

BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

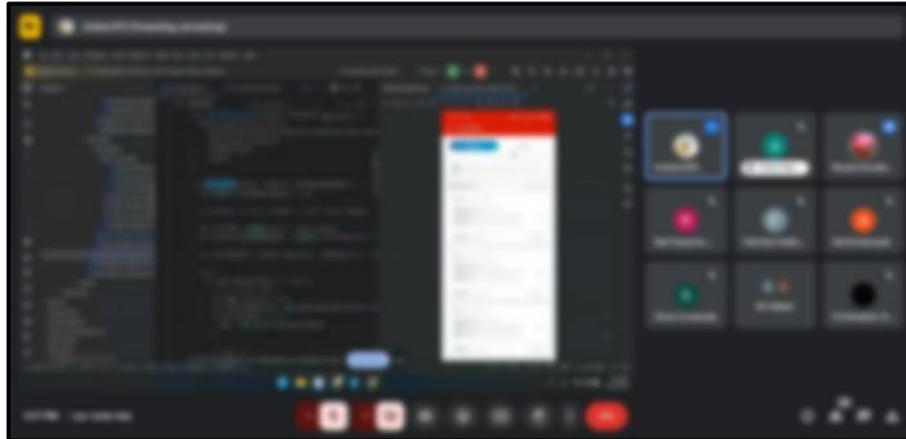
Selama menjalani program magang di PT Global Loyalty Indonesia mahasiswa menempati posisi sebagai Quality Assurance Intern dan tergabung dalam tim Product Operation bersama dengan salah satu mentor, yaitu Kak Rani Wardhani. Tim tersebut terdiri dari total 25 anggota, yang meliputi 1 manager, yang bertindak sebagai penanggung jawab dan pengarah dalam departemen, lalu terdapat 16 karyawan sebagai QA *manual*, 3 karyawan sebagai QA *automation* dan 6 karyawan magang. Karyawan magang berada di bawah supervisi langsung dari manager dengan fokus utama pada proses Quality Assurance (QA) pada mekanisme khusus *refund*, termasuk menyusun skenario pengujian dan melakukan validasi fitur sebelum aplikasi dirilis ke publik. Skenario pengujian ini memuat langkah-langkah detail, *data input*, *expected result*, serta status pengujian (*passed/failed*) yang menjadi acuan dalam proses *testing*. Pengerjaannya memerlukan pemahaman mendalam terhadap alur dan fungsi fitur. Setelah pengujian dilakukan, temuan seperti *bug* atau *error* dilaporkan kepada tim *developer* atau *product manager* untuk diperbaiki. Komunikasi aktif dengan tim menjadi bagian penting dari proses ini, agar kualitas aplikasi tetap terjaga dan dapat memberikan pengalaman pengguna yang optimal.

Koordinasi dan komunikasi dalam tim dilakukan secara fleksibel namun terstruktur. Untuk komunikasi, mayoritas anggota tim menggunakan aplikasi Telegram, baik melalui grup maupun chat pribadi, untuk membahas isu-isu yang berkaitan dengan *project* secara cepat dan responsif. Selain itu, komunikasi juga difasilitasi melalui rapat rutin dan sesi pembaruan yang sudah terjadwal melalui *platform* Google Meet, guna menjaga konsistensi dan akuntabilitas kerja setiap anggota tim. Salah satu bentuk koordinasi utama dalam tim QA di PT Global Loyalty Indonesia adalah kegiatan *daily update* yang dilakukan setiap hari kerja. Melalui sesi ini, setiap anggota tim menyampaikan progres pekerjaan dari hari

sebelumnya, sekaligus menjelaskan rencana aktivitas yang akan dijalankan pada hari tersebut. Kegiatan ini juga menjadi wadah untuk membahas hambatan atau *blocker* yang ditemukan selama proses pengerjaan *project*. Selain itu, sesi *daily* digunakan untuk menindaklanjuti bug yang telah ditemukan namun belum diperbaiki, sehingga dapat dipastikan bahwa isu-isu tersebut terus dipantau hingga terselesaikan.

Selain koordinasi harian, tim juga melakukan *weekly meeting* yang dilaksanakan setiap hari Rabu. Pertemuan mingguan ini bertujuan untuk memperbarui perkembangan masing-masing anggota terhadap *project* yang tengah dikerjakan. Dalam pertemuan ini, setiap anggota divisi Quality Assurance memberikan laporan singkat mengenai capaian kerja selama sepekan terakhir kepada *Project Manager*, sekaligus mendiskusikan rencana kerja yang akan dijalankan di minggu berikutnya. *Weekly meeting* juga menjadi kesempatan untuk membahas kendala yang belum terselesaikan serta menjadi forum bagi manajer tim untuk menyampaikan informasi mengenai *project* baru dan pembagian tugas yang menyertainya.

Sementara itu, koordinasi secara lebih mendalam dilakukan melalui sesi *sprint*, yang umumnya dilaksanakan setiap kali sebuah *project* memasuki milestone tertentu. *Sprint* dilakukan beberapa kali dalam satu *project*, bergantung pada tingkat kompleksitas dan durasi pengerjaan. Dalam sesi ini, tim membahas sistem desain *project*, termasuk penentuan ruang lingkup pekerjaan untuk setiap divisi yang terlibat. Selain itu, diskusi juga mencakup estimasi effort dan timeline *project*, serta pembahasan teknis seperti penggunaan asset, kebutuhan tambahan, dan penyesuaian terhadap *requirement* yang telah ditetapkan sebelumnya secara terbuka. Gambar 3.1 merupakan contoh *meeting sprint* ketika pencapaian atau *milestone* tertentu.



Gambar 3.1 Meeting Sprint *Project*

3.2 Tugas dan Uraian Kerja Magang

3.2.1 Tujuan Kerja Magang

Selama menjalani program magang di PT Global Loyalty Indonesia (GLI), karyawan magang ditempatkan di divisi Quality Assurance untuk berkontribusi dalam memastikan kualitas produk digital, Alfagift, sebelum dirilis ke pengguna. Tujuan kerja magang ini mencakup pemahaman alur pengujian fitur aplikasi, penyusunan skenario *testing*, serta pelaksanaan validasi terhadap fitur yang sedang dikembangkan. Karyawan magang juga berkesempatan untuk mendalami *tools* yang digunakan oleh tim QA, seperti DBeaver untuk pengelolaan *database*, Swagger untuk *testing* API, serta berbagai platform internal yang digunakan dalam pengujian. Selain melakukan pengujian fitur, karyawan magang juga dilibatkan dalam proses melaporkan *bug* dan diskusi bersama dengan tim yang bersangkutan, seperti salah satunya tim *developer*. Proses ini tidak hanya bertujuan untuk memastikan kualitas aplikasi saat perilisan, tetapi juga melatih kemampuan komunikasi tim dan pemahaman terhadap *workflow* QA yang dinamis. Dalam setiap *project* yang dikerjakan, karyawan magang dituntut untuk memahami *system design*, mampu menyusun *scenario-test case* secara kritis berdasarkan Figma, *business requirement*, hingga melakukan pengujian pada fitur yang dikembangkan sesuai dengan *scenario* yang ada.

Tabel 3.1 adalah rincian *project* serta detail pekerjaan yang dilakukan selama proses kerja magang:

Tabel 3.1 Rincian *Project* dan Detail Pekerjaan

| No. | Nama Kegiatan/ <i>Project</i> | Periode Kerja | Deskripsi Pekerjaan |
|-----|---|---|---|
| 1. | Pengenalan Lingkungan Kerja | 3 – 6 Februari 2025 | Kegiatan pengenalan lingkungan kerja, pembuka program kerja magang |
| 2. | <i>Early Guidance</i> | 5 – 10 Februari 2025 | Melakukan <i>setup tools</i> yang menunjang proses kerja magang, <i>setup database</i> , serta <i>management testing</i> |
| 3. | <i>Daily Task Update</i> | Selama <i>project</i> berlangsung | Membuat laporan singkat tentang pekerjaan yang dilakukan setiap hari kerja |
| 4. | <i>Fullcycle Rollout Version Beta</i> | 12 – 13 Februari 2025, 15 – 17 April 2025 | Melakukan pengujian untuk aplikasi <i>Alfagift Beta</i> (sebelum perilisasi menjadi versi <i>production</i>) |
| 5. | <i>Project Revamp Voucher dan Redeem A-Poin</i> | 10 Februari – 6 Maret 2025 | Membuat scenario untuk pengujian fitur katalog produk pada setiap modul, dan proses <i>refund</i> melalui <i>back office</i> |
| | | 11 Maret – 30 April 2025 | Melakukan pengujian terhadap scenario yang dibuat melalui <i>test run</i> untuk fitur katalog produk, keranjang, dan proses <i>refund</i> melalui <i>back office</i> |
| 6. | <i>Project Flash Sale Alfagift (Userguide)</i> | 4 Maret – 10 Maret 2025 | Membuat <i>user guide</i> tentang fitur <i>flash sale</i> , yang merupakan sebuah <i>development</i> baru pada tampilan <i>homepage</i> dan beberapa modul lain sesuai dengan strategi perusahaan |
| 7. | <i>Project Lock Limit Use per User VLP</i> | 8 Mei 2025 – Sekarang | Membuat skenario untuk pengujian kondisi ketika penanganan <i>voucher VLP</i> yang dibatasi penggunaannya dan kondisi penggunaan <i>voucher post-refund</i> |

3.2.2 Uraian Kerja Magang

Pada bagian ini akan dibahas penjelasan tentang pekerjaan yang dilakukan secara mendetail selama proses kerja magang di PT Global Loyalty Indonesia, seperti pada tabel 3.1.

3.2.2.1 Pengenalan Lingkungan Kerja

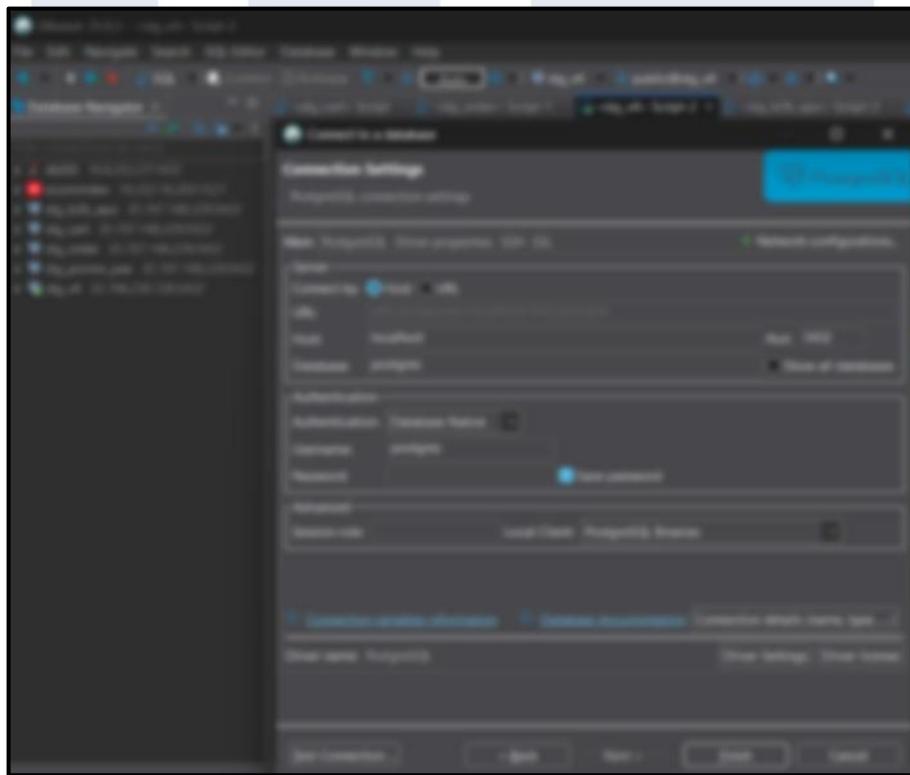
Tanggal 3 Februari 2025 merupakan hari pertama yang menjadi pembuka *program* kerja magang di PT Global Loyalty Indonesia. Karyawan magang dibekali beberapa informasi terkait perusahaan melalui kegiatan *briefing*. Kegiatan *briefing* tersebut mencakup profil perusahaan, sejarah perusahaan, informasi karyawan, hingga *tour* singkat lokasi kantor GLI di Alfa Tower, yang terletak di lantai 24 dan 29. Tentunya peraturan perusahaan juga dijelaskan kepada para karyawan magang pada kegiatan *briefing* ini. Seluruh rangkaian kegiatan ini dilead oleh departemen Human Capital GLI, MCnya adalah Ibu Satifani Mira dan Bapak Pramudita Widiatmoko. Kegiatan berlangsung selama setengah hari kerja, dan setelah kegiatan *briefing* selesai, karyawan magang akan diminta untuk segera bergabung pada departemen masing-masing. Karyawan magang QA ditempatkan pada departemen Product Operation, dibawah tim manager, Bapak Benny Chandra. Setiap karyawan magang mendapatkan mentor, sesuai arahan dari bapak manager. Mentor bertugas untuk memperkenalkan, mengajari, serta membantu karyawan magang selama program magang berlangsung.

3.2.2.2 *Early Guidance*

Sebelum mulai terlibat langsung dalam *project* yang sedang berlangsung, karyawan magang terlebih dahulu diberikan *training* yang bertujuan untuk mempersiapkan kebutuhan teknis selama proses magang. Pada tahap awal ini, dilakukan berbagai pengaturan awal, mulai dari setup perangkat lunak hingga pengenalan *tools* yang digunakan oleh tim QA dalam kegiatan sehari-hari.

Salah satu *tools* utama yang digunakan adalah DBEaver, sebuah *software database management system* (DBMS) yang digunakan untuk mengakses dan memantau data pada sistem [3], khususnya perusahaan. DBEaver memudahkan QA untuk melihat

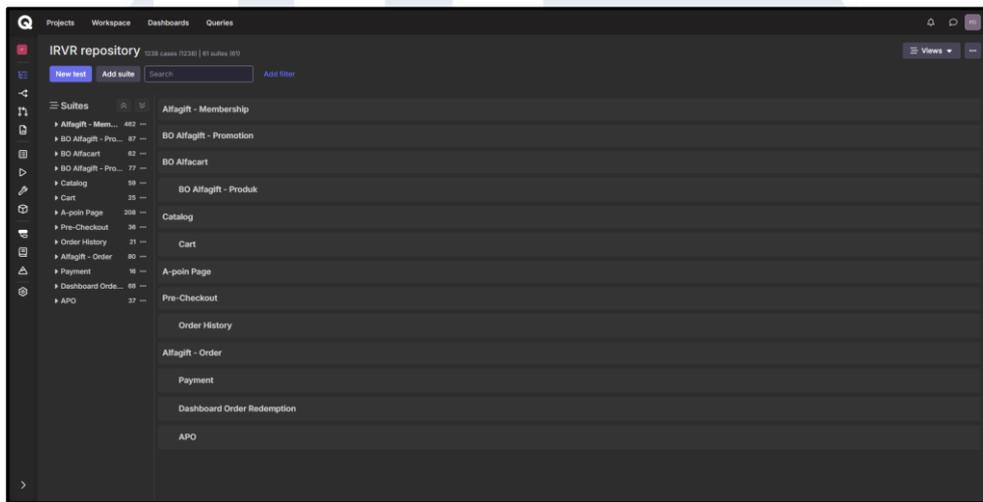
data secara *real-time* yang digunakan oleh aplikasi, terutama saat melakukan validasi *input* dan *output* dari suatu fitur. Sebelum digunakan, karyawan magang perlu menginstal aplikasi ini dan melakukan koneksi ke *database internal* perusahaan menggunakan *username* dan *password* yang telah diberikan pada *host* tertentu. Proses *setup* ini juga menjadi awal pengenalan terhadap struktur data dan skema tabel yang digunakan di sistem *backend*. Gambar 3.2 merupakan tampilan *setup database* melalui DBeaver.



