Improvement Web Dashboard Aplikasi Super dengan Menggunakan Metode *Waterfall*

Yohanes Ricky Wijaya[1]

Abstract: PT. Krakatau Karya Abadi, commonly known as the Super Application Startup, is a network of agents distributing necessities in small towns and remote areas. One of the divisions in Super Applications is the Product Manager division. The division is responsible for product planning and development. Products in the Product Manager division talk about applications and dashboard platforms used by management and super application users. The Product Manager Division will be directly responsible to the user for any development. When the product is released, a Product Manager will perform maintenance on the development and will fix if bugs or problems are found in the future. During the internship, the researcher completed several developments on the super application. In development, the method used is the waterfall method, where the work is carried out coherently from beginning to end.

Keywords: startup; product manager; waterfall methodology; product

Pendahuluan

Sembilan bahan pokok (sembako) yang ditetapkan oleh pemerintah terdiri dari beras, gula pasir, minyak goreng, daging-dagingan, telur, susu, bawang merah & putih, ikan, dan garam. Sekarang penyebutan sembako sudah mulai meluas. Tidak hanya terdiri dari 9 bahan tersebut namun sudah mencakup kebutuhan dapur dan rumah tangga lainnya. Indonesia masih menjadi negara yang Jawa sentris. Situasi rantai kebutuhan pokok sembako pada kota kedua dan ketiga tidak merata , ada kesenjangan harga. Di beberapa kota, terutama kota–kota di luar pulau jawa. Sebagai contoh di Papua harga 1 botol aqua bisa mencapai Rp 20.000,- dibandingkan di Surabaya seharga Rp 3.000,-.

Pada saat ini, skema penjualan Aplikasi Super dapat dilakukan melalui 2 cara yaitu secara mandiri dimana pelanggan bisa unduh aplikasi sendiri yang tersedia pada aplikasi *google play store* dan melakukan pembelanjaan secara *online* atau yang bisa disebut dengan *customer* mandiri yang terhitung sebagai *end user*. Cara kedua yaitu dengan melalui non-mandiri dimana Aplikasi Super terdapat agen yang bernama Super Agen sebagai mitra yang datang ke toko sembako dan toko kelontong di berbagai kota yang dilayani Aplikasi Super. Cara ini merupakan cara yang paling sering dilakukan oleh toko-toko kelontong yang ada untuk melakukan transaksi.

Penulis melakukan kegiatan magang pada Divisi *Product and Technology* khususnya membantu *product manager*. Tugas utama, Divisi *Product and Technology* melakukan perencanaan dan pengembangan produk aplikasi super. *Product Manager* akan melakukan perencanaan, pembuatan konsep dan melakukan pengembangan terhadap setiap *request* yang masuk dari *user*. Dalam melakukan pengembangan seorang *product manager* juga akan membuat sebuah dokumen yang biasa disebut dengan *Product Requirement Document* (PRD) dimana didalam PRD tersebut terdapat *design user interface/user experience* (UI/UX) yang sudah dibuat, *flowchart process*, *user stories, Definition of Done* (DoD). PRD itu akan digunakan oleh setiap *user* yang terlibat seperti tim *developer* saat akan melakukan pembuatan program baru, dan juga *user* yang melakukan *request* sehingga *user* tersebut dapat mengetahui dengan jelas bagaimana bentuk pengembangannya.

Metode Penelitian

Pada bab ini akan membahas metode penyelesaian permasalahan yang akan diulas pada makalah ini. Metode yang digunakan adalah *waterfall*

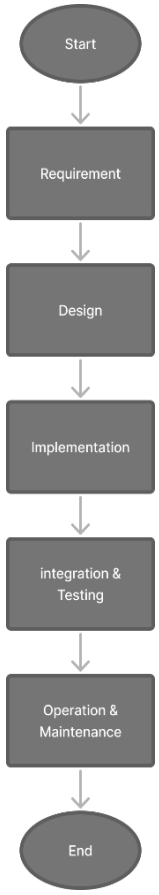
[1] Fakultas Teknologi Industri, Program Studi Teknik Industri, Universitas Kristen Petra. Jl. Siwalankerto 121-131, Surabaya 60236. Email: yohanesriicky@gmail.com

methodology. *Waterfall methodology* merupakan salah satu tools untuk melakukan pengembangan *software development* yang cukup sering digunakan. Pengembangan metode *waterfall* merupakan representatif seperti air terjun. Sehingga dalam penggunaannya metode *waterfall* merupakan penyelesaian atau pengembangan suatu permasalahan perangkat lunak yang dilakukan beruntun dan sistematis. *the waterfall methodology does allow for the software requirements, design, code or implementation, test and debug the software, presentation of the finished product and lastly follow by the maintenance mode* (Murray [1]).

