdasarkan nama
Aktor	Member
Pre_Condition	Aktor berada dalam halaman utama
Main_Flow	1. *Icon* Search ditekan oleh aktor 2. Pada aplikasi tampil *form* pencarian produk 3. Nama barang yang yang akan dicari ditekan oleh aktor 4. Pada aplikasi tampil hasil pencarian produk sesuai kata kunci yang diinputkan

Alternative_Flow	
Post_Condition	Pada aplikasi tampil hasil pencarian produk sesuai yang inputkan oleh pengguna

4.7.2.29 *Use-Case Scenario Filter* kategori

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Filter* kategori:

Tabel 4.36 Skenario *Use-Case* Filter kategori

ACM-F-29	
Objective	Aktor harus mampu melakukan pencarian produk berdasarkan kategori
Aktor	Member
Pre_Condition	Halaman utama tampil pada layar
Main_Flow	1. Dari kategori yang tersedia aktor memilih salah satu 2. Pada aplikasi tampil produk berdasarkan kategori
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil hasil pencarian produk sesuai kategori

4.7.2.30 *Use-Case Scenario Filter* Harga

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Filter* Harga:

Tabel 4.37 Skenario *Use-Case Filter* Harga

ACM-F-30	
Objective	Aktor dapat melakukan pencarian produk berdasarkan rentang harga
Aktor	*Member*
Pre_Condition	Halaman utama tampil pada layar
Main_Flow	1. Aktor menekan tombol cari berdasarkan harga 2. Pada aplikasi tampil field harga minimum dan harga maksimum 3. Aktor mengisi field harga maksimum & minimum 4. Pada aplikasi tampil hasil pencarian produk sesuai *range* harga yang sudah di atur oleh member

Alternative_Flow	-
Post_Condition	Pada aplikasi tampil hasil pencarian produk sesuai *range* harga

4.7.2.31 *Use-Case Scenario* Produk ke Keranjang

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Produk ke Keranjang:

Tabel 4.38 Skenario *Use-Case* Menambahkan Produk ke Keranjang

ACM-F-31	
Objective	Aktor dapat menambahkan produk ke keranjang
Aktor	*Member*
Pre_Condition	Halaman utama tampil pada layar
Main_Flow	1. Salah satu gambar produk yang berada di halaman utama ditekan oleh aktor 2. Pada aplikasi tampil halaman detail produk yang dipilih 3. Aktor menekan tombol "*Buy*" untuk menambahkan produk ke keranjang 4. Pada aplikasi tampil halaman Cart dan berisikan *list* produk yang pernah dipilih
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil halaman Cart yang berisikan *list* produk sesuai dengan pilihan aktor

4.7.2.32 *Use-Case Scenario* Melihat isi Keranjang

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat isi Keranjang:

Tabel 4.39 Skenario *Use-Case* Melihat isi Keranjang

ACM-F-32	
Objective	Aktor dapat melihat item yang ia masukkan ke keranjang
Aktor	Member
Pre_Condition	Aktor berada di halaman utama

Main_Flow	1. Aktor menekan *icon* keranjang 2. Pada aplikasi tampil halaman *Cart* dan berisikan *list* sesuai produk yang dipilih aktor
Alternative_Flow	2.1 Pada aplikasi tampil dialog "keranjang kosong, silahkan pilih barang yang ingin anda beli"
Post_Condition	Pada aplikasi tampil halaman Cart yang berisikan *list* produk

4.7.2.33 *Use-Case Scenario* Mengubah Jumlah Barang pada Keranjang

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Mengubah Jumlah Barang pada Keranjang:

Tabel 4.40 Skenario *Use-Case* Mengubah Jumlah Barang pada Keranjang

ACM-F-33	
Objective	Aktor harus mampu menambah / mengurangi kuantitas per barang yang ada pada keranjang
Aktor	Member
Pre_Condition	Aktor berada dalam halaman *Cart*
Main_Flow	1. Aktor menekan tombol + untuk menambah produk dan – untuk mengurangi produk pada salah satu produk 2. Pada aplikasi tampil hasil perhitungan berdasarkan harga & kuantitas barang
Alternative_Flow	-
Post_Condition	Jumlah barang di halaman *Cart* berhasil diubah sesuai keinginan actor

4.7.2.34 *Use-Case Scenario* Menghapus Barang pada Keranjang

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Menghapus Barang pada Keranjang:

Tabel 4.41 Skenario *Use-Case* Menghapus Barang pada Keranjang

ACM-F-34	
Objective	Barang pada keranjang dihapus oleh aktor
Aktor	*Member*
Pre_Condition	Aktor berada dalam halaman *Cart*

Main_Flow	1. Aktor menekan icon trash pada salah satu barang 2. Pada aplikasi tampil dialog "apakah anda ingin menhapus produk ini dari keranjang?" 3. Tombol ya ditekan oleh aktor 4. Aplikasi menghilangkan barang yang dihapus dari keranjang & merubah total harga
Alternative_Flow	3.1 Tombol batal ditekan oleh aktor maka sistem tidak mengeksekusi hapus barang pada keranjang
Post_Condition	Jumlah barang di halaman *Cart* berhasil diubah sesuai keinginan aktor & total harga berubah

4.7.2.35 *Use-Case Scenario Order*

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Order*:

Tabel 4.42 Skenario *Use-Case Order*

ACM-F-35	
Objective	Aktor harus mampu memesan barang
Aktor	*Member*
Pre_Condition	Halaman *Cart* tampil
Main_Flow	1. Tombol "*Checkout*" yang ada bagian bawah ditekan oleh aktor 2. Pada aplikasi tampil form order 3. Form order diisi oleh aktor 4. Aktor menekan tombol simpan 5. Sistem membuat pesanan & tagihan 6. Pada aplikasi tampil daftar belanja *Member*
Alternative_Flow	-
Post_Condition	*Member* berhasil melakukan pemesanan untuk selanjutnya melakukan pembayaran

4.7.2.36 *Use-Case Scenario payment*

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case payment*:

Tabel 4.43 Skenario *Use-Case payment*

ACM-F-36	
Objective	*Member* dapat melakukan pembayaran dari pesanan yang suda berhasil dibuat
Aktor	*Member*
Pre_Condition	Halaman detail pesanan yang dipilih tampil
Main_Flow	1. Tombol bayar ditekan oleh aktor 2. Sistem mengalihkan *member* ke web pembayaran 3. Aktor melakukan pembayaran diluar aplikasi mobile 4. Sistem mengubah status pesanan menjadi sudah dibayar.
Alternative_Flow	-
Post_Condition	*Member* berhasil melakukan pembayaran

4.7.2.37 *Use-Case Scenario* Melihat Daftar Belanja

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat Daftar Belanja:

Tabel 4.44 Skenario *Use-Case* Melihat Daftar Belanja

ACM-F-37	
Objective	Daftar belanja dapat dilihat oleh aktor
Aktor	Member
Pre_Condition	Halaman menu "Akun pembeli" tampil
Main_Flow	1. Aktor memilih menu daftar belanja 2. Pada aplikasi tampil daftar belanja
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil riwayat pemesanan *member* dalam bentuk *list*

4.7.2.38 *Use-Case Scenario* Melihat Daftar Pesanan

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat Daftar Pesanan untuk aktor admin:

Tabel 4.45 Skenario *Use-Case* Melihat Daftar Pesanan (Admin)

ACM-F-38	
Objective	Aktor harus mampu melihat daftar pesanan yang masuk
Aktor	Admin
Pre_Condition	Halaman "*Sell Report*" tampil pada layar
Main_Flow	1. Tanggal yang terdapat order dipilih 2. Pada aplikasi tampil Daftar Pesanan dalam bentuk *list*
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil *list order* seluruh *member* dalam bentuk *list* berdasarkan tanggal yang dipilih

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat Daftar Pesanan untuk aktor member:

Tabel 4.46 Skenario *Use-Case* Melihat Daftar Pesanan (Member)

ACM-F-38	
Objective	Aktor harus mampu melihat daftar pesanan
Aktor	Member
Pre_Condition	Halaman "Akun *Merchant*" tampil pada layar
Main_Flow	1. Menu Daftar Pesanan dipilih oleh aktor 2. Pada aplikasi tampil Daftar Pesanan dalam bentuk *list*
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil *list order member* dalam bentuk *list*

4.7.2.39 *Use-Case Scenario* Melihat Detail Pesanan

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat Detail Pesanan:

Tabel 4.47 Skenario *Use-Case* Melihat Detail Pesanan

ACM-F-39	
Objective	Aktor harus mampu melihat detail pesanan yang dipilih
Aktor	*Member* & Admin
Pre_Condition	Halaman daftar pesanan tampil

Main_Flow	1. Salah satu pesanan dipilih oleh aktor 2. Pada aplikasi tampil halaman detail pemesanan sesuai yang dipilih member
Alternative_Flow	-
Post_Condition	Pada aplikasi tampil halaman detail pemesanan sesuai yang pemesanan member

4.7.2.40 *Use-Case Scenario* Input Resi

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Input Resi*:

Tabel 4.48 Skenario *Use-Case* Input Resi

ACM-F-40	
Objective	Aktor dapat memasukan resi dari pesanan yang masuk & sudah berhasil
Aktor	*Member*
Pre_Condition	Aktor berada dalam halaman detail pemesanan, pesanan sudah dibayar oleh pembeli
Main_Flow	1. Pada aplikasi tampil tombol input resi 2. Tombol *input* resi di tekan oleh aktor 3. Aplikasi menampikan *modal* yang berisikan *field* no resi 4. Aktor mengisikan no resi 5. Tombol simpan ditekan oleh aktor 6. Aplikasi mengarahkan ke halaman daftar pesanan dengan status yang baru
Alternative_Flow	-
Post_Condition	Status pesanan menjadi "sedang dikirim" & terdapat no resi yang baru saja di isi

4.7.2.41 *Use-Case Scenario* Verifikasi Pengiriman

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Verifikasi Pengiriman:

Tabel 4.49 Skenario *Use-Case* Verifikasi Pengiriman

ACM-F-41	

Objective	Aktor dapat mengubah status pengiriman
Aktor	Member
Pre_Condition	Aktor berada dalam halaman detail pesanan, status pesanan "sedang dikirim"
Main_Flow	1. Pada aplikasi tampil tombol terima barang 2. Tombol terima barang ditekan oleh aktor 3. Pada aplikasi tampil detail pesanan yang baru
Alternative_Flow	-
Post_Condition	Status order berubah menjadi "sudah diterima"

4.7.2.42 *Use-Case Scenario* Unduh Invoice

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case Unduh Invoice*:

Tabel 4.50 Skenario *Use-Case* Unduh *Invoice*

ACM-F-42	
Objective	Aktor dapat mengunduh invoice dari pesanan yang dipilih
Aktor	Member
Pre_Condition	Aktor berada dalam halaman detail pesanan
Main_Flow	1. *Text* "lihat" pada baris invoice ditekan oleh aktor 2. Aktor diarahkan ke halaman download invoice 3. Invoice berhasil di simpan pada device member
Alternative_Flow	-
Post_Condition	Invoice berhasil di simpan pada penyimpanan member

4.7.2.43 *Use-Case Scenario* Melihat *Sell Report*

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat *Sell Report* untuk admin:

Tabel 4.51 Skenario *Use-Case* Melihat *Sell Report* (Admin)

ACM-F-43	
Objective	Laporan penjualan dapat dilihat oleh aktor sesuai dengan *range* waktu yang dipilih aktor
Aktor	Admin

Pre_Condition	Aktor berada dalam halaman utama
Main_Flow	1. Menu *sell report* dipilih oleh aktor 2. Pada aplikasi tampil halaman *sell report*
Alternative_Flow	1.1 Aktor mengubah rentang laporan penjualan yang ingin dilihat
Post_Condition	Pada aplikasi tampil laporan seluruh penjualan sesuai range waktu yang sudah ditentukan

Dibawah ini merupakan skenario penggunaan dari fitur yang terdapat pada *Use-Case* Melihat *Sell Report untuk* member:

Tabel 4.52 Skenario *Use-Case* Melihat *Sell Report (Member)*

ACM-F-43	
Objective	Laporan penjualan dapat dilihat oleh aktor sesuai dengan *range* waktu yang dipilih aktor
Aktor	*Member*
Pre_Condition	Aktor berada dalam halaman "Akun Merchant"
Main_Flow	1. Menu *sell report* dipilih oleh aktor 2. Pada aplikasi tampil halaman *sell report*
Alternative_Flow	1.1 Aktor mengubah rentang laporan penjualan yang ingin dilihat
Post_Condition	Pada aplikasi tampil laporan penjual sesuai range waktu yang sudah ditentukan

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Bab berikut membahas tentang proses perancangan dan implementasi pada pengembangan aplikasi *E-Commerce*. Hasil dari tahapan perancangan inilah yang menjadi dasar untuk proses implementasi pembangunan sistem.

5.1 Perancangan

Setelah proses analisis kebutuhan dilakukan maka hasil dari tahap tersebut akan dijadikan acuan pada tahap selanjuatnya yaitu perancangan. Pada proses perancangan akan dilakukan secara bertahap dengan tujuh buah tahapan yang masing-masing adalah perancangan arsitektur, perancangan *sequence diagram* dengan 5 sampel, perancangan Activity Diagram dengan 5 fitur utama, perancangan *class diagram*, perancangan basis data, perancangan algoritme dengan 3 sampel dan terakhir perancangan antarmuka.

5.1.1 Perancangan Arsitektur

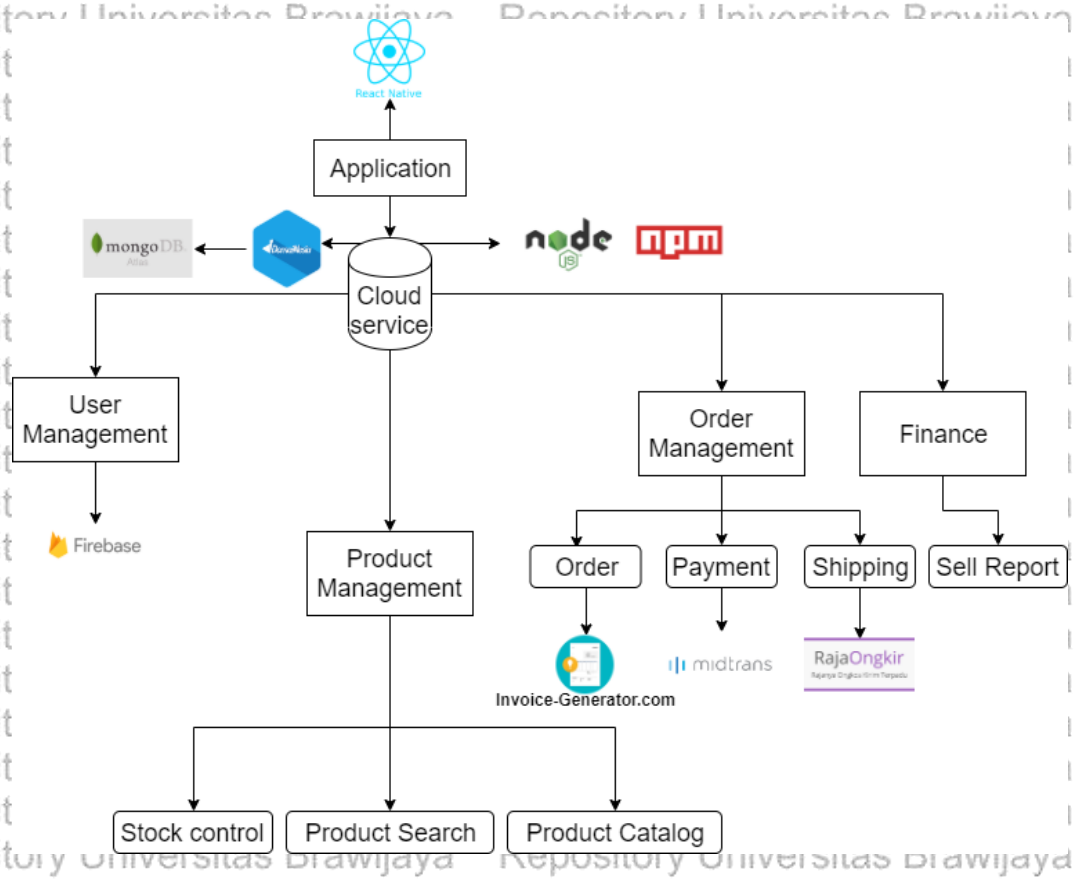
Aplikasi *E-Commerce* ini akan di kembangkan menggunakan *framework React Native*. *Framework* ini mempunyai sebuah arsitektur yang sudah banyak digunakan untuk mengembangkan aplikasi-aplikasi besar. *Service* yang dibuat akan dijalankan di *hosting domainesia* dengan menggunakan spesifikasi yang mumpuni untuk kelas *service* yang ringan pada aplikasi ini. *MongoDB* atlas digunakan sebagai *cloud database mongodb* yang di hubungkan dengan *service* yang dijalankan di *hosting domainesia.*

Aplikasi ini mjenggunakan beberapa API diantaranya adalah *Firebase Authentication, Firebase Storage, Invoice Generator, Midtrans*, RajaOngkir. API tersebut memiliki penggunaan untuk tugas-tugas yang berbeda, yaitu sebagai berikut:

1. *Firebase Authentication* digunakan untuk aksi pengguna seperti *register* & *login*. Di implementasikan pada aplikasi *mobile* & *service* pada *hosting*.
2. *Firebase Storage* digunakan untuk menyimpan gambar produk, gambar profil *member* & dokumen *invoice order*.
3. *Invoice Generator* digunakan untuk membuat *invoice* dari *order* yang sudah dibayarkan. Dokumen *invoice* memiliki format pdf.
4. *Midtrans* digunakan untuk membuat pembayaran dari *order* yang dibuat oleh member.
5. RajaOngkir digunakan untuk memnghitung ongkos kirim dari jenis kurir & *service* yang dipilih.

Pada tahap *deployment*, service akan di *install* pada *hosting domainesia*. *Service* yang diberikan cukup untuk menangani proses yang di eksekusi oleh kode program dengan perangkat lunak *nodeJS*. Agar beban *service* terbagi maka

transaksi data akan di pisah, oleh karena itu aliran data akan di tangani oleh *service* yang lain, Basis data akan di deploy *pada MongoDB atlas*.



Gambar 5.1 Perancangan Diagram Arsitektur

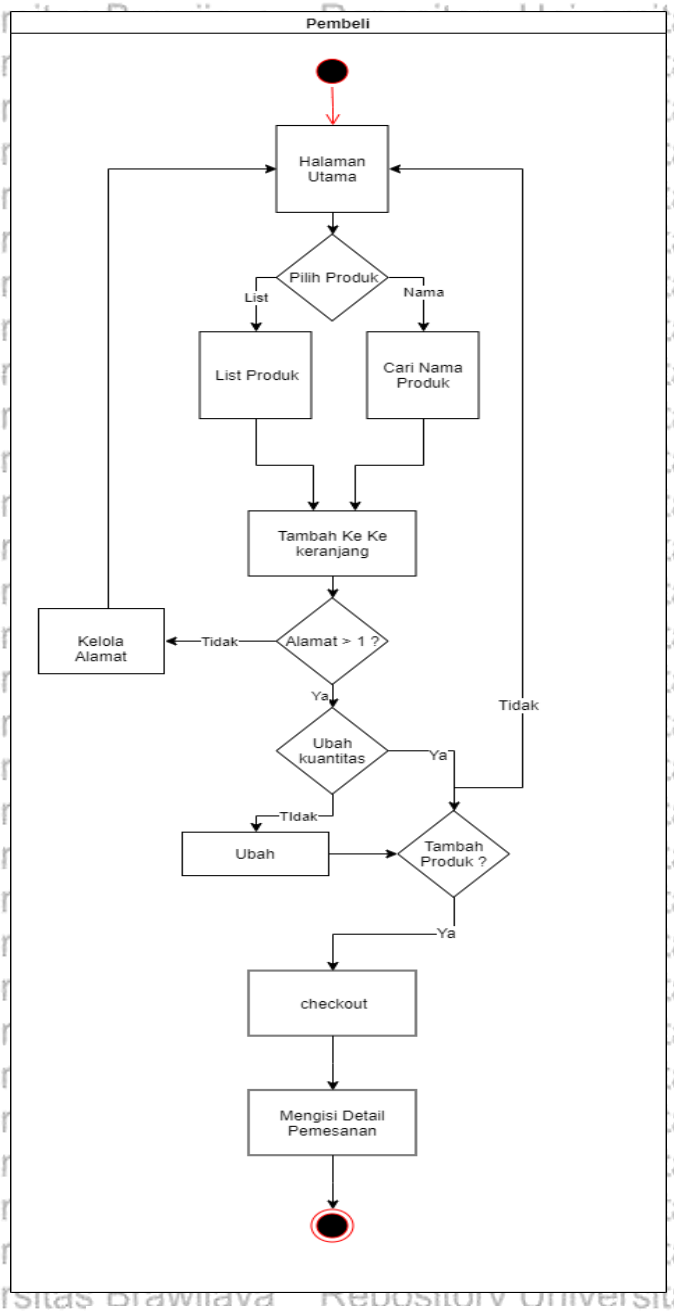
5.1.2 Perancangan *Activity Diagram*

Activity diagram dibuat untuk menggambarkan alur aktivitas yang dilakukan oleh *user* dan aplikasi secara runut. Aplikasi *e-commerce* ini memiliki 5 fitur utama yaitu *Order*, *Product Management*, *Merchant Management*, *Register* dan *Login*.

5.1.2.1 Activity Diagram Order

Proses order dilakukan oleh pembeli dimulai dengan memilih barang langsung dari halaman utama dan dapat juga dilakukan dengan cara mencari berdasarkan kategori, harga atau nama barang. Barang yang di pesan dapat lebih dari satu toko. Setelah itu barang akan masuk ke keranjang, pada kondisi tersebut kuantitas tiap barang dapat diubah selama stok masih tersedia. Ketika kuantitas sudah sesuai dengan jumlah barang yang di inginkan maka pembeli dapat melakukan proses *checkout*. Pada saat *checkout* dilakukan pembeli wajib memilih alamat pengiriman dari alamat yang sudah di daftarkan oleh pembeli. Lalu pembeli juga dapat memilih kurir yang tersedia. Jika pilihan tidak dibuat maka pilihan *default* akan dipilih. Pembeli akan diperlihatkan detail pesanan yang baru saja dibuat. Setelah seluruh proses diatas telah dilakukan maka status pemesanan akan

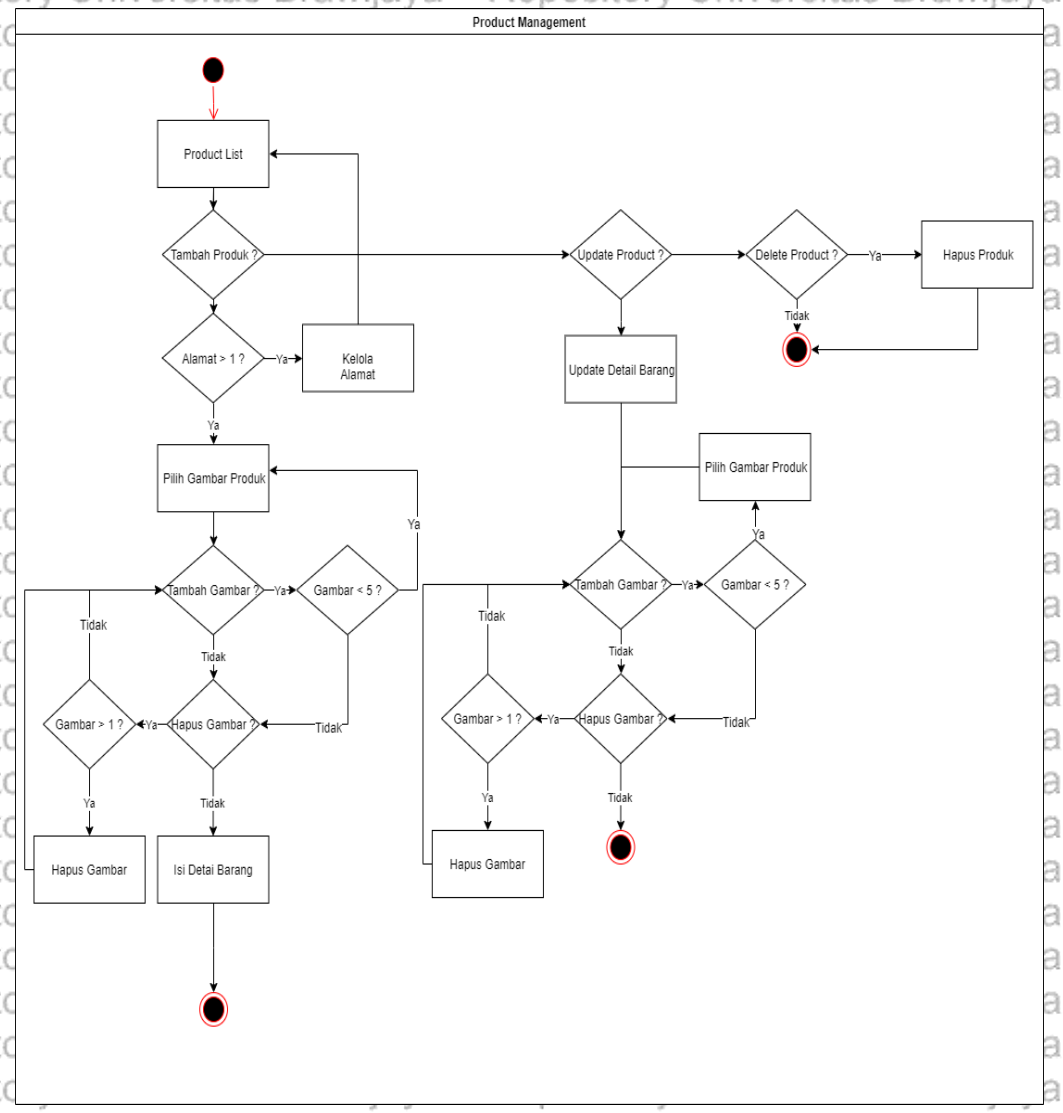
menjadi “Belum dibayar” dan pembeli dapat melakukan pembayaran. Proses pembayaran akan dijelaskan pada Sub Bab yang lainnya. Gambaran alur proses *order* dapat juga dilihat pada Gambar dibawah.



Gambar 5.2 *Activity diagram order*

5.1.2.2 Activity Diagram Product Management

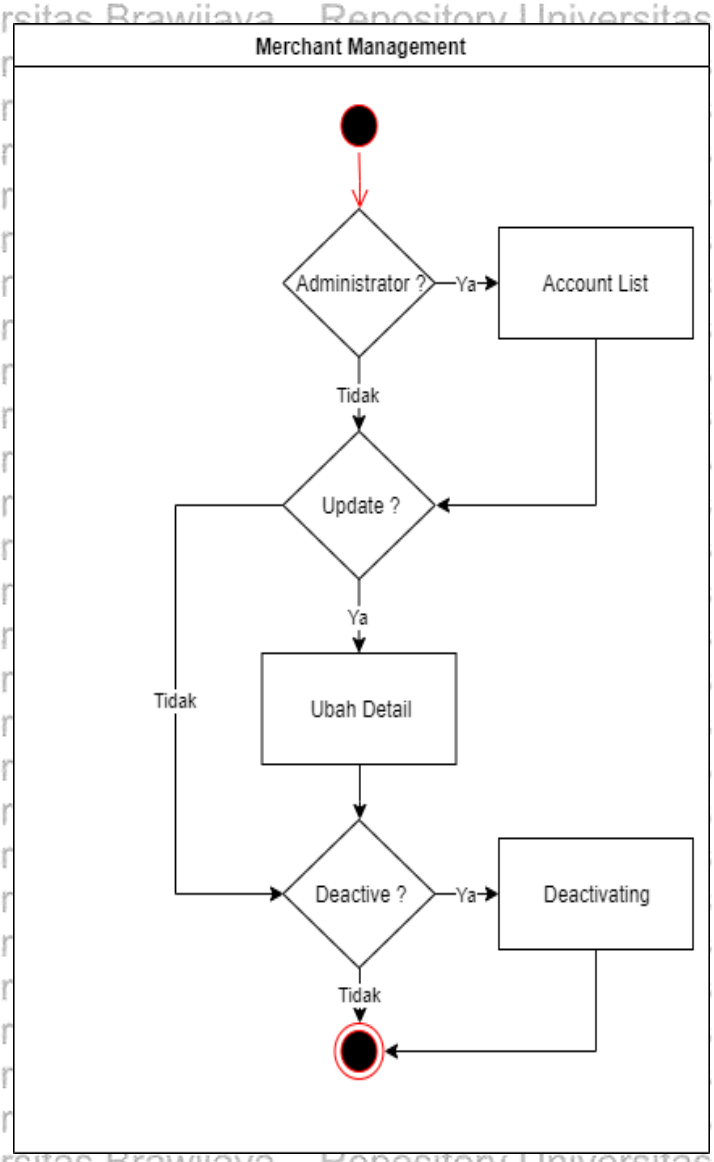
Product management terdiri dari 3 pilihan yaitu Tambah , Update dan hapus produk. Pada saat proses Tambah dan Update *member* dapat mengatur gambar yaitu proses menghapus atau menambah gambar. Gambar yang boleh ada dalam produk yaitu berjumlah lebih dari satu dan kurang dari lima. Detail produk yang dapat diatur adalah harga, nama , gambar, kuantitas, deskripsi, status, kategori. Gambaran alur proses *order* dapat juga dilihat pada Gambar dibawah.



