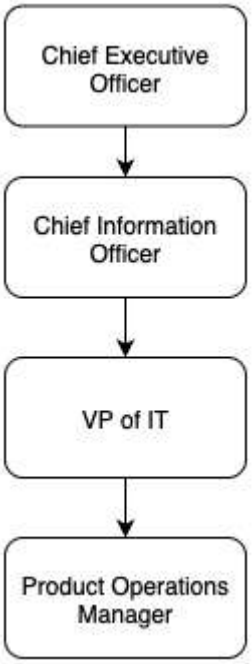
BAB III

PELAKSANAAN KERJA MAGANG

3.1. Kedudukan dan Koordinasi

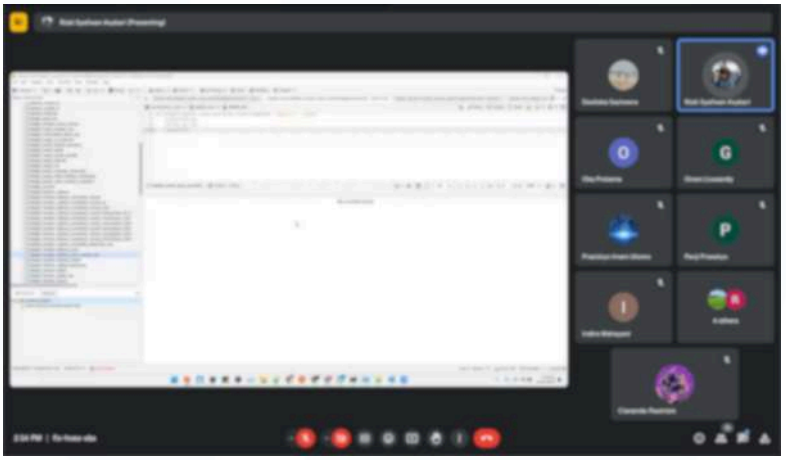


Gambar 3.1 Kedudukan Pemimpin divisi Product Operations

Penulis melaksanakan program magang dengan menempati posisi *Quality Assurance* di departemen *Product Operations* pada PT. Global Loyalty Indonesia. Dapat dilihat dari gambar 3.1 kedudukan kepemimpinan di divisi yang ditempatkan penulis untuk koordinasi terkait pekerjaan. Dalam menjalankan tugas sehari-hari, divisi QA menjalin koordinasi secara intensif melalui aplikasi Telegram dan Whatsapp sebagai media komunikasi utama, baik untuk interaksi antar anggota QA, dengan atasan atau *leader* divisi, maupun komunikasi lintas divisi yang terlibat dalam proyek yang sama. Pemanfaatan Telegram dan Whatsapp memudahkan pertukaran informasi secara cepat dan efektif, sehingga mendukung kelancaran pelaksanaan pekerjaan dan kolaborasi antar tim.

Disaat sebuah proyek mulai, akan ada pertemuan yang berupa *virtual meeting* antara beberapa divisi untuk mendiskusikan terkait proyek yang bernama

refinement. Pertemuan ini membahas alur keseluruhan proyek dan dihadiri oleh berbagai divisi, baik dari sisi bisnis maupun *developer*. Dalam pertemuan tersebut, biasanya terjadi diskusi antara divisi bisnis dan divisi IT untuk menilai tingkat kesulitan proyek serta apakah proyek tersebut memerlukan usaha tambahan atau *effort* ekstra dari pihak *developer*.

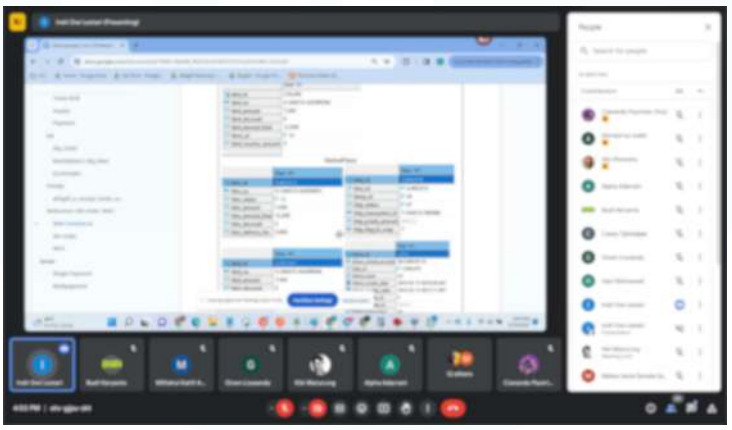


Gambar 3.2 Suasana ketika *Sprint Planning*

Setelah dilakukan *refinement*, akan ada pertemuan berbentuk *virtual* selanjutnya yang bernama *sprint planning* seperti yang dapat dilihat di gambar 3.2, dimana selama durasi proyek berjalan, akan dilakukan beberapa kali *sprint planning* tergantung dari skala proyek tersebut. Pertemuan pertama biasanya disebut *Sprint* 1, melibatkan tim *Quality Assurance* (QA), *developer*, dan *product development* untuk membahas kebutuhan bisnis, desain sistem, serta *storylist* proyek. Dalam sebuah proyek berjalan, umumnya terdapat 2 hingga 4 *Sprint*, tergantung pada skala proyek. Proyek dengan tingkat kompleksitas dan skala yang lebih besar biasanya memiliki lebih dari tiga *Sprint* untuk memastikan pembaruan perkembangan dilakukan secara berkala oleh semua divisi yang terlibat.