Dalam praktiknya yang menjadi kunci penting keberhasilan metode *waterfall* adalah pengerjaan yang harus dikerjakan ketika langkah sebelumnya dinyatakan sudah selesai. Ketika Langkah sebelumnya belum diselesaikan maka Langkah selanjutnya belum bisa dikerjakan, hal ini diharapkan akan adanya pekerjaan yang maksimal dan dapat mengurangi tingkat kesalahan atau *bug* dikemudian hari saat produk sudah di rilis.

Waterfall memiliki beberapa kelebihan seperti saat pengembangan akan meminimalisir proses pengerjaan karena setelah proses *requierement* selesai sudah terdapat dokumen yang berisi kebutuhan dari *user*, sebagai PM juga akan lebih mudah untuk melihat perkembangan dari setiap tim saat melakukan pekerjaan. Tidak hanya terdapat kelebihan dalam penggunaan *waterfall* namun juga terdapat kekurangan yaitu pengerjaan akan membutuhkan waktu lebih lama dalam melakukan pengerjaan karena diharuskan menunggu proses sebelumnya selesai dan baru bisa menjalankan proses selanjutnya, dan ketika *user* menginginkan adanya perubahan atau penambahan fitur saat proses sudah mencapai implementation maka penambahan tersebut harus menunggu *update* fitur selanjutnya atau menunggu seluruh proses yang direncanakan diawal selesai.

Tahap *requirement* akan berfokus dalam melakukan *research* dan wawancara terhadap *user* yang mengajukan *request* fitur atau produk. Tahap *design* berfokus dalam pembuatan UI/UX yang akan digunakan. Tahap *implementation* berfokus pada pengerjaan *developer* baik secara *frontend* maupun *backend*. Tahap *integration & testing* befokus dalam uji *Quality Assurance* (QA) untuk fitur atau produk tersebut untuk mengetahui apakah produk bisa di rilis atau tidak. Pada tahap terakhir yaitu *operation & maintenances* berfokus pada rilis produk dan melakukan *maintenance* jika ditemukan *bug*.



Gambar 1. *Flow of waterfall methodology*

Pada saat pengerjaan memang terdapat beberapa tambahan pengerjaan yaitu setiap penyelesaian satu langkah akan dilakukan *review* baik terhadap penanggung jawab department maupun terhadap *user*. Ketika tahap *requirement* sudah selesai akan dilakukan *review* hasil *research* dan wawancara kepada *head product*, sama hal nya dengan ketika *design* selesai akan dilakukan *review* singkat kepada *user* untuk mengetahui *feedback* dari *user* terhadap *design* dan *flow* yang sudah dibuat. Setiap hasil dari pengerjaan pada *waterfall* akan didokumentasikan menggunakan PRD.

Tahap r*equirements* pada tahap ini merupakan awal dalam pengembangan atau penyelesaian masalah perangkat lunak. Tahap ini akan melakukan analisa dan mempersiapkan kebutuhan untuk melakukan pengembangan tersebut. Untuk mendapatkan data-data dan keperluan dalam pengembangan biasanya akan dilakukan *interview* secara langsung dengan *user* yang membutuhkan. Pada penerapan untuk PRD tahap ini akan membantu dalam menjabarkan *introduction, solution, user stories, dan Definition of Done*.

Tahap *design* akan berfokus dalam merancang solusi dari data yang sudah didapatkan. Pada tahap ini akan dimulai dalam pembuatan *flowchart* untuk mengetahui bagaimana suatu proses dapat terjadi.

Ketika *flowchart* sudah dibuat maka akan dilanjutkan dengan pembuatan desain tatap muka atau yang biasa diketahui dengan *design UI*/*UX*. Desain UI/UX diharapkan dapat menggambarkan lebih jelas bagaimana tampilan *interface* akan dibuat dan dipakai oleh *user*. Tahap *design* akan membatu *user* dalam melengkapi PRD pada bagian *detail requirements* agar *user* dan *developer* dapat mengetahui lebih jelas gambaran *visual* dari apa yang diharapkan oleh *user*.

