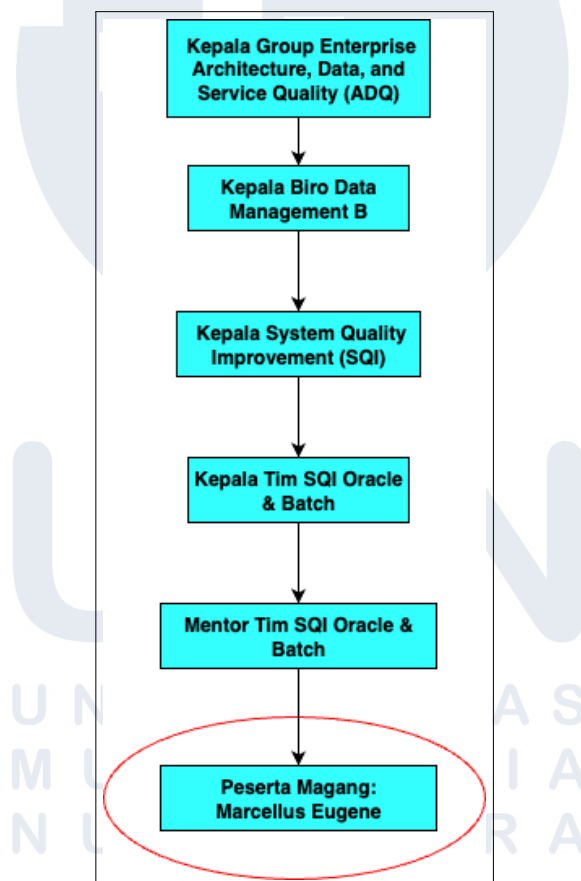


### BAB 3

#### PELAKSANAAN KERJA MAGANG

##### 3.1 Kedudukan dan Koordinasi

Selama proses pengerjaan proyek magang, bimbingan dan arahan langsung dari mentor selalu diberikan serta dari rekan kerja dalam Tim SQI *Oracle and Batch*. Kedudukan magang dalam Tim SQI *Oracle and Batch* di PT Bank Central Asia Tbk dapat dilihat pada Gambar 3.1.



Gambar 3.1. Kedudukan Peserta Magang SQI di BCA

Pada Gambar 3.1, kedudukan magang ditempatkan di bawah unit kerja *Data Management B*, tepatnya di dalam Tim SQI dan sub tim SQI *Oracle and Batch*. Unit *Data Management B* dipimpin oleh Suwandi, yang bertanggung jawab dalam

mengawasi seluruh tim pengelolaan data, termasuk Tim SQI. Adapun Tim SQI sendiri dipimpin oleh Nia Viviwaty, yang mengoordinasikan seluruh aktivitas sub tim terkait pengelolaan sistem dan aplikasi di lingkungan *Data Warehouse* BCA. Sub tim SQI *Oracle and Batch* dipimpin oleh Michael James selaku *team leader*, sedangkan pembimbing langsung sekaligus mentor adalah Rizaldi Suwandi, yang memberikan arahan serta bimbingan selama pelaksanaan kegiatan magang. Dalam prosesnya, mentor turut didukung oleh beberapa anggota tim lain di SQI *Oracle and Batch* yang membantu apabila terdapat kendala dalam pengerjaan proyek.

Koordinasi dengan Tim DIS/*Digital Innovation Solution* juga dilakukan bersama Sherly Florencia. Tim DIS adalah tim yang berperan dalam seluruh proyek yang menggunakan aplikasi *lowcode OutSystems*. Sherly secara khusus adalah PIC dalam proyek pengerjaan aplikasi DWH IRIS sehingga perlu berkoordinasi terkait proses pembuatan akun, *refresh* data, hingga membantu proses perbaikan apabila terjadi kendala pada sistem atau *environment*.

### 3.2 Tugas yang Dilakukan

Terdapat tiga tugas utama yang dilakukan dalam pengerjaan pengembangan aplikasi DWH IRIS.

1. Pemindahan atau migrasi data dari *Database Lowcode* ke *Database Oracle*. Proyek DWH IRIS pada fase sebelumnya menggunakan *database* lokal dari aplikasi *Lowcode Outsystems*. Oleh karena proses implementasi ini akan dilakukan secara rutin dalam DWH, diperlukan pengelolaan dan penyimpanan data implementasi dengan baik dan terstruktur. Tim SQI adalah tim yang berperan dalam pengelolaan dan pengecekan hal tersebut dari tahap pendaftaran hingga proses implementasi selesai.
2. Penambahan Fitur Pencarian *Dashboard* dan Kolom pada beberapa Tabel di *Oracle*. Pada bagian ini, dibutuhkan beberapa fitur tambahan, seperti pencarian *PIC Developer*, pencarian *PIC SQI*, dan penggabungan beberapa pencarian secara bersamaan dengan tujuan *filter* data pada *dashboard*.
3. Perbaikan dan Pengecekan *Flow* Proses Implementasi baik sebagai *Developer* dan *Admin SQI*. Pada bagian ini merupakan lanjutan dari tahap sebelumnya dimana *flow* implementasi apakah dapat berjalan dengan baik setelah proses migrasi data secara keseluruhan.