Gambar 5.3 *Activity diagram product management*

5.1.2.3 Activity Diagram Merchant Management

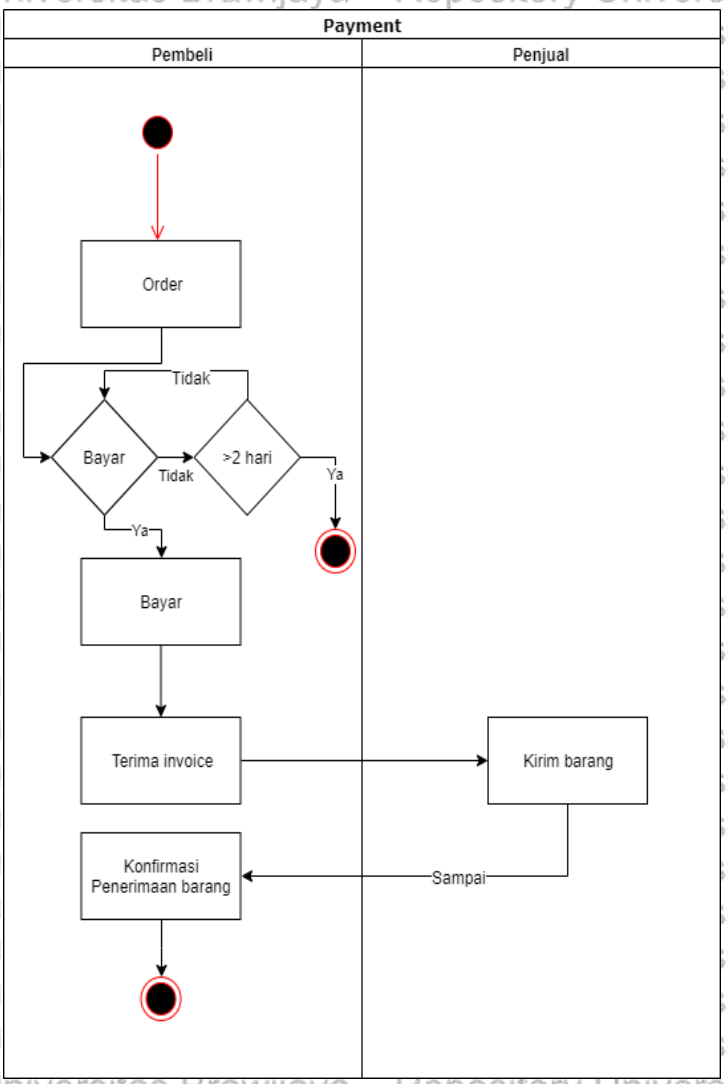
Proses berikut dapat dilakuan oleh pemilik akun tersebut ataupun administrator, detail yang dapat diubah adalah nama, tanggal lahir dan status. *Previlage* tambahan yang dapat dilakukan oleh administrator adalah tipe akun tersebut dapat menghapus akun *member* yang dipilih. Perbedaan dari menghapus dan deactive adalah akun yang sudah dihapus tidak dapat kembali dan status *member* yang tidak aktif dapat kembali. JIka status *member* tidak aktif atau terhapus *member* tidak dapat login dan produk tidak akan muncul di halaman apapun. Gambaran alur proses *order* dapat juga dilihat pada Gambar dibawah.



Gambar 5.4 *Activity diagram merchant management*

5.1.2.4 Activity Diagram Payment

Proses *payment* atau pembayaran dapat dilakukan ketika order sudah berhasil dibuat. Tagihan akan muncul pada jenis pembayaran yang telah dipilih. Total tagihan yang muncul jumlah merupakan total dari harga barang + ongkos kirim dari tiap *merchant* ke alamat penerima. Pembayaran dapat dilakukan 2x24 jam sejak tagihan dibuat. Setelah pembayaran berhasil dilakukan maka status order akan berubah menjadi "Sudah dibayar & Belum dikirim". Setelah itu penjual dapat mengisi no resi maka status order akan berubah menjadi "Sudah dibayar & Sedang dikirim". Ketika barang sudah sampai maka pembeli dapat memverifikasi penerimaan barang, saat proses tersebut dilakukan maka status order akan berubah menjadi "Sudah dibayar & Sudah sampai". Gambaran alur proses *order* dapat juga dilihat pada Gambar dibawah.



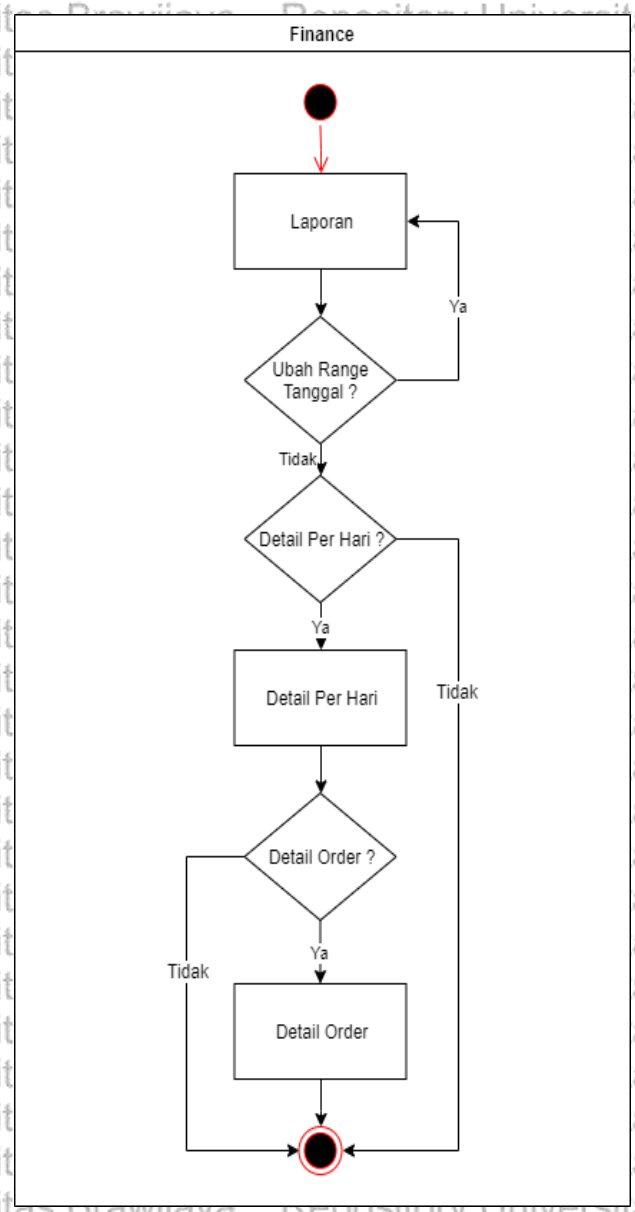
Gambar 5.5 *Activity diagram payment*

5.1.2.5 Activity Diagram Finance

Proses *finance* merupakan *sell report* atau laporan penjualan. Yang dapat dilakukan oleh administrator atau penjual itu sendiri. Laporan penjualan secara *default* akan menampilkan dari satu bulan sebelumnya. Range tanggal juga dapat diubah dengan mengubah tanggal awal dan tanggal akhir. Detail laporan yang ditampilkan adalah

1. Total penjualan selama range yang telah diatur.
2. Total penjualan perhari.
3. Detail order perhari yang dipilih.

Gambaran alur proses *order* dapat juga dilihat pada Gambar dibawah.



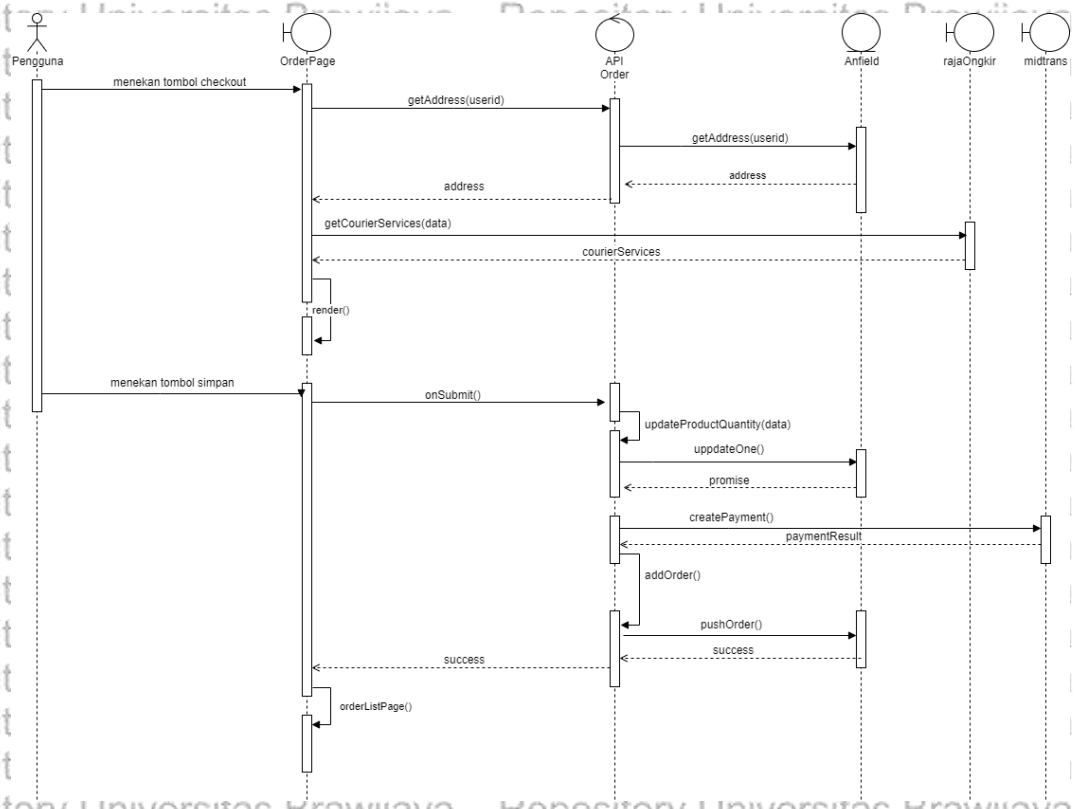
Gambar 5.6 *Use Case Diagram finance*

5.1.3 Perancangan *Sequence Diagram*

Sequence Diagram menggambarkan interaksi antar objek yang diatur berdasarkan waktu untuk memenuhi kebutuhan yang sudah di definisikan. Pembuatan *sequence diagram* didasarkan pada *use case diagram* dan *use case scenario*. Dari 43 kebutuhan fungsional yang definisikan hanya 5 yang dijadikan sampel yaitu *Order*, Tambah Produk, *Payment*, *Register* dan *Login*.

5.1.3.1 Perancangan *Sequence Diagram Order*

Pada Gambar 5.7 terdapat hasil perancangan sequence untuk *order* barang. API yang terlibat dalam sequence ini adalah rajaOngkir dan *MidTrans*, kedua API tersebut direpresentasikan dengan gambar "*entity*". *Order* hanya membutuhkan single class pada mobile karena terdapat banyak API yang terlibat dalam melakukan proses *Order*. Proses order dimulai dengan *member* memilih alamat dan kurir, setelah itu kelas *OrderPage* akan memanggil API order yang dimana API tersebut mengendalikan fungsi dan memanggil API dari raja ongkir dan *midtrans* sebagai mitra dalam pembayaran pada proses *Order*.

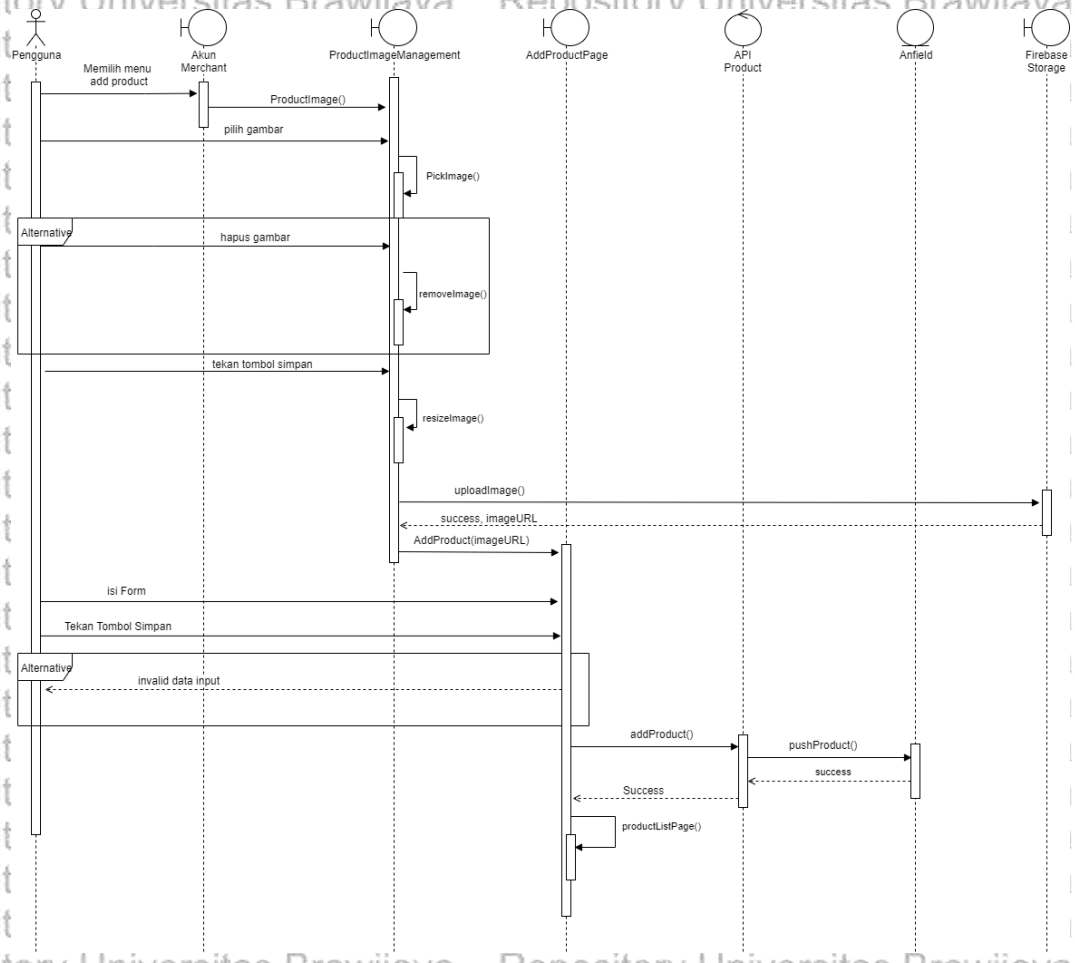


Gambar 5.7 *Sequence Diagram Order*

5.1.3.2 Perancangan *Sequence Diagram* Tambah Produk

Pada Gambar 5.8 terdapat hasil perancangan *sequence* untuk *order* barang. API yang terlibat dalam *sequence* ini adalah *firebase storage*. API tersebut

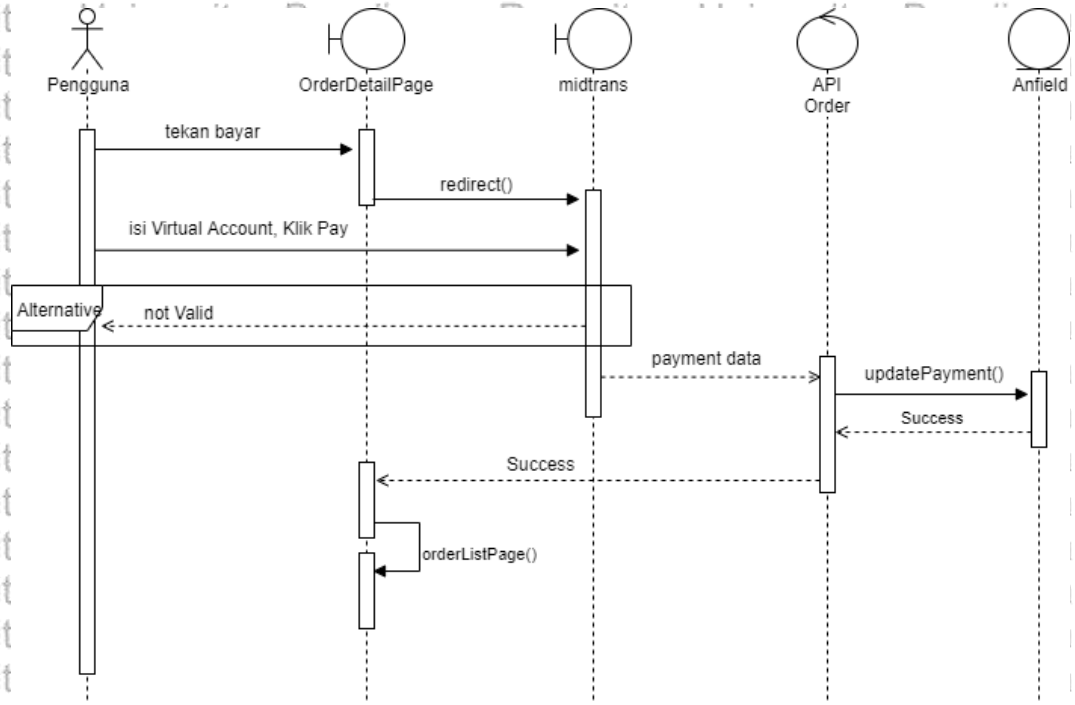
direpresentasikan dengan gambar "*entity*". Pada fitur tambah produk dilakukan melalui 3 halaman dan 3 kelas, yaitu halaman *AkunMerchant, productimagemanagement,* dan *addProductPage*. Pertama aktor memilih menu tambah produk pada halaman akun *merchant*. Kemudian aktor diarahkan menuju halaman *product image management*, disana aktor dapat menambah atau menghapus gambar produk. Setelah aktor mengelola gambar produk, aktor menekan tombol simpan, saat aktor menekan tombol simpan maka fungsi untuk *upload* gambar akan terkesekusi. Gambar produk yang ditambahkan akan di *upload* ke *firebase storage*, kembalian yang didapatkan dari *api firebase storage* adalah *object*. *Return* dari *api firebase storage* akan disimpan *value url* gambar pada *object properties* yang ada di kelas *addProductPage*. Setelah tahap tersebut aktor berada di halaman *AddProductPage*, aktor di ekspektasikan akan mengisi seluruh *field* yang ada dihalaman tersebut, setelah selesai mengisi aktor menekan tombol selesai. Tombol tersebut akan memicu fungsi untuk post data ke API order yang akan menyimpan data yang di isi *member* dan url gambar yang di*upload* ke firebase storage, lalu data akan disimpan pada db anfield. Jika seluruh tahap telah dilakukan sesuai ekspektasi maka produk berhasil ditambahkan dan aktor akan di arahkan ke halaman list produk.



Gambar 5.8 *Sequence Diagram* Tambah Produk

5.1.3.3 Perancangan *Sequence Diagram Payment*

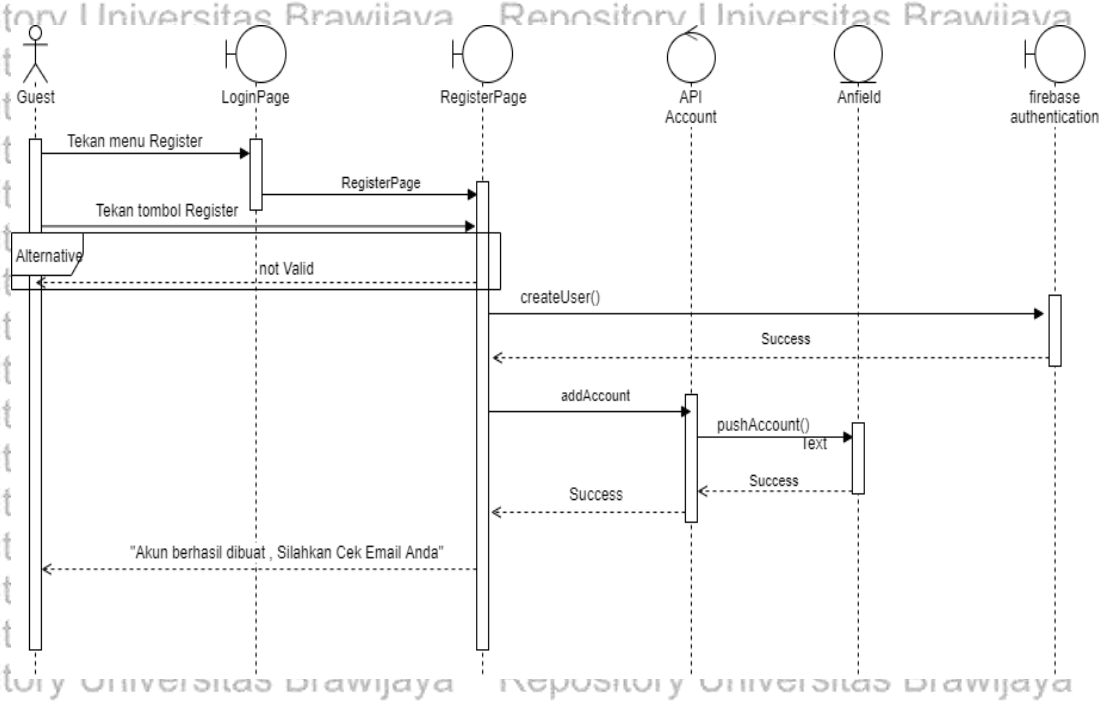
Fitur pembayaran yang dirancang dengan memperlakukan *midtrans payment gateway* sebagai mitra utama dalam menangani pembayaran yang dilakukan pada aplikasi. Pembayaran dapat dimulai dari detail order yang ditampilkan pada halaman *OrderDetailPage*. Aktor menekan tombol bayar yang ada pada halaman tersebut lalu aktor akan diarahkan pada halaman *website* simulasi pembayaran *midtrans*. Ketika aktor berhasil melakukan pembayaran lalu midtrans akan memanggil (*callback*) fungsi yang ada pada *api order*, yang dimana fungsi tersebut akan mengubah data yang disimpan pada *db anfield*. Setelah data di ubah API order akan mengembalikan status terbaru *order*.



Gambar 5.9 *Sequence Diagram Payment*

5.1.3.4 Perancangan *Sequence Diagram Register*

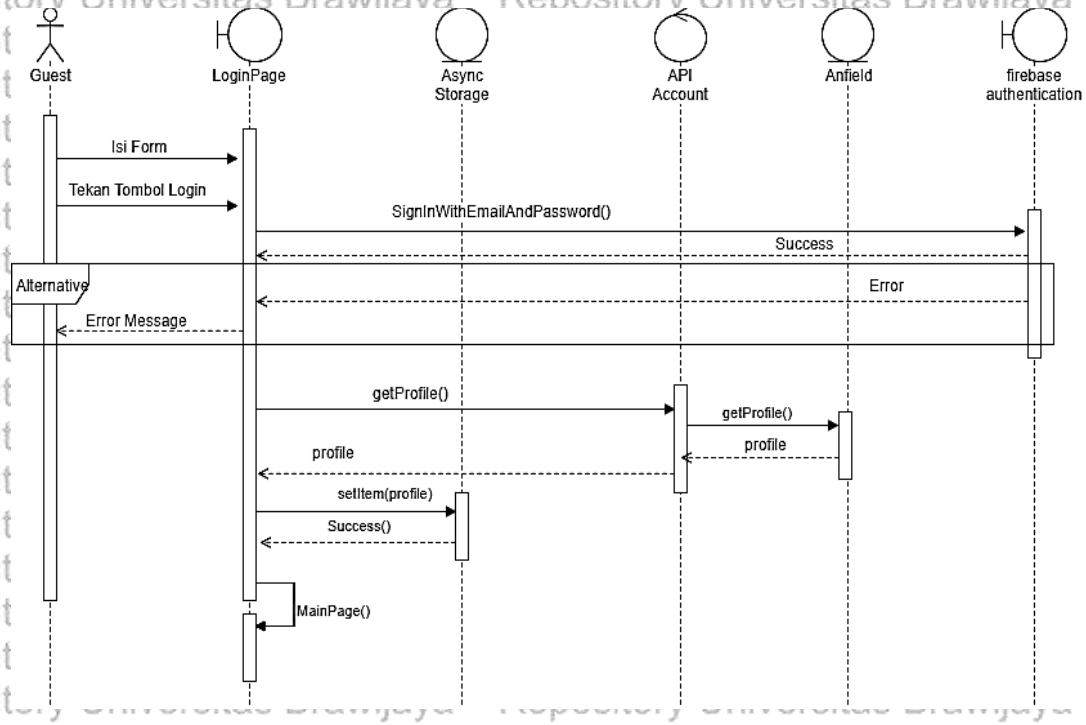
Pada Gambar 5.10 terdapat hasil perancangan *sequence* untuk *Register.* API yang terlibat dalam *sequence* ini adalah *Firebase Authentication* karena dapat melakukan banyak fungsi untuk manajemen akun. *Register* dimulai pada saat *member* berada di halaman pertama yaitu *login*. Di halaman *login* terdapat *hyperlink* untuk mengarahkan aktor ke halaman *register* (*RegisterPage*). Setelah aktor berda di dalam halaman *register* maka aktor diekspektasikan akan mengisi seluruh *field* yang ada pada form register, lalu user akan menekan tombol register. Tombol register akan memicu fungsi *CreateUser* yang ada pada *library Firebase Authentication javascript* untuk mengirim data yang telah di isi user ke api *firebase authentication*. Jika pendaftaran akun di firebase berhasil, maka pesan berhasil akan di dapatkan. Lalu fungsi *PushAccount* akan dipanggil, fungsi tersebut untuk memasukan informasi akun ke dalam db Anfield. Jika semua sequence telah dilakukan, maka aktor harus memverifikasi email yang telah didaftarkan terlebih dahulu.



Gambar 5.10 *Sequence Diagram Register*

5.1.3.5 Perancangan *Sequence Diagram Login*

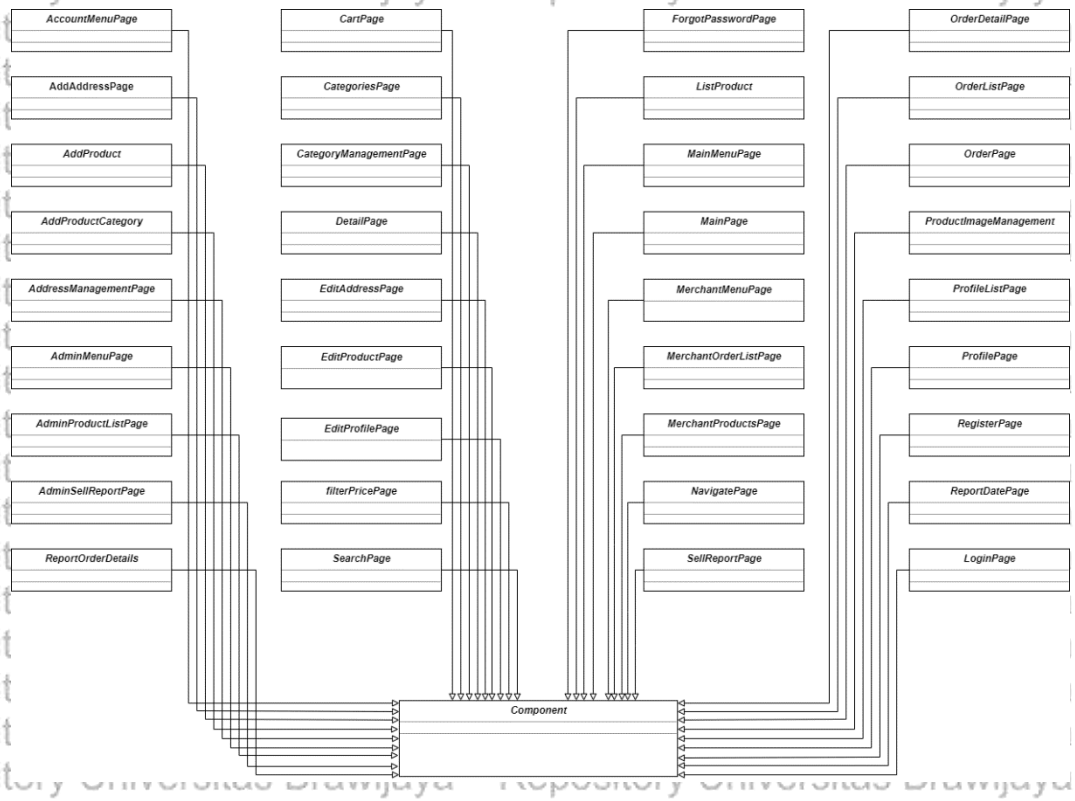
Pada Gambar 5.11 terdapat hasil perancangan sequence untuk *login*, API yang digunakan adalah *Firebase Authentication*. *Login* dimulai pada saat aktor berada di halaman *login*, lalu aktor di ekspektasikan akan mengisi seluruh *field* di *form login*. Setelah aktor mengisi form aktor menekan tombol *login*, Tombol tersebut akan mengeksekusi fungsi *SigninWithEmailAndPassord* yang di dapat dari *library firebase authentication* untuk *JavaScript*. Fungsi tersebut memanggil *Web Service firebase authentication* untuk *login*. Jika *login* gagal maka pesan error akan didapatkan, sebaliknya jika login berhasil maka pesan untuk *success* akan diberikan. Apabila sukses maka fungsi *getProfile*() akan eksekusi fungsi tersebut memanggil fungsi yang ada di API Account untuk mengambil informasi *member* yang ada di db anfield. Saat informasi berhasil di dapatkan maka fungsi untuk menyimpan data di local Storage di panggil , setelah itu aktor akan di arahkan ke *MainPage*.