Disaat suatu proyek sedang aktif berjalan, akan diadakan *meeting* yang dilakukan setiap hari yang bernama *daily update* dan dihadiri oleh *Project Manager*, *Quality Assurance*, dan *Developer* untuk memberikan *update* mengenai perkembangan pekerjaan yang dilakukan didalam proyek pada hari sebelumnya, mengidentifikasi kendala yang dialami, serta mendiskusikan berbagai

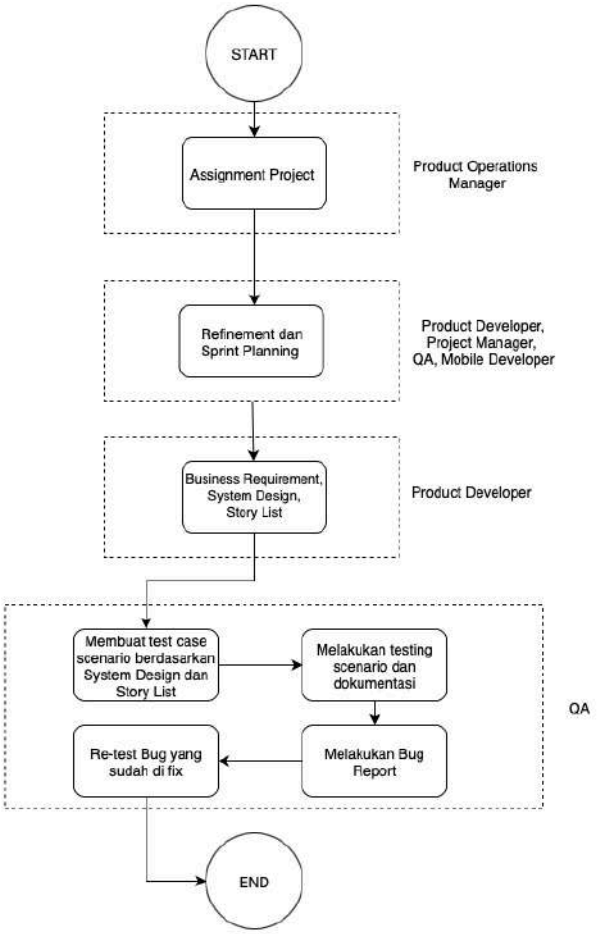
permasalahan yang muncul dalam proyek tersebut agar dapat segera dicari solusi bersama.



Gambar 3.3 Proses melakukan *sharing knowledge*

Disaat proyek telah selesai proses pengetesan, akan dilakukan pertemuan yang berbentuk presentasi *online* seperti pada gambar 3.3 dilakukan oleh *Person in Charge* (PIC) dari proyek yang telah selesai untuk menjelaskan kegiatan yang telah dilakukan dalam proyek tersebut. Tujuannya adalah untuk melihat kembali dan berbagi informasi kepada tim *Quality Assurance* (QA) lain yang tidak terlibat langsung dalam proyek tersebut. Selain itu, para PIC biasanya juga siap memberikan penjelasan dan menjawab pertanyaan dari QA lain yang ingin mengetahui lebih lanjut mengenai project yang dipresentasikan.

3.2. Tugas dan Uraian Kerja Magang



Gambar 3.4 Proses kerja QA

3.2.1 Tugas yang Dilakukan

Selama menjalani masa magang di PT. Global Loyalty Indonesia, penulis menempati posisi sebagai *Quality Assurance* di bawah departemen *Product Operations*. Dalam pelaksanaan tugas, penulis secara langsung dilibatkan dalam

sejumlah proyek, baik kecil maupun besar, sesuai arahan dari *supervisor*. Peran utama penulis sebagai QA adalah melakukan pengujian terhadap aplikasi Alfagift guna memastikan bahwa aplikasi berjalan sesuai dengan fungsi yang diharapkan dan bebas dari *bug* atau masalah sebelum dirilis ke publik melalui *platform* App Store dan Play Store.

Berdasarkan ilustrasi pada gambar 3.4, proses kerja dimulai ketika penulis menerima penugasan dari *supervisor* untuk bergabung dalam suatu proyek tertentu. Setelah masuk ke dalam proyek tersebut, penulis mengikuti sesi *refinement* dan *sprint planning* untuk memahami alur dan ruang lingkup pengetesan yang akan dilakukan. Tahap selanjutnya, penulis menerima dokumen *system design* dan rancangan antarmuka dari tim UI/UX *developer*, yang digunakan sebagai acuan dalam menyusun *test case scenario*.

Setelah aplikasi memasuki tahap *staging*, proses pengujian pun dimulai—baik secara manual maupun otomatis. Dalam tahapan ini, penulis menggunakan sejumlah *tools* seperti DBeaver, MongoDB, Solr Admin, Swagger UI, Postman, Redis Insight, Jira, dan QASE. *Tools* tersebut membantu dalam mengakses dan memverifikasi data pada *database*, berinteraksi dengan API untuk mengirim atau menerima data sesuai kebutuhan fitur, serta melakukan manipulasi data seperti pengaturan stok barang atau menambahkan promosi pada produk tertentu di aplikasi.