Gambar 3.2 Tampilan *Software Database* DBeaver

Selain itu, terdapat *tools* khusus untuk manajemen *testing* yang digunakan oleh tim QA, yaitu QASE [4]. Di platform ini, karyawan magang mulai mempelajari cara membuat *test case*, menjalankan *test run*, serta mencatat hasil pengujian apakah dinyatakan *passed*, *failed* atau lebih detail lagi *cancel* hingga *skipped* dimana masing-masing hasil menyatakan berbagai arti sehingga berguna sebagai penanda bahwa ada fitur yang berhasil atau perlu

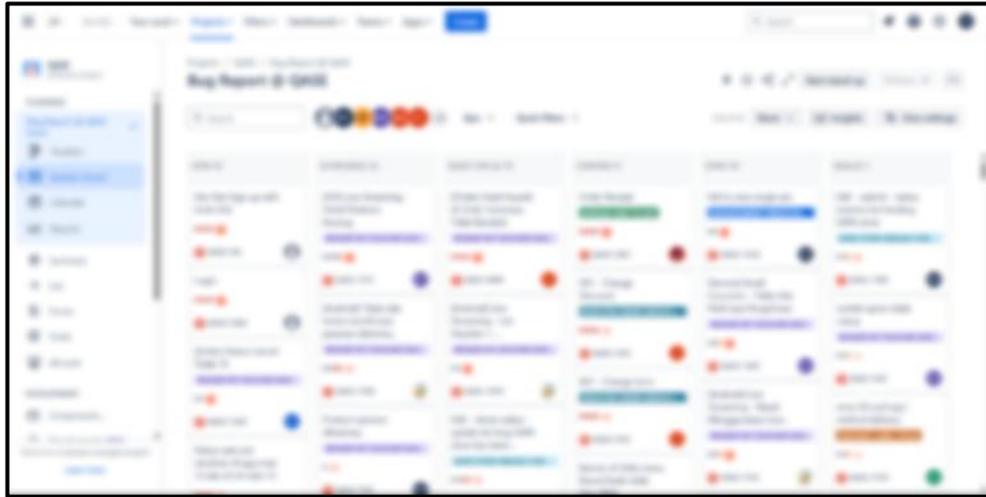
dilakukan *bug fixing*. QASE juga menyediakan struktur yang rapi untuk mencatat skenario pengujian berdasarkan modul aplikasi, sehingga sangat membantu dalam menjaga konsistensi pengujian antar *project*. Gambar 3.3 adalah tampilan situs *web* dari QASE



Gambar 3.3 Tampilan *Website* QASE

Untuk pelaporan *bug*, digunakan *platform* Jira, di mana setiap *bug* atau *error* yang ditemukan dapat dicatat secara rinci lengkap [5]. Penjelasan tersebut mencakup *step to reproduce* dari *bug*, *expected result*, dan *actual result*. Karyawan magang juga diajari cara menulis laporan *bug* yang efektif agar mudah dipahami oleh tim *developer* dan tim *testing operation*. Karyawan magangan yang sistematis di Jira sangat penting karena akan digunakan sebagai acuan tim dalam proses perbaikan fitur seperti tampilan yang terdapat pada Gambar 3.4.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



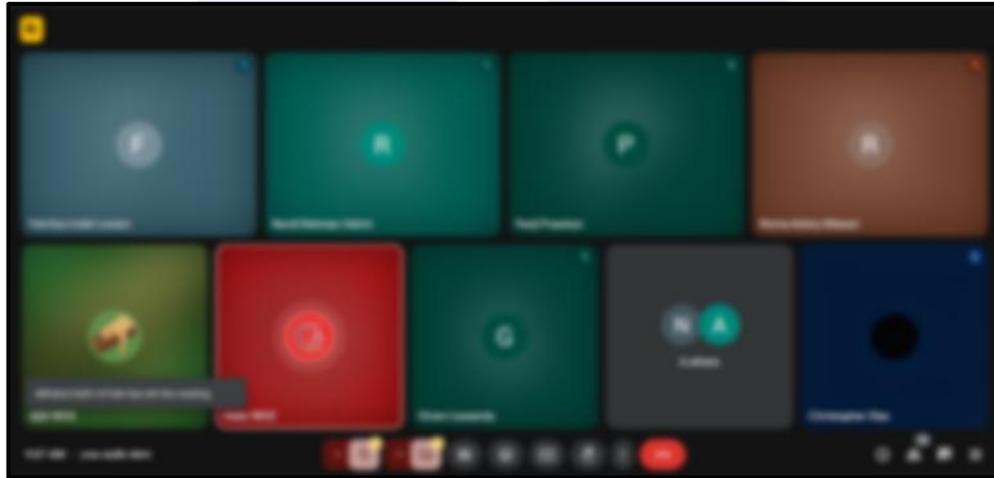
Gambar 3.4 Tampilan *Website* Jira

Tidak hanya dari fungsionalitasnya saja, karyawan magang juga diperkenalkan pada *tools* desain sistem seperti Figma dan Miro. Figma digunakan sebagai acuan UI (*user interface*) dari setiap fitur yang sedang dikembangkan. Di dalamnya, karyawan magang bisa melihat tampilan aplikasi per halaman secara interaktif, sekaligus memahami komponen visual yang harus sesuai saat melakukan pengujian UI. Sementara itu, Miro menjadi media kolaboratif untuk melihat rancangan UI *back-office* yang digunakan oleh admin nantinya pada versi *production*. Kedua *tools* ini sangat membantu dalam memahami konteks fitur yang akan diuji, terutama untuk memastikan bahwa implementasi di aplikasi sudah sesuai dengan desain awal.

3.2.2.3 Daily Task Update

Selama mengerjakan *project*, karyawan magang mengikuti rutinitas pelaporan harian yang menjadi bagian penting dalam proses kerja tim. Gambar 3.5 merupakan pertemuan yang dilakukan setiap pagi secara *online* melalui Google Meet bersama *Project Manager* dan para anggota tim lainnya. Dalam sesi ini, masing-masing orang menyampaikan progres pekerjaan yang telah diselesaikan, rencana aktivitas untuk hari tersebut, serta kendala yang mungkin sedang

dihadapi. Kegiatan ini membantu tim untuk tetap sinkron dan memastikan *project* berjalan sesuai dengan target waktu yang telah ditentukan.



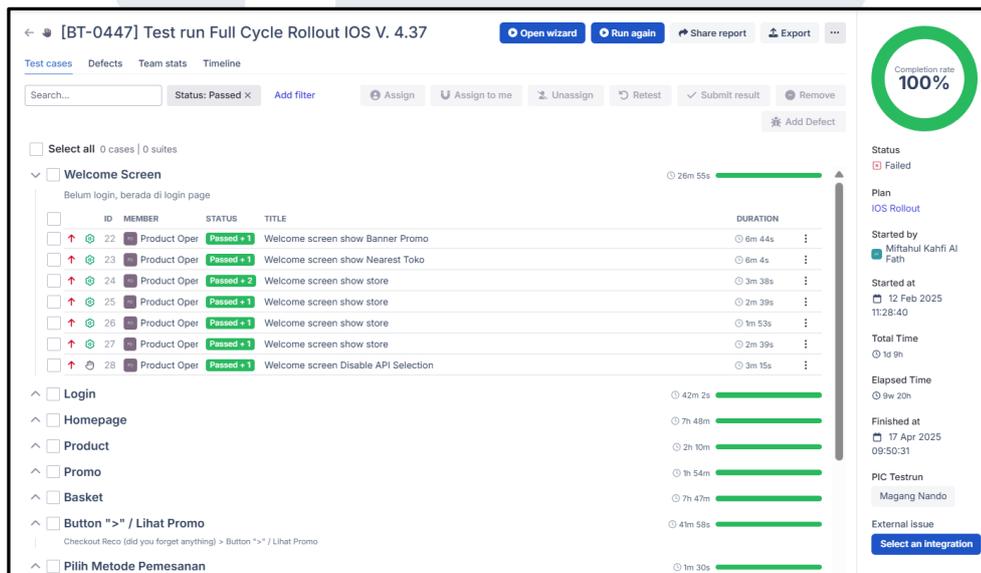
Gambar 3.5 Proses Pelaporan Daily Update

Selain pertemuan untuk membahas *project*, karyawan magang juga diwajibkan untuk melaporkan aktivitas hariannya secara rutin kepada manajer tim atau *supervisor* melalui Whatsapp. Laporan ini berisi rangkuman pekerjaan yang dilakukan dalam bentuk catatan kecil, mulai dari pengerjaan tugas *project* hingga kegiatan pembelajaran seperti *sharing knowledge*, eksplorasi sistem, atau pengenalan *tools* baru. Melalui pelaporan ini, *supervisor* dapat memantau perkembangan dan progres kerja magang secara individu, serta memberikan arahan tambahan jika diperlukan. Kegiatan pelaporan harian ini sejalan dengan praktik yang biasa digunakan dalam pendekatan *Agile*, yang menekankan pentingnya komunikasi dan transparansi dalam tim. Dengan menerapkan sistem ini, proses koordinasi menjadi lebih terarah dan membantu seluruh tim bekerja secara efisien dan saling terhubung.

3.2.2.4 *Fullcycle Rollout Beta*

Fullcycle Rollout merupakan kegiatan yang dilakukan secara rutin oleh tim QA setiap terdapat perlisian aplikasi ke publik,

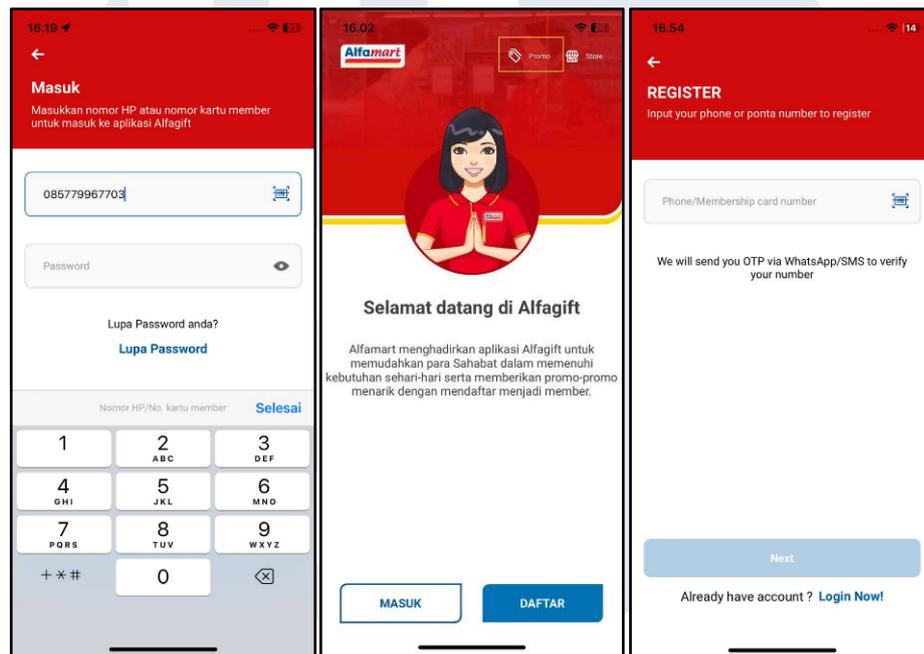
biasanya berlangsung setiap 1 bulan sekali. Kegiatan ini dilakukan untuk memastikan setiap fitur yang *terdevelop* dan fitur sebelumnya berfungsi dengan semestinya dan tidak terdapat bug ketika dilakukan proses perilisan. Seringkali pada proses *rollout* ini tester menemukan beberapa fitur yang bertabrakan dengan fitur yang baru *ter-develop*, sehingga terkadang juga diperlukan komunikasi lebih lanjut dengan tim developer terkait hal tersebut. Kegiatan *Fullcycle Rollout* dilakukan pada aplikasi dengan versi beta. Pada bagian ini, aplikasi *beta* merupakan bentuk *clone* dari versi asli/*production* dengan tambahan fitur yang ingin dirilis atau *merged*. Gambar 3.6 berikut merupakan tampilan *Fullcycle Rollout* dari website yang digunakan (QASE) untuk melakukan pengujian yang terstruktur



Gambar 3.6 *Fullcycle Rollout Beta iOS V. 4.37*

Pada Gambar 3.6, terdapat tampilan dari proses pengujian ketika melakukan *Fullcycle Rollout*, yang dibagi menjadi beberapa bagian seperti modul *Welcome Screen*, *Homepage*, hingga tata cara pemesanan produk pada modul APG (*payment gateway*). Setiap langkah pengujian yang dilakukan didesain untuk menghasilkan *output* tertentu yang dapat dibandingkan dengan *expected result*. Jika hasil yang muncul sesuai dengan yang diharapkan, maka *test*

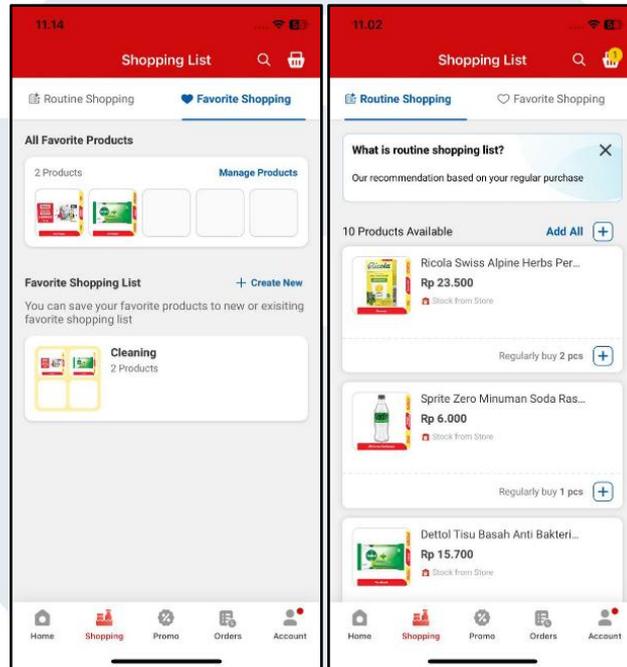
case tersebut akan diberi label *passed*. Namun, jika terdapat bug atau kesalahan, maka statusnya akan menjadi *failed*. Label khusus seperti *cancel*, *skipped* dapat digunakan ketika fitur belum siap diuji atau masih pada tahap *development*



Gambar 3.7 Modul Welcome Screen-Login

Gambar 3.7 merupakan tampilan dari UI modul yang dilakukan *testing*, salah satunya adalah modul *welcome page* dan *login* yang sering diakses *end user* ketika ingin menggunakan aplikasi Alfagift, sehingga ini menjadi salah satu aspek penting yang perlu diperhatikan ketika melakukan pengujian terhadap aplikasi yang ingin dirilis. Skenario pengujian pada modul ini dilakukan ketika *user* membuka aplikasi dan logika sederhana ketika kondisi belum *login*, maka akan muncul *welcome screen* terlebih dahulu. Lalu, ketika *user* ingin mengakses aplikasi, *user* akan diminta untuk mengisi nomor telepon serta *password* mereka terlebih dahulu sebelum masuk ke aplikasi. Jika belum memiliki akun, *user* bisa membuat akun dengan mengeklik *link button* “Daftar”. Dengan

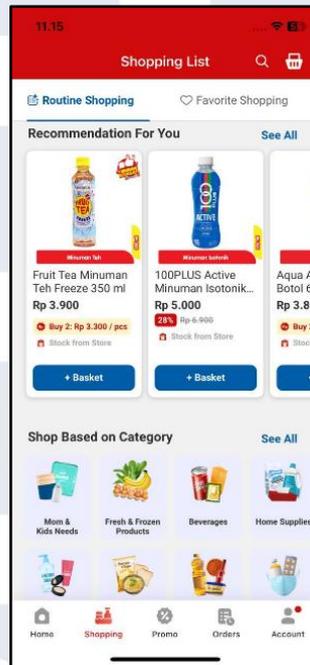
metode *manual*, pengujian modul *welcome screen* dan *login* berhasil melewati tahap uji (*passed*).