### 3.3 Uraian Pelaksanaan Magang

Pada tahap awal, diberikan sebuah materi pembelajaran terkait keamanan data, etika selama bekerja, hingga mendapatkan sertifikat sebagai syarat mendapatkan *email domain* dari perusahaan. Hal ini digunakan untuk kelancaran selama proses pelaksanaan magang. Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Minggu Ke-	Pekerjaan yang dilakukan
1	Mengerjakan <i>e-learning MyDevelopment</i> BCA
2	Instalasi aplikasi yang dibutuhkan dan belajar proses implementasi
3	Pengenalan pekerjaan tim SQI dan migrasi data ke <i>Database Oracle</i>
4	Memperbaiki desain aplikasi pada Figma dan teks serta urutan <i>descending</i> pada <i>log activity</i>
5	Memperbaiki <i>logic dropdown on change</i> dan menambahkan kolom <i>Is Active</i>
6	Memperbaiki penggunaan kolom <i>Is Additional</i> dan perbaikan <i>back button</i> pada pendaftaran
7	Membuat tabel <i>user</i> sesuai kebutuhan dan migrasi tabel dari <i>database</i> lokal
8	Memperbaiki <i>problem</i> pada tabel <i>user</i> dan pengecekan tipe data
9	Membuat <i>logic</i> baru untuk <i>create user</i> dan melakukan <i>Data Manipulation Language</i>
10	Membuat <i>filter</i> untuk <i>Developer</i> di <i>Dashboard</i> dan filter berdasarkan tanggal
11	Membuat <i>filter</i> PIC SQI dan fitur pencarian ( <i>Search Implementation</i> )
12	Memperbaiki desain Figma dan pengecekan <i>team member</i> lebih dari satu orang
13	Mengubah <i>assign</i> PIC dan penyesuaian <i>filter</i> untuk PIC yang belum ditunjuk
14	Finalisasi <i>filter team member</i> dengan validasi <i>message</i>
15	Memperbaiki <i>layout pop-up</i> agar responsif

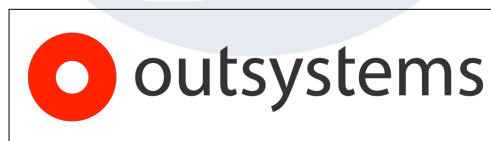
**Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang (lanjutan)**

Minggu Ke-	Pekerjaan yang dilakukan
16	Membuat <i>log activity</i> untuk <i>Notes</i> dan data <i>changes</i> serta <i>subteam</i>
17	Membuat <i>sorting</i> transaksi berdasarkan prioritas status
18	Menaikkan aplikasi dari <i>development</i> ke UAT ( <i>User Acceptance Testing</i> ) dan revisi temuan

Pelaksanaan kerja magang ini berfokus pada pengembangan dan optimalisasi jalannya aplikasi DWH IRIS secara menyeluruh. Adapun dua *software* yang digunakan untuk mendukung proses pengembangan tersebut, yakni *OutSystems* dan *Oracle SQL Developer*. *Software* tersebut merupakan aplikasi *development* yang dipakai oleh tim *Data Warehouse* sesuai *job* dan peran masing-masing.

#### 1. *OutSystems*

Logo aplikasi *OutSystems* yang digunakan hingga saat ini terdapat pada Gambar 3.2.



Gambar 3.2. Logo *OutSystems*

*OutSystems* adalah *platform lowcode* mutakhir yang menjadi fondasi utama dalam pengembangan aplikasi. *Platform* ini memungkinkan pembuatan aplikasi *web* dan seluler dengan kecepatan tinggi melalui antarmuka visual, secara drastis mengurangi waktu koding manual yang diperlukan. Keunggulannya terletak pada kemampuannya untuk mengintegrasikan logika bisnis, *user interface*, dan *database* dalam satu lingkungan terpadu, menjadikannya alat yang ideal untuk iterasi cepat.

#### 2. *Oracle SQL Developer*

*Oracle SQL Developer* berperan sebagai jembatan menuju lapisan data. Aplikasi ini adalah lingkungan pengembangan terintegrasi (IDE) yang memungkinkan untuk mengakses, menjelajahi, memanipulasi, dan menjalankan *query* pada *database Oracle*. Dalam konteks pekerjaan magang ini, *Oracle SQL Developer* sangat vital untuk memastikan integritas data, menguji performa *query* yang kompleks, dan mengelola struktur tabel yang menjadi *backend* bagi setiap aplikasi yang dibangun di *OutSystems*.

Logo aplikasi *Oracle SQL Developer* yang digunakan hingga saat ini terdapat pada Gambar 3.3.



Gambar 3.3. Logo *Oracle SQL Developer*

Selama periode magang, fokus utama pelaksanaan kerja dibagi menjadi dua fase integral. Fase pertama melibatkan pembelajaran mendalam mengenai arsitektur *OutSystems*, dimulai dari pemahaman *Service Studio* hingga penerapan pola desain *low-code* terbaik. Tugas konkret mencakup pembangunan layar input data dan implementasi logika bisnis untuk proses pendaftaran implementasi. Di fase ini, setiap proses pengembangan di *OutSystems* selalu disinkronkan dengan *Oracle SQL Developer*. Secara rutin *Oracle SQL Developer* digunakan untuk memverifikasi skema data yang dihasilkan secara otomatis oleh *OutSystems*.