Pada tahap *implementation* akan berfokus dalam ketika *design* sudah dibuat dan sudah disetujui oleh *user*, akan dilakukan tahap implementasi pengembangan dengan membuat coding baik secara *frontend* maupun *backend*. Pada tahap ini akan lebih berfokus pada hal - hal yang menyangkut sisi teknis. Jika pada tahap ini diketahui terdapat beberapa hal yang memang tidak bisa untuk dilakukan akan kembali ke tahap desain untuk melakukan *desain* ulang.

Pada tahap ini *developer* akan melakukan pengembangan dengan menggunakan PRD yang sudah dibuat untuk sehingga apa yang dibuat tidak berbeda atau tidak akan jauh berbeda dengan apa yang diharapkan *user*. Tahap *integration & Testing* memiliki peran dalam sebelum fitur atau product akan dirilis ke *user* maka akan dilakukan testing yang biasa disebut dengan tahap *test* QA.

Test QA akan dilakukan untuk mengetahui apakah modul sudah dibuat sesuai dengan fungsinya dan apakah sudah memberikan *experience* yang baik untuk *user* ketika menggunakan fitur atau *product* tersebut. Untuk melakukan *testing* tersebut akan dibuat skenario-skenario kasus. Ketika fitur tersebut sudah bisa menjawab skenario tersebut dan sudah memberikan experience yang baik maka fitur tersebut sudah dianggap siap untuk dirilis.

Tim QA akan melakukan *testing* dengan menggunakan standar yang sudah ditentukan dan diharapkan oleh *user* yang tertera pada *user stories* dan *definition of done*. Tahap terakhir yaitu *Deployment & Maintenance* memiliki tanggung jawab setelah fitur atau product sudah dirilis untuk digunakan oleh *user*, maka tahap terakhir pada *waterfall methodology* akan berjalan. Proses deployment & maintenance akan berfokus pada saat apabila terjadi bug atau permasalahan terjadi. Hal lain yang akan dilakukan pada tahap ini yaitu melakukan pengembangan lebih lanjut bila ada permintaan tambahan dari *user*.

Dashboard

Dalam perkembangan dunia digital, banyak perusahaan mulai melakukan digitalisasi terhadap perkembangan bisnis mereka untuk mendapatkan efektivitas dan produktivitas dalam melakukan pekerjaan. Dengan adanya digitalisasi dapat memudahkan semua data yang tercatat dapat ter*update* secara *realtime* dan dapat dibaca juga secara cepat tanpa menunggu waktu yang lama dikarenakan adanya integrasi antara divisi yang sudah tercatat secara *online*. Hal ini sehingga memudahkan juga dalam pengiriman surat atau dokumen dan juga dapat melihat *update* KPI secara cepat. KPI sendiri biasa ditampilkan pada sebuah *dashboard* yang akan menyajikan setiap data yang diperlukan oleh perusahaan.

Selain itu, *dashboard* adalah alat visualisasi yang memberikan kesadaran, tren, dan perencanaan serta perbandingan aktual, yang sering divisualisasikan dalam antarmuka pengguna yang disederhanakan. Mengubah data menjadi informasi yang akurat, andal, dan tepat waktu sebagai sumber untuk keputusan kritis dan penting serta mendapatkan pandangan yang jelas dan cerah tentang kinerja bisnis dapat menjadi tugas yang menantang bagi departemen manapun (Karami [2]).

Dalam perkembangannya, *dashboard* tidak hanya dapat menampilkan data saja. *Dashboard* pada aplikasi Super merupakan "*core*" yang merupakan *platform* yang tidak hanya digunakan hanya untuk mengecek data penjualan saja. *Dashboard* pada super sendiri berguna untuk menjalankan sistem jual beli, untuk melakukan PO, melakukan cek Stocking barang secara komputer. Sedangkan *dashboard* sendiri menurut Shadan Malik *The dashboard is the new face of the emerging information management field. Dashboards have become the vehicle of execution for several key initiatives being implemented among organizations worldwide* (Malik [3]).

Memang sedikit berbeda dengan penggunaan *dashboard* pada umumnya dimana hanya memunculkan data saja sedangkan pada *dashboard* super sendiri merupakan tempat untuk *management* melakukan aktivitas mereka. Pada *web dashboard* super dapat diakses oleh seluruh tim pada aplikasi super baik dari logistik hingga tim *human resources and general affair*.