Selama proses pengujian berlangsung, setiap temuan berupa *bug* atau kesalahan lainnya akan didokumentasikan. Penulis kemudian melakukan koordinasi dengan tim terkait, baik dari sisi *backend developer*, *frontend developer*, maupun tim layanan pendukung lainnya yang terlibat dalam pengembangan aplikasi Alfagift. Setelah *bug* diperbaiki oleh tim *developer*, penulis melakukan *retesting* untuk memastikan bahwa *bug* tersebut telah diselesaikan dan fungsi aplikasi berjalan sesuai harapan.

Setelah seluruh skenario pengujian dalam proyek tersebut selesai dijalankan dan semua temuan *bug* telah diperbaiki serta diuji ulang, maka QA

akan melakukan dokumentasi akhir. Dokumentasi ini bertujuan sebagai referensi untuk proyek-proyek berikutnya atau pengujian di masa mendatang. Proyek tersebut kemudian akan memperoleh status QA *Passed*, dan selanjutnya diserahkan kepada tim *Testing Operations* untuk proses lebih lanjut.

3.2.2 Uraian Kerja Magang

Penulis menjalankan kerja magang mulai dari tanggal 4 Februari 2025 sampai tanggal 4 Agustus 2025 di PT. Global Loyalty Indonesia. Pekerjaan yang dilakukan penulis selama menjalankan magang di PT. Global Loyalty Indonesia dapat dilihat di tabel 3.1.

Tabel 3.1 *Timeline* Kerja Magang

Tugas	Februari				Maret				April				Mei				Juni			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Onboarding Plan for New Joiner																				
Full Cycle Rollout																				
Project Thematic Homepage																				
Project Pembatasan Promo pada Member																				

Project AI Content Generator																			
Project Advanced Search																			

3.2.2.1 *Onboarding Plan for New Joiner* dan *Full Cycle Rollout*

Program magang di PT. Global Loyalty Indonesia berlangsung selama 6 bulan, di mana penulis mengikuti beberapa proyek selama masa tersebut. Pada hari pertama, penulis mendapatkan seorang mentor yang berperan sebagai pembimbing dan pengajar dalam menjalankan tugas di posisi *Quality Assurance*. Semua materi yang harus dipelajari telah dirangkum dalam dokumen Google Sheets berjudul *Onboarding Plan for New Joiner*. Dokumen ini berisi daftar *checklist* materi yang harus dipelajari beserta waktu mulai dan selesai pembelajaran. Dengan adanya dokumen ini, *supervisor* dapat dengan mudah memantau kemajuan anak magang dan menentukan kesiapan untuk terlibat dalam proyek yang sedang berjalan.

Full Cycle Rollout adalah tahap pengujian menyeluruh terhadap seluruh fitur pada aplikasi Alfagift sebelum versi terbaru aplikasi tersebut diluncurkan di Play Store maupun App Store. Tahap ini didominasi oleh *positive test case* yang bertujuan memastikan semua fitur berjalan dengan baik dan sesuai desain, serta memastikan tidak terjadi konflik antar fitur dalam aplikasi.

3.2.2.2 *Setup tools*

Penulis menggunakan berbagai *tools* dan aplikasi yang umum dipakai di divisi *Quality Assurance* (QA), seperti DBeaver, MongoDB, Postman, Swagger

UI, Redis Insight, QASE, serta Jira. Masing-masing tools ini memiliki fungsi spesifik yang mendukung aktivitas QA, mulai dari mengakses dan mengelola *database* (DBeaver dan MongoDB), mengirim dan menerima data melalui API (Postman dan Swagger UI), hingga melakukan pengujian dan pelacakan proyek (QASE.io dan Jira). Keseluruhan *tools* tersebut digunakan secara terpadu guna mempermudah proses pengelolaan data, pengujian aplikasi, dokumentasi, serta koordinasi antar tim dalam rangka memastikan kualitas produk yang dihasilkan. Dengan memanfaatkan berbagai *tools* ini, divisi QA dapat menjalankan tugasnya secara lebih efisien, akurat, dan terorganisir.

QASE adalah sebuah aplikasi berbasis web yang digunakan sebagai *test case management* yang digunakan penulis untuk membuat *test case*, membuat dokumentasi terhadap masing-masing *test case*, dan juga digunakan untuk proses *bug reporting*, dengan bantuan integrasi dengan Jira. QASE juga mempermudah *monitoring progress* pengetesan dari sebuah proyek karena memiliki *real-time progress update* untuk semua *test case* yang dibuat, dan juga memiliki fitur *dashboard* yang mempermudah untuk *monitoring* informasi penting terkait proyek seperti durasi proyek berjalan, jumlah *test case*, lama pengetesan masing-masing *test case*, status dari masing-masing *test case*, jumlah *defects* atau *bugs* yang ditemukan di dalam proyek, dan lainnya.