Fase kedua bergeser ke arah pemecahan masalah dan pengoptimalan performa. Hal ini mencakup identifikasi *problem* pada proses menarik data dan modifikasi struktur tabel melalui *Oracle SQL Developer* untuk mendukung indeks yang lebih efisien. Secara keseluruhan, sinergi antara kecepatan pengembangan *OutSystems* dan kontrol data yang presisi melalui *Oracle SQL Developer* memastikan bahwa produk akhir yang dihasilkan tidak hanya fungsional dan cepat dibuat, tetapi juga didukung oleh fondasi data yang kuat dan terstruktur.

### 3.3.1 Pemindahan data dari Database Lowcode ke Database Oracle

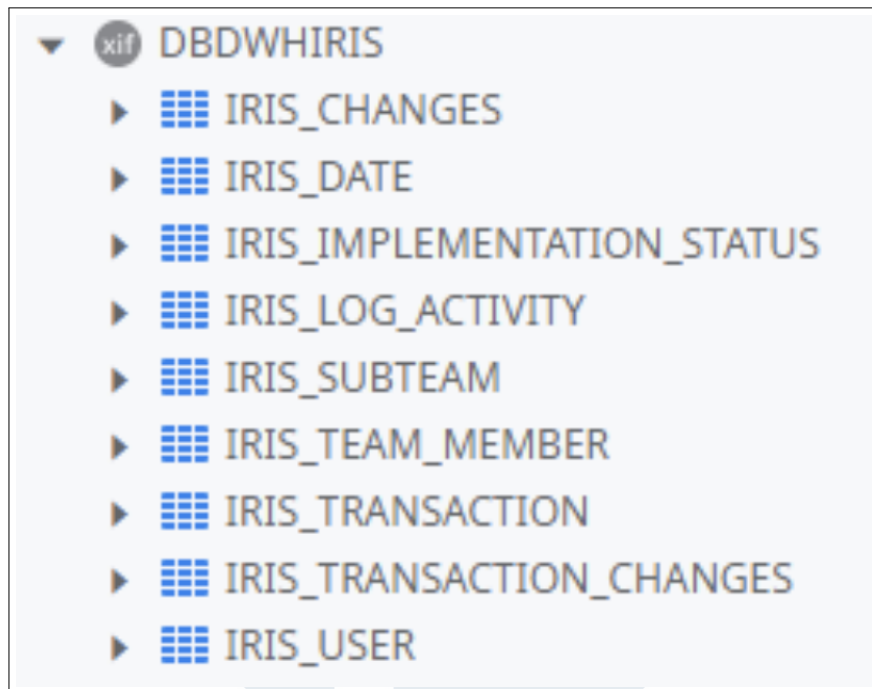
Pemindahan data ini diletakkan dalam *database development* dan membutuhkan sebuah aplikasi *Oracle SQL Developer* untuk melakukan *query*

terkait penggunaan seluruh data dan tabel. Pada proses ini, tabel-tabel disesuaikan kembali dengan data yang sebelumnya dibuat pada *database lowcode Outsystems*. Namun, terdapat beberapa tipe data yang tidak bisa dipakai persis seperti yang sebelumnya telah dibuat. Contohnya pada *Oracle* tidak bisa menggunakan tipe data *Integer* sehingga diganti menjadi tipe data *Number*. Tipe data sebelumnya seperti *Text* atau *String* juga perlu diubah menjadi *VARCHAR*.

Penyesuaian di atas menimbulkan *error* dalam beberapa *flow* yang melibatkan penarikan data dari *database*. Oleh karena itu, *error* yang ada perlu diperbaiki satu per satu. Tidak hanya terkait tipe data, penggunaan *aggregate* untuk melakukan *join* antartabel dan *filtering* data juga harus diperbaiki sesuai kebutuhan. Ketika menggunakan *database* lokal, pembuatan *id* untuk setiap tabel dapat langsung bertambah secara berurutan atau *auto increment*. Namun, tidak bisa dilakukan otomatis pada *database oracle* sehingga penambahan *logic increment* diperlukan untuk dapat menjalankan *create id* suatu transaksi baru.

Meskipun *platform lowcode* seperti *OutSystems* menyediakan *database* internal yang efisien untuk pengembangan cepat dan *prototyping*, untuk aplikasi skala *enterprise* yang membutuhkan skalabilitas, keamanan tingkat tinggi, dan integrasi terpusat dengan sistem lain, pemindahan data ke *Database Oracle* yang lebih kuat menjadi suatu kebutuhan dalam pengembangan lebih lanjut. Proses ini dimulai dengan mendefinisikan koneksi eksternal dalam lingkungan *lowcode*, secara efektif membuat tautan langsung ke *schema* tertentu di *Oracle*. Setelah koneksi berhasil dibuat, data yang awalnya tersimpan di *entity* internal *lowcode* akan dilakukan *mapping* atau dipindahkan ke tabel tujuan yang sudah disiapkan di *Oracle*. Di sinilah *Oracle SQL Developer* memainkan peran vital. Sebelum pemindahan, *tool* ini digunakan untuk mempersiapkan dan memvalidasi skema tabel tujuan, serta menjalankan *script Data Definition Language (DDL)* yang kompleks. Selama dan setelah proses migrasi, *Oracle SQL Developer* berfungsi sebagai alat verifikasi untuk memastikan integritas dan konsistensi data antara sumber dan tujuan. Dengan kata lain, migrasi ini memastikan bahwa meskipun kecepatan pengembangan tetap tinggi berkat *lowcode*, fondasi data aplikasi tetap kokoh, aman, dan siap untuk menangani volume transaksi yang besar sesuai standar *Data Warehouse*.