Hasil dan Pembahasan

Pada pengerjaan *project* terdapat beberapa pekerjaan yang harus diselesaikan yaitu penurunan *cost* terhadap biaya logistik yang tinggi, peningkatan *retention rate* penggunadan peningkatan penjualan pada aplikasi super. Dalam menangani *cost* yang terlalu tinggi karena penggunaan vendor pengiriman yang menghitung jumlah pembayaran berdasarkan jumlah *invoice* yang diterima walaupun terdapat alamat yang sama namun berbeda *invoice* , maka untuk menjawab kebutuhan tersubut dibuatlah fitur *invoice global.* Dalam peningkatan *retention rate* dan penjualan pada aplikasi super akan dibuatnya fitur diskon dan fitur *bundle* produk untuk berjualan. Berikut adalah salah satu contoh *project* yang dikerjakan menggunakan *waterfall methodology* dalam upaya penurunan *cost logistic*.

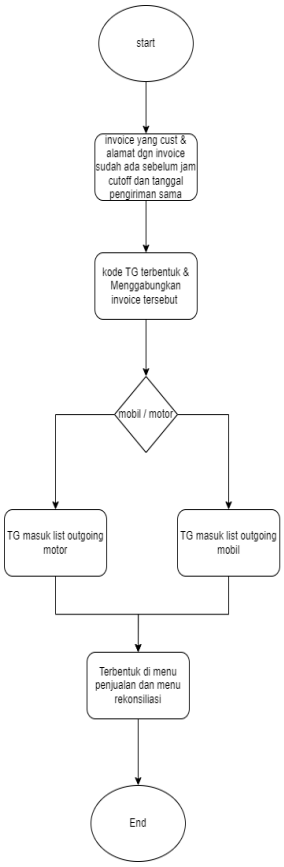
Invoice Global

Pada saat ini setiap transaksi yang masuk meskipun terdapat 2 transaksi dengan alamat yang sama maka akan menghasilkan 2 *invoice* yang berbeda. Sehingga hal tersebut membuat tim logistik mengeluarkan biaya lebih terkait nota dengan pelanggan yang sama dan alamat tujuan yang sama jika menggunakan pengiriman vendor karena vendor menghitung dari kode *invoice*.

Tim *product and tech* akan dibuat fitur pengiriman global dimana setiap terdapat 2 transaksi dengan 1 alamat yang sama secara sistem akan digabungkan di 1 kode baru yaitu kode TG sehingga akan mengurangi jumlah *invoice* yang akan diserahkan kepada pihak vendor.

Solusi yang dibuat sebagai berikut yaitu pertama sistem akan membaca *invoice* yang memiliki customer & alamat sama & tanggal pengiriman sama (dalam rentang cutoff yang sama) maka akan terbentuk transaksi global. Kedua sistem membaca nominal belanja akan dihitung secara total dari *invoice* yang tergabung di dalam *invoice global*, disesuaikan terhadap syarat minimum pesan mobil-motor yang telah diatur.

Pada setiap *invoice* yang masuk di hari yang sama dan sebelum jam *cutoff* akan membaca apakah ada alamat & nama pelanggan yang sama serta tanggal pengiriman yang sama maka akan terbentuk *invoice global*. Sistem juga akan membaca jumlah nominal gabungan dari dua *invoice* yang digabungkan untuk menentukan pengiriman menggunakan mobil atau motor setelah perhitungan jumlah *invoice* tersebut.



Gambar 2. *Flow of invoice global*

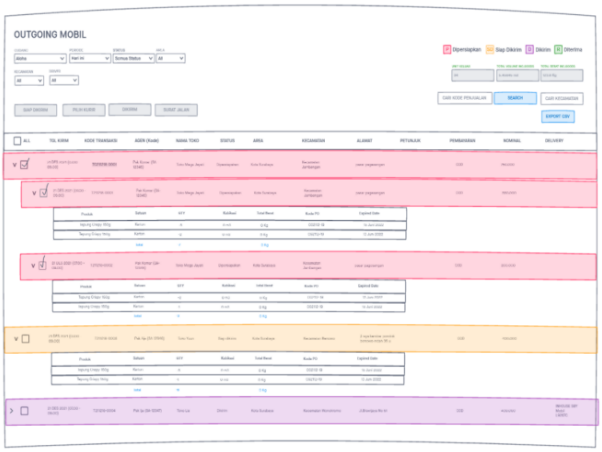
Ketika sistem membaca setiap transaksi yang sudah masuk, dan terdapat jumlah *invoice* yang >1 maka secara sistem akan terbentuk kode TG untuk menggabungkan *invoice* tersebut. Ketika *invoice* tersebut sudah digabungkan maka akan dihitung jumlah nominal dari transaksi gabungan tersebut untuk dibedakan pengiriman menggunakan motor / mobil. Ketika sudah dibedakan pengiriman mobil dan motor maka akan terbentuk menu penjualan dan menu rekonsiliasi untuk mempermudah tim logistik dalam merekap data pengiriman.