Jira adalah sebuah aplikasi berbasis web yang digunakan sebagai *project management tools* dari berbagai divisi seperti *Project Manager*, *Product Developer*, *Software Developer*, *Quality Assurance*, *Team Leaders*, *Mobile Developer*, *Backend Developer*, dan masih banyak lagi. Jira membantu mempermudah proses koordinasi seperti proses melakukan *bug report*, di mana jika sebuah *bug* ditemukan, maka akan dilakukan dokumentasi terkait *bug* tersebut dan cara *reproduce bug* tersebut. Kemudian, akan di *assign developer* terkait agar mereka dapat pemberitahuan dan dapat segera melakukan *bug fixing*. Setelah *developer* terkait selesai melakukan proses *bug fixing*, maka status dari *ticket* tersebut akan diubah menjadi *Ready to Test*, dan tim dari QA dapat

melakukan proses *testing bug* untuk memastikan bahwa *bug* tersebut sudah diperbaiki.

DBeaver merupakan aplikasi yang digunakan untuk mengakses berbagai jenis *database* seperti MySQL, PostgreSQL, MariaDB, SQLite, Apache Family, dan lainnya. Penulis memanfaatkan DBeaver untuk mengakses *database* PostgreSQL. Aplikasi ini dipilih karena mudah digunakan, mampu terhubung ke beberapa *database* secara bersamaan, dan memungkinkan pelaksanaan operasi CRUD (*Create, Read, Update, Delete*) dengan cepat dan mudah. Selain itu, DBeaver juga menyediakan fitur untuk menyimpan *query* yang sering dipakai, sehingga pengguna dapat menggunakannya kembali tanpa harus mengingat *query* tersebut secara manual.

MongoDB Compass adalah sebuah aplikasi dengan antarmuka grafis (GUI) yang dirancang untuk memudahkan akses ke *database* jenis MongoDB [9]. Aplikasi ini memiliki tampilan yang interaktif dan ramah pengguna. Beberapa fitur unggulannya meliputi *built-in query bar* yang memungkinkan pencarian data spesifik tanpa harus menulis *query* yang kompleks, serta opsi untuk menampilkan skema *database* yang membantu dalam visualisasi hubungan antar data di dalam *database* tersebut

Solr Admin merupakan antarmuka web dari Apache Solr yang berfungsi untuk memodifikasi, mengelola, dan mengkonfigurasi instance Solr secara mudah dan efisien [10]. Dalam proyek yang dikerjakan oleh penulis, Solr Admin digunakan untuk melakukan pembaruan harga produk serta mengaktifkan kembali barang yang sebelumnya dalam kondisi dinonaktifkan (*disabled*) sehingga tidak dapat ditambahkan ke keranjang belanja (*add-to-cart*). Dengan menggunakan Solr Admin, penulis dapat mengelola data produk dengan lebih cepat dan memastikan bahwa informasi yang tampil di platform selalu akurat dan *up-to-date*.

Penulis menggunakan Swagger UI sebagai alat untuk proses pengiriman dan penerimaan data melalui API yang digunakan dalam proyek. Dengan Swagger UI, penulis dapat secara langsung memeriksa respons JSON dari API

guna memastikan apakah pembaruan data telah berhasil dilakukan [11]. Selain itu, Swagger UI membantu dalam memonitor status proses tersebut, apakah berhasil atau gagal, sehingga mempermudah verifikasi dan *troubleshooting* ketika melakukan pengujian atau integrasi API. Swagger UI juga menyediakan dokumentasi yang interaktif, memungkinkan penulis dan tim *developer* untuk memahami dan menguji berbagai *endpoint* API dengan lebih efisien.

Postman adalah sebuah aplikasi yang digunakan untuk melakukan pengetesan terkait API yang digunakan untuk aplikasi Alfagift. Dengan menggunakan Postman, pengetesan dari fungsi API yang digunakan menjadi lebih mudah karena Postman menyediakan berbagai macam fungsionalitas seperti menggunakan *script* untuk memanipulasi *request* dan *response* dari API sehingga mempercepat proses pengetesan API, memvisualisasikan hasil *response* dari API terutama jika bentuk dari *response* API tersebut merupakan list dari banyak data, dan lainnya.