Entitas yang dipakai pada *Database Oracle* dapat dilihat pada Gambar 3.4.



Gambar 3.4. Entitas pada *Database Oracle* setelah migrasi

Dalam proses migrasi data, ditemukan beberapa *error* setelah penyesuaian nama kolom dan tabel yang baru pada *Database Oracle*. Untuk melakukan pengecekan, *OutSystem* memberikan fitur *Debugging* untuk membantu menemukan *error* yang sedang terjadi. Pada umumnya, tipe data dan penamaan tabel yang tidak sinkron menimbulkan *error* pada aplikasi. Hal ini terjadi karena ada beberapa tipe data yang belum cocok saat masuk ke dalam sebuah *condition* pada *logic*. Oleh karena itu, dibutuhkan proses pengubahan atau konversi tipe data yang sesuai agar proses *condition* dalam setiap *logic* dapat dilakukan dengan baik. Contohnya adalah *TextToIdentifier()* dan *IdentifierToInteger()*. *Function-function* tersebut telah disediakan dari modul *OutSystems* sehingga membantu proses konversi tipe data.

### 3.3.2 Penambahan Fitur Pencarian *Dashboard* dan Kolom pada Beberapa Tabel di *Oracle*

Dalam sebuah transaksi, terdapat data atau informasi implementasi berupa anggota atau PIC *Developer* yang mengajukan implementasi itu sendiri. Lalu, di setiap implementasi terdapat satu orang PIC dari tim SQI yang berperan sebagai



*admin*. Data tersebut telah disimpan dalam tabel *IRIS TRANSACTION* dan *IRIS TEAM MEMBER*. Untuk proses *filter* diperlukan mengambil data dari kedua tabel tersebut dengan *Aggregate*. Sebelumnya terdapat sebuah *action Get Quota* yang menghitung jumlah implementasi dalam satu hari pada bulan tersebut di *Dashboard*. Proses tersebut dilakukan dalam *Server Action* agar *action* tersebut dapat digunakan kembali pada *logic* lainnya.

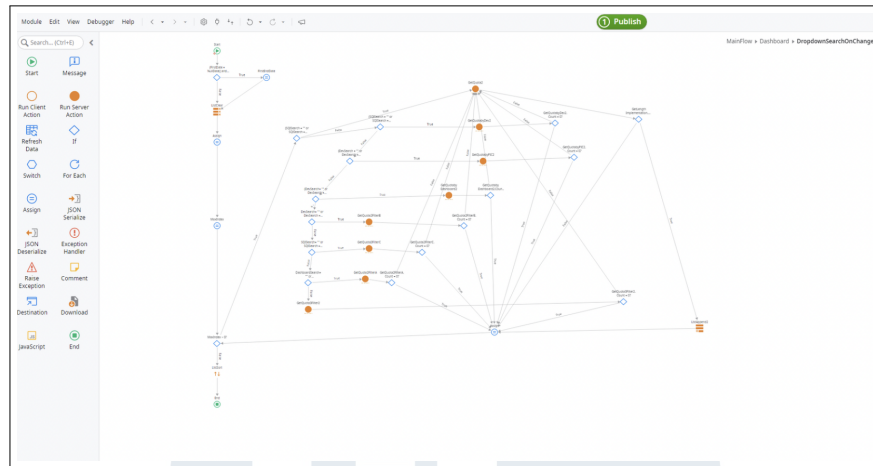
Tampilan UI(*User Interface*) dari filter PIC *Developer* dan PIC *SQI* pada *Dashboard* dapat dilihat pada Gambar 3.5.

Gambar 3.5. Tampilan *Filter* dan Pencarian pada *Dashboard*

Dalam proses pengembangan, ada beberapa kebutuhan penambahan kolom seperti pada tabel transaksi dimana ingin menandakan apakah suatu transaksi tersebut masih aktif atau tidak. Ketika implementasi sudah selesai dijalankan, maka transaksi tersebut sudah tidak aktif. Oleh karena itu, dilakukan penambahan kolom *Is Active*. Selain penambahan kolom, terdapat juga beberapa penambahan isi data pada tabel, seperti *changes* untuk implementasi. *Changes* adalah *script* atau aplikasi yang akan dipakai dalam implementasi. Pada tahap ini dilakukan pengecekan kembali terkait kelengkapan dan kesesuaian data yang sebenarnya.