Gambar 5.11 *Sequence Diagram Login*

5.1.4 Perancangan *Class Diagram*

Setelah dilakukan pemodelan 5 *sequence diagram* , maka tahap berikutnya yang dilakukan adalah pemodelan *class diagram*. Hasil dari pemodelan *sequence diagram* akan digunakan untuk melakukan pemodelan *class diagram*. *Class diagram* merupakan bagian dari *Object Oriented Design* (OOD), merupakan tahapan sebelum *Object Oriented Programing* (OOP). Pada *react native* terdapat kelas utama untuk membuat halaman yaitu kelas *Component* yang wajib di *extend.* Di kelas Component terdapat fungsi-fungsi penting untuk di gunakan seperti, fungsi *render*() menampilkan halaman pada layar, fungsi *componentDidMount*() yang terpanggil saat kelas siap di render, *setState*() untuk memberikan nilai baru ke *variable state*. Terdapat juga *variable state* yang berguna untuk menyimpan nilai yang dinamis.



Gambar 5.12 Perancangan *Class Diagram*

Dari Gambar diatas, terdapat 36 kelas yang terhubung langsung ke kelas Component. Kelas yang dibuat selalu mengimplementasikan fungsi render untuk menampilkan halaman pada layar. Fungsi yang terdapat pada kelas – kelas tersebut banyak terdapat perintah untuk eksekusi atau memanggil API yang ada diluar. Dari 36 kelas , hanya 5 klas yang akan dijelaskan sebagai sampel, *yaitu klas OrderPage , AddProduct, OrderDetailPage, LoginPage, dan RegisterPage*. Berikut penjelasan masing-masing klas tersebut :

1. **Klas *OrderPage***

Detail dari *klas AutentikasiController* akan dijelaskan pada Tabel 5.1.

Tabel 5.1 Detail Klas *OrderPage*

Nama Operasi	*Visibility*	*Tipe*
createAddressOptions()	*public*	*void*
onCourierChange(index)	*public*	*void*
componentDidMount()	*Public*	*Void*
initiateShippings()	*Public*	*Void*
onAddressChange(addressIndex)	*Public*	*Void*
getCourierServices(index)	*Public*	*Void*
getAddress(userid)	*Public*	*Void*
onSubmit()	*Public*	*Void*
render()	*public*	*Void*

2. **Klas *AddProduct***

Detail dari *klas SalesController* akan dijelaskan pada Tabel 5.2.

Tabel 5.2 Detail Klas *AddProduct*

Nama Operasi	*Visibility*	*Tipe*
componentDidMount()	*public*	*void*
groupJenis(categories)	*public*	*void*
groupBahan(categories)	*public*	*void*
groupProses(categories)	*public*	*void*
getCategories()	*Public*	*Void*
onSubmit()	*Public*	*Void*
render()	*public*	*void*

3. ***Klas OrderDetailPage***

Detail dari *klas ImporBerikatControlle*r akan dijelaskan pada Tabel 5.3.

Tabel 5.3 Detail Klas *OrderDetailPage*

Nama Operasi	*Visibility*	*Tipe*
sumShipping()	*public*	*void*
paymentCheck()	*public*	*void*
onDelivered(merchantId)	*public*	*void*
getStatus(shippingStatus)	*public*	*void*
renderDeliveryButton(status, title)	*public*	*void*
render()	*public*	*void*

4. **Klas *LoginPage***

Detail dari klas *LoginPage* yang akan dijelaskan pada Tabel 5.4.

Tabel 5.4 Detail Klas *LoginPage*

Nama Operasi	*Visibility*	*Tipe*
onLoginSuccess()	*public*	*void*
doLogin()	*public*	*void*
onPress(x)	*public*	*void*
render()	*public*	*void*

5. ***Klas RegisterPage***

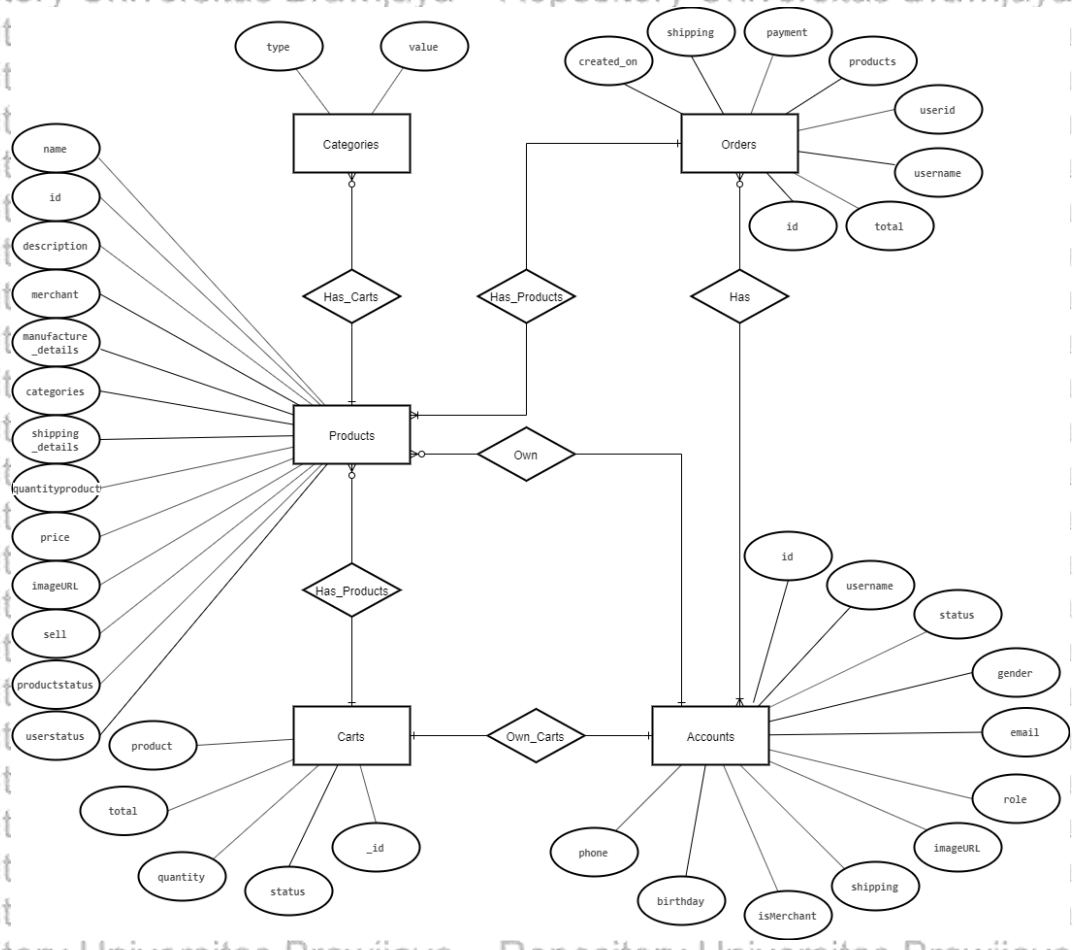
Detail dari *klas MemberController* akan dijelaskan pada Tabel 5.5.

Tabel 5.5 Detail Klas *RegisterPage*

Nama Operasi	*Visibility*	*Tipe*
componentDidMount()	*public*	*void*
createAccount()	*public*	*void*
onRegister()	*public*	*void*
render()	*public*	*void*

5.1.5 Perancangan Basis Data

Entity Relation Diagram (ERD) digunakan untuk menggambarkan hubungan antara entitas yang ada pada sistem dan akan digunakan sebagai dasar dalam implementasi basis data. Database yang digunakan adalah database *NoSQL* Anfield yang berada pada server. Perancangan ERD menggunakan tipe logical data model yang dapat dilihat pada gambar 5.8 berikut. Perancangan basis data disini hanyalah schema yang diekspektasikan akan banyak ditulis kedalam database. Banyaknya API yang digunakan membuat data yang di simpan sangat dinamis, karena jenis data yang disimpan berupa objek yang di dalamnya memiliki variable dan objek yang dinamis (*nested*), Contohnya pada field *payment* dirancang untuk menampung kembalian dari API midtrans.



Gambar 5.13 Perancangan Basis Data

5.1.6 Perancangan Algoritme

Perancangan algoritme dilakukan untuk mendefinisikan proses pada fungsi yang telah di definisikan pada kelas diagram. Perancangan dilakukan dengan menulis alur proses pada fungsi yang akan dibuat. Akan tetapi, hanya 3 fungsi yang di definisikan sebagai sample yaitu fungsi *groupBy*, *uploadImage* dan *Login*.

5.1.6.1 Perancangan Algoritme fungsi *groupBy*

Fungsi memiliki tujuan untuk melakukan pengelompokan barang berdasarkan toko. Fungsi ini berada pada kelas *cartPage*. Perancangan Algoritme yang terdapat dalam fungsi ini dijelaskan pada Tabel 5.6.

Tabel 5.6 Alogritme fungsi *groupBy*

No	Kode
1	initialize variable i = 0
2	initialize variable index = 0
3	initialize variable merchant = execute function getMerchantName(id of merchant), await until done
4	initialize variable result = [object { title:null, merchantname:null,citymerchant:null, data: [empty] }]
5	result[i].title = collection[0].merchant
6	result[i].merchantname = merchant.username
7	result[i].citymerchant = merchant.city
8	FOR value OF collection DO
9	IF (value.merchant == result[i].title) THEN
10	push value to result[i].data
11	(this class var) orderList[index].index = index
12	ELSE
13	i++
14	this.orderList[index].index = index
15	result.push({ title: null, merchantname: null, data:[]})
16	var merchant = execute getMerchantName(Merchant id)
17	result[i].merchantname = merchant.username
18	result[i].citymerchant = merchant.city
19	result[i].title = value.merchant
20	result[i].data.push(value)
21	END IF
22	index++
23	END FOR
24	(this class var) DATA = result

5.1.6.2 Perancangan Algoritme fungsi *uploadImage*

Fungsi memiliki tujuan untuk mengunggah foto menggunakan *API firebase storage*. Fungsi ini berada pada kelas *ProductImageManagement.* Perancangan Algoritme yang terdapat dalam fungsi ini dijelaskan pada Tabel 5.7.

Tabel 5.7 Alogritme fungsi *UploadImage*

No	Kode
1	initialize variable outPut = string path to phone storage
2	(this class var) [index]=firebase
3	Compress Image , THEN
4	outPut = outPut+photoUri
5	fileName= Random
6	(this class var) isUploaded[index]=false
7	FOR uploadImage[index], Upload status, DO
8	IF (Upload status success) THEN
9	(this class var) imagesUrl[index]=url Download
10	(this class var) isUploaded[index]=true
11	(this class var) totalUploaded=(this class var) totalUploaded = totalUploaded + 1
12	IF ((this class var) totalUploaded = (this class var) length of uploadImages), THEN
13	Navigate Page
14	break
15	Else
16	Nothing
17	END IF
18	Else
19	Nothing
20	END IF
21	END FOR

5.1.6.3 Perancangan Algoritme fungsi *doLogin*

Fungsi ini memiliki tujuan untuk pengguna masuk kedalam sistem bedasarkan *email & password* menggunakan *API firebase auth*. Fungsi ini berada pada kelas *login* .Perancangan Algoritme yang terdapat dalam fungsi ini dijelaskan pada Tabel 5.8.

Tabel 5.8 Alogritme fungsi *doLogin*

No	Kode
1	TRY, Then
2	execute FirebaseAuth(email,password)
3	IF (email user not verified) THEN
4	send alert to screen "email not verified"
5	break
6	ELSE
7	initialize variable user = object var { auth: { id: result.user.uid, role:'user',shipping:[] } }
8	execute getProfile PROMISE(user)
9	execute AsyncStorage.setItem('auth') PROMISE, THEN
10	initialize variable result = parse(AsyncStorage.getItem('auth'))
11	IF (global.userRole='user') THEN
12	Navigate to main page
13	ELSE
14	Navigate to Admin Page
15	END IF
16	END IF
17	CATCH
18	send error message to screen
19	END CATCH
20	END TRY

5.1.7 Perancangan Antarmuka

Perancangan antarmuka dilakukan untuk menjelaskan tentang gambaran rancangan terhadap tampilan dari Aplikasi *E-Commerce* ini. Ada 11 rancangan antar muka yang akan ditampilkan sebagai sampel yaitu antar muka *Register, Login*, Halaman Utama, Detail Produk, *Cart, Order, OrderList, OrderDetail, Profil, Sell Report*, Menu Akun.

5.1.7.1 Perancangan Antarmuka *Register*

Perancangan antarmuka untuk *Register member* akan digambarkan seperti pada gambar berikut.

Gambar 5.14 Rancangan Antarmuka *Register*

Tabel 5.9 Penjelasan Rancangan Antarmuka *Register*

No	Nama Objek	Keterangan
1	*Title*	Text untuk menampilkan judul halaman *Sign-Up*
2	*Subtitle*	Text yang berisikan sub-title dari halaman *Sign-Up*
3	*Form Title* "nama"	Nama form untuk *form input* nama
4	*Form Input* "nama"	Form input yang berisikan beberapa *attribute* nama *member* untuk melakukan *Sign-Up*
5	*Form Title "email"*	Nama form untuk *form input email*
6	*Form Input "email"*	Form input yang berisikan beberapa attribute nama *member* untuk melakukan *Sign-Up*
7	*Form Title "password"*	Nama form untuk *form input password*
8	*Form Input "password"*	Form input yang berisikan beberapa *attribute* nama *member* untuk melakukan *Sign-Up*
9	*Button "SIGN UP"*	*Button* yang digunakan untuk mendaftarkan diri kedalam sistem

5.1.7.2 Perancangan Antarmuka *Login*

Perancangan antarmuka untuk Login *member* akan digambarkan seperti pada gambar berikut.

Gambar 5.15 Rancangan Antarmuka *Login*

Tabel 5.10 Penjelasan Rancangan Antarmuka *Login*

No	Nama Objek	Keterangan
1	Title	Text yang digunakan untuk menampilkan tulisan "*Login*"
2	Subtilte	Text untuk mengenai *sub-title* dari *Login*
3	Email	*Form-input* untuk yang digunakan untuk menginput *email*
4	Password	*Form-input* untuk yang digunakan untuk menginput *Password*
5	Login	*Button* untuk *login* ke dalam sistem
6	Register	*Text & Hyperlink* menuju halaman *register*
7	Forgotten Password?	*Text & Hyperlink* menuju halaman lupa *password*

5.1.7.3 Perancangan Antarmuka Halaman Utama

Perancangan antarmuka untuk Haman Utama akan digambarkan seperti pada gambar berikut.

Gambar 5.16 Rancangan Antarmuka Haman Utama

Tabel 5.11 Penjelasan Rancangan Antarmuka Haman Utama

No	Nama Objek	Keterangan
1	*Search* Produk	*Search bar* yang digunakan untuk mencari sebuah produk
2	*Search by price*	*text* yang berisikan aksi untuk navigasi ke halaman *search by price*
3	Judul Kategori	*Text* untuk keterangan *icon Image* dari Kategori
4	*Title Section*	*Text* yang berisikan informasi mengenai produk-produk yang ada didalam sistem
5	info	Informasi yang berisikan nama produk, nama toko, lokasi toko dan harga produk

5.1.7.4 Perancangan Antarmuka Detail Produk

Perancangan antarmuka untuk Detail Produk akan digambarkan seperti pada gambar berikut.

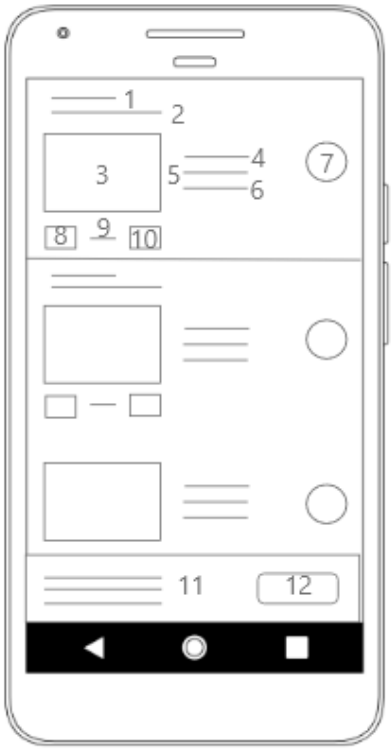
Gambar 5.17 Rancangan Antarmuka Detail Produk

Tabel 5.12 Penjelasan Rancangan Antarmuka Detail Produk

No	Nama Objek	Keterangan
1	Gambar Produk	Image untuk menampilkan Gambar Produk
2	Indikator gambar	Indicator yang menunjukan banyaknya gambar dan urutan gambar yang sedang ditampilkan
3	Nama Produk	Text untuk menampilkan nama produk
4	Detail	Text untuk menampilkan deskripsi produk
5	Kuantitas	Text untuk menampilkan angka jumlah produk yang tersedia
6	Kategori	Text yang menampilkan kategori produk
7	Harga	*Text* untuk menampilkan harga
8	*Buy*	*Button* yang digunakan untuk mamasukan produk ke dalam keranjang

5.1.7.5 Perancangan Antarmuka *Cart*

Perancangan antarmuka *Cart* produk yang dibeli akan digambarkan seperti pada gambar berikut.



Gambar 5.18 Rancangan Antarmuka *Cart*

Tabel 5.13 Penjelasan Rancangan Antarmuka *Cart*

No	Nama Objek	Keterangan
1	Nama Toko	Text yang menampilkan nama toko
2	Kota / Kabupaten Toko	Text yang menampilkan kota / kabupaten toko
3	Gambar Barang	Gambar pertama dari tiap barang
4	Nama Barang	Text yang merepresentasikan nama barang
5	Stok Barang	Text yang menampilkan stok barang
6	Harga Barang	Text yang menampilkan harga barang dikalikan dengan kuantitas
7	Icon Tempat sampah	Button yang digunakan untuk menghapus barang dari Cart
8	Tombol dash quantity	Button untuk mengurangi satu quantity yang ingin dibeli

9	Text Quantity	Text yang menampilkan kuantitas baarng yang diatur member
10	Tombol Plus quantity	Button untuk menambah satu quantity yang ingin dibeli
11	Text total	Text yang menampilkan total pembayaran
12	Tombol Checkout	Tombol yang melakukan aksi berupa navigasi ke halaman order

5.1.7.6 Perancangan Antarmuka *OrderPage*

Perancangan antarmuka *OrderPage* yang digunakan untuk melakukan pemesanan produk akan digambarkan seperti pada gambar berikut.

Gambar 5.19 Rancangan Antarmuka *OrderPage*

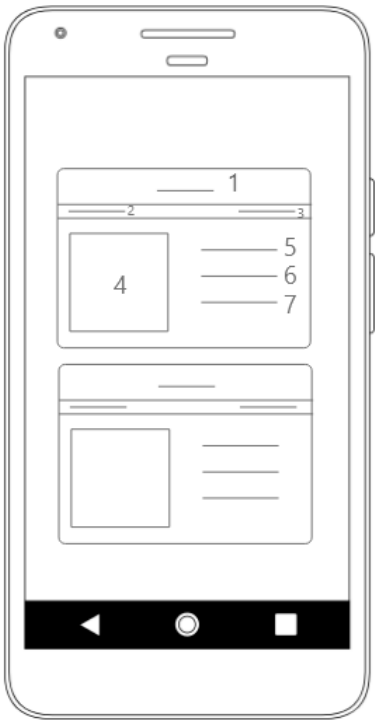
Tabel 5.14 Penjelasan Rancangan Antarmuka *OrderPage*

No	Nama Objek	Keterangan
1	Judul Halaman	Image untuk menampilkan Gambar Produk

2	Options Alamat Member	Text untuk menampilkan nama produk
3	Order	Button yang digunakan untuk membeli produk
4	Nama & Lokasi Toko	Text yang menampil kan nama took & lokasi took (kabupaten / kota)
5	Kurir	Options untuk memilih kurir
6	Service	Options untuk memilih jenis service berdasarkan kurir yang dipilih
7	Tombol Order	Tombol untuk mengeksekusi order

5.1.7.7 Perancangan Antarmuka *OrderList*

Perancangan antarmuka *OrderList* yang digunakan untuk melihat daftar pemesanan *member* akan digambarkan seperti pada gambar berikut.



Gambar 5.20 Rancangan Antarmuka *OrderList*

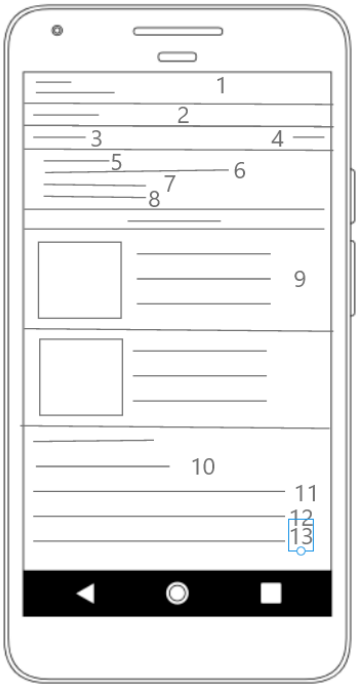
Tabel 5.15 Penjelasan Rancangan Antarmuka *OrderList*

No	Nama Objek	Keterangan
1	Status	Text untuk menampilkan status order
2	Id order	Text untuk menampilkan id order

3	Tanggal order	Text untuk menampilkan tanggal order dibuat
4	gambar	Gambar yang menunjukan produk yang dipost oleh penjual
5	Nama barang	Text untuk menampilkan nama produk
6	Jumlah toko	Text untuk jumlah took yang ada di dalam order
7	Total harga	Text yang menampilkan total pembayaran

5.1.7.8 Perancangan Antarmuka OrderDetail

Perancangan antarmuka *OrderDetail* yang digunakan untuk melihat detail pemesanan produk akan digambarkan seperti pada Gambar 5.16.



Gambar 5.21 Rancangan Antarmuka *OrderDetail*

Tabel 5.16 Penjelasan Rancangan Antarmuka *OrderDetail*

No	Nama Objek	Keterangan
1	Status	*Text* untuk menampilkan status order
2	Tanggal Order	*Text* untuk menampilkan tanggal order dibuat
3	No Invoice	*Text* untuk menampilkan kode *invoice order*
4	Link Invoice	*Hyperlink* yang mengarahkan ke halamman download invoice dibrowser
5	Nama toko	*Text* yang menampilkan nama toko dari order yang dibuat

6	Tracking number	*Text* yang menampilkan nomor resi
7	kurir	*Text* yang menampilkan kurir yang dipilih ketika order
8	Ongkos kirim	*Text* yang menampilkan angka jumlah ongkos kirim.
9	Detail produk	*Text* yang menampilkan informasi nama,kuantitas ,harga produk.
10	Informasi pembayaran	*Text* yang menampilkan jenis pembayaran & No Virtual Account.
11	Total harga barang	*Text* yang menampilkan total harga barang.
12	Total ongkos kirim	*Text* yang menampilkan total ongkos kirim.
13	Total	*Text* yang menampilkan total harga barang ditambah total ongkos kirim.

5.1.7.9 Perancangan Antarmuka Profil

Perancangan antarmuka Profil yang digunakan untuk melihat informasi *member* akan digambarkan seperti pada gambar dibawah ini.

Gambar 5.22 Rancangan Antarmuka Profil

Tabel 5.17 Penjelasan Rancangan Antarmuka Profil

No	Nama Objek	Keterangan

1	Gambar profil	Gambar profil member.
2	Nama	*Text* yang menampilkan nama member.
3	Nama Lengkap	*Text* yang menampilkan nama lengkap member.
4	Tanggal Lahir	*Text* yang menampilkan tanggal lahir member.
5	Jenis Kelamin	*Text* yang menampilkan jenis kelamin member.
6	Email	*Text* yang menampilkan Email member.
7	No Telepon	*Text* yang menampilkan No Telepon member.

5.1.7.10 Perancangan Antarmuka *Sell Report*

Perancangan antarmuka *Sell Report* yang digunakan untuk melihat informasi *member* akan digambarkan seperti pada gambar berikut.

Gambar 5.23 Rancangan Antarmuka *Sell Report*

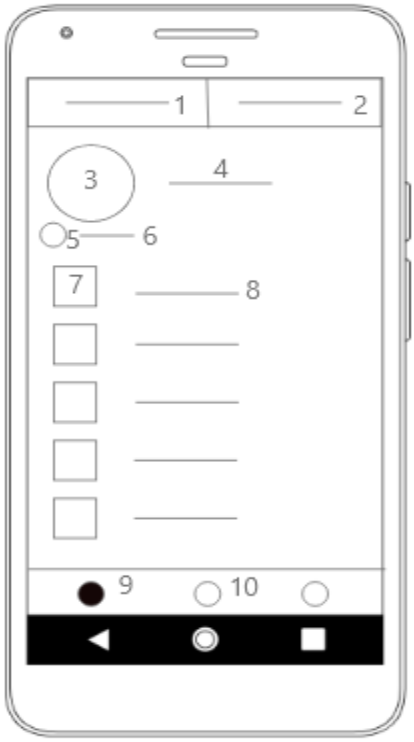
Tabel 5.18 Penjelasan Rancangan Antarmuka *Sell Report*

No	Nama Objek	Keterangan
1	Grafik penjualan	Grafik yang menunjukan performa penjualan dari range tanggal yang di atur.
2	Tanggal mulai	Tombol untuk mengubah tanggal pilihan mulai laporan penjualan.

3	Tanggal selesai	Tombol untuk mengubah tanggal pilihan mulai laporan penjualan.
4	Tombol tampilkan	*Button* untuk menampilkan laporan penjualan dari tanggal yang telah diatur.
5	Tanggal & total order	*List order* berdasarkan hari.
6	Jumlah barang	Jumlah barang dari antara tanggal yang telah diatur.
7	Total pendapatan	Total pendapatan dari antara tanggal yang telah diatur.

5.1.7.11 Perancangan Antarmuka Akun Menu

Perancangan antarmuka Akun Menu yang digunakan untuk melihat informasi *member* akan digambarkan seperti pada gambar berikut.



Gambar 5.24 Rancangan Antarmuka Akun Menu

Tabel 5.19 Penjelasan Rancangan Antarmuka Akun Menu

No	Nama Objek	Keterangan
1	Akun Pembeli	*Text* untuk menampilkan judul halaman
2	Akun *Merchant*	*Icon* palu untuk tab *merchant / icon* keranjang untuk tab pembeli.

3	Gambar profil	Gambar profil member.
4	Nama Member	*Text* untuk menampilkan nama member.
5	Ikon tab	Ikon tab yang membedakan antara tab pembeli & tab *merchant*.
6	Jenis tab	Text yang menunjukan tab pembeli & tab merchant.
7	Ikon menu	Icon dari list menu.
8	Nama menu	Text yang menampilkan nama dari *list* menu.
9	*Selected page icon*	Ikon yang dipilih akan berubah menjadi warna hitam.
10	*Page icon*	Ikon yang dipilih akan berubah menjadi warna putih.

5.2 Implementasi Sistem

Hasil dari 7 tahap perancangan yang telah dilakukan akan menjadi acuan untuk pelakukan implementasi aplikasi. Akan dilakukan 3 tahap pada proses implementasi yaitu implemetasi data berisi tangkapan gambar collection, implementasi kode program berdasarkan perancangan algoritme dan implementasi Antarmuka. Ada 3 spesifikasi sistem yang akan di jabarkan terkait perangkat yang digunakan dalam proses pengerjaan sistem. Spesifikasi yang akan dijabarkan antaralain spesifikasi perangkat keras, spesifikasi perangkat lunak dan spesifikasi sistem operasi.

Pada sisi *service* proses *deployment* dilakukan pada hosting domainesia, service yang diberikan mampu menangani proses yang dieksekusi oleh kode program. Sedangkan untuk service basis data *mongodb* di implementasikan pada *mongoDB atlas* agar beban sistem dapat terbagi.

5.2.1 Spesifikasi Sistem

Untuk melakukan implementasi dalam pengembangan sistem dibutuhkan sebuah sistem yang memiliki resource. Spesifikasi resource perangkat keras dan perangkat lunak akan didefinisikan dibawah.