3.2.2.3 Proyek pengetesan UI Thematic Homepage

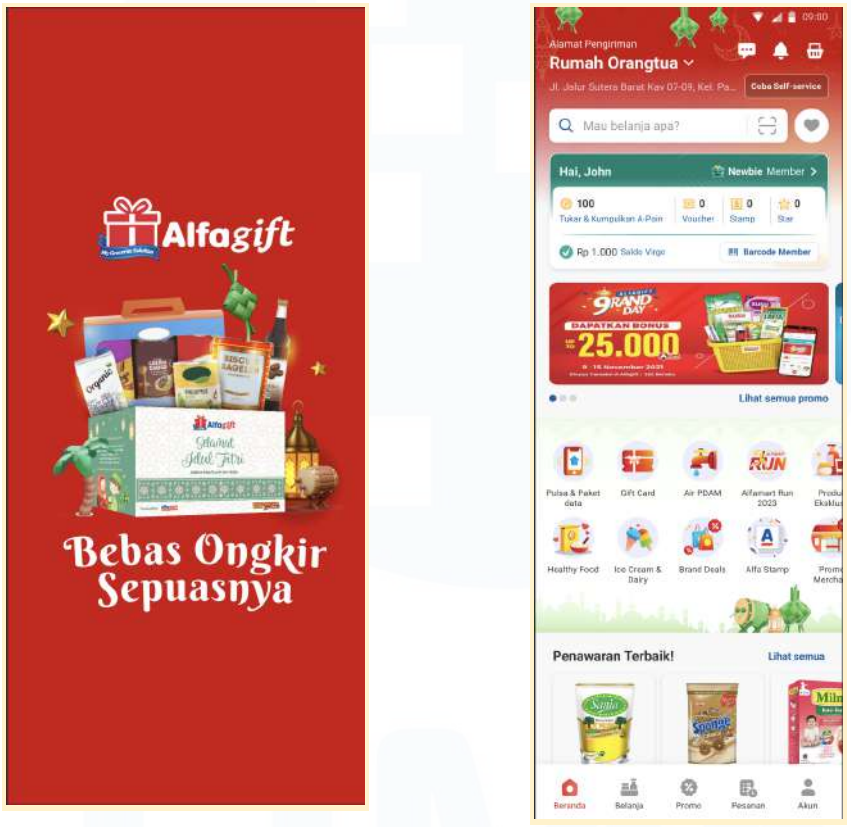
Proyek Thematic Homepage adalah proyek dimana ada penambahan tema baru ketika bulan puasa, dimana tampilan halaman *Homepage* Alfagift mendapatkan aksen-aksen seperti ketupat, bintang dan bulan. Ada juga perubahan terhadap *icon* aplikasi Alfagift dan *splash screen* disaat pengguna membuka aplikasi Alfagift yang dibuat bernuansa bulan puasa. Di dalam proyek ini juga ada penambahan fitur baru dimana akan ada animasi saat pengguna pertama kali melakukan proses *add-to-cart* terhadap suatu produk.

Proses pengetesan proyek *UI Thematic Homepage* dapat dibagi menjadi beberapa tahap:

1. Pembuatan *test case*

Di dalam proyek Thematic Homepage ini, terdapat *business requirement* dan *story list* yang memerlukan untuk memeriksa semua fitur baru tersebut dan harus dipastikan bahwa ketika waktu masuk periode bulan ramadhan, semua rancangan seperti UI *Homepage* di Alfagift, *splash screen* saat membuka aplikasi

Alfagift, dan *icon* aplikasi Alfagift sudah mengikuti tema bulan ramadhan yang telah di desain. Ada juga tambahan pengecekan fitur animasi saat melakukan proses *add-to-cart* produk-produk yang berada di katalog Alfagift. Dari keperluan ini, maka dirancang *test case scenario* yang akan mencakup semua fitur baru di proyek ini.



Gambar 3.5 Tampilan UI *Splash screen* dan *Homepage*

2. Proses *testing scenario* dan dokumentasi

Test case scenario yang dibuat untuk memastikan bahwa semua fitur baru di proyek *Thematic Homepage UI* ini telah sesuai dengan rancangan antara lain adalah *scenario* dimana *user* telah melakukan *update* aplikasi Alfagift dan periode bulan ramadhan telah mulai, maka *expected output* adalah ketika *user* selesai melakukan proses *update* aplikasi Alfagift, maka *icon* aplikasi Alfagift akan berubah menjadi *icon* yang memiliki tema bulan ramadhan. Dari *test case* ini, dibuat juga *negative case* dimana ketika *user* melakukan *update* aplikasi Alfagift dan periode belum memasuki bulan ramadhan atau ketika periode bulan ramadhan

telah selesai, maka *expected output* adalah ketika *user* selesai melakukan *update*, maka *icon* aplikasi Alfagift akan tetap menggunakan *icon default*. Begitu juga terkait *splash screen* dan UI *Homepage*, *test case scenario* dibuat ketika *user* telah melakukan *update* aplikasi Alfagift dan periode bulan ramadhan telah mulai, maka *expected output* adalah ketika *user* selesai melakukan proses *update* aplikasi Alfagift dan *user* membuka aplikasi Alfagift, maka *splash screen* Alfagift akan berubah menjadi yang memiliki tema bulan ramadhan, dan ketika *user* sampai di *Homepage* aplikasi Alfagift, maka akan ada dekorasi dan aksen bulan ramadhan seperti ketupat dan *background* berwarna hijau sesuai dengan tema bulan ramadhan seperti yang ditampilkan di gambar 3.5.

3. *Bug report* dan *retesting bug*

Saat tahap pengetesan scenario, ketika ada output dari sebuah skenario yang tidak sesuai dengan *expected output*, maka akan dibuat sebuah *ticket bug* di Jira yang berisi langkah-langkah untuk melakukan *reproduce bug*, kemudian dokumentasi dari hasil skenario tersebut dibandingkan dengan *expected output* yang diharapkan. Seperti pada skenario ketika *user* telah melakukan *update* pada aplikasi Alfagift dan sudah memasuki periode bulan ramadhan, namun setelah melakukan *update*, *icon* aplikasi tidak berubah dan *UI Homepage* Alfagift juga masih menggunakan yang *default*, maka akan dilakukan *bug report* untuk skenario ini. Setelah *bug* tersebut di *fixing* oleh *developer* terkait, maka pihak QA akan melakukan *retesting bug* untuk memastikan bahwa *bug* tersebut sudah aman dan hasil skenario sudah sesuai dengan *expected output* yang telah ditentukan.