Selanjutnya, penggunaan *filter* untuk melakukan pencarian dibutuhkan tambahan *logic* dalam beberapa *server action*. Contohnya adalah *Dropdown Search on Changed* seperti pada Gambar 3.6. Pada bagian ini, setiap perlakuan yang mengubah variabel dalam *List Dropdown* harus ditangani dalam *logic* tersebut. Yang menjadi tujuan dari pembuatan *action* ini adalah pencarian pada *Dropdown* baik *SQI* maupun *Developer* dapat berjalan secara bersamaan ketika *user* ingin mencari sebuah implementasi pada *Dashboard*. Oleh karena itu, setiap kondisi dilakukan pengecekan untuk menghitung jumlah *filter* yang sedang dipakai oleh *user* tersebut sehingga kuota yang terhitung sesuai dengan *filter* tersebut seperti pada Gambar 3.6.



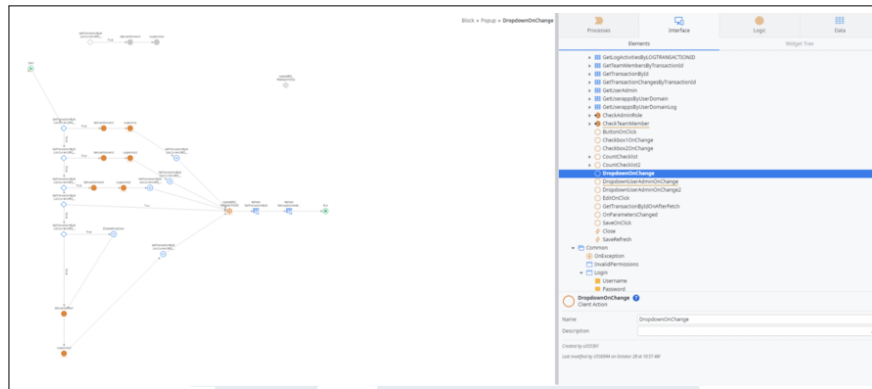


Gambar 3.6. *Logic Dropdown Search On Changed* sebagai *Filter* pada *Dashboard*

### 3.3.3 Perbaikan dan Pengecekan *Flow* Proses Implementasi baik sebagai *Developer* dan *Admin SQI*

Adapun sebuah temuan seperti pada status implementasi yang tidak berjalan dengan benar. Status awal tepat setelah implementasi didaftarkan oleh *developer* adalah *Backlog*. Tim SQI akan menugaskan seorang PIC SQI yang bertanggung jawab dalam implementasi. Setelah PIC sudah ditunjuk, statusnya akan berubah menjadi *Preparation*. PIC SQI akan melakukan *checklist* seluruh *changes* yang dibutuhkan apakah sudah siap digunakan hingga statusnya berubah dari *Preparation* menjadi *Ready*. Apabila implementasi sudah selesai dikerjakan, status seharusnya akan berubah otomatis menjadi PIR atau *Post Implementation Review*. Ketika terjadi sebuah *problem* setelah proses tersebut, maka status berubah menjadi *Waiting*. Sebaliknya jika tidak ada masalah, status dapat diubah menjadi *Closed* oleh PIC SQI.

Perbaikan terkait dengan *flow* tersebut terjadi pada halaman *Pop Up* atau halaman yang berisi detail dari sebuah implementasi. Status sebuah implementasi berada dalam *container dropdown* yang menggunakan *logic Dropdown On Change* seperti pada Gambar 3.7.



Gambar 3.7. Logic Dropdown On Change pada Pop Up

Pada saat melakukan *testing*, ditemukan bahwa perubahan status tidak sesuai dengan *requirement* seperti yang telah dijelaskan sebelumnya. Setelah melakukan *debugging*, *error* pada *flow* tersebut muncul karena tidak melakukan *assign value* dari status *id* sebelum melakukan *update* ke *database*. Dibutuhkan *assignment value* status *id* agar terjadi perubahan status berdasarkan setiap pengecekan kondisi pada *logic Dropdown On Change*.

Untuk melakukan pengecekan status, diperlukan *query* ke *database Oracle* agar dapat melihat status *id* pada suatu transaksi. Pada Gambar 3.8, *query* dilakukan dengan menggunakan *SELECT \* FROM Tabel Transaksi WHERE Release Number* yang akan dicari atau dapat menambahkan *ORDER BY*. Beberapa *query* ke tabel yang lain dilakukan juga untuk mempermudah melihat kesesuaian dengan tabel yang melakukan *JOIN* seperti pada Gambar 3.8.

```

1 | select * from iris_implementation_status;
2 |
3 | select t.transaction_release_number, t.transaction_project_name, i.implementation_status_label from iris_transaction t, iris_implementation_status i
4 | where t.transaction_implementation_status_id =
5 | i.implementation_status_id
6 | and trunc(t.transaction_implementation_date) = trunc(to_date('14-oct-2025', 'DD-mon-yyyy'))
7 | ORDER BY i.implementation_status_order ASC;
8 |
9 |
10 | select * from iris_transaction t
11 | where trunc(t.transaction_implementation_date) = trunc(to_date('24-oct-2025', 'DD-mon-yyyy'));
12 |
13 | select * from iris_transaction
14 | where transaction_release_number = 22229;
15 |
16 | select * from iris_changes;
17 |
18 | select * from iris_user;