5.2.1.1 Spesifikasi Perangkat Lunak

Spesifikasi yang berisikan informasi tentang bahsa pemrograman yang digunakan dan juga versi yang digunakan selama pengembangan aplikasi terdapat pada Tabel 5.20.

Tabel 5.20 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
Editor Dokumentasi	*Microsoft Office Word* 2019
Editor Perancangan	*Draw.io & Adobe XD 17.0.12.11*

Editor Pemograman	*Visual Studio Code*
FrameWork	*NodeJS, React Native*

5.2.1.2 Spesifikasi Perangkat Keras

Selama pengembangan aplikasi digunakan juga *resource* perangkat keras yang digunakan selama pengembangan aplikasi. Spesifikasi perangkat keras tersebut terdapat pada Tabel 5.21.

Tabel 5.21 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
Laptop	1. ASUS Zephyrus M GU502GU 2. Processor Intel I7-9750H 3. RAM 16gb DDR4, 2666 Mhz 4. GTX 1660Ti 6gb

5.2.1.3 Spesifikasi Sistem Operasi

Bagian ini berisikan informasi spesifikasi sistem operasi yang mendukung dan dapat digunakan selama tahap pengembangan sistem. Perangkat keras tersebut dijelaskan pada Tabel 5.22.

Tabel 5.22 Spesifikasi Sistem Operasi

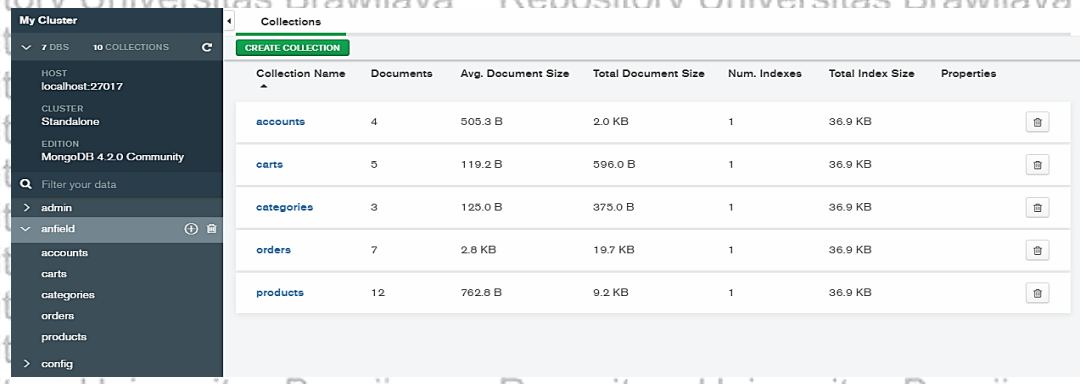
Nama Komponen	Spesifikasi
Sistem Operasi	Windows 10 Home Edition 64-Bit

5.2.2 Implementasi Basis Data

Struktur basis data pada *mongoDB* terdiri dari macam-macam *collection* dimana *collection* tersebut merupakan entitas yang telah di definisikan pada perancangan basis data. *Field* yang di definisikan merupakan atribut yang ada pada *ERD*. *MongoDB* bersifat dinamis oleh karena itu nilai dari atribut dapat memiliki atribut lagi bahkan array atribut. Sedangkan relasi yang didefinisikan pada tahap perancangan di definisikan pada kode program.

5.2.2.1 DB Collection List

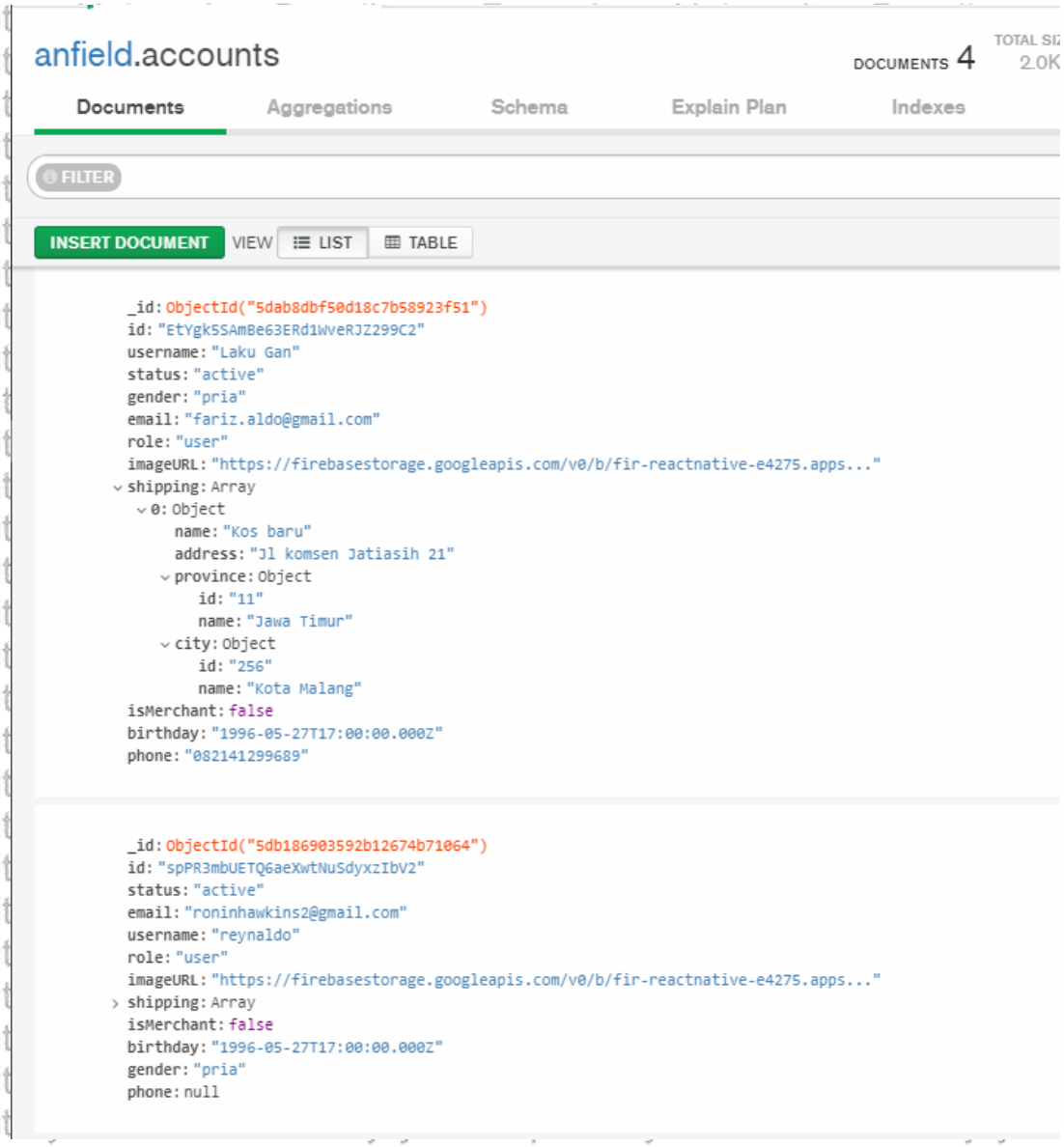
Berikut adalah daftar collection yang di implementasikan pada DB anfield yang telah dibuat, dapat dilihat pada Gambar 5.25.



Gambar 5.25 Daftar *Collection* Pada *DB Anfield*

5.2.2.2 Collection Account

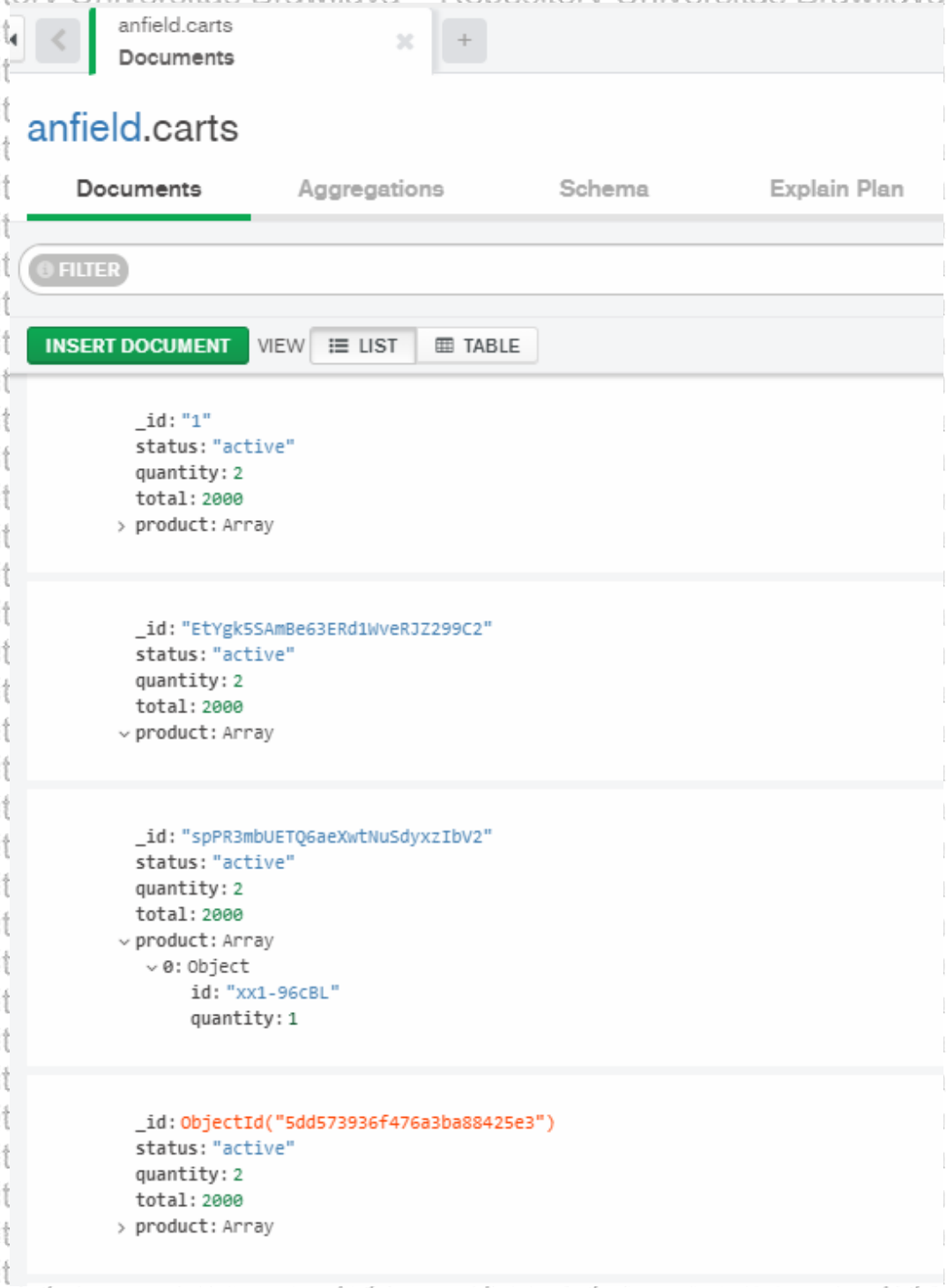
Dari perancangan yang telah dilakukan, salah satu hasil implementasi basis data adalah *collections accounts*, yang dapat dilihat pada Gambar 5.26.



Gambar 5.26 *Collection Accounts*

5.2.2.3 Collection carts

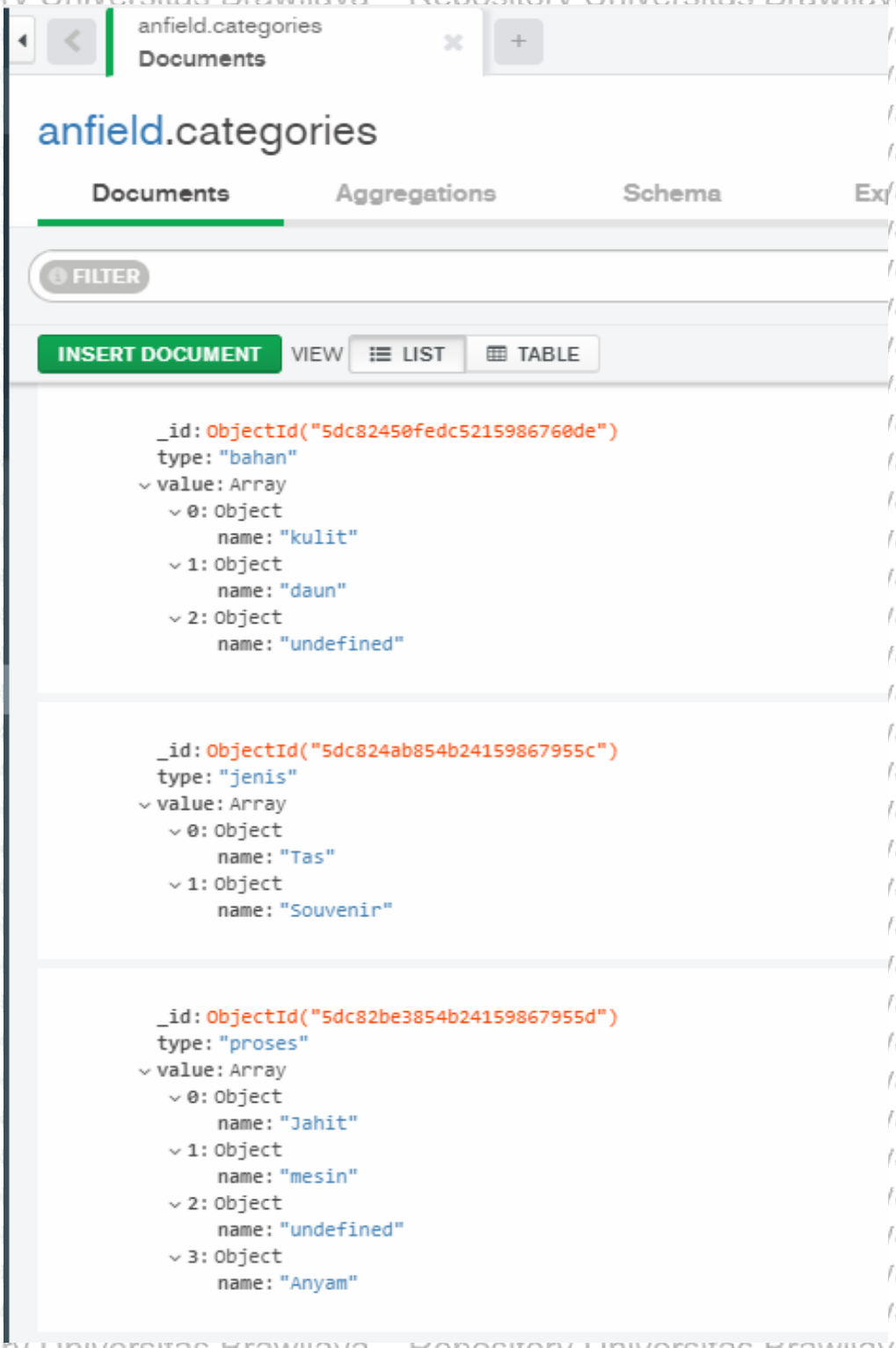
Dari perancangan yang telah dilakukan, salah satu hasil implementasi basis data adalah *collections carts*, yang dapat dilihat pada Gambar 5.27.



Gambar 5.27 *Collection Carts*

5.2.2.4 Collection categories

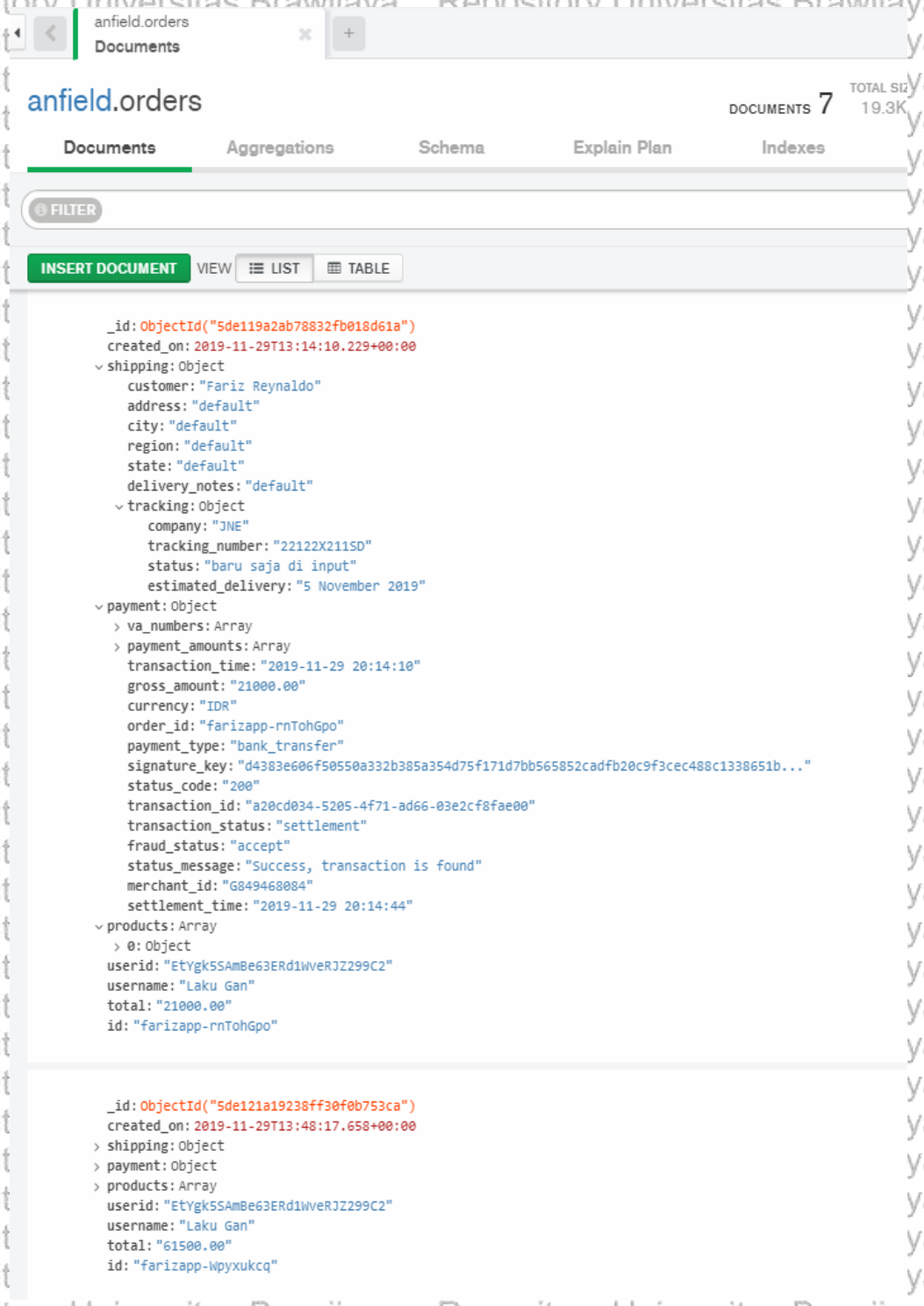
Dari perancangan yang telah dilakukan, salah satu hasil implementasi basis data adalah *collections categories*, yang dapat dilihat pada Gambar 5.28.



Gambar 5.28 *Collection categories*

5.2.2.5 Collection orders

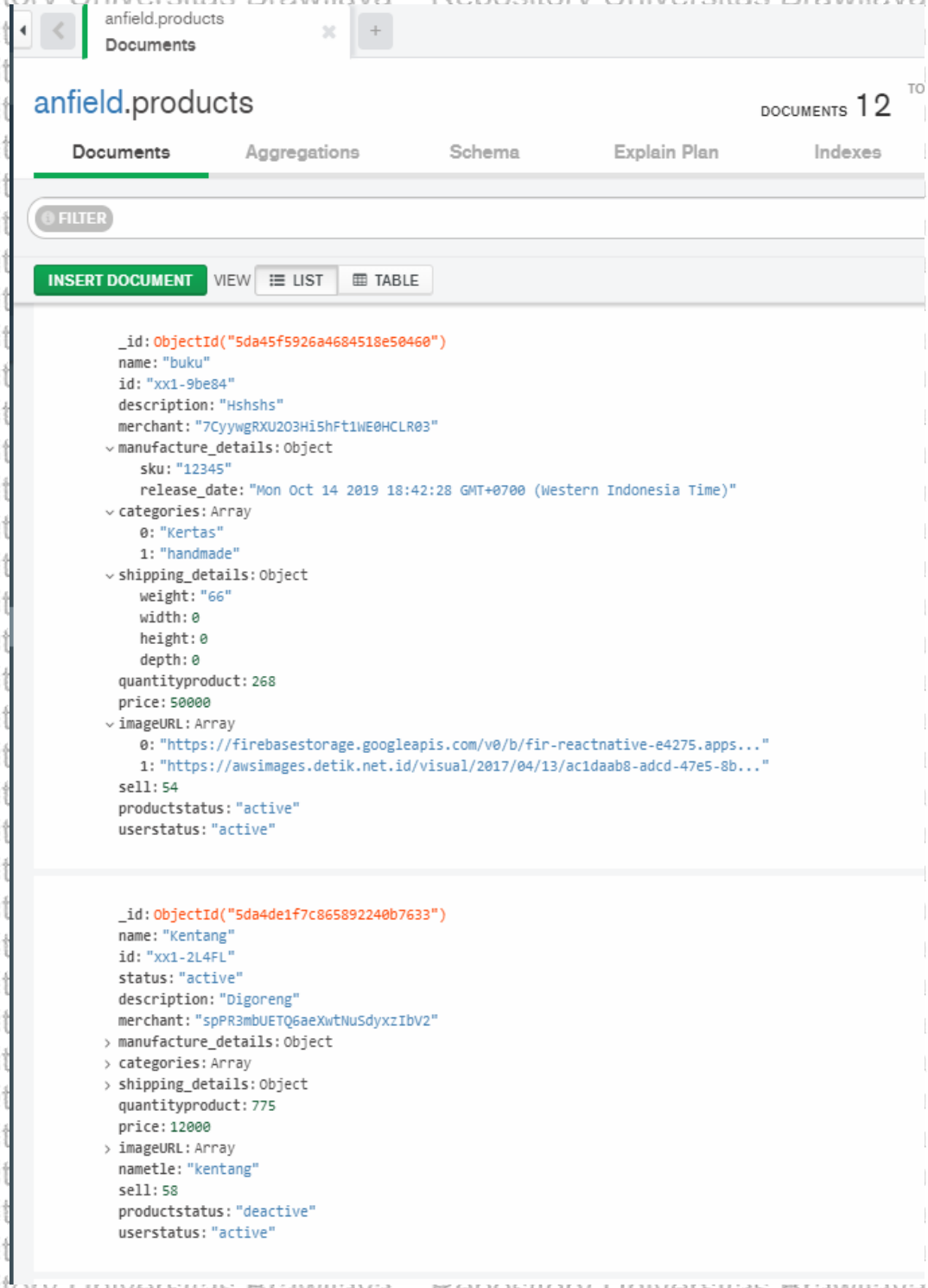
Dari perancangan yang telah dilakukan, salah satu hasil implementasi basis data adalah *collections orders* yang dapat dilihat pada Gambar 5.29. Pada field *payment* dirancang untuk menampung kembalian dari *API midtrans*.



Gambar 5.29 *Collection orders*

5.2.2.6 DB Collection products

Dari perancangan yang telah dilakukan, salah satu hasil implementasi basis data adalah *collections products*, yang dapat dilihat pada Gambar 5.30.



Gambar 5.30 *Collection products*

5.2.3 Implementasi Kode Program

Implementasi kode program dilakukan menggunakan referensi dari perancangan algoritme yang dibuat sebelumnya. Kode program yang di implementasikan adalah fungsi *groupBy* pada kelas *cartPage*, *uploadImage* pada kelas *ProductImageManagement*, lalu fungsi *doLogin* dan kelas *login*.

5.2.3.1 Implemetasi kode Program Fungsi "groupBy"

Fungsi memiliki tujuan untuk melakukan pengelompokan barang berdasarkan toko. Isi kode program yang terdapat dalam fungsi ini dijelaskan pada Tabel 5.23.

Tabel 5.23 Kode Program Fungsi "*groupBy*"

No	Kode
1	`async groupBy(collection) {`
2	`var i = 0;`
3	`var index = 0`
4	`var merchant = await this.getMerchantName(collection[0].merchant)`
5	`let result = [{ title: null,`
6	`merchantname: null,`
7	`citymerchant:null,`
8	`data: [] }];`
9	`result[i].title = collection[0].merchant;`
10	`result[i].merchantname = merchant.username;`
11	`result[i].citymerchant = merchant.city;`
12	`for (var value of collection) {`
13	`if (value.merchant == result[i].title) {`
14	`result[i].data.push(value);`
15	`this.orderList[index].index = index; }`
16	`else {`
17	`i++;`
18	`this.orderList[index].index = index;`
19	`result.push({ title: null,`
20	`merchantname: null,`
21	`data: [] })`
22	`var merchant = await this.getMerchantName(value.merchant);`
23	`result[i].merchantname = merchant.username;`
24	`result[i].citymerchant = merchant.city;`
25	`result[i].title = value.merchant;`
26	`result[i].data.push(value); }`
27	`index++; }`
28	`this.DATA = result;`
29	`return result; }`

5.2.3.2 Implemetasi kode Program Fungsi "uploadImage"

Fungsi memiliki tujuan untuk mengunggah foto menggunakan *API firebase storage*. Isi kode program yang terdapat dalam fungsi ini di jelaskan pada Tabel 5.24.

Tabel 5.24 Kode Program Fungsi "*uploadImage*"

No	Kode
1	`async uploadImage(photoUri,index) {`
2	`  var outPut = "/storage/emulated/0/";`
3	`  this.multiFirebase[index]=firebase;`
4	`  await  ImageResizer.createResizedImage(photoUri, 640, 480, 'JPEG', 80).then((res) => {`
5	`    outPut=outPut+photoUri.toString();`
6	`    fileName= '/'+this.uuidv4()+'.jpeg';`
7	`    this.isUploaded[index]=false;`
8	`    this.multiFirebase[index].storage().ref(fileName)`
9	`    .putFile(res.path)`
10	`.on(this.multiFirebase[index].storage.TaskEvent.`
11	`STATE_CHANGED, snapshot=>{`
12	`      if (snapshot.state === this.multiFirebase[index].storage.TaskState.SUCCESS && this.isUploaded[index]==false) {`
13	`        this.imagesUrl[index]=snapshot.downloadURL;`
14	`        this.isUploaded[index]=true;`
15	`        this.totalUploaded=this.totalUploaded+1;`
16	`        if(this.totalUploaded==this.uploadImages.length){`
17	`          this.navigatePage();`
18	`        return;  }`
19	`      }`
20	`    })`
21	`  })`

5.2.3.3 Implemetasi kode Program Fungsi "doLogin"

Fungsi ini memiliki tujuan untuk pengguna masuk kedalam sistem bedasarkan *email & password* menggunakan *API firebase auth*. Isi kode program yang terdapat dalam fungsi ini di jelaskan pada Tabel 5.25.

Tabel 5.25 Kode Program Fungsi "*doLogin*"

No	Kode
1	`doLogin() {`

```
2  firebase.auth().signInWithEmailAndPassword(this.state.email,
   this.state.password).then(function (result) {
3    if (!result.user.emailVerified) {
4      Alert.alert('Email Belum Diverifikasi');
5      return; }
6    var user = { auth: { id: result.user.uid,
7                         role:'user',
8                         shipping:[] } }
9    fetch(ip + '/getProfile', {
10       method: 'POST',
11       headers: {
12         'Accept': 'application/json',
13         'Content-Type': 'application/json'},
14       body: JSON.stringify({userId : result.user.uid})
15   }).then((response)                                  =>
   response.json()).then((responseJson)  => {
16     user.auth.shipping= responseJson.shipping;
17     user.auth.role= responseJson.role;
18     user.auth.username= responseJson.username;
19     AsyncStorage.setItem('auth', JSON.stringify(user)
   async () => {
20       var result = JSON.parse(await
   AsyncStorage.getItem('auth'));
21       global.userid = result.auth.id;
22       global.username= result.auth.username;
23       global.cartChanged = true;
24       global.userRole = result.auth.role;
25       global.shipping = result.auth.shipping;
26       global.userChanged = 0;
27       if (global.userRole=='user'){
28         Actions.replace('tabbarx');  }
29       else{
30         Actions.replace('AdminMenuPage')  }
31       });
32     }).catch((error) => {
33       Alert. (error.message);
34   });
35 }).catch(function (error) {
36   var errorCode = error.code;
37   var errorMessage = error.message;
38   console.log(error.message);
39   return Alert.alert(errorMessage); });
40 }
```

5.2.4 Implementasi antarmuka

Implementasi antarmuka berisikan hasil implementasi dari perancangan antarmuka Aplikasi *E-commerce* untuk Produk Kerajinan telah dibuat. Dari 37 halaman yang dibuat hanya 11 halaman yang dijadikan contoh implementasi antarmuka diantaranya adalah antarmuka *Register*, *Login*, Haman Utama, Detail Produk, *Cart*, *OrderPage*, *OrderList*, *OrderDetail*, *Profile*, *Sell Report* dan Akun Menu.