Gambar 3.8 Modul Shopping Page

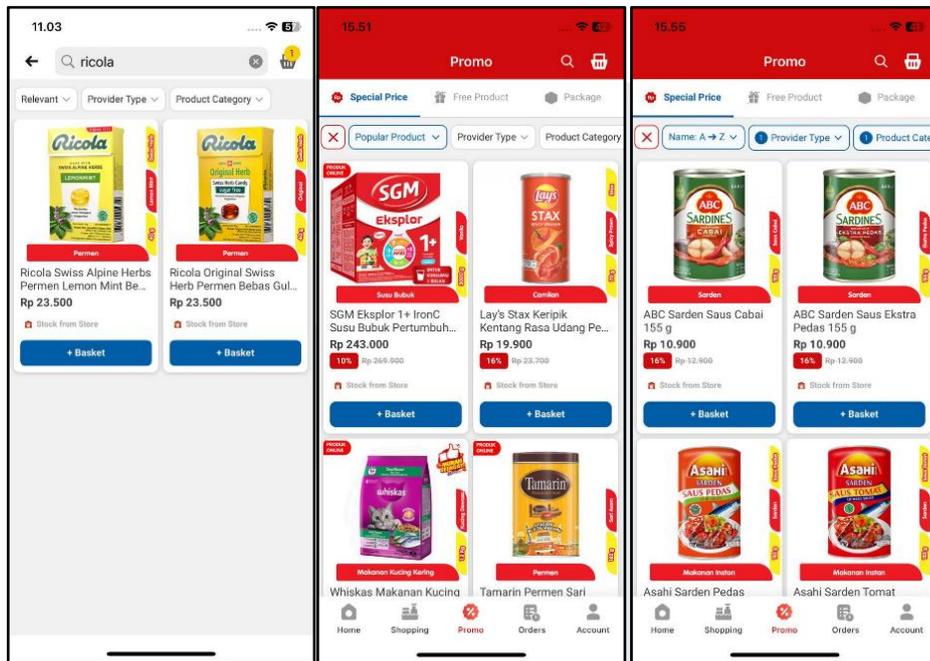
Gambar 3.8 merupakan tampilan modul utama dari *shopping page* yang juga diuji oleh karyawan magang pada *project fullcycle rollout beta*. Pada modul ini, terdapat 2 *tab* yang diuji, yaitu *Routine Shopping* dan *Favorite Shopping*. Kedua submodul tersebut memiliki katalog produk, yang masing-masing memiliki fungsi untuk menyimpan pembelian produk rutin dan produk favorit pelanggan. Pada modul ini, *end user* dapat melakukan interaksi seperti *add to cart* (ATC) produk yang rutin dibeli/favorit dengan menekan tombol + atau “Add All” untuk menambahkan semua produk rutin/favorit. Pengujian yang sama pentingnya adalah ketika karyawan magang perlu menguji proses ATC dalam status tertentu, seperti mengurangi kuantitas produk yang sudah di ATC, memastikan *toast* yang muncul sesuai seperti *system design*, kondisi tidak ada jaringan, dan lainnya. Konsep sederhana diatas

menjelaskan bahwa proses *rollout* pada modul ini menyeluruh pada halaman *shopping* dan hasil dari proses pengujian tersebut adalah *passed* dengan metode pengujian *manual*.



Gambar 3.9 Submodul Produk Rekomendasi dan Kategori Produk

Pada bagian *tab Routine Shopping*, terdapat bagian Produk Rekomendasi dan juga Kategori Produk seperti pada Gambar 3.9. bagian ini juga perlu dilakukan pengujian untuk memastikan fitur tersebut dapat *direct* ke modul yang sesuai dengan *system design*, seperti dapat melihat detail produk jika menekan salah satu produk pada bagian Produk Rekomendasi, lalu dapat membuka halaman kategori produk ketika menekan salah satu *button* pada bagian Kategori Produk. Proses pengujian berhasil (*passed*) melalui metode pengujian *manual*.



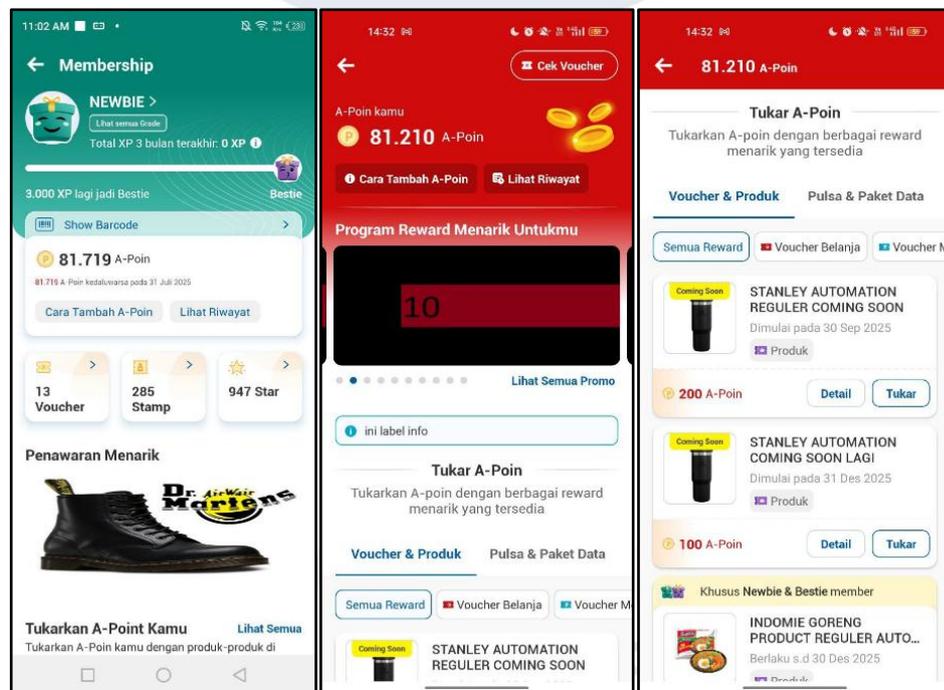
Gambar 3.10 Modul Search dan Filter Produk

Gambar 3.10 merupakan beberapa modul tambahan seperti *search* dan *filter product* juga diuji oleh karyawan magang karena berhubungan erat dengan modul sebelumnya, yaitu *shopping page*. Pada modul ini, fitur yang ada seperti *filtering* dari tipe *promo*, asal penyedia barang, hingga kategori produk, yang juga terdapat pada fitur *search* pada modul *shopping page*. Pengujian fitur harus dilakukan dengan teliti, agar fungsi dapat menampilkan produk sesuai dengan filter yang dipilih oleh *end user* nantinya. Metode pengujian manual pada modul tersebut memberikan hasil *passed*.

3.2.2.5 Project Revamp Voucher dan Redeem A-poin

Project pertama yang ditugaskan kepada karyawan magang adalah *project revamp voucher* dan *redeem A-poin*, pada minggu kedua magang. Umumnya, setiap *project* memiliki beberapa aspek penting yang perlu dipahami oleh setiap tim yang terlibat, salah satunya adalah *system design* seperti *activity diagram*, *class diagram*, dan *wireframe* dari *project* yang dikerjakan. Lebih khusus

lagi pada tim QA, mereka bertugas membuat perancangan untuk *testing scenario* berdasarkan tampilan UI yang sudah dirancang dan membuat segala kemungkinan yang dapat terjadi (*positive* dan *negative case*) dalam modul-modul yang terdapat pada fitur berdasarkan *system design*. *Test scenario* yang sudah terbuat, baik itu proses *testing automation* ataupun *manual*, akan di-import menuju *website* manajemen pengujian, yaitu QASE. Pengujian dapat dilakukan setelah fitur *enhancement*/fitur sudah selesai dikembangkan oleh tim *developer* dan siap dilakukan pengujian. Tim QA segera melakukan sebuah proses pengujian yang disebut dengan *testrun*, yang berisikan *test scenario* yang telah dibuat sebelumnya, memastikan fitur yang telah dikembangkan sesuai dengan *flow requirement* dan bebas dari *bug*. Proses ini dapat berlangsung paralel, terutama ketika *tester* menemukan *bug* dan perlu menunggu perbaikan oleh dari tim *developer* terlebih dahulu



Gambar 3.11 Perubahan Tampilan UI Halaman *Membership* dan *A-poin*

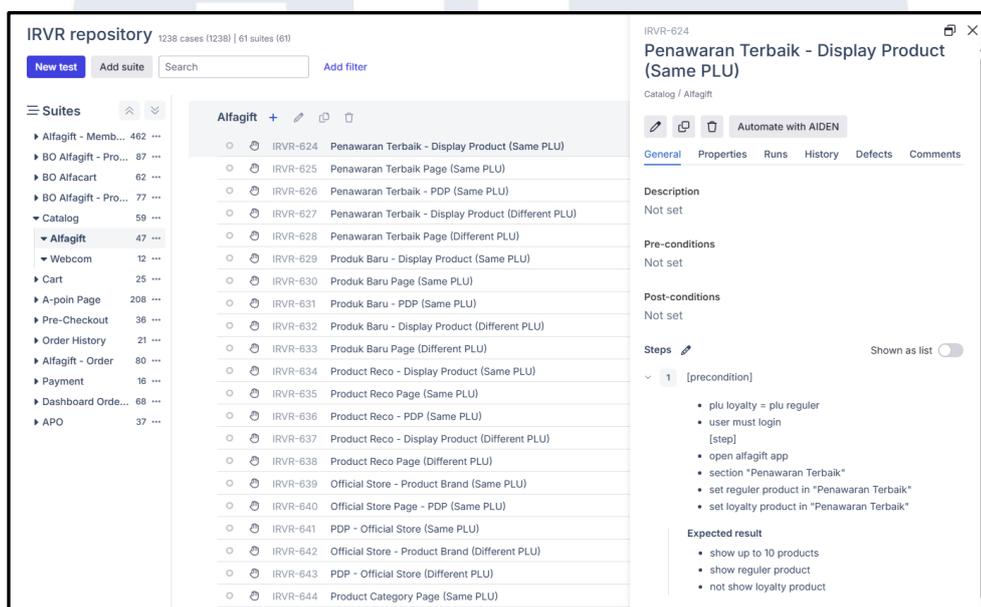
Project ini merupakan sebuah *enhancement* atau pembaruan untuk pemisahan halaman A-poin, yang sebelumnya merupakan modul gabungan dengan halaman *membership*. *Scope* atau cakupan utama pada *project* ini adalah untuk merancang halaman baru serta fitur penukaran poin untuk produk *loyalty* seperti pada Gambar 3.11. Produk *loyalty* memiliki mekanisme khusus, dimana produk tersebut hanya dapat diakses pada halaman A-poin dan penukarannya dibatasi untuk member Alfagift. Selain itu, juga dilakukan *revamp* fitur *voucher* dari fungsi sebelumnya, dengan menambahkan opsi *voucher* yang dapat di *redeem* atau tukar menggunakan A-poin.

The image shows a screenshot of a spreadsheet used as a test plan. The spreadsheet has multiple columns and rows. The columns are color-coded: blue, yellow, grey, orange, and red. The rows contain text, likely representing test cases or project details. The spreadsheet is viewed through a web browser window, with the address bar and browser tabs visible at the top.

Gambar 3.12 *Test Plan Project Revamp*

Gambar 3.12 merupakan tampilan dari sebuah *test plan project*, yang berisikan beberapa detail seperti tentang modul apa saja yang akan terlibat dalam perancangan fitur dan pengujian, *platform* yang digunakan pada fitur yang dikembangkan, status fitur, hingga *person in charge* dari modul atau fitur sesuai pembagian tim internal QA yang terlibat. *Test plan* ini juga mencakup tenggat waktu pekerjaan tim QA akan berlangsung, sesuai dengan koordinasi dari *project manager* pada *meeting sprint*. Dalam

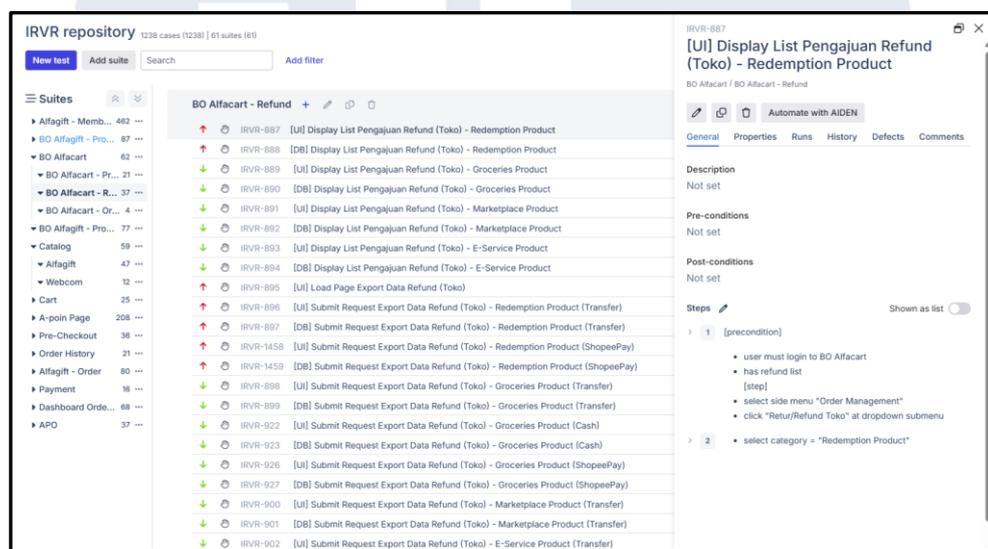
kegiatan ini, dilakukan pembagian tugas, dan karyawan magang berkontribusi dengan mengembangkan *test scenario* khusus pada modul katalog produk Alfagift. Fokus utamanya adalah memastikan bahwa integrasi produk *loyalty* yang sedang dikembangkan tidak menyebabkan konflik atau mengganggu alur yang sudah berjalan pada produk regular seperti produk *groceries*, *marketplace*, dan lainnya.