```

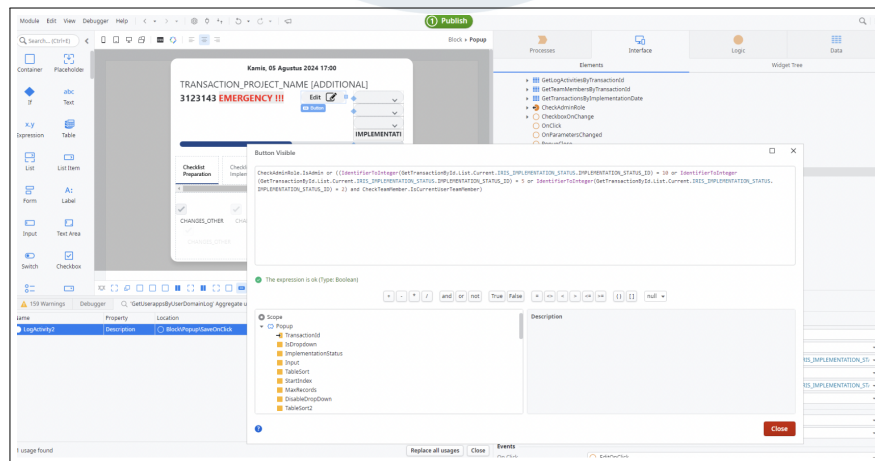
TRANSACTION_ID	TRANSACTION_IMPLEMENTATION_STATUS_ID	TRANSACTION_RELEASE_NUMBER	TRANSACTION_PROJECT_NAME	TRANSACTION_IMPLEMENTATION_DATE	IS_ADDITIONAL	TRANSACTION_UPDATED_DATE	IS_PENDING	TRANSACTION_APPROVAL_DATE	IS_APPROVED	TRANSACTION_STATUS	STATUS_ID	STATUS_LABEL
40	9	22229	YNYV	2025-10-25 17:26:18	0	2025-10-25 17:26:18	0	2025-10-25 17:26:18	0	0	0	0

Gambar 3.8. Pengecekan status transaksi yang masuk dalam Database Oracle

Selain *flow* implementasi, pengecekan terhadap hak akses antara *developer* dan *SQI* juga dilakukan. *Developer* adalah anggota tim yang mengajukan

implementasi dan tim SQI dalam hal ini bisa menjadi *Developer* atau *Admin* SQI. SQI adalah anggota tim yang berperan utama sebagai admin dalam aplikasi DWH IRIS. Oleh karena itu, SQI berperan lebih besar dalam mengatur aktivitas dalam proses implementasi, seperti aktivitas *edit* implementasi, *update* status, dan penunjukan PIC SQI. *User* yang menjadi *developer* atau SQI telah ditandai dalam sebuah kolom *Is SQI* dengan tipe data *Boolean*. Jika SQI akan bernilai 1 atau *true*, sedangkan bernilai 0 atau *False* untuk tim *developer* selain SQI. Hal ini dibutuhkan agar hak akses dapat dijaga dengan baik sesuai *user* yang sedang *login* dalam aplikasi DWH IRIS.

Selain itu, hak akses khusus diberikan pada *user* yang menjadi anggota *developer* dari implementasi yang didaftarkan. Ditambahkan sebuah *logic IsCurrentUserTeamMember*. Pengecekan ini dengan melihat pada tabel *IRIS TEAM MEMBER* apakah *user* yang sedang *login* merupakan *member* atau *developer* dari transaksi tersebut. Hal ini bertujuan agar *user* yang menjadi *member* dari transaksi dapat melakukan *editing* pada implementasi yang telah didaftarkan dan pada saat statusnya masih *Backlog* atau *Preparation*. Pengaturan *logic* dapat diaplikasikan pada properti *Visible* pada *OutSystems* seperti pada Gambar 3.9.



Gambar 3.9. Pengecekan *Role User* untuk Akses terhadap *Edit Button*

### 3.3.4 Hasil Pengembangan Aplikasi DWH IRIS

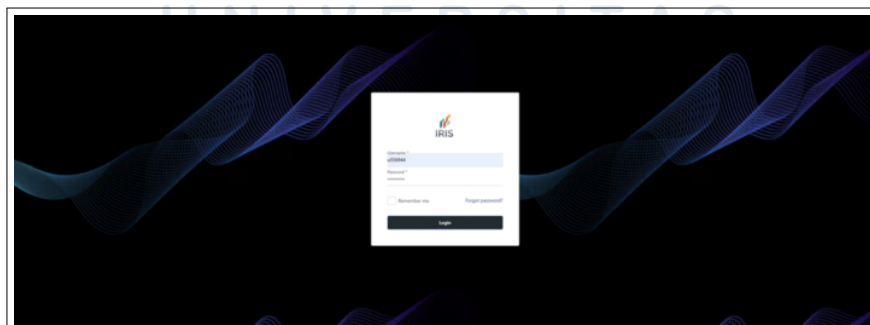
Setelah menjalani beberapa pengerjaan baik dari *front-end* maupun *back-end*, aplikasi DWH IRIS mulai memasuki tahap UAT (*User Acceptance Testing*) untuk tahapan setelah *development*. Pada tahap UAT, koordinasi dengan tim *Digital*

*Factory* diperlukan untuk melakukan *request* menaikkan aplikasi DWH IRIS dari *development* ke UAT. *Request* ini dilakukan melalui *email* dan terdapat proses *Type-In* untuk menghubungkan *Database Oracle* pada *environment* UAT.