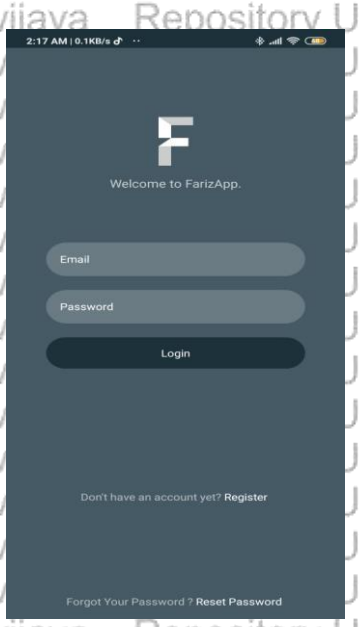
5.2.4.1 Implemetasi antarmuka *register*

Tangkapan gambar dari hasil implementasi antarmuka *Register* dapat dilihat pada gambar dibawah ini.

Gambar 5.31 Antarmuka *Register*

5.2.4.2 Implemetasi antarmuka *register*

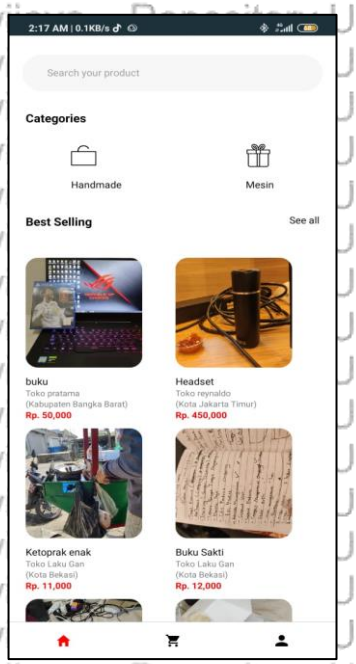
Tangkapan gambar dari hasil implementasi antarmuka Login dapat dilihat pada gambar dibawah ini.



Gambar 5.32 Antarmuka *Login*

5.2.4.3 Implemetasi antarmuka *halaman utama*

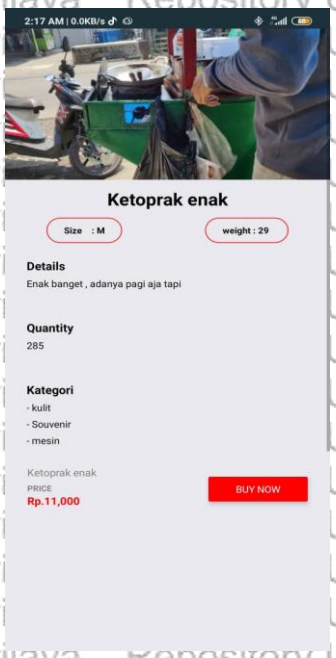
Tangkapan gambar dari hasil implementasi antarmuka halaman utama dapat dilihat pada pada gambar dibawah ini.



Gambar 5.33 Antarmuka Halaman Utama

5.2.4.4 Implemetasi antarmuka *detail produk*

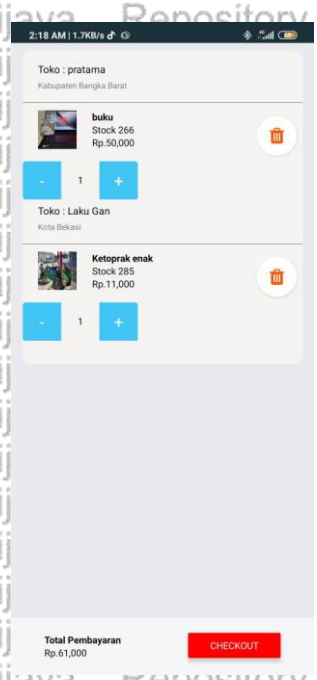
Tangkapan gambar dari hasil implementasi antarmuka *Detail* Produk dapat dilihat pada gambar dibawah ini.



Gambar 5.34 Antarmuka Detail Produk

5.2.4.5 Implemetasi antarmuka *cart*

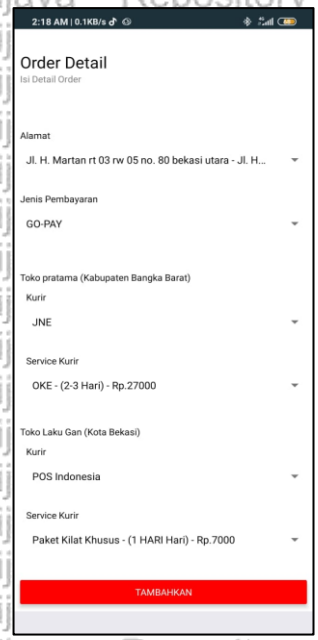
Tangkapan gambar dari hasil implementasi antarmuka *Cart* dapat dilihat pada Gambar gambar dibawah ini.



Gambar 5.35 Gambar Antarmuka *Cart*

5.2.4.6 Implemetasi antarmuka *OrderPage*

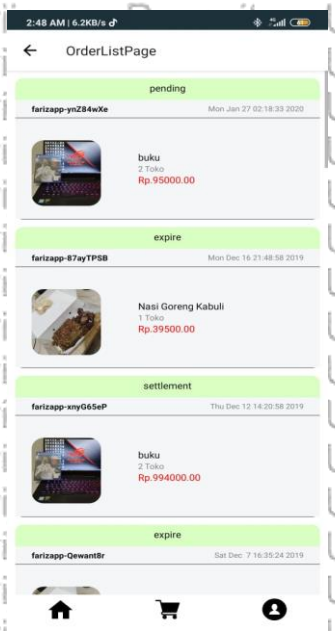
Tangkapan gambar dari hasil implementasi antarmuka *OrderPage* dapat dilihat pada Gambar gambar dibawah ini.



Gambar 5.36 Antarmuka *OrderPage*

5.2.4.7 Implemetasi antarmuka *OrderList*

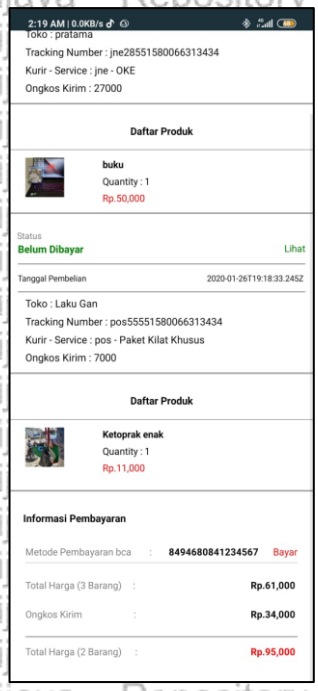
Tangkapan gambar dari hasil implementasi antarmuka *OrderList* dapat dilihat pada Gambar gambar dibawah ini.



Gambar 5.37 Antarmuka *OrderList*

5.2.4.8 Implemetasi antarmuka *OrderDetail*

Tangkapan gambar dari hasil implementasi antarmuka *OrderDetail* dapat dilihat pada gambar dibawah ini.



Gambar 5.38 Antarmuka *OrderDetail*

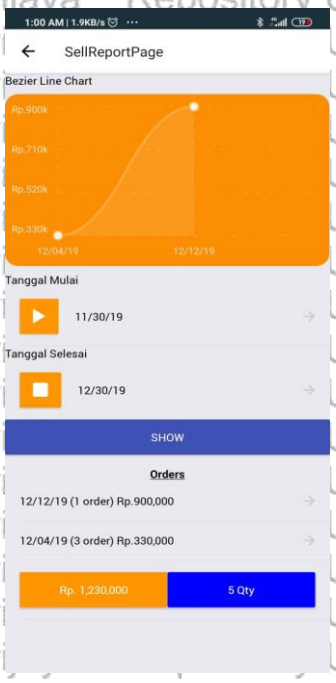
5.2.4.9 Implemetasi antarmuka profil

Tangkapan gambar dari hasil implementasi antarmuka *Profil* dapat dilihat pada Gambar gambar dibawah ini.

Gambar 5.39 Antarmuka *Profil*

5.2.4.10 Implemetasi antarmuka sell report

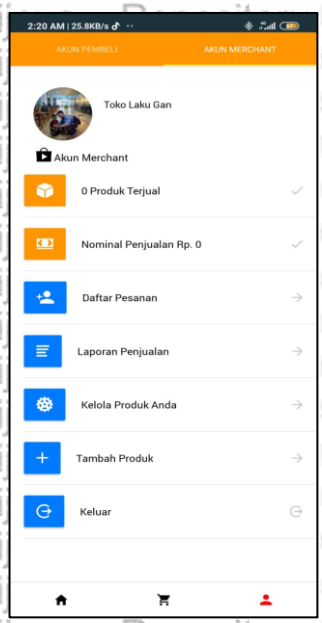
Tangkapan gambar dari hasil implementasi antarmuka *Sell Report* dapat dilihat pada gambar dibawah ini.



Gambar 5.40 Antarmuka *Sell Report*

5.2.4.11 Implemetasi antarmuka akun menu

Tangkapan gambar dari hasil implementasi antarmuka Akun Menu dapat dilihat pada gambar dibawah ini.



Gambar 5.41 Antarmuka Akun Menu

BAB 6 PENGUJIAN

Bab 6 menjelaskan tentang pengujian hasil yang diperoleh dari proses implementasi sistem yang diuji. Tujuan melakukan pengujian adalah untuk mengetahui hasil perancangan apakah sudah sesuai dengan kebutuhan sistem. Proses pengujian yang akan di lakukan pada aplikasi ini adalah menggunakan metode *black-box*, *white-box* dan terakhir pengujian *usability*. Penggunaan metode *white*-box dilakukan pada saat pengujian unit sedangkan *black-box* dilakukan pada proses validasi. Pengujian *Usability* dilakukan dengan meberikan *PSSUQ* kuesioner secara acak kepada pengguna.

6.1 Pengujian Unit

Pengujian unit dilakukan dengan cara memverifikasi unit-unit yang terdapat pada aplikasi yang dikembangkan. Unit-Unit tersebut merupakan fungsi ataupun langkah-langkah yang dijalankan untuk mendapatkan hasil yang diinginkan. Dengan melakukan pengujian dapat mengetahui apakah fungsi telah sesuai dengan hasil perancangan. Pengujian *white-box* yang dilakukan dengan *basis path testing*, yaitu menguji jalur dari alur kode program untuk mendapatkan suatu hasil.

6.1.1 Pengujian Unit Fungsi *GroupBy*

Fungsi *GroupBy* memiliki tujuan untuk melakukan pengelompokan barang yang ada di keranjang berdasarkan toko, Fungsi ini berada di kelas cartPage. Fungsi ini hanya dapat dieksekusi secara tidak langsung oleh pengguna. Psuedocode fungsi GroupBy dapat dilihat pada tabel dibawah ini.

1. Psuedocode fungsi GroupBy.

Tabel 6.1 Pseudocode Fungsi *GroupBy*

No	Kode	Node
1	`initialize variable i = 0`	1
2	`initialize variable index = 0`	1
3	`initialize variable merchant = execute function getMerchantName(id of merchant), await until done`	1
4	`initialize variable result = [object { title:null, merchantname:null,citymerchant:null, data: [empty] }]`	1
5	`result[i].title = collection[0].merchant`	1
6	`result[i].merchantname = merchant.username`	1
7	`result[i].citymerchant = merchant.city`	1
8	`FOR value OF collection DO`	2
9	`IF (value.merchant == result[i].title) THEN`	3
10	`push value to result[i].data`	4
11	`(this class var) orderList[index].index = index`	4
12	`ELSE`	5
13	`i++`	5

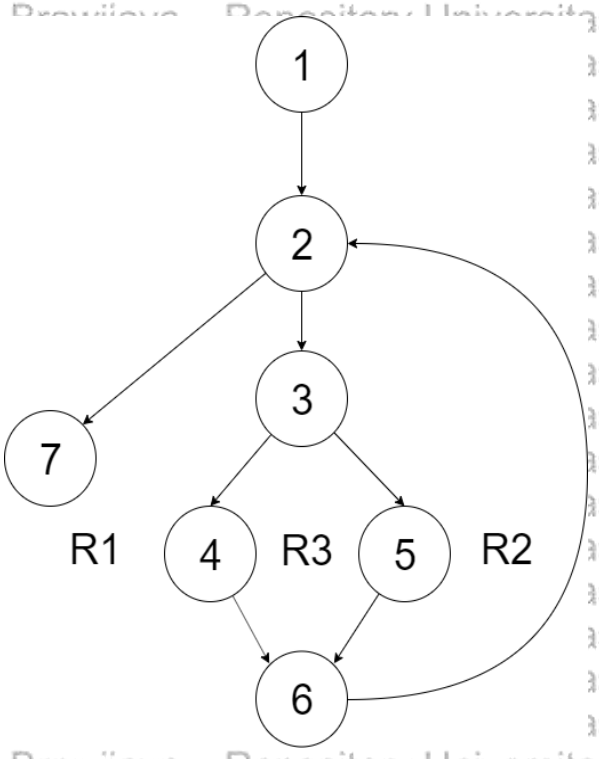
14	this.orderList[index].index = index	5
15	result.push({ title: null, merchantname: null, data:[]})	5
16	var merchant = execute getMerchantName(Merchant id)	5
17	result[i].merchantname = merchant.username	5
18	result[i].citymerchant = merchant.city	5
19	result[i].title = value.merchant	5
20	result[i].data.push(value)	5
21	END IF	6
22	index++	6
23	END FOR	7
24	(this class var) DATA = result	7

2. Basis Path Testing

a. Flow Graph

Hasil *flow graph* yang didapatkan dari *pseudocode*

Fungsi GroupBy digambarkan pada Gambar dibawah ini.



Gambar 6.1 *Flow Graph* fungsi *GroupBy*

b. Cyclomatic Complexity

- V(G) = jumlah *region* = 3
- V(G) = (jumlah *edge* - jumlah *node*) + 2 = (8 – 7) + 2 = 3

- V(G) = jumlah *predicate node* + 1 = 2 + 1 = 3

3. Test Case

Dari hasil *Basis path testing* didapatkan 3 *independent path*. Masing-masing *path* merupakan alur kode program yang mungkin terjadi. Dari jumlah alur kode program yang mungkin terjadi terdapat beberapa hasil pengujian yang dapat dilihat pada Tabel 6.2.

Tabel 6.2 Hasil Pengujian Unit fungsi *GroupBy*

No	No jalur	Prosedur Uji	Expected Result	Result	Status
1	1-2-7	Memanggil fungsi *GroupBy* dengan parameter *array* kosong	Var kelas DATA akan kosong, setelah keluar dari fungsi maka akan tampil alert "Keranjang kosong"	Var kelas DATA kosong dan tampil alert "Keranjang kosong"	*Valid*
2	1-2-3-4-6-2-7	Memanggil fungsi *GroupBy* dengan parameter *array* 2 objek produk & id *merchant* sama.	Halaman keranjang akan menampilkan 2 produk yang telah dikelompokan dengan 1 merchant.	Tampil 2 produk dengan 1 merchant yang sama.	*Valid*
3	1-2-3-5-6-2-7	Memanggil fungsi GroupBy dengan parameter *array* 3 objek produk &2 id *merchant* yang berbeda.	Halaman keranjang akan menampilkan 3 produk yang telah dikelompokan dengan 2 merchant yang berbeda.	Tampil 3 produk dengan 2 merchant yang berbeda.	*Valid*

6.1.2 Pengujian Unit Fungsi *UploadImage*

Fungsi UploadImage memiliki tujuan untuk mengunggah gambar yang dipilih pada *local storage* pengguna. Fungsi ini berada di kelas *ProductImageManagement*. Fungsi ini hanya dapat dieksekusi secara tidak langsung oleh pengguna atau admin. Psuedocode fungsi GroupBy dapat dilihat pada table dibawah ini.

1. Pseudocode fungsi UploadImage.

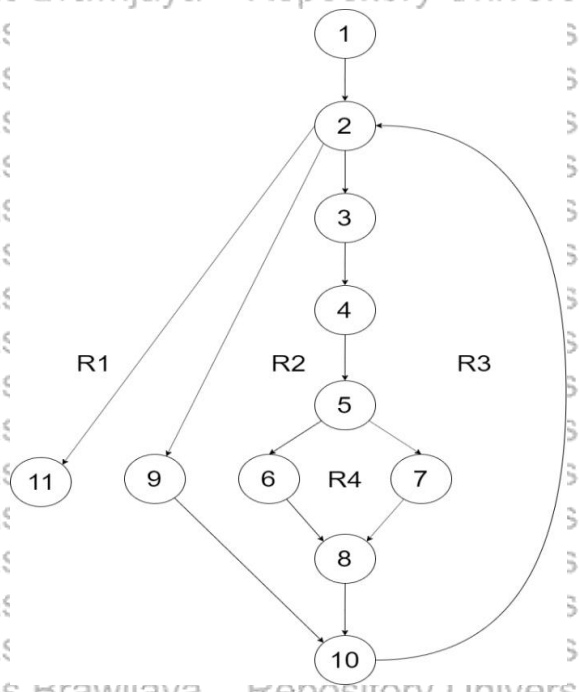
Tabel 6.3 *Pseudocode* Fungsi *UploadImage*

No	Kode	Node
1	initialize variable outPut = string path to phone storage	1
2	(this class var) [index]=firebase	1
3	Compress Image , THEN	1
4	outPut = outPut+photoUri	1
5	fileName= Random	1
6	(this class var) isUploaded[index]=false	1
7	FOR uploadImage[index], Upload status, DO	2
8	IF (Upload status success) THEN	3
9	(this class var) imagesUrl[index]=url Download	4
10	(this class var) isUploaded[index]=true	4
11	(this class var) totalUploaded=(this class var) totalUploaded = totalUploaded + 1	4
12	IF ((this class var) totalUploaded = (this class var) length of uploadImages), THEN	5
13	Navigate Page	6
14	break	6
15	Else	7
16	Nothing	7
17	END IF	8
18	Else	9
19	Nothing	9
20	END IF	10
21	END FOR	11

2. Basis Path Testing

a. Flow Graph

Hasil *flow graph* yang didapatkan dari *pseudocode* fungsi UploadImage digambarkan pada Gambar dibawah ini.



Gambar 6.2 *Flow Graph* fungsi *UploadImage*

b. Cyclomatic Complexity

- V(G) = jumlah *region* = 4
- V(G) = (jumlah *edge* - jumlah *node*) + 2 = (13 – 11) + 2 = 4
- V(G) = jumlah *predicate node* + 1 = 3 + 1 = 4

3. Test Case

Dari hasil *Basis path testing* didapatkan 4 *independent path*. Masing-masing *path* merupakan alur kode program yang mungkin terjadi. Dari jumlah alur kode program yang mungkin terjadi terdapat beberapa hasil pengujian yang dapat dilihat pada Tabel 6.4

Tabel 6.4 Hasil Pengujian Unit fungsi *UploadImage*

No	*Path*	*Test Case*	*Expected Result*	*Result*	*Status*
1	1-2-11	Memanggil fungsi dengan array objek 0 gambar.	Fungsi akan keluar dan alert “anda belum memilih gambar ” akan tampil	Fungsi selesai dan tampil alert “anda belum	*Valid*

				memilih gambar"	
2	1-2-3-4-5-6-8-10-2-11	Memberi nilai array objek 2 gambar, lalu memanggil fungsi *UploadImage()* dengan gambar kedua.	Halaman upload image akan pindah ke halaman selanjutnya	Halaman berpindah ke halaman selanjutnya.	*Valid*
3	1-2-3-4-5-7-8-10-2-11	Memberi nilai array objek 2 gambar, lalu memanggil fungsi *UploadImage()* dengan gambar pertama.	Fungsi *UploadImage* akan selesai, dan Fungsi UploadImage akan terpanggil kembali dengan gambar selanjutnya.	Satu gambar terupload dan fungsi terpanggil kembali dengan gambar yang berbeda.	*Valid*
4	1-2-3-9-10-2-11	Memanggil fungsi *UploadImage()* dengan kondisi tidak terhubung ke internet.	Akan tampil pesan *error.*	Tampil pesan *error.*	*Valid*

6.1.3 Pengujian Unit Fungsi *Login*

Fungsi *Login* memiliki tujuan agar *Guest* dapat masuk kedalam aplikasi dan mendapatkan otorisasi sebagai admin maupun member, Fungsi ini berada di kelas *Login*. Fungsi ini hanya dapat dieksekusi secara tidak langsung oleh pengguna. Psuedocode fungsi *Login* dapat dilihat pada tabel dibawah ini.

1. Pseudocode fungsi Login.

Tabel 6.5 Pseudocode Fungsi *doLogin*

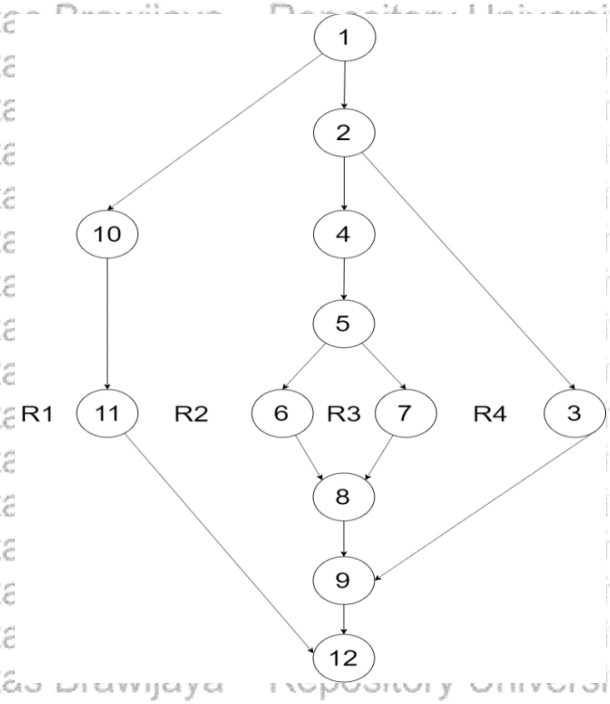
No	Kode	Node
1	`TRY, Then`	1
2	`execute FirebaseAuth(email,password)`	2
3	`IF (email user not verified) THEN`	2
4	`send alert to screen "email not verified"`	3
5	`break`	3
6	`ELSE`	4

7	initialize variable user = object var { auth: { id: result.user.uid, role:'user',shipping:[] } }	4
8	execute getProfile PROMISE(user)	4
9	execute AsyncStorage.setItem('auth') PROMISE, THEN	4
10	initialize variable result = parse(AsyncStorage.getItem('auth'))	4
11	IF (global.userRole='user') THEN	5
12	Navigate to main page	6
13	ELSE	7
14	Navigate to Admin Page	7
15	END IF	8
16	END IF	9
17	CATCH	10
18	send error message to screen	10
19	END CATCH	11
20	END TRY	12

2. Basis Path Testing

a. Flow Graph

Hasil *flow graph* yang didapatkan dari *pseudocode* 6.3 digambarkan pada Gambar dibawah ini.



Gambar 6.3 *Flow Graph* fungsi *doLogin*

b. Cyclomatic Complexity

- V(G) = jumlah *region* = 4
- V(G) = (jumlah *edge* - jumlah *node*) + 2 = (14 – 12) + 2 = 4
- V(G) = jumlah *predicate node* + 1 = 3 + 1 = 4

3. Test Case

Hasil dari *Basis path testing* didapatkan 4 *independent path*. Masing-masing *path* merupakan alur kode program yang mungkin terjadi. Dari jumlah alur kode program yang mungkin terjadi terdapat beberapa hasil pengujian yang dapat dilihat pada Tabel 6.6.

Tabel 6.6 Hasil Pengujian Unit fungsi *doLogin*

No	*Path*	*Test Case*	*Expected Result*	*Result*	*Status*
1	1-2-3-9-12	Memberi nilai "aa@a.a" (tidak ada pada colDB) pada var email lalu var pass "aaaaaaaa" lalu memanggil fungsi *doLogin()*	Pesan email tidak terdaftar akan tampil.	Tampil Pesan email tidak terdaftar.	*Valid*
2	1-10-11-12	Memberi nilai "aaa.a" pada var email lalu var pass "aaaaaaaa" lalu memanggil fungsi *doLogin()*	Pesan format email salah akan tampil.	Tampil Pesan format email salah	*Valid*
3	1-2-4-5-6-8-9-12	Memberi nilai "anfield1@kbscertification.co.id" pada var email lalu var pass "Qwerty" (email terverifikasi & password cocok pada *colDB*) lalu memanggil fungsi *doLogin()*	Halaman utama akan tampil.	Tampil halaman utama.	*Valid*
4	1-2-4-5-7-8-9-12	Memberi nilai "anfield1@kbscertification.co.id" pada var email lalu var pass "Qwerty" (email terverifikasi & password cocok) lalu memanggil fungsi *doLogin()*	Halaman admin akan tampil.	Tampil halaman admin.	*Valid*

6.2 Pengujian Validasi

Pengujian Validasi dilakukan untuk menguji fungsi-fungsi dari kebutuhan yang terdapat pada aplikasi. Pengujian ini dilakukan untuk memastikan bahwa setiap fungsi telah memenuhi kebutuhan yang telah didefinisikan. Pada penelitian ini pengujian validasi dilakukan dengan menggunakan metode *black-box.* Dari pengujian *black-box* diharapkan akan mendapatkan kemungkinan hasil yang diperoleh dari input pada kebutuhan yang sudah didefinisikan.