Gambar 3.13 *Test Scenario* untuk Katalog Produk

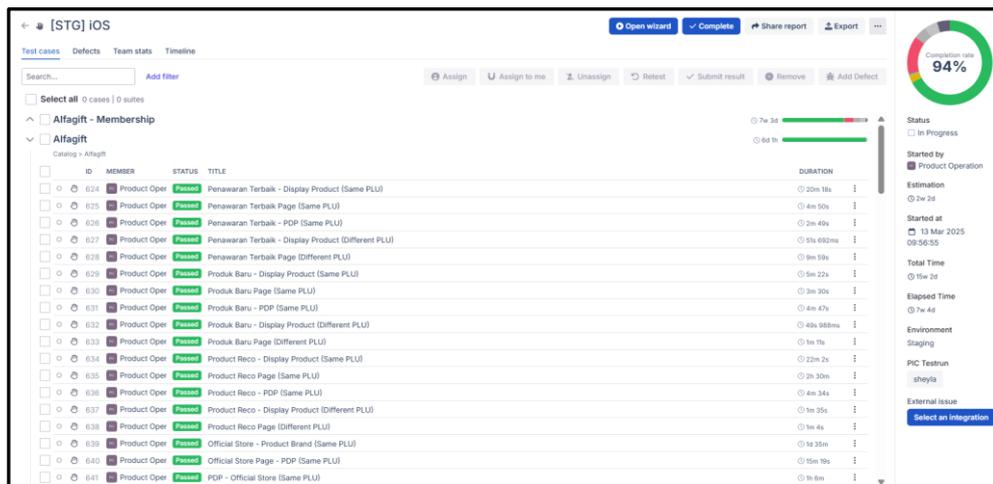
Gambar 3.13 merupakan tampilan dari *test scenario* yang disusun secara sistematis oleh tim Quality Assurance (QA) yang terlibat dalam *project* ini. *Test scenario* tersebut terdiri dari berbagai *test case* yang masing-masing dirancang untuk menguji fitur-fitur spesifik dari modul pada aplikasi atau *platform* yang sedang dikembangkan. Modul katalog produk yang diuji mencakup beberapa bagian utama, seperti daftar produk rekomendasi yang muncul di halaman utama (*homepage*), serta produk yang ditampilkan di halaman promosi (*promo page*). Salah satu skenario pengujian yang cukup krusial adalah pengecekan terhadap sebuah ID, yang disebut sebagai PLU. PLU atau *price lookup* adalah sebuah

identifikasi unik yang mengambil informasi harga produk dari *database* sistem dan menampilkannya di aplikasi. Tes ini bertujuan untuk mengamati bagaimana sistem bereaksi jika ada dua produk yang memiliki PLU yang sama, atau sebaliknya jika PLU-nya berbeda. Hal ini penting karena berpengaruh terhadap logika tampilan produk di dalam aplikasi, seperti produk apa yang akan muncul dan yang disembunyikan.



Gambar 3.14 *Test Scenario* Proses Refund melalui *Back-office*

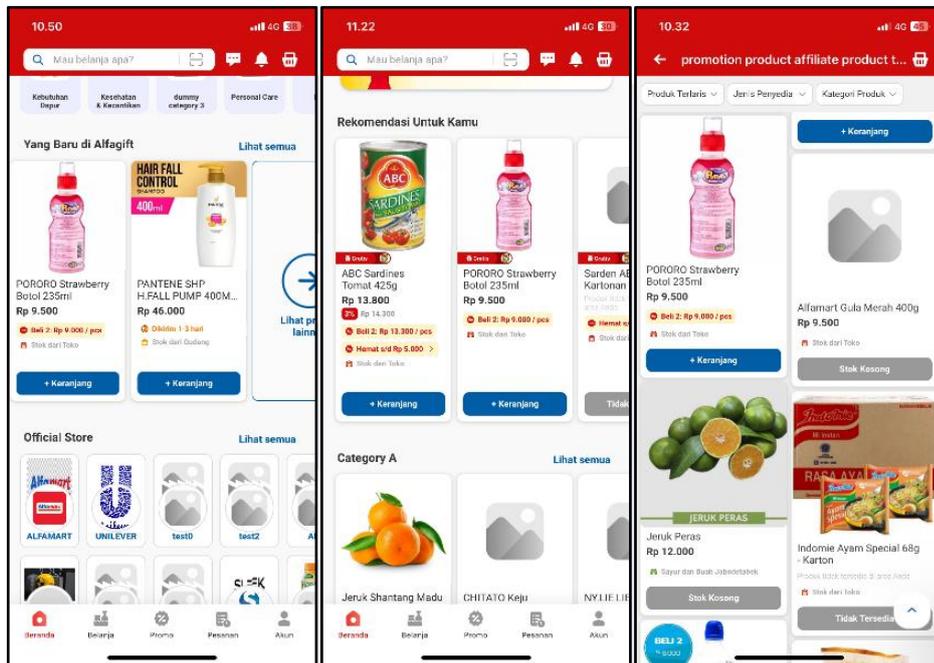
Gambar 3.14 merupakan salah satu contoh bagian kerja karyawan magang pada *project* ini, dengan menyusun *test scenario* untuk modul *refund page* pada sistem *back-office*. Pembaruan ini difokuskan pada metode pengembalian untuk jenis produk *loyalty*, yang hanya dapat diakses oleh tim *customer service* ketika menerima keluhan atau permintaan pengembalian dari pelanggan. *Test scenario* tersebut mencakup serangkaian langkah pengujian yang berfokus pada beberapa aspek penting, seperti pengajuan *refund* dalam berbagai status pesanan, pengembalian poin atau dana secara penuh kepada pelanggan, serta memastikan bahwa fitur baru yang dikembangkan tidak mengganggu proses *refund* yang telah berjalan sebelumnya.



Gambar 3.15 Tampilan *Testrun* untuk *Device iOS*

Setelah *test scenario* selesai terbuat dan fitur sudah dikembangkan atau setidaknya dasar fungsinya sudah dapat dilakukan pengujian, maka tahapan kerja selanjutnya adalah melakukan *test run*. Gambar 3.15 merupakan beberapa *test scenario* yang siap diuji dalam modul *testrun*. Tahap tersebut mencakup proses pengujian dengan memperhatikan kesesuaian dengan *test scenario* dan *system design*. Sama seperti kegiatan *fullycycle rollout*, *test run* menggunakan *website QASE* untuk melakukan manajemen pengujian. Terdapat delapan modul utama yang diuji pada versi Alfagift Staging, seperti halaman A-poin, katalog produk, keranjang, dan lainnya. *Testrun* tersebut berjumlah 72 *cases* dari keseluruhan modul, yaitu 904 *cases*

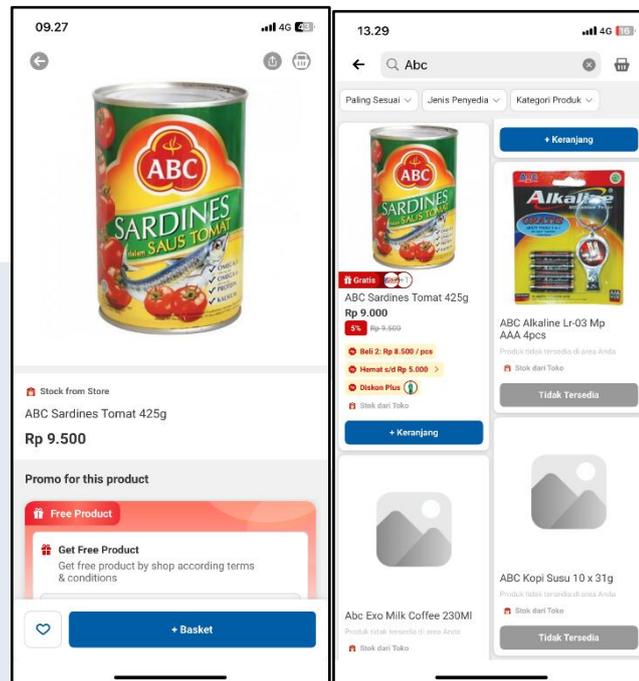
U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



Gambar 3.16 Testrun Modul Katalog (Homepage)

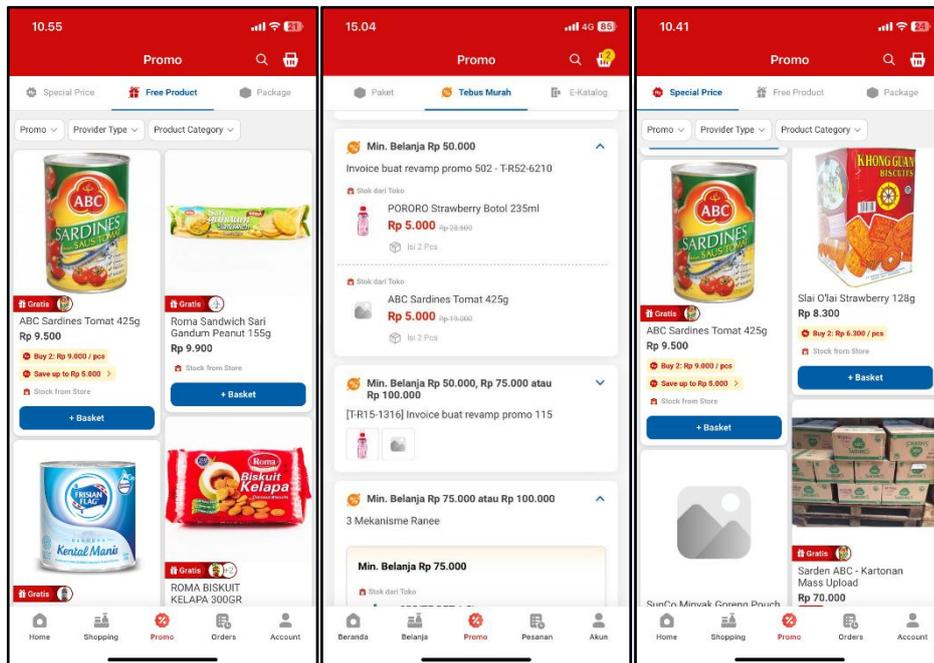
Gambar 3.16 adalah sekumpulan katalog produk yang sudah ter-setting untuk menguji fitur yang ditambahkan pada sistem. Beberapa modul diantaranya seperti *homepage*, *promo page*, dan *shopping page*. Karyawan magang perlu memastikan produk yang tersusun pada katalog modul terkait agar tidak memunculkan produk sejenis *loyalty* jika produk memiliki *value* PLU yang sama. *Homepage* memiliki beberapa submodul didalamnya, seperti *slide menu*, “Yang Baru di Alfagift” *card*, “Official Store”, “Rekomendasi untuk Kamu” *card*, dan lainnya. Masing-masing submodul memiliki daftar produk yang berkemungkinan memiliki PLU yang sama juga dengan produk *loyalty*

UNIVERSITAS
MULTIMEDIA
NUSANTARA



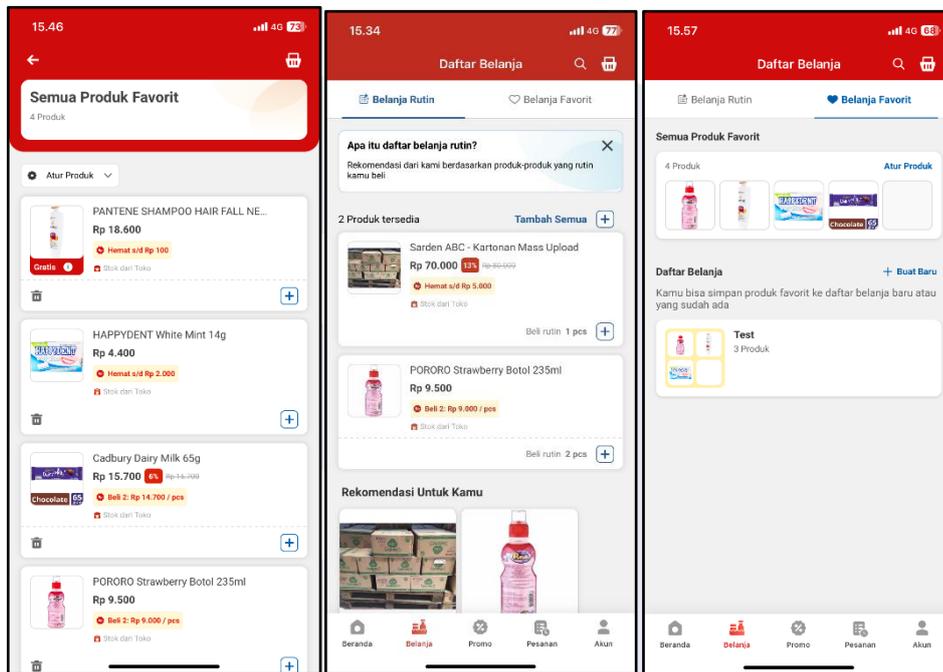
Gambar 3.17 Testrun Modul Katalog (Search - PDP)

Gambar 3.17 juga menunjukkan pengujian katalog yang muncul pada fitur lain seperti *search* dan *product detail page* (PDP). Pada fitur yang menampilkan katalog perlu diuji lebih menyeluruh, karena katalog produk *groceries* bersinggungan dengan katalog milik produk *loyalty*. Halaman *search* memiliki sebuah *search bar* yang dapat digunakan juga untuk mencari produk berdasarkan nama, oleh karena itu pula fitur ini perlu dilakukan *enhancement* dimana produk *loyalty* tidak dapat dicari menggunakan fitur ini. Sesuai dengan kebutuhan bisnis, produk *loyalty* hanya dapat diakses pada halaman A-poin. Sedangkan untuk PDP, jika produk yang ditampilkan memiliki *value* PLU yang sama, maka detail produk yang ditampilkan harus mengambil data sesuai dengan informasi yang dimiliki oleh produk *groceries* terkait.



Gambar 3.18 Testrun Modul Katalog (*Promo Page*)

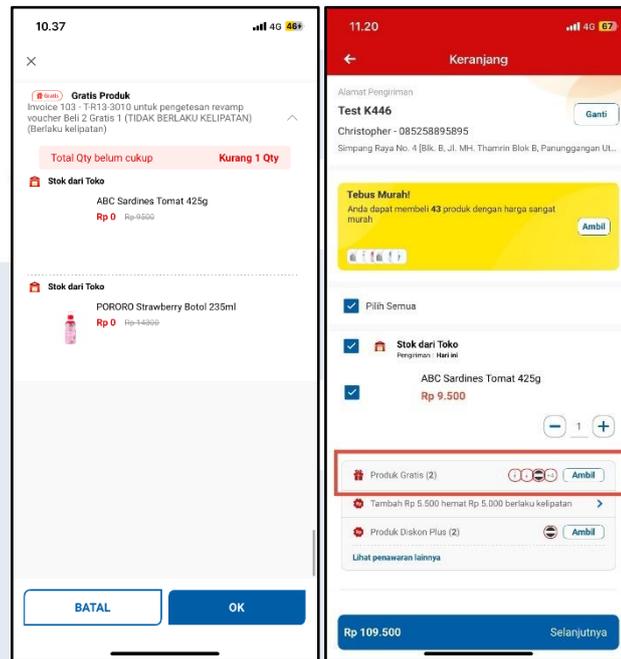
Gambar 3.18 merupakan modul katalog yang berada pada halaman *promo*. Terdapat 5 bagian utama pada halaman *promo*, yang dibagi kedalam *tab* atau *chips*, seperti “Special Price”, “Free Product”, “Paket”, “Tebus Murah”, dan “E-catalogue”. Secara garis besar, pengujian yang dilakukan sama seperti pada modul katalog sebelumnya, yaitu memastikan jika produk *loyalty* tidak ditampilkan pada katalog produk *groceries*. Pada kasus tertentu, seperti pada *tab* “Tebus Murah”, terdapat daftar produk *benefit*, yang juga memiliki kemungkinan *value* PLU yang sama seperti produk *loyalty*. Oleh karena itu, diperlukannya pengujian lebih pada submodul tersebut agar fitur dapat berjalan dengan baik tanpa mengalami *bug* yang mengganggu sistem/fitur yang sudah ada.