Detail Requirement

Pada bagian ini akan menjelaskan tentang *prototype* yang sudah dibuat. Diharapkan melalui bagian ini akan membantu *developer* untuk memenuhi kebutuhan lebih jelas. Selain itu, diharapkan bagian ini bisa dapat membantu *end user* sebagai pedoman dalam menggunakan fitur yang sudah dibuat.

Pada tampilan *outgoing* maka akan terdapat *list invoice* yang terkirim. Dapat dilihat pada urutan paling atas terdapat satu kode TG yang terbentuk dikarenakan terbaca terdapat 2 *invoice* yang memiliki pengirim, alamat dan jam kirim yang sama. Apabila dilakukan *checklist* pada tampilan kode TG maka 2 *invoice* yang

lainnya akan otomatis ter*chekclist* dan jika melakukan print surat jalan maka yang terbentuk adalah surat jalan TG. Namun, apabila yang dilakukan checklist pada kedua *invoice* tunggal bukan pada kode TG maka akan di *print* surat jalan tunggal untuk masing - masing *invoice*.



Gambar 3. Halaman *list invoice global*

User Stories

User stories akan membantu tim dalam menjelaskan hal apa saja yang harus dimiliki pada fitur yang akan dibuat. Hal yang dapat dilakukan oleh *user* adalah pertama sebagai *user*, saya dapat melakukan *checklist invoice global* dan anak *invoice* otomatis ter*checklist*. Kedua sebagai *user*, apabila saya melakukan *checklist invoice global* tetapi anak *invoice* ada yang tidak tercentang minimum 1 maka saya tidak bisa *update* status *invoice* tersebut. Ketiga sebagai *user*, saya dapat melakukan *checklist* anak *invoice* dan tidak mempengaruhi *invoice global*. Keempat sebagai *user*, saya dapat melihat, dan melakukan *print* surat jalan *invoice global* serta *invoice* tunggal secara massal. Kelima sebagai *user*, saya dapat melihat dan *print* surat jalan masing-masing dari *invoice* yang ada di dalam transaksi *global* secara massal. Keenam sebagai *user*, saya dapat *checklist all data outgoing* dan print surat jalan. Ketujuh sebagai *user*, saya dapat mengubah status *invoice global* pada halaman *outgoing* dan berpengaruh pada anak *invoice* dibawahnya. Kedelapan sebagai *user*, saya dapat *export* CSV dengan penambahan format kolom baru yaitu kolom *invoice global*.

Definition of Done

Pertama invoice global akan digabung jika *invoice* terbuat pada hari yang sama, pada tanggal pengiriman yang sama (dalam satu *cutoff* yang sama), nama *customer* & alamat yang sama. Kedua *Invoice global* akan memiliki kode *prefix* dengan awalan "TGxxx". Ketika Penyesuaian di *outgoing*: Jika di dalam *invoice* global terdapat *invoice* dengan nominal tertinggi masuk dalam syarat nominal *order* motor, maka *invoice global* masuk *outgoing* motor. Keempat Jika di dalam *invoice global* terdapat *invoice* dengan nominal tertinggi masuk dalam syarat nominal order mobil, maka *invoice global* masuk *outgoing* mobil.

Definition of done pada menu *outgoing* yaitu pertama ketika dilakukan *checklists invoice global* maka anak *invoice* juga ter*checklist*. Kedua ketika *checklist* anak *invoice* maka kotak ceklis *invoice global* tidak berpengaruh. Kedua ketika *user checklist* semua maka semua *invoice* ter*checklist* semua (*global* dan tunggal). Ketiga jika terdapat *invoice global* yang dan anak *invoice* di dalamnya ter *checklist* maka *print* format *invoice global*. *Keempat* ketika jika terdapat anak *invoice* saja yang terceklis dan *invoice* global tidak ter*checklist* maka *print* format *invoice* tunggal. Kelima jika *invoice global* berstatus P, SD dan D dan jika terdapat pembatalan *invoice* maka sistem akan menyesuaikan Selama minimal >1 anak *invoice* yang memiliki *customer* sama, alamat pengiriman sama, dan tanggal pengiriman sama maka *invoice global* tetap terbentuk.