3.2.2.4 Proyek pengetesan mekanisme Pembatasan Promo pada Member

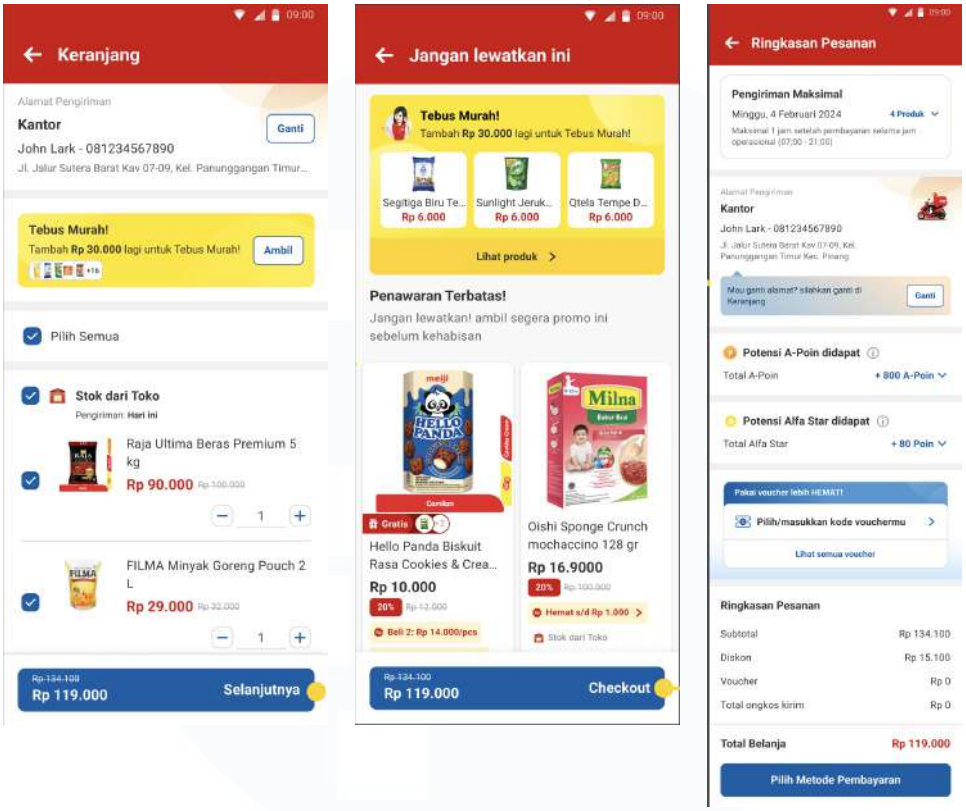
Proyek Pembatasan Promo pada *Member* adalah proyek yang memunculkan mekanisme baru dimana masing-masing *member* memiliki kuota untuk 2 tipe promo yang dapat digunakan, yaitu promo A dan promo B. Masing-masing dari promo A dan B memiliki syarat berbeda yang harus dipenuhi dan juga produk syarat yang berbeda. Mekanisme ini memastikan bahwa pada

setiap *member* di dalam suatu jangka waktu tertentu, hanya ada beberapa kesempatan atau kuota untuk menggunakan promo A dan promo B, dan jika kuota pada *member* tersebut sudah habis, maka member tidak dapat menikmati *benefit* dari promo tersebut.

Proses pengetesan proyek mekanisme Pembatasan Promo pada Member dapat dibagi menjadi beberapa tahap:

1. Pembuatan *test case*

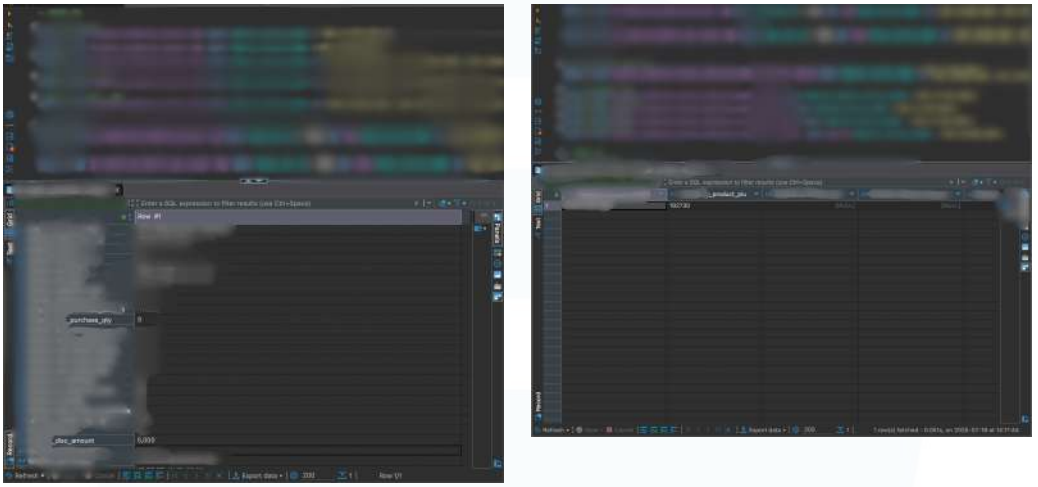
Di dalam proyek ini, terdapat *business requirement* dan *story list* yang memerlukan untuk pengetesan mekanisme baru ini, terutama terkait mekanisme pembatasan atau jumlah kuota promo A dan promo B yang dapat digunakan oleh *member* di dalam suatu periode waktu tertentu. Proses pengetesan berawal dengan perancangan dan pembuatan *test case scenario* yang menggunakan *tools* QASE. Di dalam *web app* QASE, pembuatan *test case scenario* dapat diorganisir dengan rapi sesuai dengan modul yang ditentukan dan proses pengetesan menjadi lebih rapi dan efisien karena proses pengetesan dan dokumentasi dapat dilakukan di satu *platform* yang sama. QASE juga memiliki fitur *dashboard*, yang dapat digunakan untuk *monitoring progress* dari proyek saat proses pengetesan dimulai, agar proyek dapat selesai sesuai dengan *timeline* yang ditentukan. Semua *test case scenario* yang dibuat memiliki komponen yang sama, yaitu *pre-conditions*, *steps*, dan *expected output*. *Pre-conditions* adalah kondisi yang harus dipenuhi sebelum pengetesan *scenario* tersebut dimulai, *steps* adalah langkah-langkah untuk melakukan *scenario* tersebut, dan *expected output* adalah hasil atau sesuatu yang diekspektasikan terjadi saat langkah-langkah dijalankan.