Proses UAT merupakan tahapan wajib bagi aplikasi yang baru dibuat sebelum naik ke *production* atau digunakan oleh tim *Data Warehouse*. Di bagian ini, terdapat *Form Testing* dan *Form Verifikasi* yang harus dilengkapi sebagai dokumentasi UAT. Pada *form testing* dilakukan 44 skenario dalam melakukan pengecekan jalannya aplikasi DWH IRIS. Skenario tersebut terbagi dalam dua kategori, yakni kategori normal dan kategori negatif. Contoh skenarionya adalah *Submit Implementasi Normal*. *User* akan melakukan *testing* mendaftarkan implementasi sesuai dengan *project* yang dikerjakan. Implementasi tersebut harus dapat diinput ke dalam setiap kolom informasi pendaftaran yang tersedia mulai dari Pendaftaran A hingga Pendaftaran C.

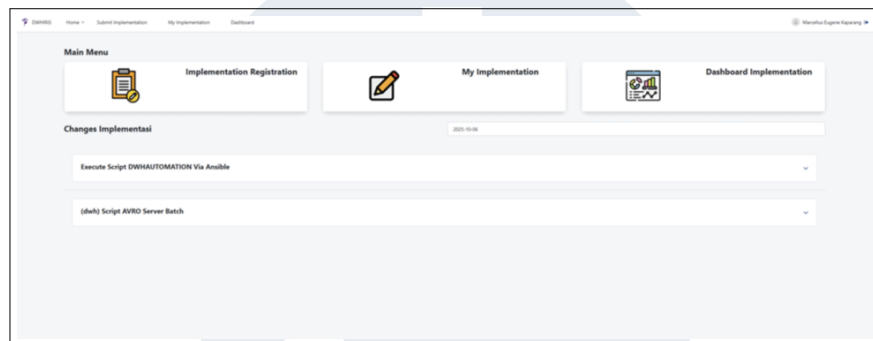
Skenario *submit implementasi* adalah aktivitas yang paling krusial dalam pembuatan aplikasi DWH IRIS karena akan menggantikan cara pendaftaran dari *Form Microsoft Office* ke dalam aplikasi tersebut. Setiap *developer* dapat melakukan pendaftaran implementasi secara normal atau *emergency*. Implementasi normal adalah implementasi yang didaftarkan paling lambat pada satu minggu sebelum tanggal implementasi, sedangkan implementasi *emergency* dapat didaftarkan jadwalnya kurang satu minggu dari tanggal implementasi dan dapat langsung diproses oleh tim SQL.

Sebelum menggunakan aplikasi DWH IRIS, setiap *user* akan masuk melalui *login page* dengan menggunakan *user domain* dan *password*. Tampilan dari halaman *login* dapat dilihat pada Gambar 3.10. Setelah berhasil *login*, pengguna akan diarahkan ke halaman *Home*.



Gambar 3.10. Tampilan *Login Page* Aplikasi DWH IRIS

Tampilan *Home* dari aplikasi DWH IRIS dapat dilihat pada Gambar 3.11. Pada bagian ini, *developer* dapat melihat tiga pilihan menu, yaitu *Implementation Registration*, *My Implementation*, dan *Dashboard Implementation*. Di bawah menu, terdapat *List Changes* Implementasi yang muncul berdasarkan *filter* tanggal. Fitur ini membantu tim SQI untuk mengecek implementasi apa saja yang akan dan sudah dikerjakan dalam suatu hari atau tanggal tersebut.



Gambar 3.11. Tampilan *Home* Aplikasi DWH IRIS

Selanjutnya, *developer* dapat melakukan registrasi implementasi dengan mengisi *Form* Pendaftaran A terlebih dahulu seperti pada Gambar 3.12. Pada tahap ini, *developer* wajib mengisi nomor *release*, nama *project*, dan tanggal implementasi. *Developer* dapat mengubah *switch button emergency* apabila implementasi yang didaftarkan bersifat *emergency*. Dengan begitu, pemilihan tanggal akan terbuka kurang tujuh hari dari tanggal implementasi dan tidak terhitung dalam kuota harian. Implementasi dalam satu harinya akan dibatasi maksimal sejumlah 10 kuota. Selain itu, *developer* yang melakukan pendaftaran akan otomatis terdaftar sebagai PIC Implementasi. Apabila terdapat anggota lain yang menjadi PIC, dapat menekan tombol *Add Member* untuk menambahkan kolom nama PIC dan *subteam* baru.