Gambar 3.19 Testrun Modul Katalog (*Shopping Page*)

Selanjutnya, modul katalog juga terdapat pada halaman belanja (*shopping page*). Gambar 3.19 merupakan modul katalog yang berada pada *shopping page*. Pada halaman ini, terdapat dua bagian utama, yaitu “Belanja Rutin” dan “Belanja Favorit”. Kedua bagian tersebut juga memiliki katalog didalamnya, dimana pengguna dapat membuat daftar belanja pribadi pada bagian belanja favorit. Tentunya hal ini dapat menimbulkan masalah jika pengguna bisa menambahkan produk *loyalty* menjadi bagian dari daftar belanja mereka, sehingga diperlukan juga pengujian pada katalog *shopping page* sehingga produk *loyalty* tidak dapat diakses melalui katalog pada halaman ini

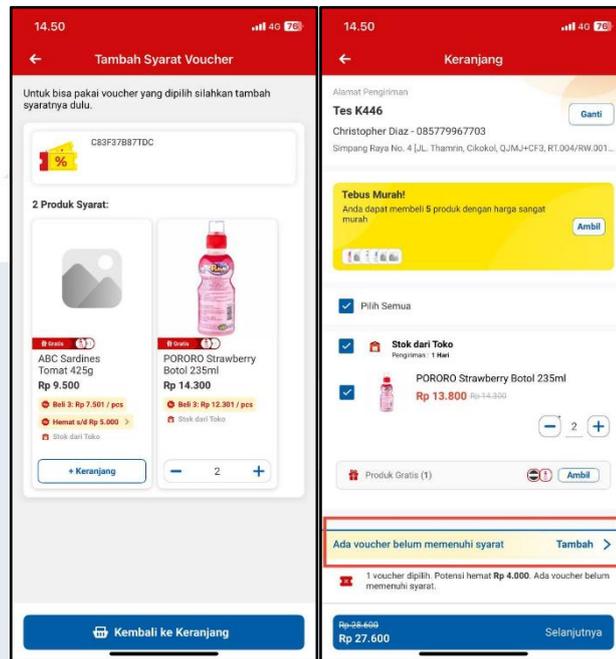
UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.20 *Testrun* Modul *Cart* untuk Penggunaan *Promo*

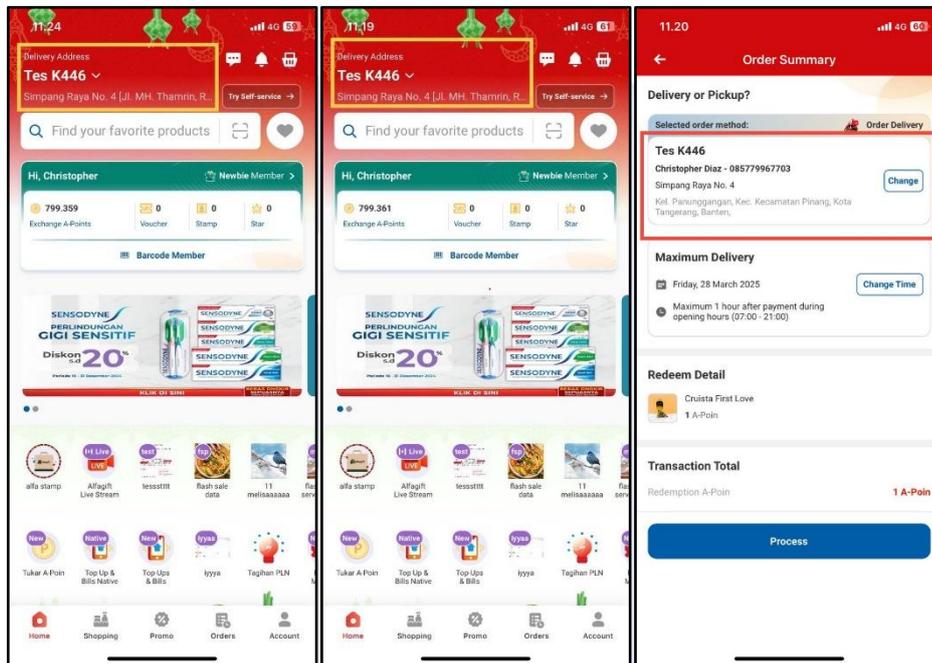
Gambar 3.20 adalah contoh pengujian yang melibatkan penggunaan *promo*, seperti pembelian gratis produk, persentase diskon, hingga potongan harga. Pada modul tersebut digunakan *promo* produk gratis yang didapatkan ketika pengguna membeli lebih dari sekian jumlah untuk produk tertentu. Hal yang perlu dipastikan ketika melakukan pengujian ketika produk bonus/*benefit* memiliki *value* PLU yang sama dengan produk *loyalty*.

U M W N
 U N I V E R S I T A S
 M U L T I M E D I A
 N U S A N T A R A



Gambar 3.21 *Testrun* Modul *Cart* untuk Penggunaan *Voucher*

Selain penggunaan *voucher*, pada *project* ini juga dilakukan pengujian terhadap penggunaan *voucher* yang berpotensi memunculkan produk *loyalty* pada persyaratan penggunaannya. Pada Gambar 3.21, *voucher* digunakan dalam kondisi yang belum terpenuhi, sehingga pengguna akan diarahkan untuk menambah produk ke dalam keranjang sehingga *voucher* dapat digunakan pada pesanan belanja. Potensi munculnya produk persyaratan yang memiliki *value* PLU yang sama dengan salah satu produk *loyalty* bisa saja terjadi, sehingga perlunya dilakukan pengujian khusus pada penggunaan *voucher* ketika berbelanja.



Gambar 3.22 *Testrun* Modul *Cart* Cek Alamat Sebelum dan Sesudah *Redeem*

Gambar 3.22 merupakan pengujian yang berfokus pada perubahan Alamat yang ditampilkan pada *homepage* ketika melakukan penukaran A-poin, karena berdasarkan sistem yang dibangun, terdapat pembuatan alamat baru yang berbeda dengan alamat pada umumnya digunakan ketika berbelanja produk *groceries* Alfagift. Seperti pada tampilan tersebut, data alamat yang ditampilkan pada *homepage* harus tetap sama setelah dilakukannya *redeem* atau penukaran A-poin.

U
M
N
UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.23 Perbandingan *Chucker Response* Aplikasi dengan *PLU Loyalty*

Selanjutnya, metode pengecekan *value* PLU diperlukan sebuah *chucker*, yaitu sebuah *library* untuk menginspeksi *HTTPS request* dan *response* dari aplikasi. Pada katalog tersebut terdapat produk regular yang memiliki *value* PLU yang sama dengan produk *loyalty*, yaitu produk minuman “PORORO Strawberry”. Berdasarkan Gambar 3.23, *value* PLU produk minuman tersebut sama dengan PLU yang ada pada salah satu produk *loyalty*.

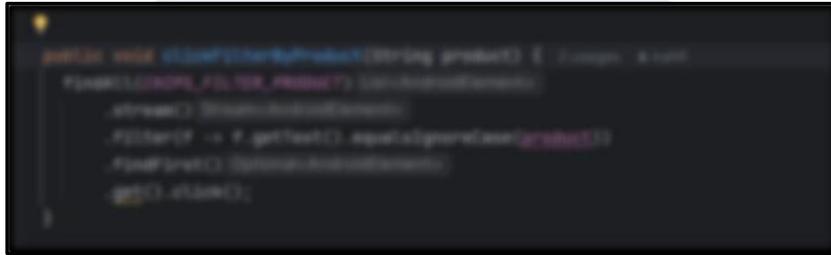


Gambar 3.24 *Software* Android Studio dan Appium

Karyawan magang juga diarahkan untuk melakukan persiapan *automation testing* terhadap aplikasi Alfigift yang sudah dilakukan tahap pengembangan, khususnya untuk fitur *voucher* dan penukaran poin. Proses diawali dengan melakukan *setup environment*, termasuk *emulator* Android sebagai perangkat virtual untuk menjalankan aplikasi menggunakan Android Studio. Di sisi lain, Appium yang merupakan *framework* otomatisasi yang dikembangkan berdasarkan arsitektur Selenium, dipasang sebagai *tools* utama untuk melakukan *automation testing* melalui identifikasi UI. Gambar 3.24 merupakan *software* yang digunakan dalam melakukan *automation testing*. Melalui *emulator*, aplikasi Alfigift di-*install* dan dijalankan agar bisa berinteraksi dengan Appium dalam proses pengujian. Sebelum mulai menjalankan tes, karyawan magang perlu memastikan bahwa koneksi perangkat ke sistem pengujian sudah terdeteksi melalui perintah pada CMD. Dari hasil pengecekan ini, nama perangkat digunakan sebagai identifikasi dalam *script automation*.

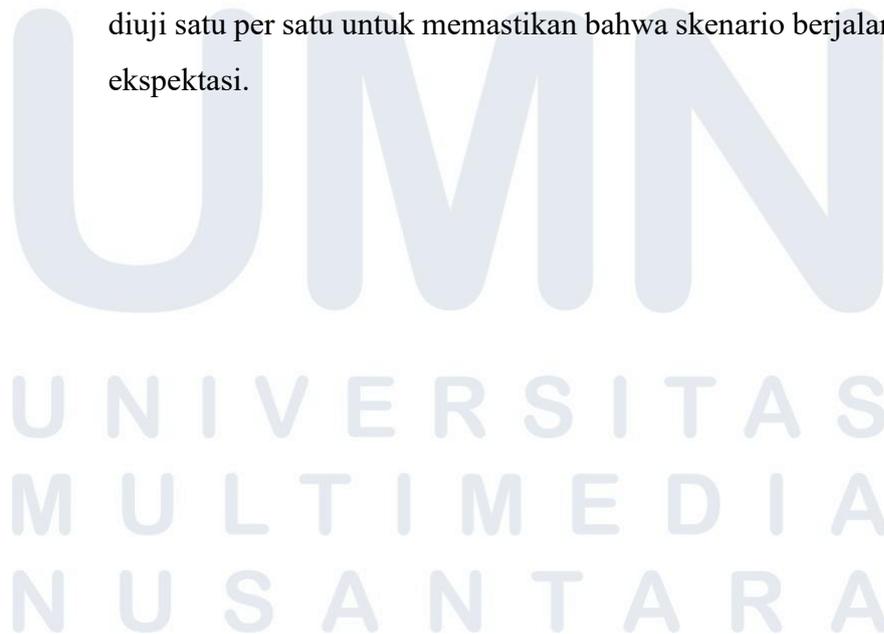
Setelah perangkat sudah siap, tahap selanjutnya adalah melakukan konfigurasi awal pada *file automation code*. Di sini, karyawan magang perlu mengisi informasi seperti nama perangkat (UDID), sistem operasi, jalur file APK aplikasi, serta pengaturan *driver automation* seperti penggunaan UIAutomator. Penyesuaian ini penting untuk memastikan bahwa script bisa berjalan lancar sesuai dengan *target environment* yang disediakan. Kegiatan ini juga mengenalkan karyawan magang pada konsep dasar *automation framework* serta bagaimana menghubungkan antar komponen sistem agar pengujian berjalan secara otomatis. Jika pengaturan sudah benar, sistem akan mampu menjalankan aplikasi secara otomatis ke tampilan awal aplikasi seperti *splash screen* dan *homepage*,

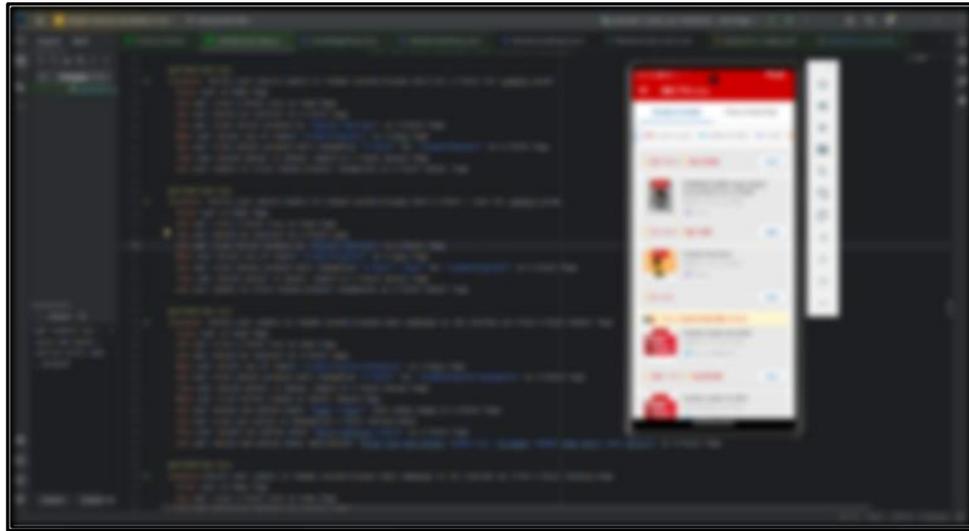
sehingga *test case* terkait pada modul tertentu dapat mulai dieksekusi pada *test run*.



Gambar 3.25 *Function* untuk melakukan deteksi *chips* dan menekan filter

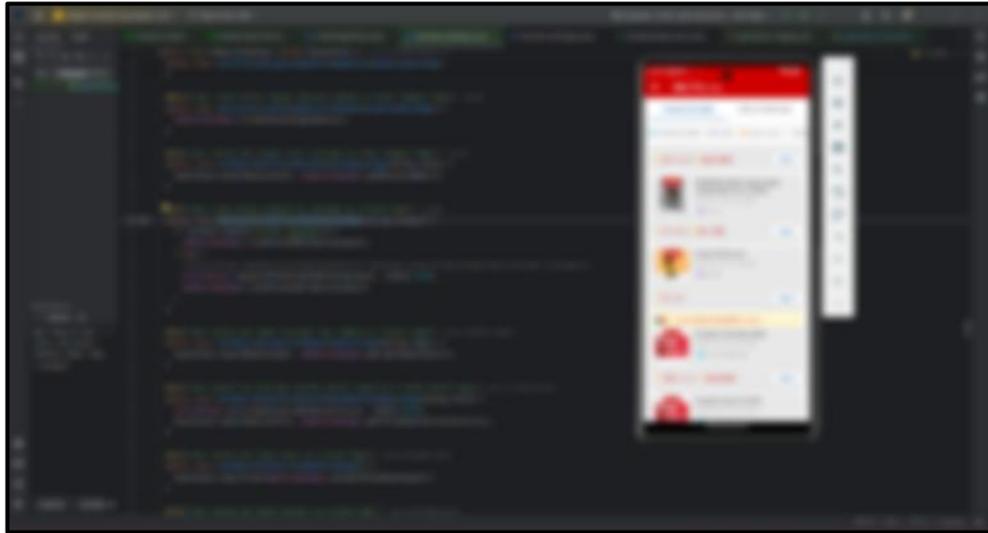
Untuk mengenali elemen yang akan dites, digunakan Appium Inspector sebagai alat bantu untuk mendeteksi komponen-komponen UI (*locator*) di dalam aplikasi seperti contoh pada Gambar 3.25. Proses ini cukup krusial karena elemen-elemen seperti tombol, teks, dan *input field* harus diidentifikasi terlebih dahulu agar bisa dijadikan *locator* dalam *script* pengujian. Dengan menggunakan Inspector, karyawan magang dapat menekan langsung bagian UI tertentu pada aplikasi yang berjalan di *emulator*, lalu mendapatkan informasi seperti ID, *class*, atau *xpath* elemen tersebut. Semua elemen yang diperlukan untuk proses automation dicatat dan diuji satu per satu untuk memastikan bahwa skenario berjalan sesuai ekspektasi.