Gambar 3.6 Tampilan proses pembelian produk di Alfagift

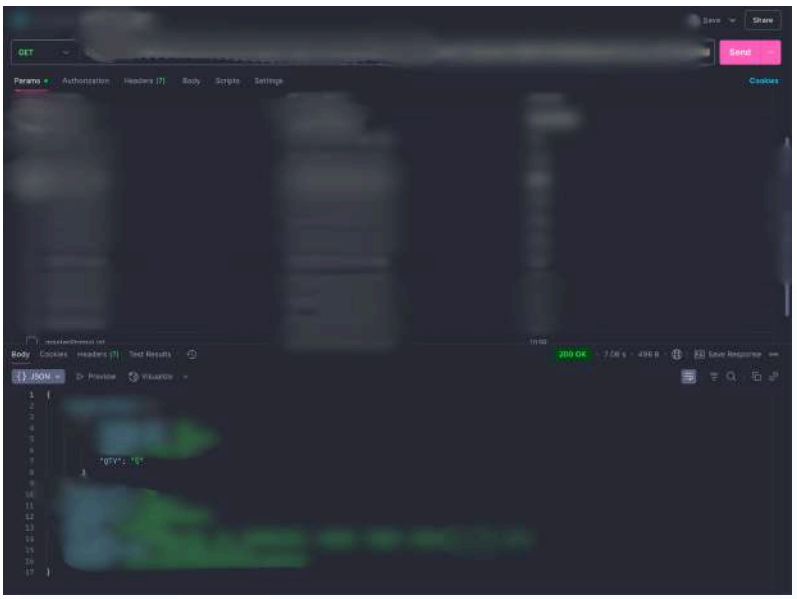
2. Proses *testing scenario* dan dokumentasi

Salah satu contoh dari *test case scenario* yang dibuat dalam proyek ini adalah ketika *member* memiliki kuota sebanyak 3 untuk menggunakan promo A dan melakukan transaksi yang dapat menggunakan promo A tersebut. *Pre-conditions* untuk scenario ini adalah *member* harus memiliki kuota sebanyak 3 untuk promo tipe A, *member* harus melakukan transaksi pembelian produk syarat promo tipe A sesuai dengan *settingan* di dalam *database*, dan *member* harus memenuhi syarat minimal pembelian untuk menggunakan promo A disaat transaksi dilakukan. Proses *setting* atau mengatur produk syarat dari promo A dan syarat pembelian dari promo A dilakukan menggunakan kombinasi dari *tools* seperti MongoDB, Solr Admin, dan DBeaver untuk mengakses dan melakukan proses CRUD (*Create, Read, Update, Delete*) di *database* yang terkait seperti di gambar 3.7.



Gambar 3.7 Hasil *setting* informasi promo menggunakan Dbeaver

Ada juga penggunaan *tools* seperti Postman dan Swagger UI untuk proses pengecekan kuota dari member untuk menggunakan promo A, karena proses ini memerlukan untuk mengirim *request* yang berisi id dari *member* ke API dan melihat *response* yang berisi kuota promo A untuk *member* dari API tersebut seperti di gambar 3.8. *Steps* yang perlu dilakukan saat menjalankan *scenario* ini adalah *member* melakukan proses *add-to-cart* produk syarat promo A, *member* membeli produk A hingga memenuhi minimal pembelian Rp. 10.000, dan *member* melakukan transaksi hingga pembayaran selesai. *Expected output* dari *scenario* ini adalah disaat *member* telah memenuhi syarat untuk promo A, disaat sebelum melakukan pembayaran, maka *member* akan mendapatkan potongan harga sebesar Rp. 1.000. Setelah *member* berhasil melakukan pembayaran dan berhasil mendapatkan *benefit* dari promo A, maka kuota penggunaan promo A akan berkurang menjadi 2. *Flow* dari proses transaksi yang dilakukan *member* dapat dilihat dari gambar 3.6.