Pada saat ini untuk *invoice global* sudah bisa membantu tim logistik dalam menurunkan *cost* yang berlebih saat melakukan pengiriman ke konsumen. Selama penggunaan belum ditemukan *bug* atau permasalahan yang diterima oleh tim logistik.

Product Bundling* Sisi *Vanguard

Bundling digunakan untuk meningkatkan *retention rate* dan *active users* di aplikasi Super. *Product bundling* merupakan gabungan dari beberapa produk yang akan dijual dalam bentuk satu paket (*bundle*). Selain itu *Product Bundling* diharapkan dapat meningkatkan minat beli pengguna aplikasi super. Product *bundling* pada sisi *vanguard* akan membantu tim *core* untuk mengembangkan desain dan fitur yang akan dinikmati oleh *end user* atau *customer*.

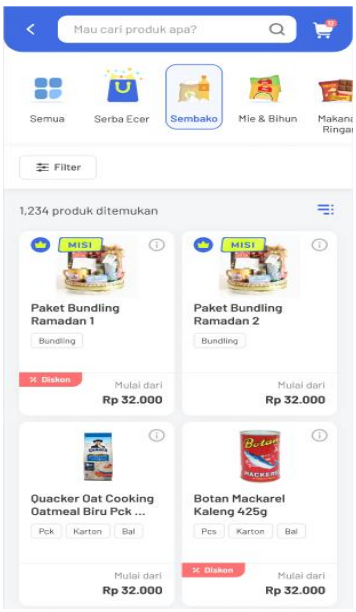
Detail Requirement

Pada bagian ini akan menjelaskan tentang *prototype* yang sudah dibuat. Diharapkan melalui bagian ini akan membantu *developer*

untuk memenuhi kebutuhan lebih jelas. Selain itu, diharapkan bagian ini bisa dapat membantu *end user* sebagai pedoman dalam menggunakan fitur yang sudah dibuat.

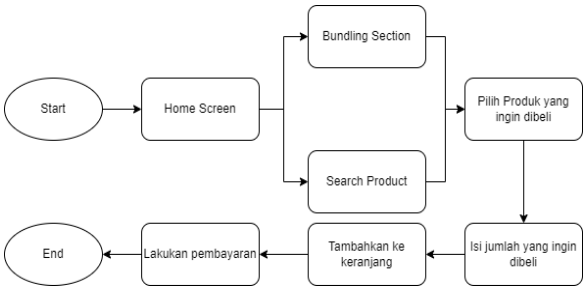
Gambar 4. *Section product bundling*

Bundling adalah salah satu fitur untuk meningkatkan *engagement* dan juga *retention user* Super. Satu *bundling* ini memiliki beberapa produk yang dikemas dalam satu paket (satu sku) yang sesuai dengan dibuat dalam *dashboard* Super. Untuk mencari produk *bundling* dapat dicari melalui *section* yang disediakan di halaman awal aplikasi, selain itu dapat juga dicari melalui kategori yang ada pada *product gallery* dan dapat dicari pada *search bar* dengan menuliskan *keyword "Bundle:"*.



Gambar 5. *Product gallery category* sembako

Pada tampilan *product gallery*, produk *bundling* akan selalu ditampilkan dipaling atas. Namun jumlah produk *bundling* yang dikeluarkan hanya akan 2 – 3 produk saja agar produk yang dicari berdasarkan katergori tetap terlihat. Produk *bundling* akan di utamakan karena memang fokus utama dari *marketing* untuk menjual produk tersebut.



Gambar 6. *Flow of user product bundling*

Berdasarkan *flow* diatas, dapat dilihat fitur untuk produk *bundle* ini ada 2 macam, yaitu pertama *section bundling* hanya akan muncul jika sedang ada stok *bundle*, jika tidak ada stok *bundle* maka section *bundling* akan hilang. Section ini bisa ditamproduk *bundle*, maka akan muncul semua *list* produk *bundle*. Kedua *Suggestion produk bundle* ketika cari nama produk *bundle* ini juga dapat ditemukan pada search bar dan pada bagian *product gallery*. Ketika *user* melakukan pencarian pada produk dan terdapat salah satu dari isi paket *bundling*, maka akan ditampilkan suggestion *product bundling*. Tampilan suggestion *product bundling* hanya akan dimunculkan 2 produk *bundling* saja sesuai dengan kategori product tersebut.