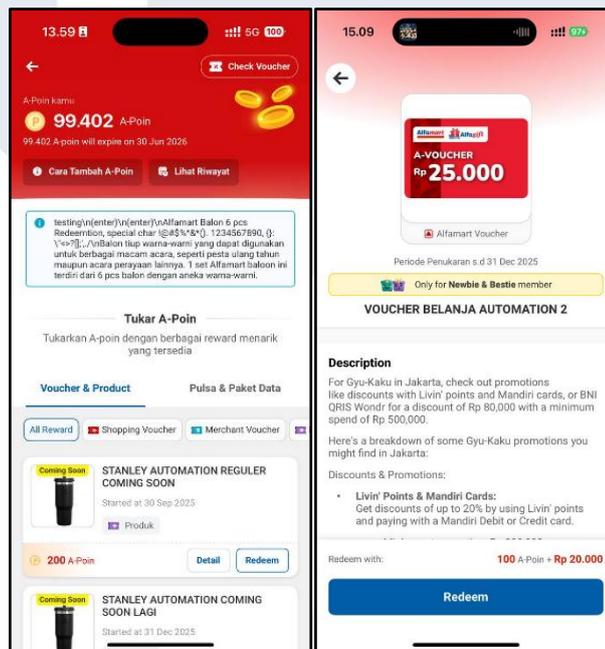


Gambar 3.26 *Automation Scenario Redeem Voucher*

Gambar 3.26 adalah proses pembuatan *scenario* pada *automation testing* untuk memastikan alur *redeem voucher* dan *A-poin* berjalan sesuai dengan ketentuan *project* dalam beberapa kondisi tertentu. Proses tersebut menerapkan pendekatan yang dikenal sebagai BDD (*Behaviour Driven Development*), yaitu metode karyawan magangan skenario pengujian yang memadukan kebutuhan bisnis dengan teknis pengujian dalam format yang mudah dipahami oleh semua pihak. BDD disusun berdasarkan enam aspek utama, yang meliputi *Feature* (fitur yang diuji), *Scenario* (skenario spesifik yang ingin diuji), *Given* (kondisi awal), *When* (aksi pengguna), dan *Then* (hasil yang diharapkan). Pendekatan ini sangat membantu tim QA karena selain mudah dibaca oleh tim teknis, struktur BDD juga memudahkan komunikasi dengan pihak non-teknis seperti tim bisnis atau *Project Manager*. Skenario yang disusun di tahap awal jadi pondasi utama dalam *automation testing*, karena semua tahapan berikutnya, termasuk implementasi dan validasi, akan mengacu pada *scenario* tersebut



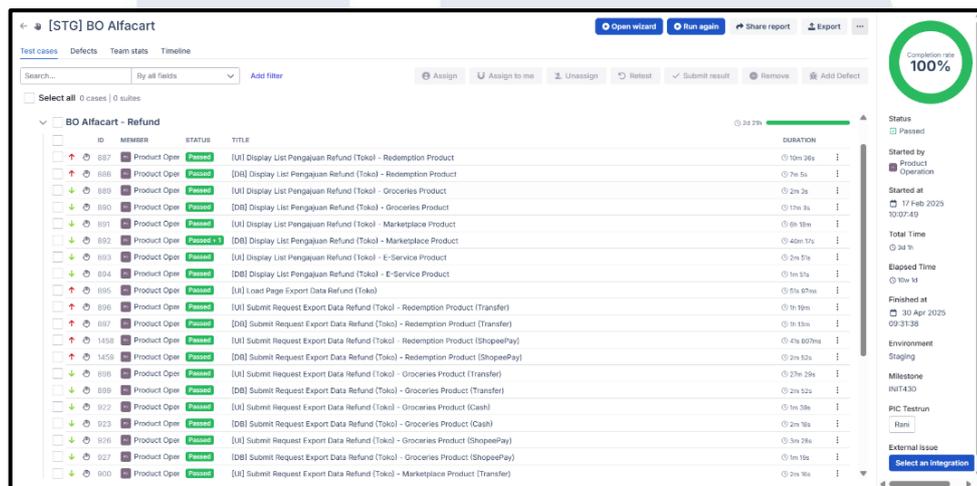
Gambar 3.27 Automation Step Redeem A-Poin



Gambar 3.28 Testrun Modul Halaman A-poin

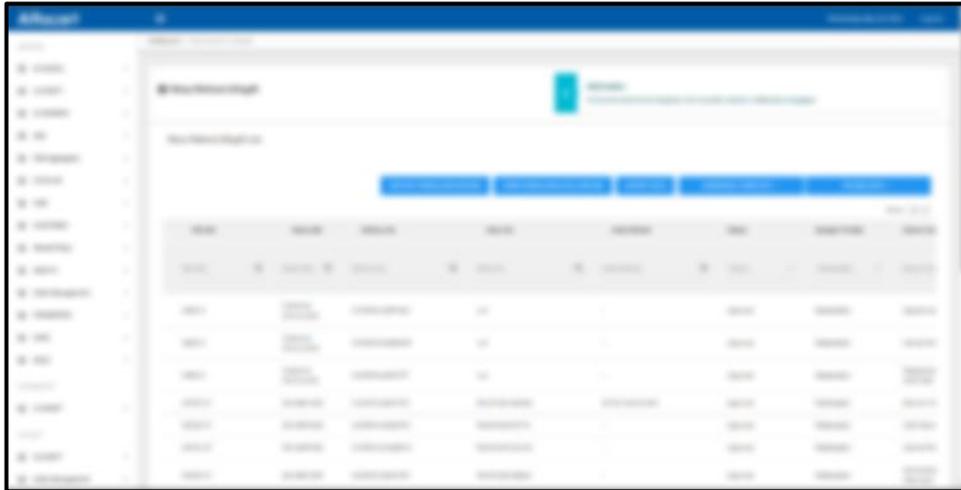
Gambar 3.27 merupakan sebuah *script* kode yang digunakan untuk melakukan pengujian terhadap *scenario automation* yang telah terbuat sebelumnya. Berbasis pada Java, *script* tersebut menguji beberapa fitur yang ada pada halaman A-poin, seperti melihat *value* dari A-poin yang dimiliki pengguna dan mengeklik tombol untuk melihat detail produk *loyalty*. Gambar 3.28

merupakan tampilan halaman yang diuji pada *testing automation* ini. Ketika kode dieksekusi, *emulator* akan diarahkan untuk melakukan *action* sesuai *step* yang tertulis pada *script* kode. Setelah seluruh pengujian selesai dilakukan, sistem akan menampilkan hasil atau *output*, seperti *completion rate*, *passed test case*, *failed test case*, dan *success rate*.



Gambar 3.29 Tampilan *Testrun* untuk *Refund* Melalui *Back-office*

Selanjutnya, karyawan magang juga mengerjakan *testrun* setelah membuat *scenario* untuk pengujian proses *refund* melalui *back-office*. Gambar 3.29 merupakan beberapa daftar *scenario* yang akan diuji dalam *testrun*. Pada modul ini, proses *refund* akan dibagi menjadi tiga bagian utama, yaitu menampilkan *list* pengajuan retur/*refund*, *export data* pengajuan *refund*, dan melakukan pengajuan *refund* melalui *back-office*. Masing-masing bagian memiliki halaman yang dapat diakses dengan menekan pada tombol tertentu. *Testrun* ini berjumlah 29 *cases* dari total 50 *cases* untuk modul *back-office*. Secara keseluruhan, proses *refund* menjadi modul yang cukup krusial pada *project* ini.



Gambar 3.30 Testrun Back-office Retur/Refund Page

Gambar 3.30 merupakan tampilan dari halaman *refund* yang diuji pada *back-office*. Karyawan magang perlu memastikan setiap pesanan, terutama dengan produk *loyalty*, yang dilakukan *refund* dapat ditampilkan pada halaman ini sesuai dengan kondisi yang berlaku. Terdapat tiga jenis produk *loyalty* yang dapat dipesan oleh pengguna, yaitu *regular*, *GLI*, dan *digital*. Masing-masing produk memiliki kondisi yang wajib dipenuhi sebelum melakukan proses *refund*, diantaranya harus sudah pada tahap pengiriman. Pengajuan *refund* yang sudah disetujui akan dapat tampil pada daftar pesanan di *retur/refund page*. Selain itu, pengujian regresi juga diperlukan untuk proses *refund* produk selain *loyalty*, sehingga fitur yang terbuat tidak mengganggu fitur yang sudah ada.

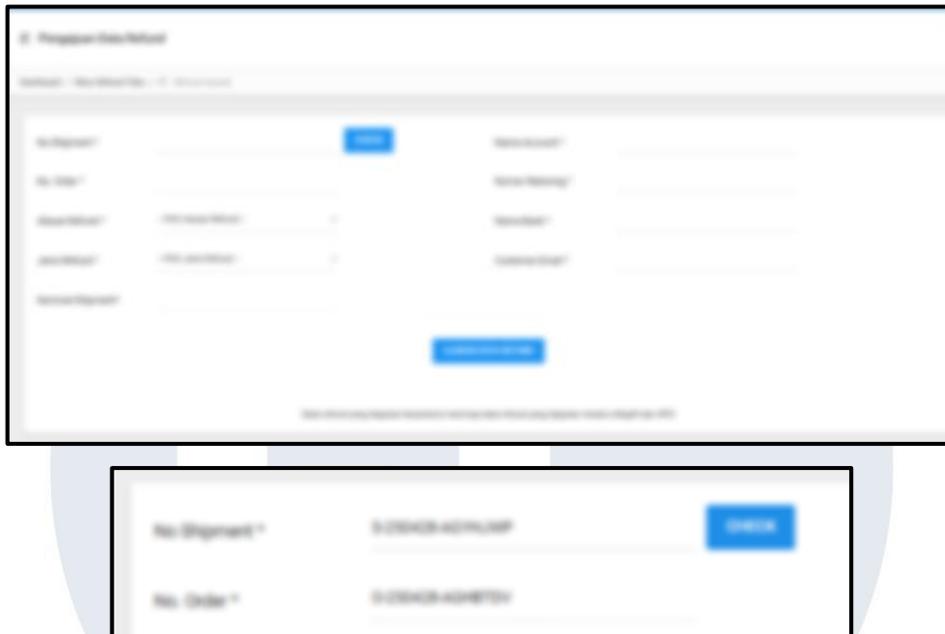
U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



Gambar 3.31 Testrun Back-office Export Data Refund

Gambar 3.31 menampilkan halaman *export data* yang digunakan oleh *customer service* untuk mengambil data dan mengubahnya ke format *.xlsx* sebelum dikirimkan melalui *email*. Terdapat beberapa *field* yang perlu dipilih, seperti jenis *refund*, kategori produk, atau diisi jika memerlukan data spesifik, seperti nomor pengiriman, tanggal pesanan, dan tentunya *email* tujuan pengiriman *data export*. File akan dikirimkan segera setelah tombol “Send” ditekan. Tujuan dari pengujian tersebut adalah untuk memastikan jika produk berjenis *loyalty* (yang ditandai dengan kategori *redemption*) dapat dilakukan proses *export* menjadi file berformat *.xlsx* seperti halnya produk kategori lain.

UIN
UNIVERSITAS
MULTIMEDIA
NUSANTARA



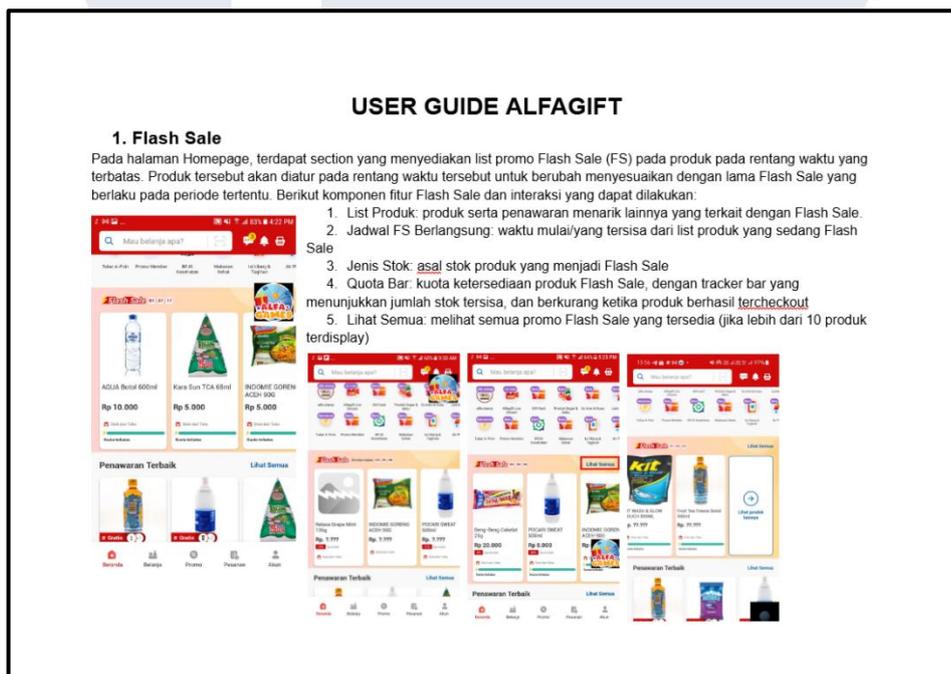
Gambar 3.32 *Testrun Back-office Form Pengajuan Refund*

Gambar 3.32 merupakan halaman *form* untuk melakukan pengajuan *refund* melalui *back-office*. Untuk melakukan *refund*, *customer service* dapat memasukkan nomor pengiriman untuk dilakukan pengecekan terlebih dahulu apakah pesanan dapat dilakukan *retur/refund* atau tidak. Pesanan yang dapat dilakukan *refund* ditandai jika nomor pesanan muncul setelah *customer service* mengisi nomor pengiriman dan menekan tombol “Check”. Setelah mengisi nomor pengiriman dan memenuhi persyaratan, *customer service* perlu mengisi *field* lainnya seperti “Alasan Refund”, “Jenis Refund”, hingga informasi tujuan pembayaran *refund*.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A