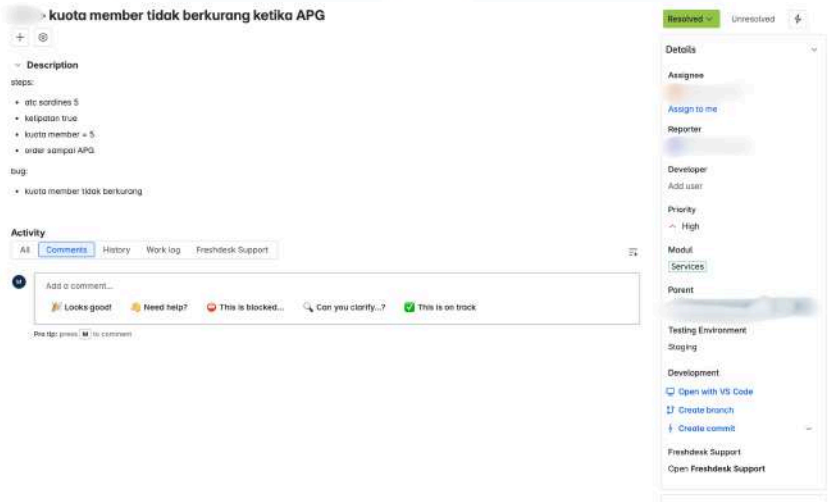
Gambar 3.8 Pengecekan kuota *member* menggunakan API di Postman

Proses pengetesan dilakukan sekaligus dengan mendokumentasi hasil dari masing-masing langkah yang dilakukan sehingga mendapatkan *expected output*.

3. *Bug Report* dan *retesting bug*

Ketika proses pengetesan sebuah scenario dilakukan, dan hasil dari *output* tidak sesuai dengan *expected output* yang telah ditentukan, maka harus dilakukan proses yang dinamakan *bug report*. *Bug report* dilakukan menggunakan *tools* seperti Jira, dimana untuk setiap *ticket bug* yang dibuat, akan dicantumkan *test case scenario* yang terkait, langkah-langkah untuk melakukan *reproduce bug*, kemudian dokumentasi dari hasil scenario tersebut dibandingkan dengan *expected output* yang diharapkan seperti yang dapat dilihat di gambar 3.9. Di dalam *ticket bug* ini juga terdapat informasi penting lainnya seperti *platform* dari *device* yang digunakan saat menemukan *bug* seperti Android atau iOS, nama dari *reporter* atau QA yang menemukan *bug*, nama dari *Developer* yang di *assign* untuk melakukan *fixing* untuk *bug* terkait, tingkat prioritas *bug* seperti *low, medium, high*, dan *blocker* yang akan menentukan *effort* dari *Developer* untuk melakukan *fixing* terhadap *bug* tersebut. Terdapat juga status dari *ticket bug* berupa *Open, In*

Progress, Ready to Test, dan *Done*. Status dari *ticket bug* ini mempermudah untuk pihak QA dan *Developer* untuk *tracking* proses *fixing* dan *retesting* dari sebuah *bug* hingga tuntas.



Gambar 3.9 Contoh tampilan *ticket bug*

Proses pengetesan, dokumentasi, *bug report*, dan *retesting bug* ini dilakukan untuk semua *test case scenario* di dalam proyek ini dan QA harus memastikan bahwa semua scenario yang telah melalui proses pengetesan telah berhasil yang berarti bahwa *output* akhir dari masing-masing scenario telah sesuai dengan *expected output* yang ditentukan. Semua *test case scenario* juga memiliki dokumentasi dari hasil pengetesan masing-masing langkah yang menjadi bukti bahwa *test case scenario* tersebut telah melalui pengetesan dan berstatus *PASSED*. Ketika semua *scenario* di dalam proyek sudah bersifat *PASSED*, maka proyek dapat dikatakan telah berstatus *QA PASSED*, dimana setelah proyek selesai maka tim QA akan melakukan sosialisasi kepada *user* dari masing-masing proyek.

3.2.3 Kendala yang Ditemukan

a. Mendapatkan proyek yang kompleks yang harus selesai sesuai dengan *timeline* yang singkat
b. Banyak hal baru yang harus dipelajari dalam waktu singkat seperti cara menggunakan tools QASE, Jira, Postman, dan lainnya.

3.2.4 Solusi atas Kendala yang Ditemukan

a. Penulis secara rutin berdiskusi dengan rekan QA, *developer*, *product developer*, dan *project manager* terkait *progress* di dalam proyek maupun kendala-kendala yang ditemukan agar proyek tetap dapat selesai sesuai dengan *timeline* yang ditentukan.
b. Untuk mendukung proses pembelajaran, penulis membuat catatan selama mempelajari cara melakukan intervensi terhadap *database* dan menjalankan *query*. Catatan tersebut berguna sebagai bahan *review* setelah menerima penjelasan dari mentor atau rekan kerja. Penulis juga mencatat langkah-langkah penting yang perlu diingat agar dapat menjalankan tugas serupa di kemudian hari tanpa harus bertanya ulang kepada mentor.