User yang melakukan pencarian berdasarkan *suggestion* label *bundle* akan berada di produk yang berhubungan dengan *bundle* yang diikuti oleh *user*. Label *bundle* ini akan ditampilkan di *product gallery*, *product detail,* dan *cart.*

User Stories

User stories merupakan penjelasan mengenai apa saja yang bisa dilakukan oleh *user* pada fitur atau *improvement* yang dibuat. Berikut adalah *user* stories dari *improvement* fitur diskon yang pertama sebagai *user*,saya dapat melihat *section bundling* di *home page*. Kedua sebagai *user*, saya dapat melihat *list* semua *product bundling* di *section bundling*.ketiga sebagai *user*, saya dapat *search* nama produk, lalu muncul *suggestion* produk *bundling* di *category* asal sku *bundling*. Keempat sebagai *user*, saya dapat klik semua kategori produk, lalu muncul *list* produk *bundle* di *produk gallery*.

Kelima sebagai *user*, saya dapat klik *detail* produk *bundling*. Keenam sebagai *user*, saya

dapat lihat *detail deskripsi* produk *bundling*. Ketujuh sebagai *user*, saya dapat lihat stok habis produk *bundle* dan mengingatkan kembali jika stok sudah tersedia. Kedelapan sebagai *user*, saya dapat transaksi *product bundle* dengan *voucher*.Terakhir kesembilan sebagai *user*, saya ketika gagal transaksi melalui *Virtual Account*, maka stok pembelian kembali ke stock *bundling*.

Definition of Done (DoD)

Definition of done merupakan standar atau prasyarat yang harus dipenuhi untuk menyatakan bahwa *project improvement* sudah siap untuk dirilis. Berikut adalah DoD dari *improvement* fitur diskon yaitu yang pertama *bundle section* akan dipakai di section produk di *home page*. Kedua jika di klik semua kategori pada kategori produk, maka akan diarahkan ke halaman *product gallery* yang berisikan semua *list bundle* yang terletak di pin paling atas dan diurutkan berdasarkan abjad.

Ketiga jika di *search product*, lalu akan muncul *suggestion produk* dan rekomendasi produk *bundling* (sesuai produk yang dicari). Keempat Hasil *search product*, maka akan diarahkan ke halaman *product gallery* yang berisi rekomendasi produk *bundle* dengan kategori asal dari produk *bundling* tersebut dan *suggestion* produk yang dicari. Contoh *Bundling* Ramadhan 1 (Beras dan Susu), jadinya produk tersebut termasuk ke kategori Sembako dan Minuman Instan.

Kelima tampilan produk *bundle* di kategori asal dari produk *bundling* terletak di pin paling atas dan ditampilkan semua produk *bundle*. Keenam jika di *filter* pada halaman Produk Gallery, maka akan ditampilkan sesuai dengan hasil *filter*. Produk *bundling* yang di pin atas mengikuti urutan *filter* yang diterapkan. Adapun contoh sebagai berikut yaitu *Filter* Z-A, kalau nama product *bundling*nya "Bundling Ramadhan" jadinya produk tersebut ada di bagian bawah karena *filter*nya Z – A. Ketujuh ketika transaksi pembayaran produk *bundle* bisa menggunakan *voucher*. Terakhir kedelapan Harga *product bundling* memiliki beberapa *tier*.

Produk *bundle* sisi *vanguard* masih dalam pengembangan tim *developer* namun ketika dilakukan *sprint planning* dan penjelasan tentang fitur kepada *user* dan tim yang terkait sudah dinyatakan bisa membantu menjawab kebutuhan dari *user* dalam meningkatkan *retention rate* dan penjualan dari aplikasi super terutama ketika terdapat beberapa *event* tertentu yang dapat membantu penjualan.

Learn and Growth

Dalam mengerjakan sebuah *project* pasti dibutuhkan sebuah prosedur yang jelas supaya mendapatkan *output* yang sesuai dengan yang diharapkan. Dalam pengerjaan *project* pun dibutuhkan beberapa proses yang penting untuk dilakukan. Ketika prosedur sudah dibuat dan dapat menjadikan prosedur yang baku hal itu dapat mempermudah pengerjaan baik seorang *product manager*. Penggunaan metode *waterfall* bisa menjadi salah satu cara untuk mengerjakan setiap *project* agar lebih efisien.