3.2.2.6 Project Flash Sale Alfagift (Userguide)

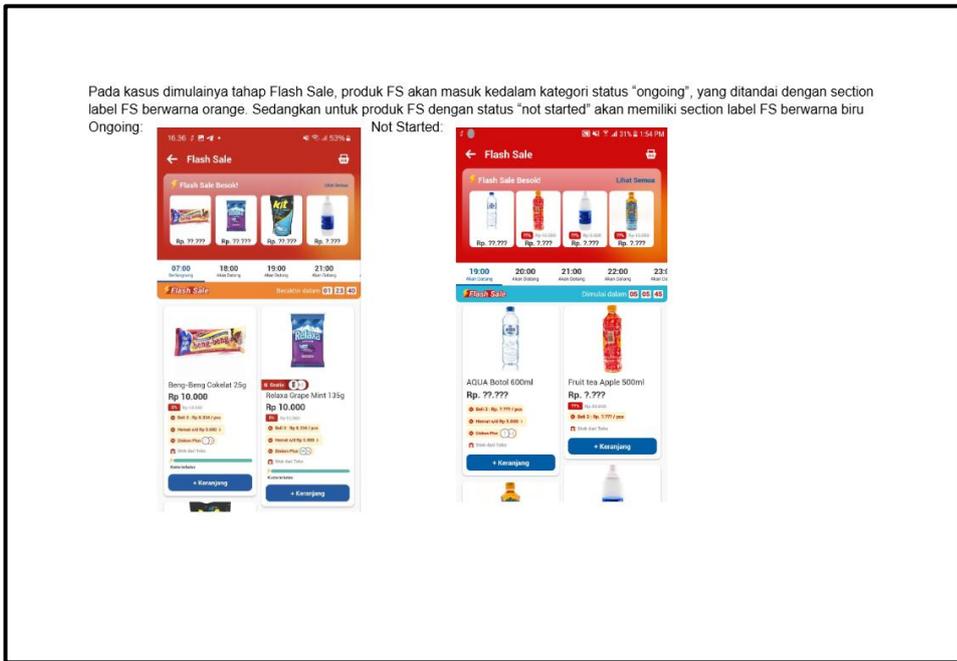
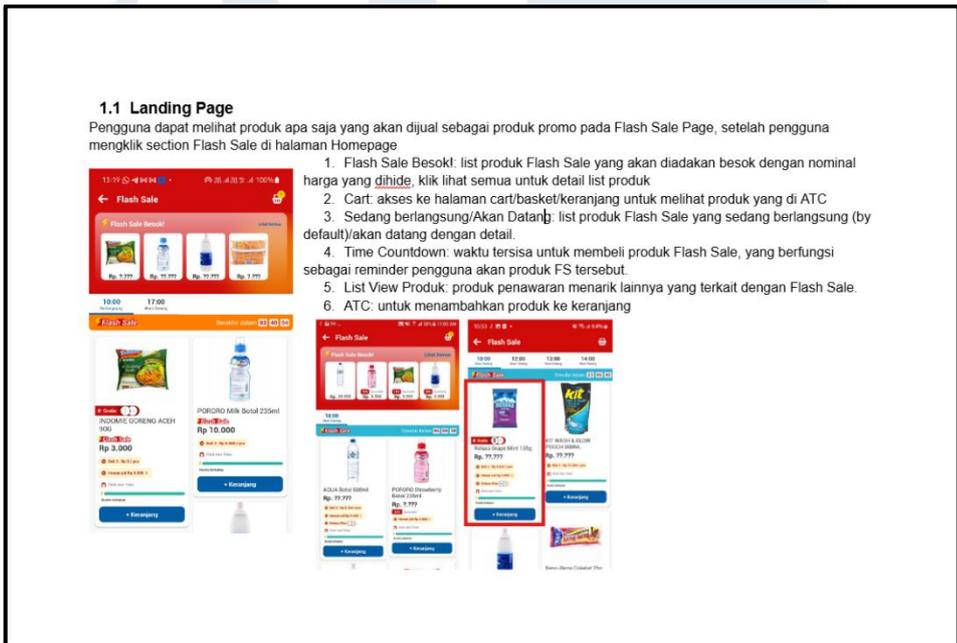
Selama masa magang, karyawan magang juga diberikan kesempatan untuk terlibat dalam *project flash sale* Alfagift. *Project* ini merupakan pengembangan sebuah fitur baru yang menawarkan penawaran eksklusif dengan harga spesial kepada pelanggan, mencakup berbagai produk mulai dari kebutuhan sehari-hari hingga layanan digital, yang hanya tersedia dalam periode waktu terbatas. Dalam *project* ini, karyawan magang berkontribusi dengan menyusun user guide yang mendokumentasikan fitur-fitur utama serta menyediakan panduan bagi tim selama proses pengembangan.*project*



Gambar 3.33 Userguide Flash Sale Homepage

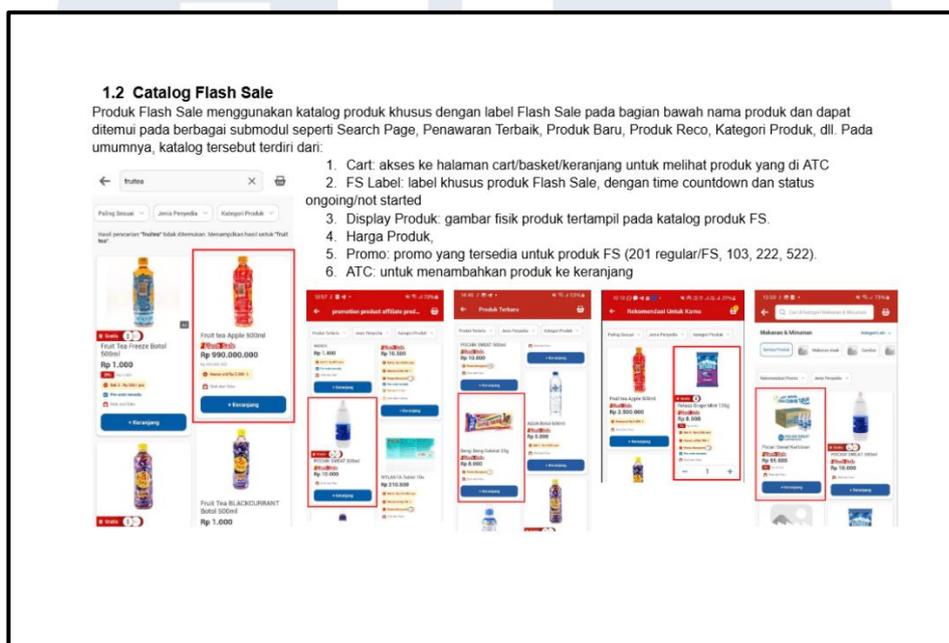
Gambar 3.33 merupakan *user guide* untuk modul *flash sale* didalam *homepage*, khususnya pada tampilan *homepage*. Halaman ini berfungsi sebagai tampilan awal dari fitur *flash sale*, yang menampilkan beberapa produk pilihan yang sedang mendapatkan *promo*. Selain itu, bagian *flash sale* ini juga menyajikan elemen-

elemen interaktif bagi pengguna, seperti status ketersediaan produk, waktu hitung mundur (*countdown timer*), serta kondisi-kondisi *promo* lainnya ketika belum dimulai, sedang berlangsung, dan sudah habis terjual/*promo* berakhir.



Gambar 3.34 Userguide Flash Sale Landing Page

Pada Gambar 3.34 dijelaskan bahwa modul *flash sale* memiliki sebuah *landing page*, yang berisikan detail, jadwal, hingga daftar produk pilihan baik yang sedang berlangsung maupun yang akan datang. *Landing page* tersebut memiliki fitur tambahan selain dari yang terdapat pada halaman *homepage*, seperti menambahkan produk ke dalam keranjang (*add to cart*), melihat detail produk, dan tentunya mengakses keranjang melalui halaman tersebut.

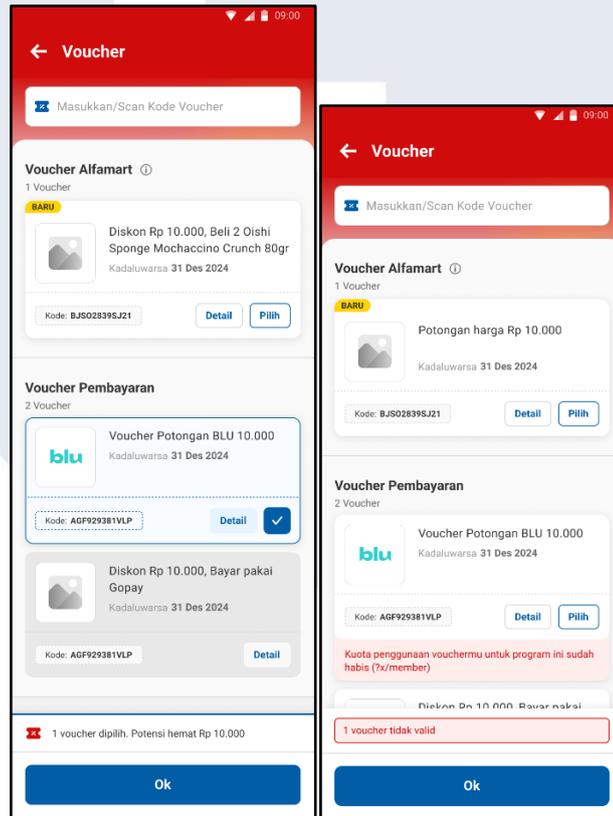


Gambar 3.35 Userguide Flash Sale Catalogue

Gambar 3.35 merupakan tampilan *userguide* untuk katalog produk *flash sale* yang dapat ditemukan di beberapa bagian dalam aplikasi. Katalog ini menampilkan produk-produk dengan label khusus *flash sale*, yang biasanya muncul di berbagai submodul seperti *search page*, *promo page*, *shopping page*, maupun pada bagian katalog pada *homepage*. Setiap item ditampilkan secara visual lengkap dengan gambar produk, harga, serta tombol untuk menambahkan produk ke keranjang (*add to cart*). Pada tampilan tersebut, pengguna juga dapat melihat berbagai elemen seperti label *flash sale* pada *card* produk, sehingga dengan adanya katalog ini,

pengguna bisa langsung mengenali produk-produk yang sedang masuk dalam periode *flash sale* dan melakukan pembelian dengan lebih cepat tanpa perlu masuk ke halaman khusus seperti *landing page* sebelumnya.

3.2.2.7 Project Lock Limit Use per User VLP



Gambar 3.36 Beberapa Kondisi Pemakaian *Voucher* VLP

Pada awal bulan Mei, karyawan magang diajak untuk berkontribusi dalam proses perancangan *project lock limit* penggunaan VLP, yang dirancang dengan maksud untuk membatasi penggunaan *voucher* pembayaran yang disebut sebagai VLP. Pemberlakuan ini ditujukan untuk mengontrol *voucher* yang digunakan setiap transaksi yang dilakukan pengguna. Terdapat beberapa fungsi yang diimplementasikan pada *project* ini, dari batas penggunaan per *member*, penggunaan dengan *voucher* lain, hingga

mekanisme pengembalian *voucher* ketika dilakukan *refund* terhadap produk yang sudah dibeli pengguna.

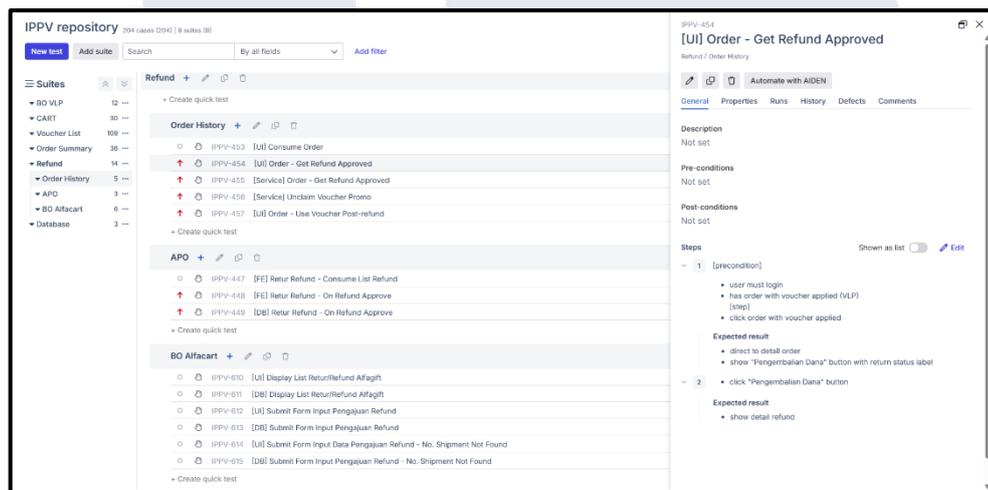
Seperti pada Gambar 3.36, limitasi penggunaan *voucher* dapat terjadi ketika pengguna sudah menggunakan *voucher* lebih dari batas yang sudah ditentukan sebelumnya. Penggunaan VLP juga tidak secara langsung dapat digunakan di setiap produk yang tersedia pada aplikasi, melainkan hanya produk yang diperbolehkan (*whitelist*) saja. Produk yang diperbolehkan dapat berupa produk *groceries* dan jenis produk *e-service* seperti pulsa, pembayaran tagihan, dan lainnya.



Gambar 3.37 Proses Pembuatan Skenario *Refund* VLP untuk *Testing*

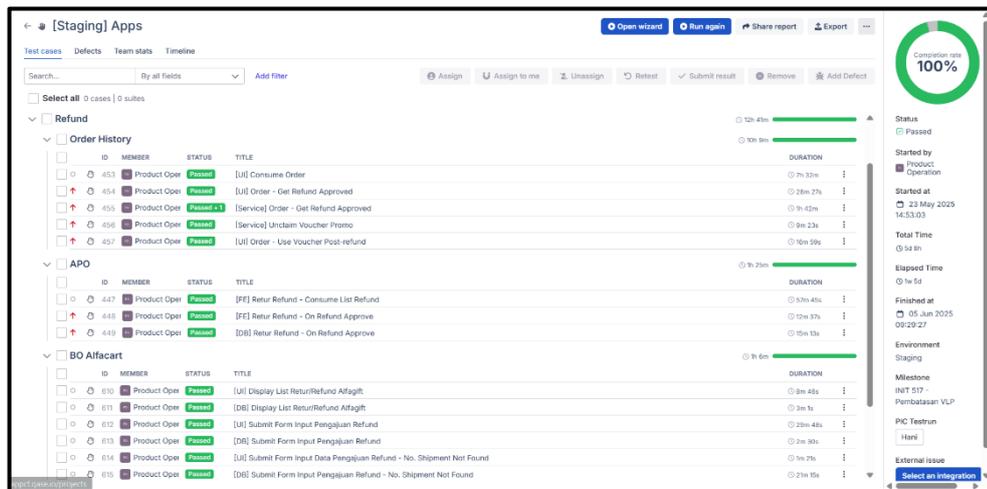
Gambar 3.37 menjelaskan proses pembuatan skenario yang umumnya dibuat menggunakan Excel atau Google Spreadsheet yang lalu di *export* menjadi *file* dengan *format* .CSV agar dapat di *import* pada *platform* manajemen pengujian yaitu QASE. Pada *project* ini, karyawan magang menangani untuk *case* pengembalian *limit* penggunaan *voucher* dan penggunaan *voucher* kembali setelah proses *refund*. Terdapat 3 modul yang terlibat pada implementasi fungsi *refund* untuk VLP, yaitu pada “Order History” dimana pengguna dapat melihat/melacak proses *refund* pada bagian bawah

halaman, lalu APO atau aplikasi pesanan *online*, yang dapat digunakan oleh kurir atau kru pengiriman untuk melakukan *refund* ketika terjadi hal yang memungkinkan pesanan tidak dapat dikirim ke alamat pelanggan, dan “Back Office” untuk melakukan proses *refund* melalui *web* oleh *customer service* berdasarkan permintaan pelanggan.