Tabel 6.7 Pengujian Validasi

No	*Feature*	*Test Case*	*Expected Result*	*Result*	*Status*
1	*Register*	User Mengisi seluruh form dengan sesuai lalu menekan tombol register	Aktor harus mampu daftarkan dirinya ke sistem.	Pada aplikasi tampil dialog register sukes dan aktor terdaftar sebagai member	*Valid*
2	Verifikasi Email	User Menekan tombol register.	Sistem harus dapat mengirimkan url verifikasi ke email aktor	Aktor berhasil melakukan verifikasi email	*Valid*
3	Reset kata sandi	User mengisi fiel email dan menekan tombol reset password.	Pengguna dapat melakukan Reset kata sandi	Pada aplikasi tampil dialog sukses dan kata sandi berhasil di *update*	*Valid*
4	*Login*	User mengisi seluruh form login dengan data yang benar lalu menekan tombol login.	Pengguna dapat masuk ke sistem	Pada aplikasi tampil halaman utama yang berarti aktor sudah masuk ke sistem	*Valid*
5	*Logout*	User menekan tombol logout.	Pengguna harus mampu mengeluarkan informasi	Pada aplikasi tampil login yang berarti aktor sudah keluar sistem	*Valid*

			akun dari sistem		
6	Melihat Profil	User memilih menu profil	Aktor dapat melihat profil yang berisikan informasi diri member	Pada aplikasi tampil halaman profil yang berisikan informasi terkait data diri member	*Valid*
7	Mengubah profil member	User mengubah salah satu field lalu menekan tombol selesai.	Aktor dapat mengubah infromasi diri dari member	Pada aplikasi tampil *dialog* sukses menyimpan perubahan dan informasi akan di perbarui di *database*	*Valid*
8	Melihat list member	Admin memilih menu list member	list *member* akan tampil	list *member* tampil	*Valid*
9	Menghapus Member	User memilih salah satu list dari *member* lalu menekan tombol delete	*Member* yang dipilih akan terhapus.	*Member* yang dipilih terhapus.	*Valid*
10	Mengubah status member	User mengubah field status, lalu menekan tombol ubah.	Status *member* yang dipilih akan berubah.	Status *member* yang dipilih berubah.	*Valid*
11	Tambah Alamat	User form dengan benar lalu menekan tombol ubah	Alamat *member* akan bertambah.	Alamat *member* bertambah.	*Valid*
12	Lihat Alamat	User memilih menu kelola alamat.	List Alamat akan tampil.	List Alamat tampil.	*Valid*
13	Edit Alamat	User memilih salah satu alamat, lalu menekan tombol edit,	Informasi alamat akan berubah.	Informasi alamat berubah.	*Valid*

		setelah itu mengubah field alamat dan menekan tombol edit.			
14	Hapus Alamat	User memilih salah satu alamat, lalu menekan tombol delete, setelah itu menekan tombol yes.	Alamat yang dipilih akan terhapus.	Alamat yang dipilih terhapus.	*Valid*
15	Tambah kategori	Admin memilih menu kelola kategori lalu memilih salah satu jenis kategori setelah itu mengisi field dan menekan tombol selesai.	Kategori produk yang dipilih dan inputannya akan bertambah.	Kategori produk yang dipilih dan inputannya bertambah.	*Valid*
16	Melihat *List* Kategori	Admin memilih menu kelola kategori lalu memilih salah satu jenis kategori.	List kategori akan tampil.	List kategori tampil.	*Valid*
17	*Edit* kategori	Admin memilih menu kelola kategori lalu memilih salah satu jenis kategori, setelah itu mengubah field yang ada dan menekan tombol selesai.	Kategori yang dipilih akan berubah.	Kategori yang dipilih berubah.	*Valid*
18	Hapus kategori	Admin memilih menu kelola kategori	Kategori yang dipilih akan terhapus.	Kategori yang dipilih terhapus.	*Valid*

		lalu memilih salah satu jenis kategori, setelah itu menekan icon sampah dan menekan tombol yes.			
19	Tambah Produk	User memilih menu tambah produk lalu, kelola gambar untuk produk setelah itu mengisi seluruh field dengan format yang tepat, setelah itu menekan tombol simpan.	Data produk akan bertambah.	Data produk bertambah.	*Valid*
20	*Manage product image*	User menekan tombol kelola gambar pada halaman edit produk lalu, user menambahkan satu gambar dan menghapus gambar yang lain lalu, menekan tombol selesai.	Gambar yang di pilih untuk dihapus akan terhapus dan gambar yang dipilih akan bertambah.	Gambar yang di pilih untuk dihapus akan terhapus dan gambar yang dipilih bertambah.	*Valid*
21	Tambah Gambar	User menekan tombol kelola gambar pada halaman edit produk lalu, user menambahkan satu gambar	Gambar akan bertambah 1.	Gambar bertambah 1.	*Valid*

		dan menekan tombol simpan.			
22	Hapus Gambar	User menekan tombol kelola gambar pada halaman edit produk lalu, user memilih satu gambar dan menekan tombol ya untuk hapus.	Gambar yang dipilih akan terhapus.	Gambar yang dipilih terhapus.	*Valid*
23	Melihat List Produk	User memilih menu kelola produk.	List produk akan tampil.	List produk tampil.	*Valid*
24	Mengubah informasi Produk	User memilih menu kelola produk dan memilih salah satu produk setelah itu menekan tombol edit, mengubah salah satu field dan menekan tombol simpan.	Informasi produk yang dipilih akan berubah sesuai dengan isian.	Informasi produk yang dipilih berubah sesuai dengan isian.	*Valid*
25	Menghapus produk	User memilih menu kelola produk dan memilih salah satu produk setelah itu menekan tombol hapus, dan menekan tombol ya.	Produk yang dipilih akan terhapus.	Produk yang dipilih terhapus.	*Valid*
26	Melihat detail Produk	Pada halam utama user memilih salah satu produk.	Detail produk akan tampil.	Detail produk tampil.	*Valid*

27	Mengubah status Produk	User mengubah status pada field status produk lalu menekan tombol simpan.	Produk yang dipilih akan tidak aktif.	Produk yang dipilih tidak aktif.	*Valid*
28	Cari produk	Pada halaman utama *user* mengisi kolom search lalu menekan ikon kaca pembesar pada *keyboard.*	Produk dengan keyword yang di inputkan akan tampil.	Produk dengan keyword yang di inputkan tampil.	*Valid*
29	*Filter* Kategori	Pada halaman utama user memilih salah satu menu kategori.	Produk dengan kategori yang dipilih akan tampil.	Produk dengan kategori yang dipilih tampil.	*Valid*
30	*Filter* harga	Pada menu filter harga user mengisi field lebih dari dan kurang dari dengan nilai yang rasional.	Produk dengan range harga yang di atur akan tampil.	Produk dengan range harga yang di atur tampil.	*Valid*
31	Menambahkan barang ke keranjang	User memilih 1 produk pada menu utama, lalu pada menu detail produk memilih tombol tambah ke keranjang	Produk yang di pilih akan masuk pada keranjang.	Produk yang di pilih masuk pada keranjang.	*Valid*
32	Melihat Keranjang	User menekan icon keranjang pada footer menu.	Produk yang ada dalam keranjang akan tampil.	Produk yang ada dalam keranjang tampil.	*Valid*

33	Mengubah Jumlah Barang pada Keranjang	Pada halaman keranjang, user menekan ikon plus dua kali dan menekan tombol minus satu kali.	Kuantitas pada barang yang dipilih akan bertambah 2 lalu berkurang 1, maka kuantitas akhir akan bertambah 1 dari kuantitas sebelumnya.	Kuantitas pada barang yang dipilih bertambah 2 lalu berkurang 1, maka kuantitas akhir akan bertambah 1 dari kuantitas sebelumnya.	*Valid*
34	Menghapus barang pada keranjang	Pada halaman keranjang, *user* menekan salah satu *icon* keranjang sampah.	Barang yang dipilih akan terhapus.	Barang yang dipilih terhapus.	*Valid*
35	Order	Pada halaman keranjang user menekan tombol order dan mengisi seluruh field dengan sesuai lalu menekan tombol order.	Order akan terbuat.	Order terbuat.	*Valid*
36	Bayar	Pada halaman *detail order* yang berstatus belum dibayar *user* menekan tombol bayar, pada web simulasi *midtrans user* mengisi no virtual account dan menekan tombol *pay*.	Status pembayaran akan berubah menjadi sudah di bayar.	Status pembayaran berubah menjadi sudah di bayar.	*Valid*

37	Melihat Daftar Belanja	Pada menu Akun pembeli, user memilih menu daftar belanja.	Daftar belanja akan tampil.	Daftar belanja tampil.	*Valid*
38	Melihat Daftar Pesanan	Pada menu Akun *merchant*, *user* memilih menu daftar belanja.	Daftar pesanan yang masuk ke *merchant* akan tampil.	Daftar pesanan yang masuk ke *merchant* tampil.	*Valid*
39	Melihat detail Pesanan	Pada menu Akun *merchant*, *user* memilih menu daftar belanja. Setalah itu user memilih salah satu pesanan yang ada.	Detail pesanan akan tampil.	Detail pesanan tampil.	*Valid*
40	Input Resi	Pada halaman pesanan yang berstatus sudah dibaya & belum dikirim, user menekan tombol input resi dan mengisi field tersebut. Setelah itu menekan tombol selesai.	Status pesanan akan berubah menjadi sedang dikirim dan no resi akan tersimpan.	Status pesanan berubah menjadi sedang dikirim dan no resi akan tersimpan.	*Valid*
41	Verifikasi Pengiriman	Pada daftar belanja user memilih satu daftar yang berstatus sedang dikirim, padah halaman selanjutnya user menekan	Status pengiriman akan berubah menjadi sudah diterima.	Status pengiriman berubah menjadi sudah diterima.	*Valid*

		tombol sudah sampai.			
42	Unduh *Invoice*	Pada halaman detail order user menekan tombol download pada field invoice.	File invoice akan tersimpan pada device.	File invoice tersimpan pada device.	*Valid*
43	Melihat *sell report*	Pada menu merchant user memilih menu *sell report*. Setelah itu mengganti tanggal dengan rasional.	Laporan penjualan akan tampil.	Laporan penjualan tampil.	*Valid*

6.3 Pengujian *Usability*

Pengujian *usability* dilakukan dengan cara memberikan aplikasi yang akan diuji coba kepada beberapa responden yang dipilih secara acak untuk mencoba menggunakan aplikasi tersebut. Kualitas dari *usability* aplikasi dinilai menggunakan *post-study system usability questionnaire* (PSSUQ). Pada PSSUQ ada aspek-aspek yang akan diuji yaitu kualitas dari sistem, kualitas antarmuka, kualitas informasi pada sistem dan kualitas aplikasi secara keseluruhan.

6.3.1 Prosedur Pengujian

Pengujian *usability* berlangsung dengan beberapa tahapan.Tahapan pertama yaitu memberikan kuesioner kepada 15 responden yang dipilih secara acak. Responden - responden tersebut akan terlebih dahulu mencoba aplikasi pada kumpulan aktifitas jual & beli (member). sebelum memberikan nilai pada kuesioner. Pada penilaian *usability* aplikasi dilakukan dengan metode *PSSUQ*. Pada metode tesebut terdapat 16 buah pertanyaan yang mewakili 3 aspek penilaian. Jawaban dari pertanyaan berupa skala 1-7 dimana semakin kecil yang diberikan oleh responden maka responden dianggap tidak setuju dengan penyataan yang diajukan dan begitu pula sebaliknya.

6.3.2 Hasil Pengujian

Setelah seluruh kuesioner sudah diisi oleh 15 responden yang mencoba aplikasi dengan *role member* maka respon yang diberikan akan dianalisis berdasarkan 3 aspek dan juga secara keseluruhan yang dijadikan dasar penilaian.

Pada Tabel 6.8 nilai di presentasikan berdasarkan pertanyaan dan jumlah resonden yang memberikan berdasarkan nilai 1-7.

Pada Tabel 6.9 data diolah menjadi rata-rata nilai berdasarkan kategori penilaian yaitu Kegunaan aplikasi (*SysUse*), kualitas informasi (*InfoQual*), kualitas antarmuka (*InterQual*) dan kepuasan secara kesuluruhan (*Overall*). Rata-rata diambil dari tiap-tiap responden. Kemudian secara keseluruhan diambil nilai rata-rata dari seluruh responden.

Tabel 6.8 Hasil Kuesioner terhadap pengguna

No	Pertanyaan	1	2	3	4	5	6	7
1	Secara Keseluruhan, Saya puas karena dapat menggunakan aplikasi anfield collection dengan mudah.	0	0	0	0	2	9	7
2	Aplikasi anfield collection sederhana untuk digunakan.	0	0	0	0	3	5	7
3	Sangat mudah untuk membuat iklan atau Sangat mudah untuk mencari barang pada aplikasi anfield collection.	0	0	0	2	3	8	2
4	Saya sangat nyaman untuk menggunakan aplikasi ini.	0	0	0	2	3	7	3
5	Aplikasi anfield collection mudah untuk dipelajari.	0	0	0	0	2	7	6
6	Saya yakin dapat menjual atau membeli barang dengan menggunakan aplikasi ini.	0	0	0	4	3	7	5
7	Aplikasi dapat menampilkan pesan & letak kesalahan ketika saya keliru saat menggunakan aplikasi ini.	0	0	0	4	5	5	5
8	Saat saya melakukan kesalahan pada aplikasi, saya dapat cepat mengkoreksi.	0	0	2	2	4	6	3
9	Informasi yang ditampilkan pada aplikasi sangat jelas.	0	0	1	0	2	10	2

10	Sangat mudah menemukan apa yang saya cari ketika menggunakan aplikasi anfield collection.pembelian	0	1	0	1	2	7	4
11	Informasi yang diberikan sangat efektif untuk membantu saya dalam menggunakan aplikasi ini.	0	0	0	1	6	6	2
12	Letak informasi berada di tempat yang tepat.	0	0	1	1	4	6	3
13	Antarmuka pada aplikasi ini sangan ramah penggunaanya.	0	0	1	1	4	9	0
14	Saya suka antarmuka aplikasi ini.	0	0	0	1	4	8	2
15	Aplikasi ini memiliki fitur sesuai dengan ekspektasi saya.	0	1	0	0	5	4	5
16	Secara keseluruhan, saya puas selama menggunakan aplikasi anfield collection.	0	0	0	0	1	12	2

Tabel 6.9 Rata-rata Nilai Per Jenis Tanggapan PSSUQ

No Responden	Jenis Tanggapan PSSUQ			
	Overall	SysUse	Infoqual	Interqual
1	6.00	5.83	6.17	6.00
2	6.67	7.00	6.50	6.50
3	5.67	6.00	5.50	5.50
4	5.47	5.67	5.50	5.25
5	6.78	6.83	7.00	6.50
6	5.69	5.83	6.00	5.25
7	5.86	6.00	5.83	5.75
8	6.00	6.00	6.00	6.00
9	6.75	7.00	7.00	6.25
10	5.50	5.50	5.50	5.50
11	4.47	5.50	4.17	3.75
12	5.53	5.50	4.83	6.25
13	5.81	6.50	5.17	5.75
14	5.67	5.83	5.17	6.00
15	5.64	5.50	5.67	5.75
Rata-rata	5.83	6.03	5.73	5.73

Tabel 6.10 Hasil Akhir Penilaian *Usability* Aplikasi

Aspek Penilaian	Total Nilai
Kegunaan Aplikasi	0.82
Kualitas Informasi	0.81
Kualitas Antarmuka	0.86
Keseluruhan	0.83

Setelah nilai rata-rata yang telah diambil, lalu data di olah menjadi standar penilaian pada *PSSUQ* yaitu range 1-7 menjadi range 0-1 untuk rata-rata keseluruhan. Data yang telah diolah dapat dilihat pada Tabel 6.10.

Dari total nilai yang telah di dapat, lalu disimpulkan berdasarkan Kegunaan Aplikasi, Kualitas Informasi, Kualitas Antarmuka, dan secara keseluruhan, yaitu:

1. Aspek kegunaan aplikasi mendapatkan penilaian yang memuaskan dari pengguna dengan nilai 0.82. Namun nilai tersebut menjadi yang paling rendah dibandingkan aspek yang lainnya.

2. Aspek kualitas informasi mendapat nilai 0.81. Nilai tersebut menjadi nilai yang paling rendah relatif sama seperti aspek kualitas informasi. Namun nilai ini adalah nilai yang baik /memuaskan untuk pengguna.
3. Aspek antarmuka menjadi aspek yang mendapatkan nilai paling baik dibandingkan yang lainya. Aspek memiliki nilai 0.86 nilai ini masuk kategori sangat memuaskan.
4. Secara keseluruhan nilai diambil dari rata-rata 3 aspek, didapatkan nilai 0.83, yaitu nilai yang sangat memuaskan.

Dapat disimpulkan dari berbagai aspek dari nilai yang diberikan oleh pengguna relatif sama yaitu nilai yang memuaskan. Dengan begitu aplikasi yang telah dibuat sudah dianggap layak oleh pengguna secara umum untuk role *member*.

6.4 Pengujian *Compatibility*

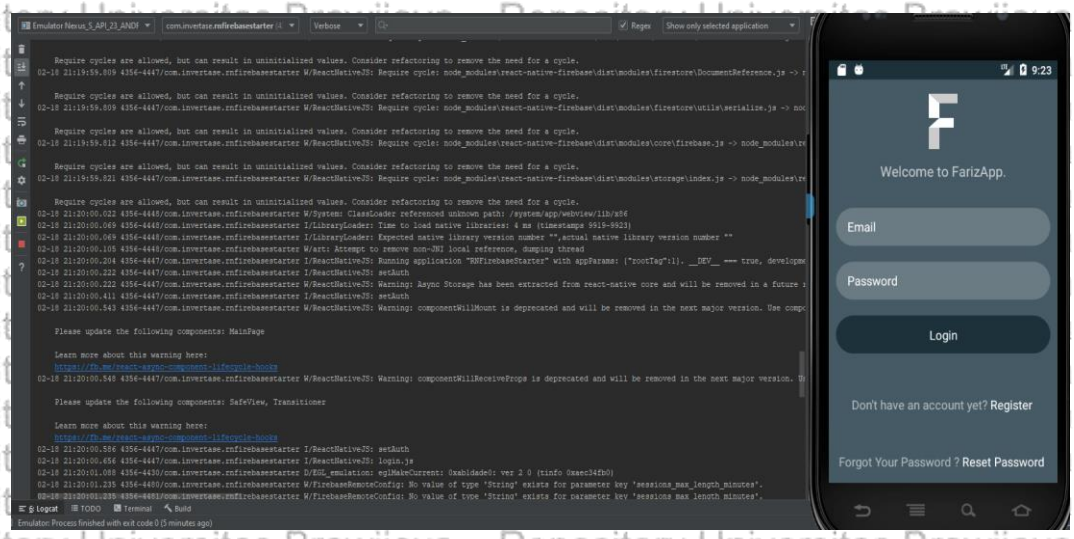
Sesuai dengan kebutuhan non fungsional yang di deklarasikan maka kompabilitas untuk aplikasi yang telah dibuat harus dapat dijalankan pada *android* 6 (*Marshmallow*). Versi Sistem Operasi Android terbaru yang tersedia untuk public saat ini (18 februari 2020) adalah versi 10 (*Beta-Public*).

Pengujian dilakukan dengan cara menjalankan fungsi login sebagai sample pada E*mulator Android Virtual Devices* dengan OS 6, 7, 8, 9, 10 dan juga 4 macam jenis *Mobile Devices* yang berbeda, karena seluruh *library* dan *devDependencies* di dalam paket sudah di definisikan dengan *min OS target value* 6. Saat melakukan perintah variant *running debug* maka *script* untuk menguji seluruh *dependency* dilakukan secara *default* sesuai dengan target *os* pada *debug device*. Jadi cukup melakukan instalasi pada target seluruh komponen untuk menjalankan aplikasi sudah di cek oleh *default script debug react native*.

Berikut merupakan hasil pengujian *Compability* dengan menjalankan aplikasi pada *android virtual devices* dengan perintah *"react-native run-android variant=debug"*.

6.4.1 Pengujian Pada OS Android 6 (*Marshmellow*)

Pengujian dilakukan dengan menjalankan *Emulator Android Virtual Devices "Emulator Nexus_S_API_23_ANDROID6"* yang telah dibuat lalu di folder development menjalankan *CMD* dengan perintah *"react-native run-android variant=debug". Logcat* untuk Emulator dan log pada *development terimal* juga terus dimonitor sampai aplikasi selesai dijalankan. Hasil tangkapan gambar pada pengujian ini dapat dilihat pada gambar dibawah.

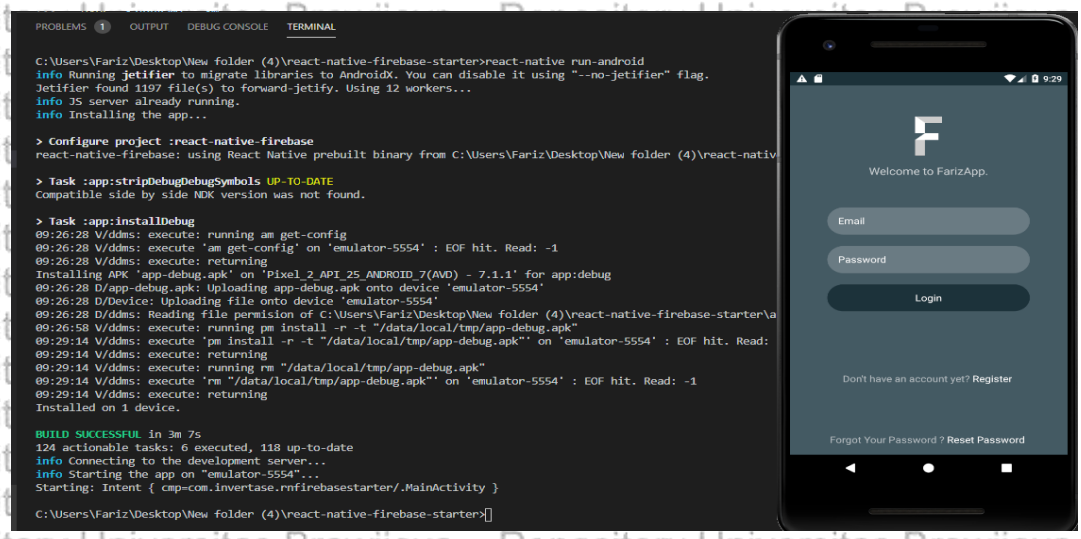


Gambar 6.4 Pengujian Aplikasi pada *OS Android 6*

Dari hasil pengujian kompabilitas pada *android* 6 dengan memperhatikan *log* & *terminal* untuk *emulator* dan *debug* dapat disimpulkan aplikasi yang dibuat kompatibel dengan versi *android* 6.

6.4.2 Pengujian Pada OS Android 7 (*Nougat*)

Pengujian dilakukan dengan menjalankan *Emulator Android Virtual Devices "Emulator PIXEL_2_API_25_ANDROID7_AVD* yang telah dibuat lalu di folder development menjalankan CMD dengan perintah *"react-native run-android variant=debug". Logcat* untuk Emulator dan log pada *development terimal* juga terus dimonitor sampai aplikasi selesai dijalankan. Hasil tangkapan gambar pada pengujian ini dapat dilihat pada gambar dibawah.

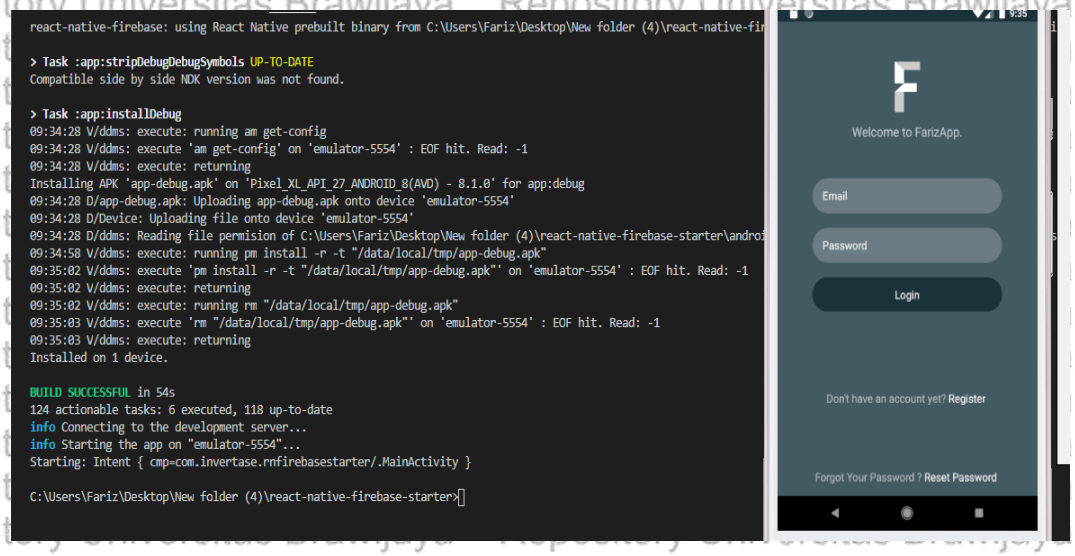


Gambar 6.5 Pengujian Aplikasi pada *OS Android* 7

Dari hasil pengujian kompabilitas pada *android* 7 dengan memperhatikan *log* & *terminal* untuk *emulator* dan *debug* dapat disimpulkan aplikasi yang dibuat kompatibel dengan versi *android* 7.

6.4.3 Pengujian Pada OS Android 8 (*Oreo*)

Pengujian dilakukan dengan menjalankan *Emulator Android Virtual Devices "Emulator PIXEL_XL_API_27_ANDROID_8_AVD* yang telah dibuat lalu di folder development menjalankan CMD dengan perintah *"react-native run-android variant=debug"*. *Logcat* untuk Emulator dan log pada *development terimal* juga terus dimonitor sampai aplikasi selesai dijalankan. Hasil tangkapan gambar pada pengujian ini dapat dilihat pada gambar dibawah.

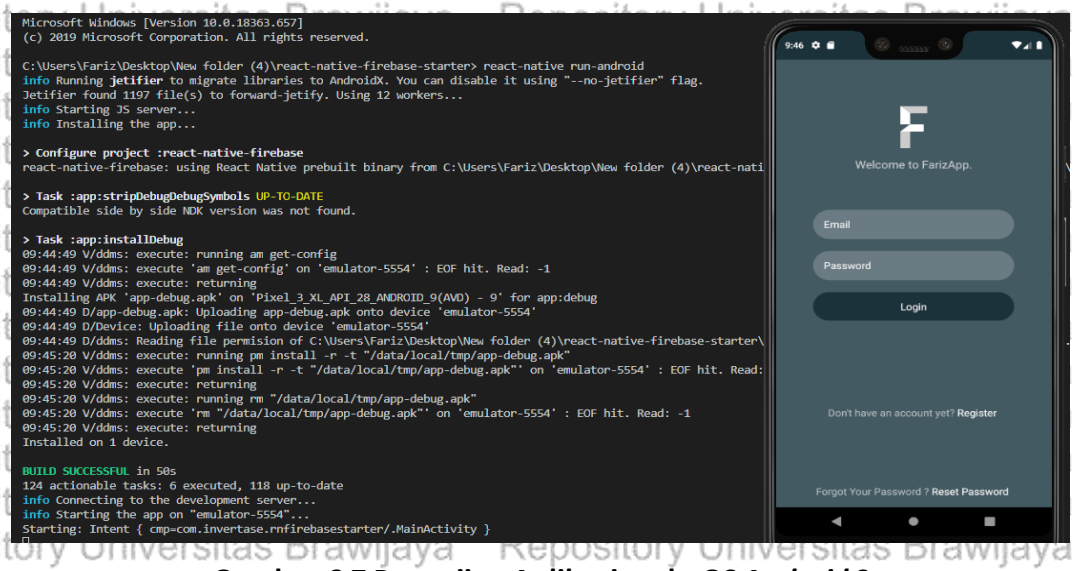


Gambar 6.6 Pengujian Aplikasi pada *OS Android* 8

Dari hasil pengujian kompabilitas pada *android* 8 dengan memperhatikan *log* & *terminal* untuk *emulator* dan *debug* dapat disimpulkan aplikasi yang dibuat kompatibel dengan versi *android* 8.

6.4.4 Pengujian Pada OS Android 9 (*Pie*)

Pengujian dilakukan dengan menjalankan *Emulator Android Virtual Devices "Emulator PIXEL_3_XL_API_28_ANDROID_9(AVD) – 9"* yang telah dibuat lalu di folder development menjalankan CMD dengan perintah *"react-native run-android variant=debug"*. *Logcat* untuk Emulator dan log pada *development terimal* juga terus dimonitor sampai aplikasi selesai dijalankan. Hasil tangkapan gambar pada pengujian ini dapat dilihat pada gambar berikut.

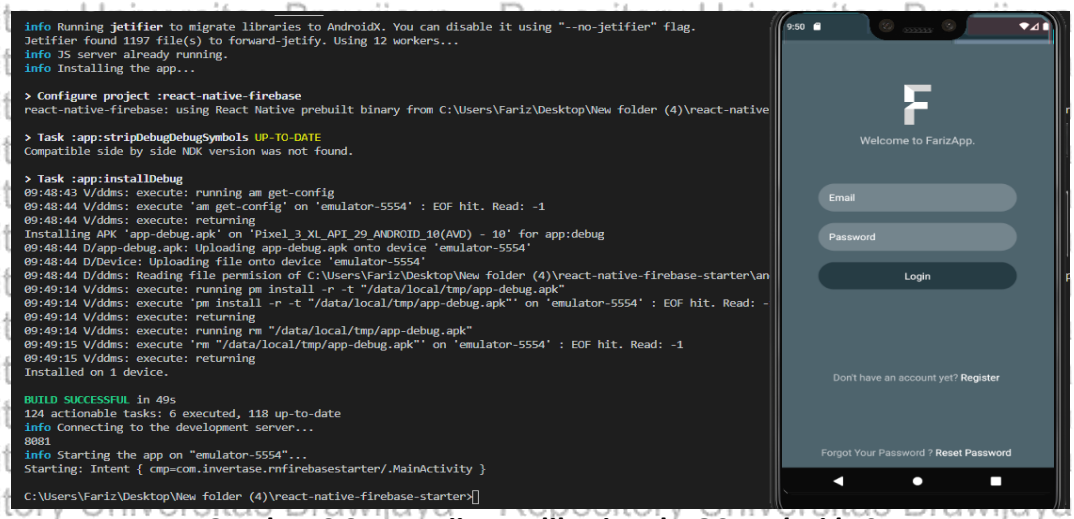


Gambar 6.7 Pengujian Aplikasi pada *OS Android* 9

Dari hasil pengujian kompabilitas pada *android* 9 dengan memperhatikan *log* & *terminal* untuk *emulator* dan *debug* dapat disimpulkan aplikasi yang dibuat kompatibel dengan versi *android* 9.

6.4.5 Pengujian Pada OS Android 10

Pengujian dilakukan dengan menjalankan *Emulator Android Virtual Devices "Emulator PIXEL_3_XL_API_29_ANDROID_10(AVD) – 10"* yang telah dibuat lalu di folder development menjalankan CMD dengan perintah *"react-native run-android variant=debug"*. *Logcat* untuk Emulator dan log pada *development terimal* juga terus dimonitor sampai aplikasi selesai dijalankan. Hasil tangkapan gambar pada pengujian ini dapat dilihat pada gambah dibawah.



Gambar 6.8 Pengujian Aplikasi pada *OS Android* 10

Dari hasil pengujian kompabilitas pada *android* 10 dengan memperhatikan *log* & *terminal* untuk *emulator* dan *debug* dapat disimpulkan aplikasi yang dibuat kompatibel dengan versi *android* 10.