Dimulai dengan ketika mendapatkan *request* dari *user*, maka sebaiknya kita menjadwalkan untuk melakukan wawancara secara langsung terhadap *user* untuk mengetahui sejauh apa keinginan yang diharapkan oleh *user* dan hal apa saja yang ingin mereka dapatkan. Pada saat wawancara ini harus melakukan pertanyaan yang detail dan cukup dalam sehingga tidak akan membutuhkan wawancara dalam waktu yang banyak cukup hanya sekali. Selanjutnya, sebagai *product manager* sebaiknya melakukan research terhadap *request* yang masuk untuk mengetahui gambaran apa yang akan dikerjakan. Setelah melakukan *research* dapat dilanjutkan dengan melakukan *design wireframe* untuk memberikan gambaran yang jelas ketika akan diberikan kepada tim UI/UX. *Product manager* dapat melanjutkan dengan mulai melakukan penulisan guna membantu setiap tim atau *user* yang akan melakukan pekerjaan atau *user* yang akan menggunakan fitur tersebut.

Ketika *design wireframe* sudah selesai, dapat dilanjutkan dengan melakukan *sprint planning* dimana hal itu akan dengan mengundang beberapa stakeholder seperti *user* yang melakukan *request*, tim UI/UX, tim *developer* dan beberapa tim lainnya yang terlibat saat nantinya akan menggunakan fitur tersebut. Jika semua stakeholder sudah menyetujui terhadap *design* yang sudah dikerjakan maka dapat fitur tersebut dinyatakan sudah siap untuk di kerjakan oleh tim *developer* dan siap untuk dirilis nantinya. Sebelum fitur tersebut dirilis bisa dilaksanakan *user acceptance test* terhadap fitur tersebut apakah sudah sesuai dan menjawab apa yang dibutuhkan oleh *user*. Ketika fitur sudah dirilis maka kita dapat melakukan *maintenance* dan *checking* untuk setiap bug jika dibutuhkan. Idealnya

keseluruhan proses meskipun sudah diberikan kepada tim UI/UX dan *developer* maka kita sebagai PM juga harus tetap melakukan *update report* setiap harinya guna mengetahui *update* perkembangan dari setiap tim yang sedang melakukan pekerjaan.

Simpulan

Dalam penggunaan *Waterfall methodology* dalam setiap pengerjaan object akan membuat pengerjaan setiap object akan lebih mudah. Dalam startup tuntutan untuk melakukan perkembangan dalam waktu yang cepat akan membutuhkan metode yang tepat dalam pengerjaannya. Metode *Waterfall* akan dapat membantu dalam melakukan pekerjaan lebih tepat dimana kita akan dituntut untuk melakukan pekerjaan secara runtut sehingga akan meminimalisir dalam melakukan kesalahan. Metode *Waterfall* juga berfokus pada pembangunan dasar yang lebih kuat sehingga ketika menjalankan setiap proses - proses tidak prrlu kembali ke proses sebelumnya. Dalam menjalankan empat projek yang dikerjakan oleh penulis, secara keseluruhan menggunakan metode *Waterfall* dimana hal tersebut membantu dalam setiap proses pengerjaan. Dalam mempelajari hal yang baru memang sangat sulit, terutama pada *startup* dimana perubahan bisa terjadi kapan saja dalam waktu yang tidak tepat. Namun, pada Aplikasi Super sudah terbagi dalam berbagai kuartal sehingga dalam setiap awal kuartal bisa membiasakan penggunaan metode.

Penggunaan metode W*aterfall* mungkin memang akan terasa sulit di awal namun jika sudah terbiasa untuk setiap orang nya maka akan sangat mudah dan akan sangat membantu setiap PM dalam melakukan *improvement*. Maka dari itu, diperlukan kemampuan manajerial yang mumpuni untuk memastikan pelaksanaan metode dapat berjalan baik sehingga perkembangan pesat perusahaan dapat terjadi.

Daftar Pustaka

1. Murray, A. P.*, He Complete Software Project Manager: Mastering Technology from Planning to Launch and Beyond*, 2016.
2. Karami, M., *Operational Radiology Dashboard: A Tool for Optimizing Workflow Management,* 2012.
3. Malik, S., *Enterprise Dashboard: Design and Best Practice for IT*, 2005.