Gambar 3.38 Test Scenario untuk Proses Refund VLP

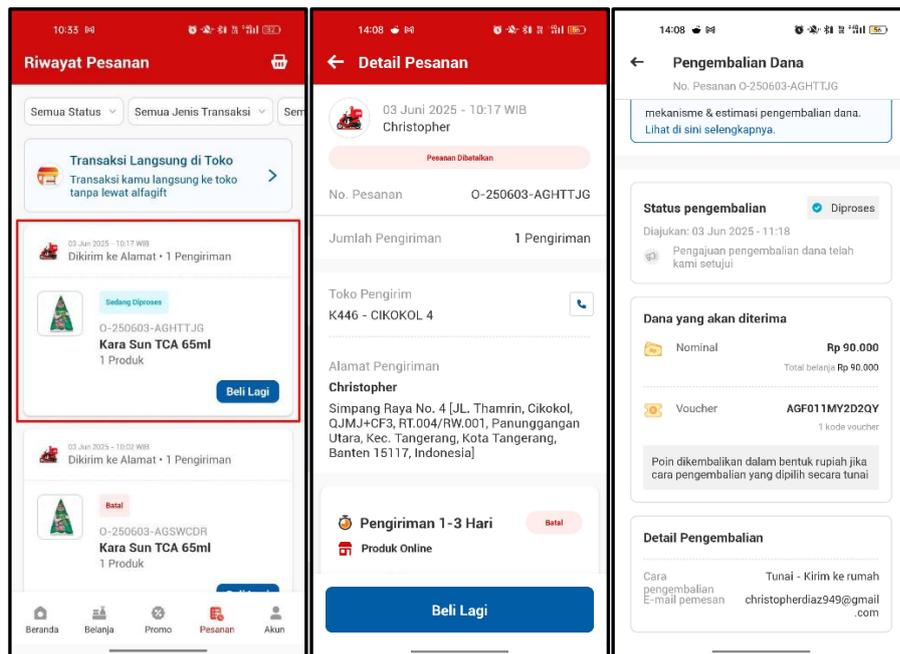
Gambar 3.38 merupakan sebuah *test scenario* yang terdiri dari sekumpulan *test case*, salah satunya untuk proses pengembalian *voucher* setelah *refund* produk telah dilakukan. Seperti pada saat membuat skenario, proses pengembalian limit penggunaan akan dibahas pada pengujian ini. Terdapat beberapa kondisi *voucher* yang perlu diperhatikan, misalnya pada kondisi *voucher* terpakai, *voucher* akan dipastikan tidak dapat dipakai/*redeem* kembali, kecuali setelah melewati proses batal beli atau *refund*. Melalui proses *refund*, kondisi *voucher* akan diubah menjadi tidak terpakai, sehingga pengguna dapat menggunakannya kembali ketika membuat sebuah pesanan baru.



Gambar 3.39 Test Run untuk Proses Refund VLP

Gambar 3.39 merupakan proses lanjutan setelah membuat *test case* untuk kondisi *voucher* setelah proses *refund* pada skenario melalui *platform* manajemen QASE. Secara keseluruhan, pengujian pada *project* ini melibatkan 5 modul utama, yaitu “Cart”, “Voucher List”, “Order Summary”, dan tentunya “Refund”. Pada modul *refund*, terdapat total 14 *cases* yang terdiri dari “Order History”, “APO”, dan “BO Alfacart” seperti yang sudah dibuat oleh karyawan magang sebelumnya pada tahap *test scenario*.



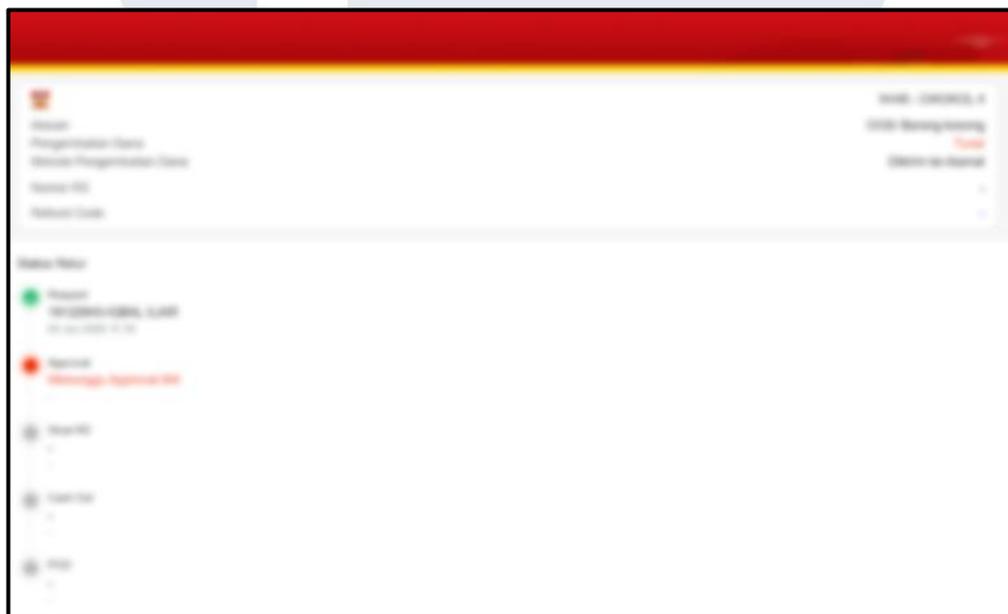


Gambar 3.40 *Testrun* Modul *Order History*

Gambar 3.40 adalah halaman *order history*, yang menampilkan status setiap pesanan yang dibuat oleh pelanggan, seperti menunggu pembayaran, sedang dalam pengiriman, selesai pengiriman, hingga batal pemesanan. Pengguna dapat menggunakan fitur pengajuan *refund* pada halaman ini, dengan mengklik tombol “Pengembalian Dana” pada bagian bawah halaman detail pesanan. Pada pengujian ini, pesanan yang dilakukan *refund* akan dilabeli “Batal” dan akan memiliki detail *refund* pada halaman pengembalian dana, yang dapat diakses melalui *button* yang terletak pada halaman bawah detail pesanan. Hal yang perlu dipastikan pada pengujian ini adalah informasi pengembalian yang akan diterima oleh pelanggan sesuai proses *refund* dilakukan, seperti informasi dana dan *voucher* VLP yang digunakan dalam bentuk kode yang sebelumnya dipakai.



Gambar 3.41 Testrun Modul APO Retur/Refund



Gambar 3.42 Detail Pengajuan Retur/Refund

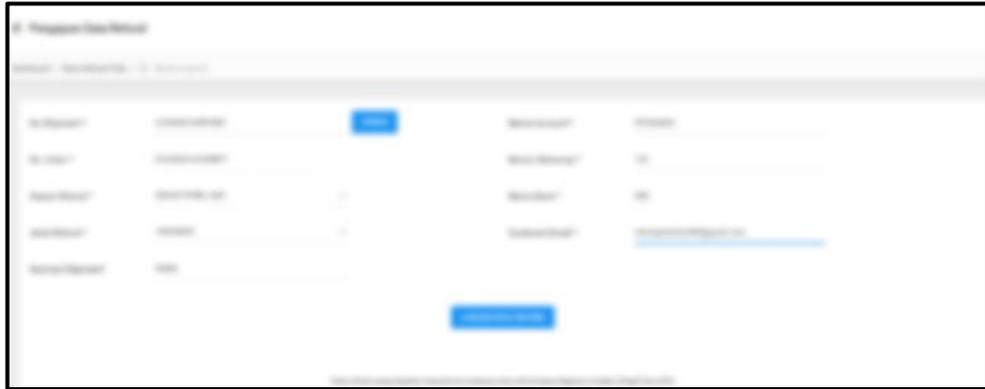
Pengujian berikutnya dilakukan pada halaman *return refund* pada aplikasi pesanan *online* (APO), yang berfungsi untuk menampung seluruh permintaan pengajuan *refund*, termasuk pesanan dengan *voucher* VLP, seperti yang ditampilkan pada Gambar 3.41, Terdapat tiga *tab* pada halaman *return refund*, *tab* “Baru” yang berisi pengajuan *refund* terbaru yang diajukan melalui

berbagai *platform*, seperti aplikasi oleh pelanggan, *back-office* oleh *customer service* atau *admin*, dan aplikasi pesanan *online* itu sendiri oleh kurir/kru pengiriman. Berikutnya, setiap pengajuan yang telah disetujui oleh manajer cabang toko akan dipindahkan ke *tab* “Proses”. *Tab* ketiga, yaitu *tab* “Selesai” mencakup seluruh pesanan yang berhasil dilakukan *retur/refund*. Apabila manajer cabang mengklik salah satu nomor pengiriman yang tersedia pada *tab* yang ada, aplikasi akan mengarahkan pengguna ke halaman detail pengajuan *retur/refund*, seperti pada Gambar 3.42. Pengujian pada modul ini menjadikan proses *refund* sangat krusial, mengingat tahap persetujuan memainkan peranan penting dalam proses *refund* berikutnya.



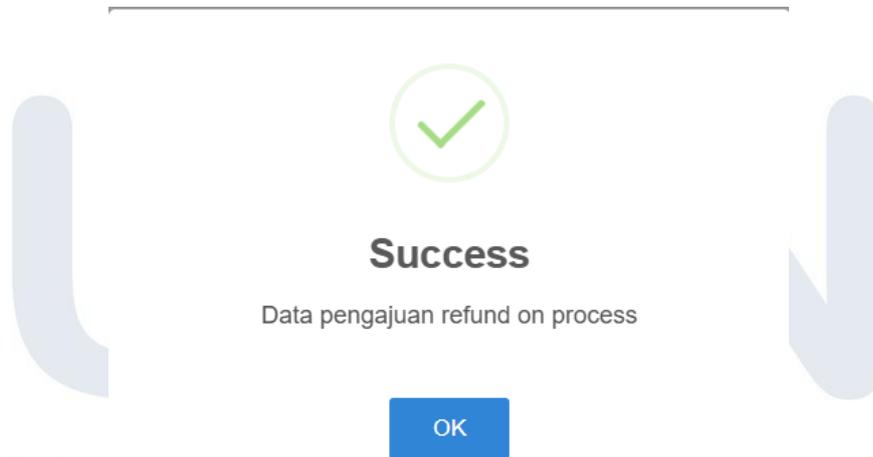
Gambar 3.43 *Testrun Modul Back-office Retur/Refund*

Sama seperti pada *project* sebelumnya, proses *refund* juga dapat dilakukan pada *back-office*. Gambar 3.43 adalah tampilan halaman *retur/refund* yang terdapat pada *back-office* dengan nomor pengiriman yang menggunakan *voucher* VLP. Halaman ini akan menampilkan seluruh pesanan *refund* baik itu menggunakan *voucher* VLP ataupun tidak.



Gambar 3.44 *Testrun Modul Back-office Pengajuan Refund*

Gambar 3.44 merupakan halaman *form* untuk mengajukan *refund* melalui *back-office*. Pengujian pada halaman ini digunakan untuk *case* proses *refund* produk dengan *voucher* VLP yang dilakukan oleh *customer service* atau *admin*. Sama seperti fungsi yang sudah ada sebelumnya, setelah mengisi pengajuan melalui *form* tanpa *error message*, sistem akan membuat status pengiriman produk pesanan berubah, serta *voucher* yang dipakai akan masuk kedalam status *released* sesuai dengan aturan *project* ini.



Gambar 3.45 *Popup Modal Success Pengajuan*

Setelah *form* diisi dan valid, maka langkah selanjutnya untuk *post* data dari pengajuan *refund* tersebut adalah mengklik *button* “Ajukan Data Refund”. Jika pengajuan berhasil di *post*, maka sistem akan memberikan *response modal* seperti pada Gambar 3.45.

3.3 Kendala yang Ditemukan

Berikut ini adalah beberapa kendala utama yang dihadapi karyawan magang selama melaksanakan magang di PT Global Loyalty Indonesia:

1. Sulit mengejar pemahaman terhadap alur sistem dan dokumentasi *project*: Sistem yang digunakan telah berjalan cukup lama dan kompleks, dan terkadang terdapat keterkaitan dengan *project* sebelumnya. Hal ini membuat karyawan magang membutuhkan waktu lebih lama untuk memahami alur kerja dan relasi antar fitur dalam aplikasi.
2. Kesulitan menggunakan *tools* teknis seperti Solr dan Swagger: Di awal magang, penggunaan *tools* tersebut menjadi tantangan karena belum familiar dengan cara kerja *query database* yang kurang umum maupun pengujian dengan melakukan intervensi tertentu. Hambatan teknis ini sempat menghambat proses *testing* pada beberapa *project*.
3. Kendala istilah khusus dalam pembahasan *project*: Proses koordinasi yang mayoritas dilakukan secara *online* seringkali mengalami miskomunikasi bagi karyawan magang, terutama ketika membahas *bug* atau kendala teknis. Perbedaan istilah dan kurangnya visualisasi langsung membuat lemahnya pemahaman karyawan magang dan perlunya bertanya lebih lanjut.

3.4 Solusi atas Kendala yang Ditemukan

Untuk mengatasi kendala yang muncul selama masa magang, dilakukan beberapa langkah penyelesaian sebagai berikut:

1. Mempelajari dokumentasi internal dan melakukan *sharing knowledge* dengan mentor: Untuk memahami alur sistem yang belum terdokumentasi secara lengkap, karyawan magang aktif mencatat ulang alur berdasarkan hasil observasi dan materi dari sprint meeting. Selain itu, sesi diskusi dengan QA senior seperti mentor, serta developer dilakukan untuk memperjelas bagian-bagian sistem yang belum dipahami.
2. Belajar mandiri penggunaan *tools* melalui praktik langsung dan dokumentasi: Dalam menghadapi kesulitan menggunakan *tools* seperti Solr dan Swagger, karyawan magang mengalokasikan waktu tambahan untuk

membaca dokumentasi dan mencoba menggunakan *tools* tersebut dalam contoh kasus. Hal ini membantu dalam mempercepat proses adaptasi terhadap *tools* yang digunakan tim QA.

3. Mengambil inisiatif dalam komunikasi dan meminta klarifikasi langsung saat diperlukan: Ketika terjadi hambatan dalam komunikasi, karyawan magang berinisiatif menghubungi rekan kerja secara personal untuk meminta penjelasan tambahan. Selain itu, penggunaan *screenshot* ketika melakukan inisiatif diskusi terbukti efektif untuk memperjelas konteks permasalahan.

UMMN

UNIVERSITAS
MULTIMEDIA
NUSANTARA