BAB 7 KESIMPULAN

7.1 Kesimpulan

Kesimpulan yang diambil merupakan hasil dari seluruh tahapan penelitian yang sudah dilakukan dan merupakan jawaban dari rumusan masalah yang didefinisikan. Berikut merupakan kesimpulan yang diambil:

1. Pada tahap analisis kebutuhan yang dilakukan dengan cara wawancara dan observasi dapat di ambil dan dikembangkan menjadi 43 kebutuhan fungsional dan 1 kebutuhan non fungsional. Aktor yang dapat dikaitkan adalah *member* dan *administrator (admin).* Yang semua digambarkan menjadi *usecase diagram* dan *use case scenario.*
2. Hasil dari proses perancangan menjadi dalam bentuk *Class Diagram, Sequence Diagram, Entity Relation Diagram (ERD),* perancangan komponen dan perancangan antarmuka. *Class diagram* Digambar dengan kelas-kelas kosong tetapi ada 5 kelas yang di deskripsikan sebagai sampel. Perancangan Sequence juga dibuatkan 5 sebagai sample. Kemudian *ERD* dibuat terdiri dari 5 *collection*. Komponen diagram terdapat 3 algoritma yang dirancang. Terakhir perancangan antarmuka dibuat 11 macam halaman.
3. Di tahap implementasi, kode yang dibuat menggunakan framework NodeJs pada *server* dan *React Native* pada *mobile*. Terdapat 4 *API* yang digunakan untuk simplifikasi fitur yang dibuat yaitu, *Firebase Authentication* untuk managemen akun, *Firebase Storage* untuk penyimpanan media, Raja ongkir untuk daftar kurir dan perhitungan ongkos kirim, *Invoice-Generator* untuk membuat invoice secara otomatis, Midtrans sebagai gerbang pembayaran yang lengkap.
4. Tahap pengujian sistem dilakukan dengan 4 macam, yaitu Pengujian unit, pengujian validasi, pengujian *Usability* dan pengujian *compability*. Pengujian unit hanya dilakukan dengan 3 sampel sesuai dari perancangan algoritme. Kemudian pengujian validasi dilakukan dengan seluruh daftar kebutuhan fungsional yang terdapat 43 kebutuhan. Pengujian *Usability* mendapatkan hasil memuaskan dari rata-rata pengguna untuk *role member*. Terakhir pengujian compability, kebutuhannya adalah aplikasi dapat digunakan di *android* 6 keatas, maka pengujian dilakukan dengan menggunakan secara random pada *android* 6, 7, 8, 9 dan *android* terakhir saat ini yaitu *android* 10 dengan *Android Virtual Devices (AVD)* yang tersedia pada *android studio*.

7.2 Saran

Berdasarkan pengalaman dan hasil penilitan yang telah dibuat maka penulis membuat beberapa saran perbaikan & ide untuk penelitian selanjutnya yaitu:

1. Penambahan fitur agar aplikasi yang dibuat menjadi lebih baik lagi yaitu fitur notifikasi secara *realtime*, Rekomendasi produk untuk pembeli dengan cara mempelajari kebiasaan pengguna yang dilakukan di dalam aplikasi. Terakhir

adalah fitur dengan kemudahan login menggunakan akun *google, facebook atau token SMS* demi mengikuti perkembangan zaman.

2. Pengembangan aplikasi menggunakan paradigma functional programming dengan *react native*.
3. Pada pengimplementasian *API Payment Gateway Midtrans* penulis masih menggunakan *Sandbox* sebagai simulasi pembayaran karena beberapa keterbatasan. Apabila penelitian selanjutnya ingin membuat aplikasi lebih akurat yaitu dengan melakukan pengujian / percobaan dengan pembayaran yang sungguhan menggunakan beberapa jenis pembayaran yang disediakan oleh *midtrans.*
4. Mengintegrasikan aplikasi dengan layanan yang menyediakan *API* lacak resi, agar pembeli dan penjual dapat melacak resi dengan keadaaan yang sesungguhnya.
5. Evaluasi dan pengukuran dampak pasca aplikasi diterapkan pada bisnis kerajinan.

DAFTAR PUSTAKA

Alshamrani, A., & Bahattab, A. (2015). A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model. *IJCSI International Journal of Computer Science Issues*, 106–111. Retrieved from:<https://www.academia.edu/10793943/A_Comparison_Between_Three_SDLC_Models_Waterfall_Model_Spiral_Model_and_Incremental_Iterative_Model>

Azizah, N., Mahendra, D., & Lofian, B. (2019). Pemanfaatan *E-Commerce* untuk Peningkatan Strategi Promosi dan Penjualan UMKM Tas di Kabupaten Kudus. *Jurnal Pengabdian kepada Masyarakat*, 96-99.

Bassil, Y. (2012). *A Simulation Model for the Waterfall Software Development Life Cycle*. *2*(5). Retrieved from http://arxiv.org/abs/1205.6904.

Bisnis.com, 2018. 75% Transaksi Online di Aplikasi Mobile. Retrieved December 15 2019. Tersedia di <https://ekonomi.bisnis.com/read/20181109/12/858146/75-transaksi-online-di-aplikasi-mobile>.

Bevan, N., 1995. *Usability AS Quality of Use. Software Quality Journal*, 4, pp.115–130.

Dierx, P. (2016). A Beginner's Guide to npm – the Node Package Manager.. Retrieved 22 July 2016. Tersedia di : <https://www.sitepoint.com/beginners- guide-node-package-manager/>.

Detik, 2019. Meningkatkan Ekspor Kerajinan Tangan. Retrieved January 22, 2020, Tersedia di : < *https://news.detik.com/kolom/d-4676540/meningkatkan-ekspor-kerajinan-tangan*>.

Domainesia, 2020. *We are DomaiNesia*. Retrieved March 3, 2020, Tersedia di : < *https://www.domainesia.com/about/*>.

Firebase Auth, 2020. *About Firebase Authentication*. Retrieved January 22, 2020, Tersedia di : < *https://firebase.google.com/docs/auth*>.

Firebase Storage, 2020. *About Firebase Cloud Storage*. Retrieved January 22, 2020, Tersedia di : < *https://firebase.google.com/docs/storage*>.

Flanangan, D. (2011). *JavaScript: The Definitive Guide* (edisi ke-6). O'Reilly & Associates.

IBM Knowledge Center, 2020. *Decision Center*. Retrieved January 9, 2020. Tersedia di : < https://www.ibm.com/support/knowledgecenter/SSQP76_8.7.1/com.ibm.odm.dcenter.overview/topics/odm_dcenter_overview.html

Imam, F., & Azhari, S. (2008). *Proses Pemodelan Software Dengan Metode Waterfall Dan Extreme Programming : Studi Perbandingan*.

Invoice-Generator, 2020. About Invoice-Generator. Retrieved March 6, 2020, Tersedia di : < *https://invoice-generator.com/help/>*.

Kurniawan, T. A. (2018). Pemodelan Use Case (UML): Evaluasi Terhadap beberapa Kesalahan dalam Praktik. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK), Vol. 5, No. 1,* 77-86.

Maulana, S. M., Susilo, H., & Riyadi. (2015). Implementasi *E-Commerce* Sebagai Media Penjualan Online. *Jurnal Administrasi Bisnis*, *29*(1), 1–9.

Mayangsari, R., & Ariesta, A. (2019). Penerapan *E-Commerce* untuk Meningkatkan Penjualan Pada Toko Tas Batam No.1. *Jurnal Idealis Vol.2 No.2*, 167-173.

Medium, 2020. *Belajar Transaksi Core API Midtrans Tanpa Menulis Satu Baris Kode*. Retrieved January 23, 2020, Tersedia di : < *https://medium.com/belajar-transaksi-core-api-midtrans-tanpa-menulis-satu-baris-kode-1adff30514bb/>*.

MnEducation, 2008. *Glossary of Key Terms.* Retrieved January 17, 2020. Tersedia di : < http://highered.mheducation.com/sites/0077110005>

Midtrans, 2020. *Kami adalah Midtrans*. Retrieved January 8, 2020, Tersedia di : < *https://www.midtrans.com/about*>.

MongoDB, 2020. *The database for modern applications*. Retrieved January 22, 2020, Tersedia di : < *https://www.mongodb.com/>*.

Nodejs org, 2020. *About NodeJs.* Retrieved January 19, 2020, Tersedia di : < https://nodejs.org/en/about/>

Nuraeni, N., & Astuti, P. (2019). Rancang Bangun Sistem Informasi Penjualan *Online* (*E-Commerce*) Pada Toko Batik Pekalongan Dengan Metode Waterfall. *Jurnal Teknik Komputer AMIK BSI*, 59-64.

Pressman, R. S., Maxim, B. R. 2015. Software Enginering A Practitioner's Approach (8th ed.). New York: McGraw-Hill.

Permatasari, R. D., Studi, P., Informatika, T., Aset, M., & Mobile, W. (n.d.). *Sistem Informasi Manajemen Aset dengan Metode SDLC (Software Development Life Sycle) (Studi Kasus STT Ibnu Sina Batam)*. *2*(04), 73–90.

RajaOngkir, 2020. Tentang RajaOngkir. Retrieved March 5, 2020, Tersedia di : < *https://rajaongkir.com/tentang/>*.

ReactJS, 2020. *Getting Started ReactJS*. Retrieved January 22, 2020, Tersedia di : < *https://reactjs.org/docs/getting-started.html*>

ReactNative, 2020. *React Native. Learn once, write anywhere*. Retrieved January 22, 2020, Tersedia di : < *https://reactnative.dev/>*

Santoso, & Hutahaean, J. (2018). Aplikasi Toko Buku *Online* Berbasis Mobile *E-Commerce*. *Seminar Nasional Royal (SENAR)*, *9986*(September), 339–344.

Sauro, J. and Lewis, J.R., 2012. *QUANTIFYING THE USER EXPERIENCE PRACTICAL STATTISTICS FOR USER RESEARCH. 1st ed.*

Sobey, A. J., Software Testing, http://www.erau.edu/research/veritas/pdf/SoftwareTesting.pdf, 17 Januari 2007

Sommerville, Ian. 2011. *Software Engineering* (Rekayasa Perangkat Lunak). Jakarta: Erlangga.

Study, 2019. *Organizational Adaptation Theory: Definition & Application*. Retrieved february 22, 2020, Tersedia di : < *https://study.com/academy/lesson/organizational-adaptation-theory-definition-application.html/>*

Sumanto, A. 2011. Pendidikan Senirupa di Sekolah Dasar. Malang: FIP UM.

Sumanto. 2015. Kerajinan Tangan di Blitar sebagai Sumber Belajar Seni Budaya dan Prakarya (SbdP) Sekolah Dasar. Jurnal Sekolah Dasar tahun 26 Nomor 2, Nov 2015. hal 121. Malang. Jurusan KSDP FIP UM.

Sulistyorini, P. 2009. Pemodelan Visual Dengan Menggunakan *UML* Dan *Rational Rose*. *Jurnal Teknologi DINAMIK, XIV(1)*, 23-29.

TutorialsPoint, 2017. *Learn SDLC*. Retrieved January 19, 2020. Tersedia di : < https://www.tutorialspoint.com/sdlc/>

Wienclaw, Ruth A. (2013) *"E-Commerce." Research Starters: Business.*

Nuraeni, N., & Astuti, P. (2019). Rancang Bangun Sistem Informasi Penjualan Online (E-Commerce) Pada Toko Batik Pekalongan Dengan Metode Waterfall. *Jurnal Teknik Komputer AMIK BSI*, 59-64.

Azizah, N., Mahendra, D., & Lofian, B. (2019). Pemanfaatan E-Commerce untuk Peningkatan Strategi Promosi dan Penjualan UMKM Tas di Kabupaten Kudus. *Jurnal Pengabdian kepada Masyarakat*, 96-99.

Cadle, J., Paul, D., & Turner, P. (2010). *Business Analysis Techniques: 72 Essential.* UK: BISL.

Hume, D. A. (2017). *Progressive Web Apps.* Manning.

Khamidah, K., & Triyono, R. A. (2013). Pengembangan Aplikasi E-learning Berbasis Web dengan PHP dan MySQL Studi Kasus SMPN 1 Arjosari. *Indonesian Jurnal on Networking and Security* , 1-7.

Kosasi, S. (2015). Perancangan E-learning untuk Meningkatkan Motivasi Belajar Guru dan Siswa. *Prosiding Seminar Nasional Pendidikan Teknik Informatika*, 82-87.

Kurniawan, B. (2008). *Desain Web Praktis dengan CSS.* Jakara: PT Elex Media Komputindo.

Kurniawan, T. A. (2018). Pemodelan Use Case (UML): Evaluasi Terhadap beberapa Kesalahan dalam Praktik. *Jurnal Teknologi Informasi dan Ilmu Komputer*, 77-86.

Lanusi , D. H. (2018). Penerapan Kelas Digital Edmodo Untuk Meningkatkan Minat Belajar Dan hasil Belajar Siswa . *Jurnal Dikdaktika Pendidikan Dasar*, 58-72.

Mayangsari, R., & Ariesta, A. (2019). Penerapan E-Commerce untuk Meningkatkan Penjualan Pada Toko Tas Batam No.1. *Jurnal IDEALIS Vol.2 No.2*, 167-173.

Pressman, R. S. (2015). *Rekayasa Perangkat Lunak: Pendakatan Praktisi Buku I.* Yogyakarta: Andi.

Sommerville, I. (2011). Software Engineering. In *Software Engineering 9th.* USA: Pearson Education.

Sulistyorini, P. (2009). Pemodelan Visual dengan Menggunakan UML dan Rational Rose. *Jurnal Teknologi Informasi DINAMIK* , 23-29.

What is Waterfall Model in SDLC? Advantages & Disadvantages. (n.d.). Retrieved from Guru99: https://www.guru99.com/what-is-sdlc-or-waterfall-model.html

Winarno, E., & Zaki, A. (2014). *Pemrograman Web Beerbasis HML5,PHP,dan JavaScript*. Jakarta: PT Elex Media Komputindo.

Yudistira Sugandi, B. P. (2018). Pengembangan Sistem Aplikasi Portal Informasi Perguruan Tinggi di Indonesia Berbasis Website.

LAMPIRAN A HASIL WAWANCARA

Informan 1

Tanggal wawancara : 3 Oktober 2019

Tempat waktu : Desa Ringin Anom, Pademangan.

Indentitas Informan 1

Nama : Ibu Hepy

Umur : 41 Tahun

Jenisn kelamin : Perempuan

Pekerjaan : Pengelola Toko Anfield *Collection*

Hasil Wawancara

1. Apa sih Anfield *Collection* itu Bu?

   Jawab:

   Anfield *Collection* itu seperti rumah kerajinan atau tempat produksi kecil salah satunya produk tas dari bahan dasar rot imitasi. Bahan tersebut kemudian dibuat pola, digunting dan di tempel di spon lalu dibentuk menjadi produk akhir.

2. Bagaimana proses pemasaran nya Bu?

   Jawab:

   Sejak awal kita bingung cara pemasaran nya bagaimana karna dari desa yang pelosok (paling ujung) dan kita memasarkan nya hanya dari mulut ke mulut atau titip gambar produk dan juga promosi lewat status di WA (*WhatsApp*).

3. Apa kendala dalam menjalankan usaha ini?

   Jawab:

   Banyak kendala dari awal mulai usaha, seperti:

   a. Tenaga kerja nya harus yang terapil jadi perlu dipilih-pilih
   b. Dalam mempromosikan barang kurang meluas dikarenakan ketinggalan dalam perkembangan teknologi juga termasuk jaringan sinyal yang agak sulit sehingga kita bingung untuk mempromosikannya
   c. Untuk daftar harga juga tidak tercantum sehingga masih menggunakan sistem tanya harga secara langsung

4. Bagaimana proses pencatatan pesanan dan sebagainya?

Jawab:

Masih manual, karena belum mengetahui teknologi dalam pemesanan menggunakan internet sehingga masih dicatat di kertas. Kemudian jika ada pemesanan kita harus datang ke rumah pembeli untuk menanyakan pesanan nya yang bagaimana dan kemudian masih menggunakan kwitansi juga.

5. Apa yang ibu harapkan kedepannya dalam membangun usaha ini?

Jawab:

Tentunya kita ingin mempunyai suatu aplikasi khusus Anfield *Collection* untuk mempermudah mempromosikan produk-produk kita dimana didalamnya sudah ada gambar produk yang disertai harga barangnya dan juga kita tidak perlu datang ke rumah-rumah lagi untuk menanyakan model pesanannya yang bagaimana.

LAMPIRAN B DATA PRODUK

Kategori	Nama Produk	Deskripsi	Ket.	Bahan	Kuantitas	Harga
Toko Wijaya, JL. Pasar PIIP Kota Blitar Blok IIIB Email: wijayatoko12@gmail.com						
Tas	Tas bulat	Tas yang cocok digunakan untuk pergi ke tempat wisata	Dijahit	Kulit	2	28k
	Tas kerang	Tas yang yang dibuat untuk menyimpang banyak barang	Diukir	Handmade	5	21k
	Tas jorum	Tas jorum sangat cocok digunakan untuk membawa keperluan banyak	Dijahit	Kulit imitasi	2	28k
	Tas oval	Tas oval cocok digunakan untuk keperluan belanja	Dijahit	Kulit imitasi	10	16k
	Tas kendi	Tas kendi cocok digunakan untuk keperluan pergi ke tempat-tempat wisata	Dijahit	Kulit imitasi	5	18k
	Tas modern kecil	Tas modern adalah tas yang didesain untuk anak muda	Dijahit	Kulit imiasi	3	15k
	Tas kepang kecil	Tas modern adalah tas yang didesain untuk anak muda	Dianyam	Kulit kayu	6	22k
	Tas love	Tas love adalah tas yang dibuat khusus untuk pesta-pesta keluarga	Dijahit	Kulit imitasi	5	16k
	Tas kombinasi	Tas kombinasi sangat cocok untuk keperluan multi guna	Dianyam	Kulit kayu	10	21k
	Tas make up	Tas makeup adalah tas praktis yang cocok digunakan untuk membawa keperluan make up anda	Dijahit	Kulit imitasi	5	35k
	Tas tali kur	Tas tali kur adalah tas yang di desain untuk anak muda ke pesta ulang tahunan	Dirajut	Benang khusus	1	25k

	Tas Ayaman	Tas ayaman adalah tas yang cocok digunakan untuk pergi ke tempat-tempat wisata	Dianyam	Kulit kayu	5	30k
	Tas enceng	Tas enceng adalah tas yang di desain untuk orang yang mempunyai barang yang minim	Diayam	Kulit kayu	2	16k
	Tas mendong	Tas mendong adalah tas yang didesain untuk orang yang banyak barang	Dianyam	Kulit kayu	3	23k
	Tas passport songket	Tas ini didesain untuk menyimpan semua kartu penting yang dimiliki	Di rajut	Benang khusus	1	21k
	Tas tabung	Tas tabung adalah tas yang di gunakan untuk menyimpang alat-alat tulis yang	Dijahit	Kulit imitasi	3	13k
	Tas ransel	Tas ransel adalah tas yang digunakan untuk anak sekolah masa kini	Handmade	Kulit khusus	2	50k
Souvenir	Dompet	Dompet yang cocok untuk dimiliki karena modelnya yang simple dan bervariatif	Dijahit	Kulit imitasi	3	15k
	Tempat tissue	Sebuah souvenir unik yang digunakan sebagai tempat tissue di ruang keluarga	Dijahit	Kulit khhusus	4	14k
	Tempat pensil	Suvenir yang cocok untuk menyimpan alat-alat tulis karena bentuknya yang unik	Dijahit	Kulit imitasi	2	12k
	Tempat bunga	Suvenir yaang cocok untuk hadiah ulang tahun perkawinan	Dijahit	Kulit imitasi	1	16k
	Gantung kunci	Suvenir yang cocok untuk sebagai cindera mata dari daerah tertentu	Diukir	Khusus	3	10k
	Sendal kuno	Sendal yang digunakan pada jaman kuno atau sendal tradisional daerah jawa	Dibor	Kayu	2	22k
	Topeng	Topeng kesenian yang biasa digunakan untuk hiasan dinding dan sebagai simbol kebudayaan daerah	Di ukir	Batok kelapa	1	32k
	Taplak meja	Banyak motif dan warna yang beranekaragam dan mempunyai berbagai jenis ukuran	Di rajut	Batok kelapa	5	15k

Toko Batok koi, JL. pasar PIIP Kota Blitar Email: wijayatoko12@gmail.com						
Tas	Tas kendi	Tas kendi cocok digunakan untuk keperluan pergi ke tempat-tempat wisata	Dijahit	Kulit imitasi	5	18k
	Tas modern kecil	Tas modern adalah tas yang didesain untuk anak muda	Dijahit	Kulit imiasi	3	15k
	Tas kepang kecil	Tas modern adalah tas yang didesain untuk anak muda	Dianyam	Kulit kayu	6	22k
	Tas love	Tas love adalah tas yang dibuat khusus untuk pesta-pesta keluarga	Dijahit	Kulit imitasi	5	16k
	Tas kombinasi	Tas kombinasi sangat cocok untuk keperluan multi guna	Dianyam	Kulit kayu	10	21k
	Tas make up	Tas makeup adalah tas praktis yang cocok digunakan untuk membawa keperluan make up anda	Dijahit	Kulit imitasi	5	35k
	Tas kondangan	Tas yang didesain untuk acara-acara keluarga	Dijahit	Kulit imitasi	6	22k
	Tas amplop kecil	Tas yang dibuat untuk keperluan kecil atau acara ulang tahun	Dijahit	Kulit imitasi	2	16k
	Tas amplop besar	Tas yang dibuat untuk keperluan kecil atau acara ulang tahun	Dijahit	Kulit imistasi	2	21k
	Tas batok	Tas batok adalah tas yang di gunakan untuk keperluan hiasan dirumah	Dibor dan diukir	Batok	1	25k
	Tas segi 6	Tas segi enam adalah sebuah tas yang didesain untuk ibu rumah tangga	Dijahit	Kulit imitasi	10	13k
	Tas wajik	Tas wajik adalah tas yang cocok digunakan untuk suvenir acara keluarga	Dijahit	Kulit imitasi	2	16k
	Tas slempang etnik	Tas slempang adalah tas yang di desain untuk anak muda melakukan traveling	Dijahit	Kulit imitasi	5	30k

	Tas passport songket	Tas ini didesain untuk menyimpan semua kartu penting yang dimiliki	Di rajut	Benang khusus	1	21k
	Tas tabung	Tas tabung adalah tas yang di gunakan untuk menyimpang alat-alat tulis yang	Dijahit	Kulit imitasi	3	13k
	Tas ransel	Tas ransel adalah tas yang digunakan untuk anak sekolah masa kini	Handmade	Kulit khusus	2	50k
Souvenir	Dompet	Dompet yang cocok untuk dimiliki karena modelnya yang simple dan bervariatif	Dijahit	Kulit imitasi	4	15k
	Tempat tissue	Sebuah souvenir unik yang digunakan sebagai tempat tissue di ruang keluarga	Dijahit	Kulit khhusus	2	15k
	Tempat pensil	Suvenir yang cocok untuk menyimpan alat-alat tulis karena bentuknya yang unik	Dijahit	Kulit imitasi	1	12k
	Tempat bunga	Suvenir yaang cocok untuk hadiah ulang tahun perkawinan	Dijahit	Kulit imitasi	3	5k
	Gantung kunci	Suvenir yang cocok untuk sebagai cindera mata dari daerah tertentu	Diukir	Khusus	2	10k
	Sendal kuno	Sendal yang digunakan pada jaman kuno atau sendal tradisional daerah jawa	Dibor	Kayu	1	12k
	Topeng	Topeng kesenian yang biasa digunakan untuk hiasan dinding dan sebagai simbol kebudayaan daerah	Di ukir	Batok kelapa	1	40k
Toko Ibu Sugiarto, JL. pasar PIIP Kota Blitar Email: tokosugiarto@gmail.com						
Tas	Tas bulat	Tas yang cocok digunakan untuk pergi ke tempat wisata	Dijahit	Kulit	2	28k
	Tas kepang kecil	Tas modern adalah tas yang didesain untuk anak muda	Dianyam	Kulit kayu	6	22k
	Tas love	Tas love adalah tas yang dibuat khusus untuk pesta-pesta keluarga	Dijahit	Kulit imitasi	5	16k

	Tas kombinasi	Tas kombinasi sangat cocok untuk keperluan multi guna	Dianyam	Kulit kayu	10	21k
	Tas make up	Tas makeup adalah tas praktis yang cocok digunakan untuk membawa keperluan make up anda	Dijahit	Kulit imitasi	5	35k
	Tas ayaman	Tas ayaman adalah tas yang cocok digunakan untuk pergi ke tempat-tempat wisata	Dianyam	Kulit kayu	5	30k
	Tas passport songket	Tas ini didesain untuk menyimpan semua kartu penting yang dimiliki	Di rajut	Benang khusus	1	21k
	Tas tabung	Tas tabung adalah tas yang di gunakan untuk menyimpang alat-alat tulis yang	Dijahit	Kulit imitasi	3	13k
	Tas ransel	Tas ransel adalah tas yang digunakan untuk anak sekolah masa kini	Handmade	Kulit khusus	2	50k
	Tas bulat	Tas yang cocok digunakan untuk pergi ke tempat2 wisata	Dijahit	Kulit	2	28k
Souvenir	Dompet	Dompet yang cocok untuk dimiliki karena modelnya yang simple dan bervariatif	Dijahit	Kulit imitasi	4	16k
	Tempat tissue	Sebuah souvenir unik yang digunakan sebagai tempat tissue di ruang keluarga	Dijahit	Kulit khusus	2	12k
	Tempat pensil	Suvenir yang cocok untuk menyimpan alat-alat tulis karena bentuknya yang unik	Dijahit	Kulit imitasi	1	10k
	Tempat bunga	Suvenir yaang cocok untuk hadiah ulang tahun perkawinan	Dijahit	Kulit imitasi	3	12k
	Gantung kunci	Suvenir yang cocok untuk sebagai cindera mata dari daerah tertentu	Diukir	Khusus	2	5k
	Sendal kuno	Sendal yang digunakan pada jaman kuno atau sendal tradisional daerah jawa	Dibor	Kayu	1	10k
	Topeng	Topeng kesenian yang biasa digunakan untuk hiasan dinding dan sebagai simbol kebudayaan daerah	Di ukir	Batok kelapa	1	40k

Toko Ibu Surati, JL. pasar PIIP Kota Blitar IIIB Email: wijayatoko12@gmail.com						
Tas	Tas bulat	Tas yang cocok digunakan untuk pergi ke tempat wisata	Dijahit	Kulit	2	28k
	Tas oval	Tas oval cocok digunakan untuk keperluan belanja	Dijahit	Kulit imitasi	10	16k
	Tas kendi	Tas kendi cocok digunakan untuk keperluan pergi ke tempat-tempat wisata	Dijahit	Kulit imitasi	5	18k
	Tas modern kecil	Tas modern adalah tas yang didesain untuk anak muda	Dijahit	Kulit imiasi	3	15k
	Tas kepang kecil	Tas modern adalah tas yang didesain untuk anak muda	Dianyam	Kulit kayu	6	22k
	Tas love	Tas love adalah tas yang dibuat khusus untuk pesta-pesta keluarga	Dijahit	Kulit imitasi	5	16k
	Tas kombinasi	Tas kombinasi sangat cocok untuk keperluan multi guna	Dianyam	Kulit kayu	10	21k
	Tas make up	Tas makeup adalah tas praktis yang cocok digunakan untuk membawa keperluan make up anda	Dijahit	Kulit imitasi	5	35k
	Tas tabung	Tas tabung adalah tas yang di gunakan untuk menyimpang alat-alat tulis yang	Dijahit	Kulit imitasi	3	13k
	Tas ransel	Tas ransel adalah tas yang digunakan untuk anak sekolah masa kini	Handmade	Kulit khusus	2	50k
Souvenir	Dompet	Dompet yang cocok untuk dimiliki karena modelnya yang simple dan bervariatif	Dijahit	Kulit imitasi	4	15k
	Tempat tissue	Sebuah souvenir unik yang digunakan sebagai tempat tissue di ruang keluarga	Dijahit	Kulit khhusus	2	12k
	Tempat pensil	Suvenir yang cocok untuk menyimpan alat-alat tulis karena bentuknya yang unik	Dijahit	Kulit imitasi	1	10k
	Tempat bunga	Suvenir yaang cocok untuk hadiah ulang tahun perkawinan	Dijahit	Kulit imitasi	3	12k

	Gantung kunci	Suvenir yang cocok untuk sebagai cindera mata dari daerah tertentu	Diukir	Khusus	2	5k
	Sendal kuno	Sendal yang digunakan pada jaman kuno atau sendal tradisional daerah jawa	Dibor	Kayu	1	10k
	Topeng	Topeng kesenian yang biasa digunakan untuk hiasan dinding dan sebagai simbol kebudayaan daerah	Di ukir	Batok kelapa	1	41k
Toko Surya, JL. pasar PIIP Kota Blitar Email: Suryatoko@gmail.com						
Tas	Tas oval	Tas oval cocok digunakan untuk keperluan belanja	Dijahit	Kulit imitasi	10	16k
	Tas kendi	Tas kendi cocok digunakan untuk keperluan pergi ke tempat-tempat wisata	Dijahit	Kulit imitasi	5	18k
	Tas modern kecil	Tas modern adalah tas yang didesain untuk anak muda	Dijahit	Kulit imiasi	3	15k
	Tas kepang kecil	Tas modern adalah tas yang didesain untuk anak muda	Dianyam	Kulit kayu	6	22k
	Tas love	Tas love adalah tas yang dibuat khusus untuk pesta-pesta keluarga	Dijahit	Kulit imitasi	5	16k
	Tas kombinasi	Tas kombinasi sangat cocok untuk keperluan multi guna	Dianyam	Kulit kayu	10	21k
	Tas make up	Tas makeup adalah tas praktis yang cocok digunakan untuk membawa keperluan make up anda	Dijahit	Kulit imitasi	5	35k
	Tas ayaman	Tas ayaman adalah tas yang cocok digunakan untuk pergi ke tempat-tempat wisata	Dianyam	Kulit kayu	5	30k
	Tas enceng	Tas enceng adalah tas yang di desain untuk orang yang mempunyai barang yang minim	Diayam	Kulit kayu	2	16k
	Tas tabung	Tas tabung adalah tas yang di gunakan untuk menyimpang alat-alat tulis yang	Dijahit	Kulit imitasi	3	13k
	Tas ransel	Tas ransel adalah tas yang digunakan untuk anak sekolah masa kini	Handmade	Kulit khusus	2	50k

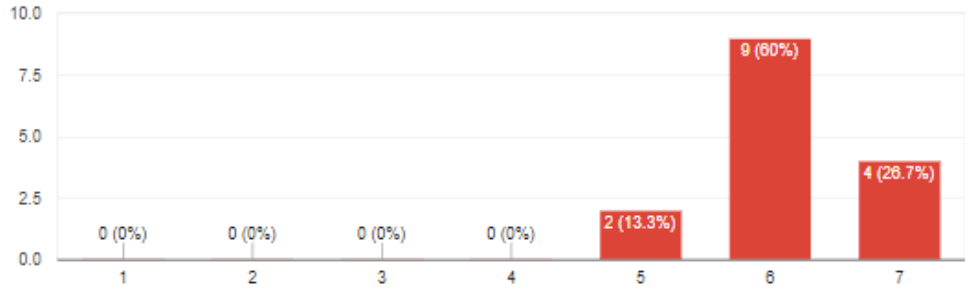
Souvenir	Dompet	Dompet yang cocok untuk dimiliki karena modelnya yang simple dan bervariatif	Dijahit	Kulit imitasi	4	15k
	Tempat tissue	Sebuah souvenir unik yang digunakan sebagai tempat tissue di ruang keluarga	Dijahit	Kulit khusus	2	12k
	Tempat pensil	Suvenir yang cocok untuk menyimpan alat-alat tulis karena bentuknya yang unik	Dijahit	Kulit imitasi	1	10k
	Tempat bunga	Suvenir yaang cocok untuk hadiah ulang tahun perkawinan	Dijahit	Kulit imitasi	3	12k
	Gantung kunci	Suvenir yang cocok untuk sebagai cindera mata dari daerah tertentu	Diukir	Khusus	2	5k
	Sendal kuno	Sendal yang digunakan pada jaman kuno atau sendal tradisional daerah jawa	Dibor	Kayu	1	10k
	Topeng	Topeng kesenian yang biasa digunakan untuk hiasan dinding dan sebagai simbol kebudayaan daerah	Di ukir	Batok kelapa	5	41k
	Taplak meja	Banyak motif dan warna yang beranekaragam dan mempunyai berbagai jenis ukuran	Di rajut	Khusus	1	41k
Toko Sumber Lancar, JL. pasar PIIP Kota Blitar Email: Sumberlancar12@gmail.com						
Tas	Tas bulat	Tas yang cocok digunakan untuk pergi ke tempat wisata	Dijahit	Kulit	2	28k
	Tas oval	Tas oval cocok digunakan untuk keperluan belanja	Dijahit	Kulit imitasi	10	16k
	Tas kendi	Tas kendi cocok digunakan untuk keperluan pergi ke tempat-tempat wisata	Dijahit	Kulit imitasi	5	18k
	Tas modern kecil	Tas modern adalah tas yang didesain untuk anak muda	Dijahit	Kulit imiasi	3	15k
	tas kepang kecil	Tas modern adalah tas yang didesain untuk anak muda	Dianyam	Kulit kayu	6	22k
	Tas love	Tas love adalah tas yang dibuat khusus untuk pesta-pesta keluarga	Dijahit	Kulit imitasi	5	16k

Tas kombinasi	Tas kombinasi sangat cocok untuk keperluan multi guna	Dianyam	Kulit kayu	10	21k
Tas make up	Tas makeup adalah tas praktis yang cocok digunakan untuk membawa keperluan make up anda	Dijahit	Kulit imitasi	5	35k
tas tali kur	Tas tali kur adalah tas yang di desain untuk anak muda ke pesta ulang tahunan	Dirajut	Benang khusus	1	25k
Tas Ayaman	Tas ayaman adalah tas yang cocok digunakan untuk pergi ke tempat-tempat wisata	Dianyam	Kulit kayu	5	30k
tas tabung	Tas tabung adalah tas yang di gunakan untuk menyimpang alat-alat tulis yang	Dijahit	Kulit imitasi	3	13k
tas ransel	Tas ransel adalah tas yang digunakan untuk anak sekolah masa kini	Handmade	Kulit khusus	2	50k

LAMPIRAN C HASIL *POST-STUDY SYSTEM USABILITY QUESTIONNAIRE*

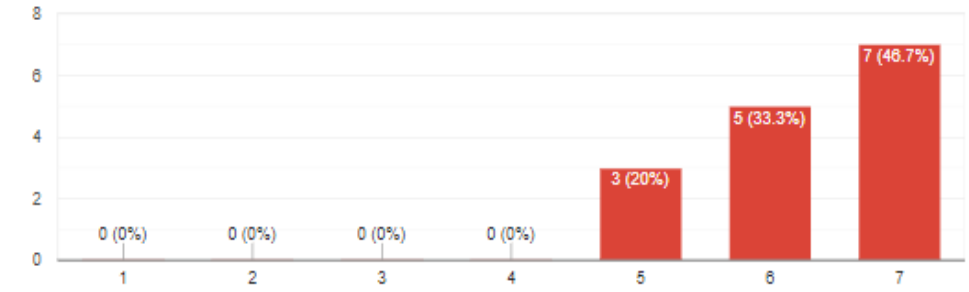
Secara Keseluruhan, Saya puas karena dapat menggunakan aplikasi anfield collection dengan mudah.

15 responses



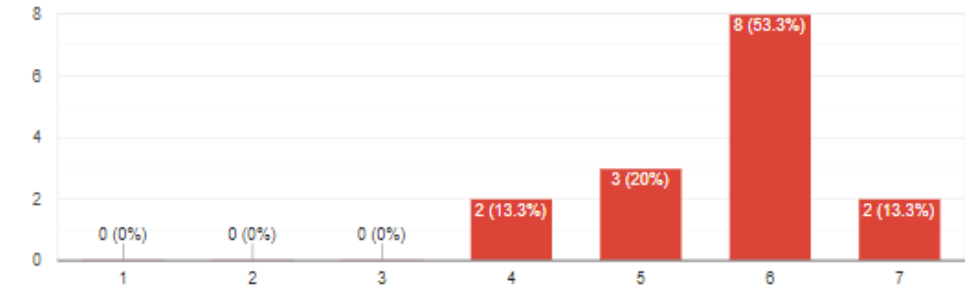
Aplikasi anfield collection sederhana untuk digunakan.

15 responses



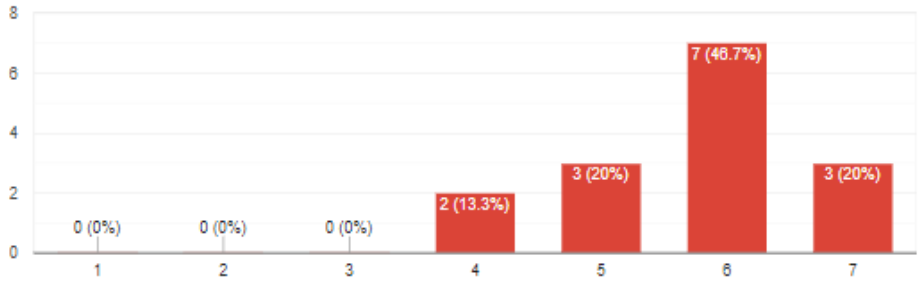
Sangat mudah untuk membuat iklan atau Sangat mudah untuk mencari barang pada aplikasi anfield collection.

15 responses



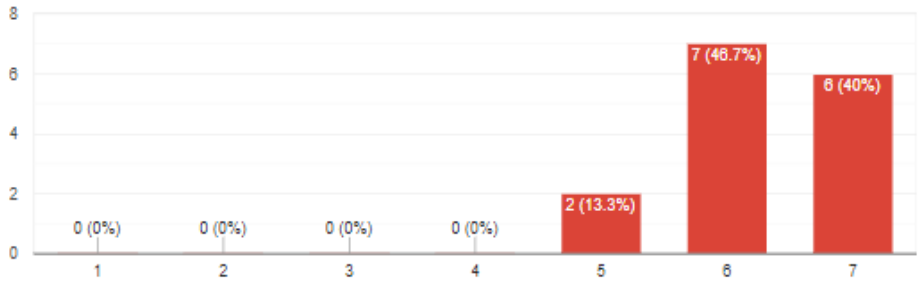
Saya sangat nyaman untuk menggunakan aplikasi ini.

15 responses



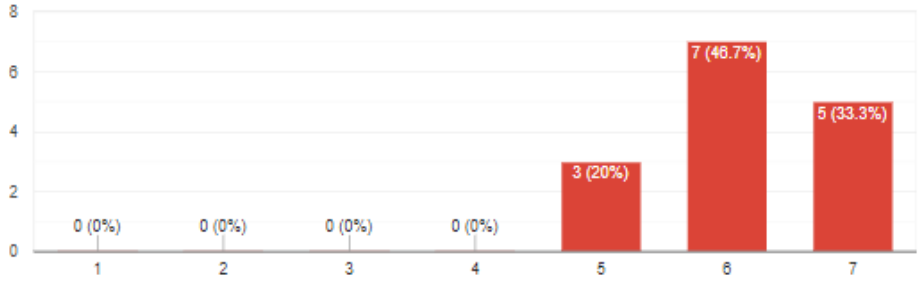
Aplikasi anfield collection mudah untuk dipelajari.

15 responses



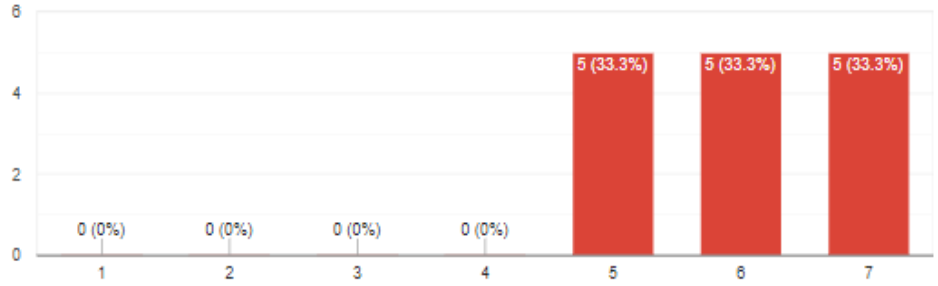
Saya yakin dapat menjual atau membeli barang dengan menggunakan aplikasi ini.

15 responses



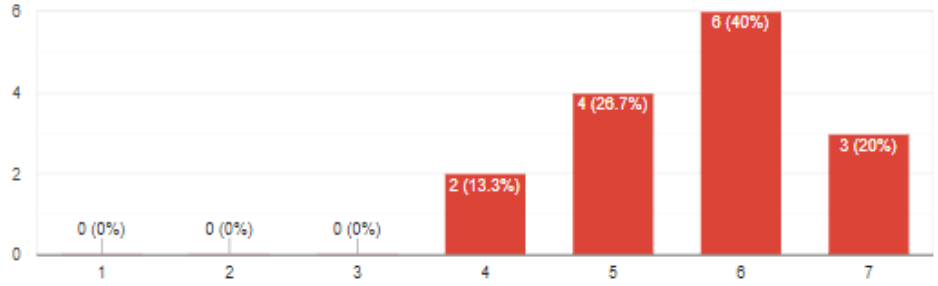
Aplikasi dapat menampilkan pesan & letak kesalahan ketika saya keliru saat menggunakan aplikasi ini.

15 responses



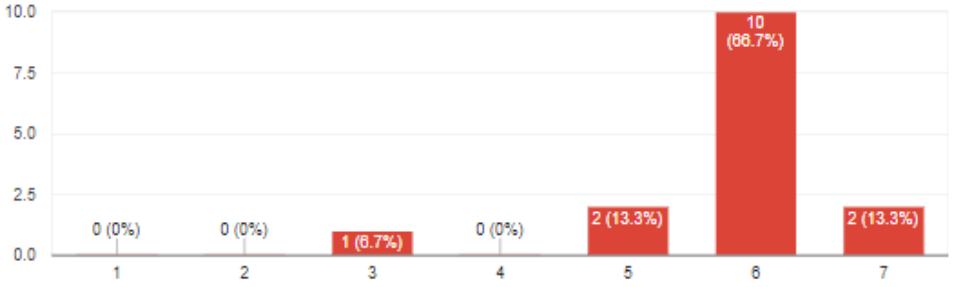
Saat saya melakukan kesalahan pada aplikasi, saya dapat cepat mengkoreksi.

15 responses



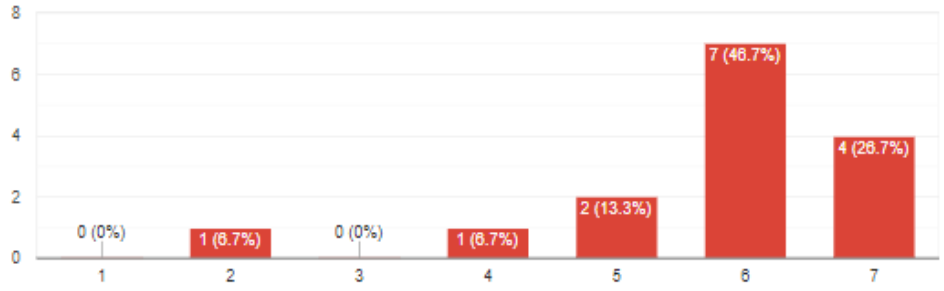
Informasi yang ditampilkan pada aplikasi sangat jelas.

15 responses



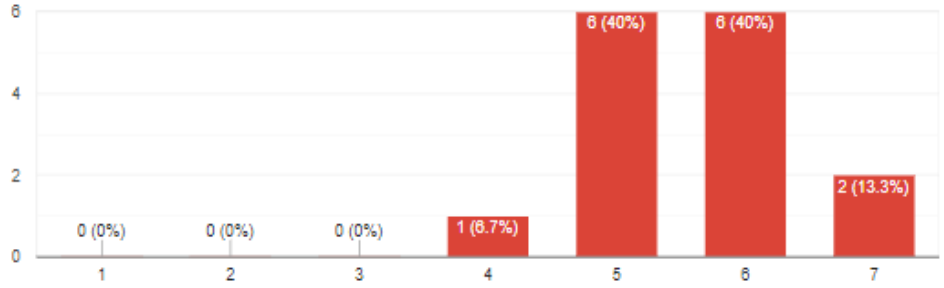
Sangat mudah menemukan apa yang saya cari ketika menggunakan aplikasi anfield collection.

15 responses



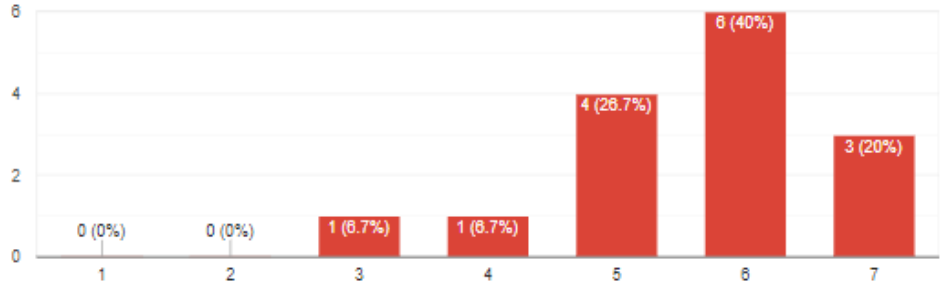
Informasi yang diberikan sangat efektif untuk membantu saya dalam menggunakan aplikasi ini.

15 responses



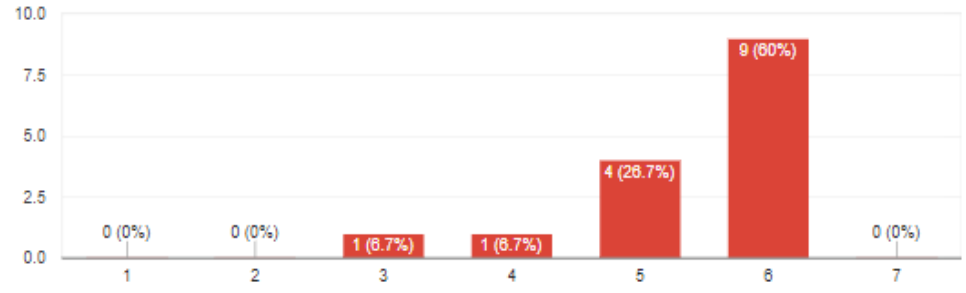
Letak informasi berada di tempat yang tepat.

15 responses



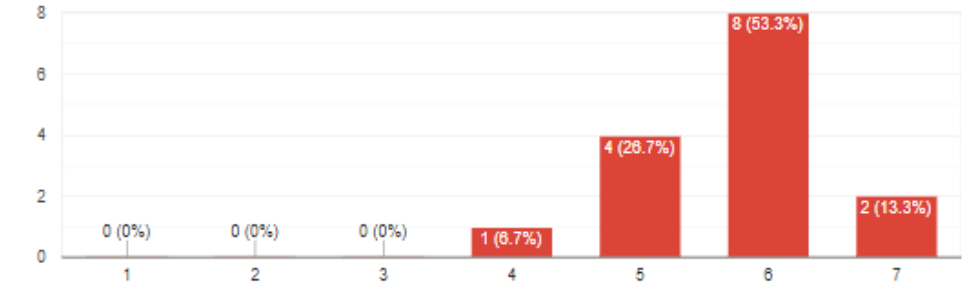
Antarmuka pada aplikasi ini sangan ramah penggunaanya

15 responses



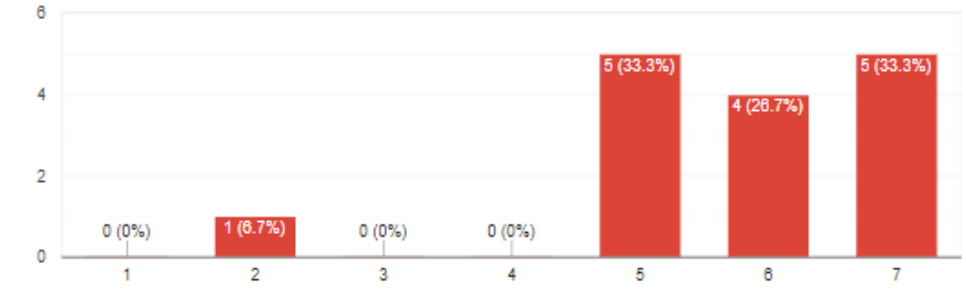
Saya suka antarmuka aplikasi ini.

15 responses



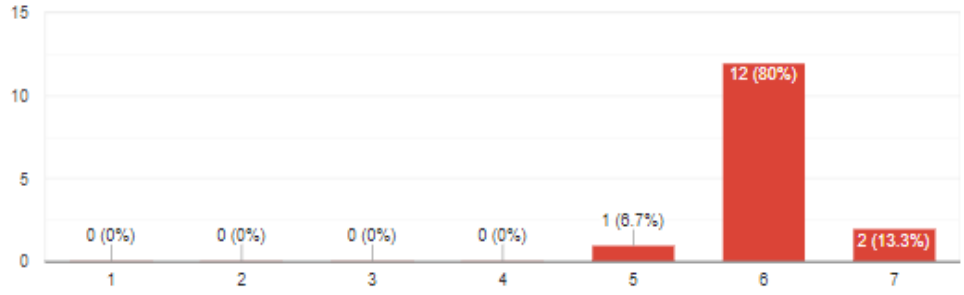
Aplikasi ini memiliki fitur sesuai dengan ekspektasi saya.

15 responses



Secara keseluruhan, saya puas selama menggunakan aplikasi anfield collection.

15 responses



Nama anda ?

15 responses

Widya Ayu Suryaningtyas
Ahmad Wahyu
Naily Zakiyatil Ilahiyah
Gheasani Safira Gunawan
Hendi Odang
Reza Mawardi
Freddy ajaz
Faris abdi El hakim
Aji Prasetyo

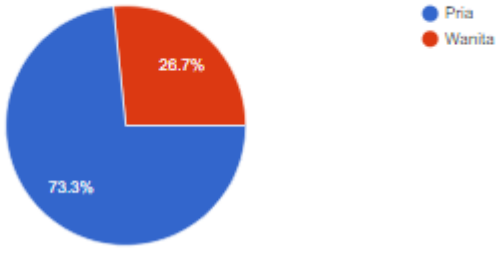
Tanggal lahir ?

15 responses

Jul 1994	8
Apr 1995	25
May 1995	4
Jun 1995	4, 15, 26, 28
Jul 1995	22
Aug 1995	16
Dec 1995	19
Feb 1996	29
Aug 1996	6
Apr 1997	19
Aug 1997	14
Feb 1998	28

Jenis Kelamin ?

15 responses



LAMPIRAN D *ROADMAP*

Nama : Fariz Reynaldo
Aplikasi : Anfield *Collection*
Status : 100%

No	Deskripsi *Task*	*Milestone*	*Progress*	*Status*
1	***Account*** - *Registration* - *Verification* - *Login* - *Forget Password*	- *Research about best practice for firebase auth* - *Make UI prototype from scratch* - *Code* - *Test*	100%	*Finished*
2	***Shopping Leather Bag*** - *Categories Product* - *List Product* - *Detail Product*	- *Research for async storage & firebase storage* - *Make UI prototype* - *Code* - *Test*	100%	*Finished*
	Filter Leather Leather Bag - *Search based name Product* - *Filter by price*	- *Make UI prototype* - *Code* - *Test*	100%	*Finished*
	Payment - *List Cart Product* - *Checkout Product* - *Payment method*	- *Research about midtrans implementation & RajaOngkir* - *Make UI prototype* - *Code* - *Test*	100%	*Finished*
3	***Warehouse Management*** - *Product Record* - *List Product* - *Finance*	- *Make UI prototype* - *Code* - *Test*	100%	*Finished*
	Total Progress		100%	*Finished*

Deskripsi *Task*:

1. ***Account***

Task ini merupakan fitur untuk pengguna.Untuk dapat menggunakan aplikasi ini, pengguna harus memiliki akun terlebih dahulu.Berikut sub fitur account:

- *Registration*

Sub fitur ini untuk berfungsi untuk melakukan register pada pengguna agar dapat melakukan proses bisnis yang terdapat pada aplikasi ini.

- *Verification*

Sub fitur ini digunakan oleh pengguna setelah melakukan registration.Pengguna akan diberi notification berubah email yang berisi email dan password yang di inputkan saat melakukan registration.

- Login

Pada Sub Fitur ini, pengguna memasukkan email dan password yang sudah didapatkan dari proses sebelumnya

- *Forget Password*

Sub fitur ini untuk pengguna yang lupa password karena dengan adanya fitur ini tidak akan mempersulit pengguna ketika lupa dengan password accountnya.

- *Profil Page*

Sub fitur ini menampilkan informasi pengguna seperti account pengguna, *wistlist* dan lain lain

2. ***Shop***

Task ini merupakan fitur untuk Pengguna maupun *member* untuk melakukan *online shopping*.Berikut adalah sub menu yang ada didalamnya :

- *Categories Product*

Sub fitur ini untuk menampilkan kategori product yang disediakan oleh toko Anfield Collection sehingga product yang di tampilkan sesuai dengan kebutuhan pengguna.

- *List Product*

Sub fitur ini digunakan oleh pengguna untuk menampilkan semua produk dilengkapi dengan gambar dan informasi lainnya seperti nama produk dan harganya.

- *Detail Product*

Sub fitur ini digunakan pengguna untuk menampilkan produk dengan informasi yang lengkap dan terdapat fitur *Add Cart* yang berfungsi menambahkan product ke keranjang belanja.

Filter Leather Bag

- *Filter by price*

Sub fitur ini digunakan untuk memfilter apabila pengguna ingin menampilkan product sesuai kriteria tertentu,misalnya dengan menentukan rentang harga sesuai kebutuhan.

- *Search Product*

Sub fitur ini untuk melakukan pencarian berdasarkan kata kunci (nama produk) agar mempermudah pengguna untuk menemukan product yang ingin dibeli.

Payment Leather Bag

- *List Cart Product*

Sub fitur ini digunakan oleh user untuk menampilkan product yang di tambahkan oleh pengguna.Pada Fitur ini juga dapat menghapus product dari list cart dan juga menjadi sebuah *product* menjadi *wistlist*

- *Checkout Product*
  Sub fitur ini digunakan melakukan *checkout* semua produk yang ditambahkan oleh pengguna untuk di bawa ke pembayaran
- *Payment Timeline*
  Sub menu ini digunakan oleh pengguna untuk mengisi data diri pengguna lalu alamat lengkap dan terakhir memilih metode pembayaran.

3. *Warehouse Management*

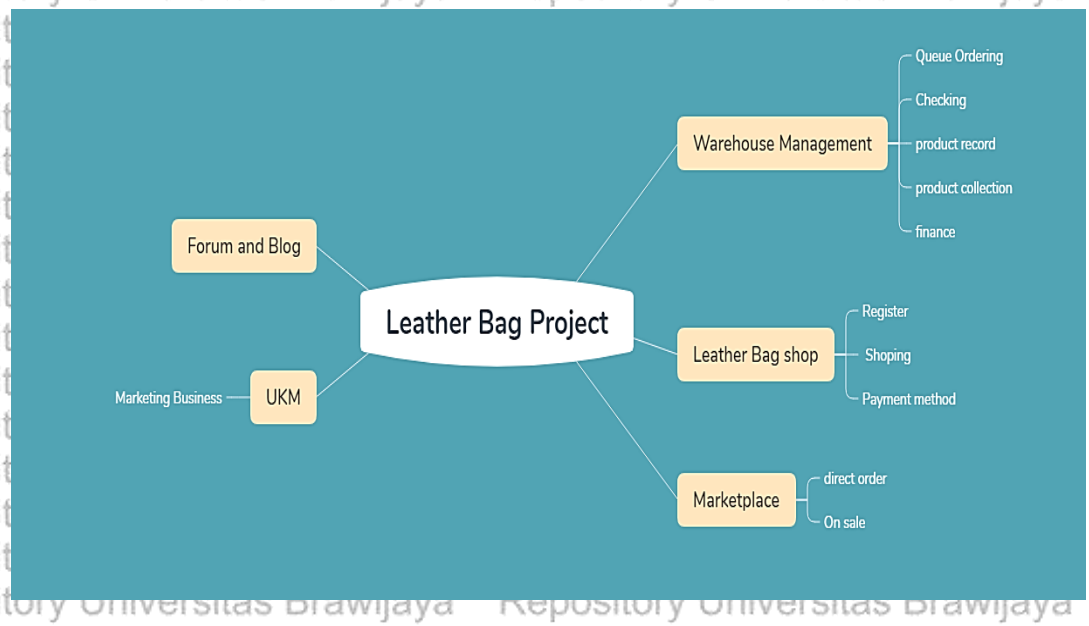
Task ini merupakan fitur untuk *project manajer* maupun *member* untuk masuk dalam team proyek tertentu. Didalam fitur ini memiliki sub fitur :

- *Product record*
  Sub fitur ini untuk melalukan perekaman product yang meliputi menambah,menghapus serta meng-*update* data produk
- *List Product*
  Sub fitur ini digunakan oleh admin untuk menampilkan semua produk dan informasi terkait data produk
- *Finance warehouse management*
  Sub fitur ini digunakan oleh admin untuk melihat laporan penjualan dan laporan keuangan dalam kurun waktu tertentu dan seberapa banyak produk yang terjual dalam kurun waktu tertentu

LOG:

- **7 Oktober 2019**
  - Riset NodeJS, React Native dan API Firebase Auth, Firebase Storage, Midtrans, RajaOngkir
  - Riset rancangan DB
- **14 OKtober 2019**
  - Implementasi DB
  - Implementasi dengan Firebase Auth
  - Fitur Register
  - Fitur Verifikasi Email
  - Fitur Login
  - Fitur Lupa Password
  - Fitur List Produk
  - Fitur List Cart
  - Fitur Kategori
- **21 Oktober 2019**
  - Fitur Cari Produk
  - Implementasi Order
- **28 Oktober 2019**
  - Implementasi Payment dengan midtrans
- **12 November 2019**
  - Implementasi Address Management
  - Implementasi Product Image Management
- **18 November 2019**
  - Fitur Laporan Penjualan
  - Account Management
- **20 November 2019**
  - Admim Feature
  - Deploy pada Domainesia & MongoDB atlas

***Work Breakdown Structure* (WBS)**



Malang, 17 Maret 2020

Pembimbing 1,	Pembimbing 2,
Nurudin Santoso, S.T., M.T.	Lutfi Fanani, S.Kom., M.T., M.Sc.
NIP. 197409162000121001	NIK. 201607 890217 1 001

Penulis,

Fariz Reynaldo

Fariz.aldo@gmail.com