Gambar 3.12. Tampilan Pendaftaran Implementasi Bagian A

Setelah menyelesaikan Pendaftaran A, *developer* akan diminta untuk mengisi *changes* implementasi apa saja yang dibutuhkan pada *Form* Pendaftaran B. *Changes* dapat dipilih lebih dari satu *item* dan dapat menambahkan *changes* yang belum ada dalam daftar pada kolom *Other Changes* seperti pada Gambar 3.13

Gambar 3.13. Tampilan Pendaftaran Implementasi Bagian B

Setelah menyelesaikan kelengkapan *changes* yang dibutuhkan, *developer* dapat melihat *review* implementasi yang telah diisi seperti pada Gambar 3.14. Hal ini digunakan untuk mengecek kembali apakah data yang dimasukkan telah benar dan sesuai. Apabila ada kekurangan atau perbaikan, *developer* dapat menekan *back button* ke Pendaftaran sebelumnya. Selain itu, terdapat tombol *Save As Draft* untuk menyimpan *form* implementasi yang belum akan dikumpulkan. Setelah *developer* menekan tombol *Submit*, implementasi akan langsung terdaftar dalam *My Implementation Submitted* dan statusnya akan menjadi *Backlog*.

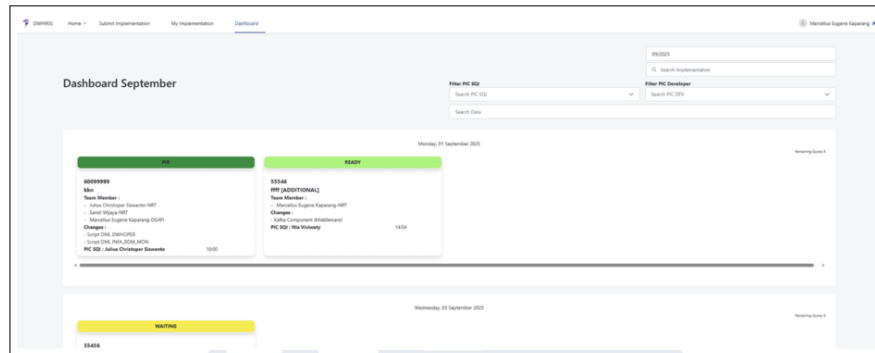


<a href="#">Home</a>	<a href="#">Submit Implementation</a>	<a href="#"><b>My Implementations</b></a>	<a href="#">Dashboard</a>		Mentorship Eugene Espinosa
<b>My Implementation</b>					
Implementation Submitted	Implementation Drafts				
No Release #	Project Name #	Implementation Status	Implementation Date #	PIC ID #	Update On #
222384	DAMAMA	closed	28 Jun-2025	Ronald Szwedki	15-Jul-2025, 08:48
888888	adwafatf [JACDTCYONAL][JACDTCYONAL]	closed	20-Jun-2025	Ronald Szwedki	22-Jul-2025, 09:37
777777	UUDKDFP [JACDTCYONAL][JACDTCYONAL]	closed	20-Jun-2025	Matthew Christian Nadiapanyia	27-Jul-2025, 16:05
99	tgh	progressing	20-Jun-2025	Mentorship Eugene Espinosa	22-Jul-2025, 15:40
7676	vssv	closed	20-Jun-2025	Ronald Szwedki	18-Jul-2025, 11:38
222229	wxyx	pH	20-Jun-2025	Mentorship Eugene Espinosa	22-Jul-2025, 17:26
427174	xjhe	progressing	21-Jun-2025	Ronald Szwedki	14-Jul-2025, 09:12
7676	Ivdex [JACDTCYONAL][JACDTCYONAL]	warning	21-Jun-2025	Matthew Christian Nadiapanyia	22-Jul-2025, 09:41
1577778	jshf	pH	21-Jun-2025	Jenny Charles Gallo	22-Jul-2025, 09:04
45454545	Hgdytgd	warny	21-Jun-2025	Jenny Charles Gallo	20-Jul-2025, 19:07
2012	mwa	closed	21-Jun-2025	Ronald Szwedki	21-Jul-2025, 13:17
1623	wezd [JACDTCYONAL]	progressing	21-Aug-2025	Ronald Szwedki	23-Aug-2025, 13:50

Gambar 3.15. Tampilan Halaman Daftar *My Implementation*

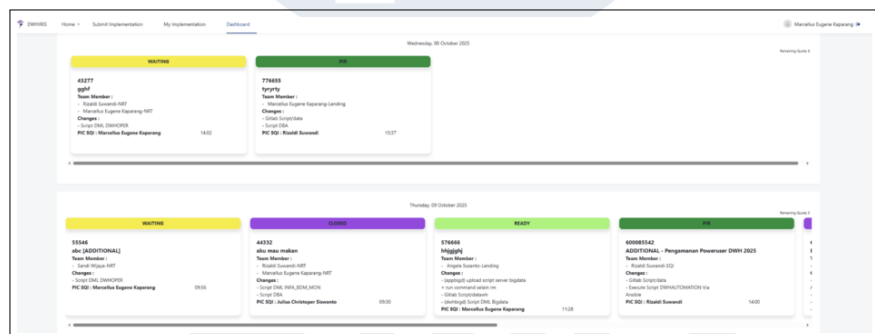
23





Gambar 3.16. Tampilan Halaman *Dashboard* Implementasi

Dalam setiap tanggal, berisi *Block Card List* yang sudah dikelompokkan berdasarkan tanggal implementasi masing-masing seperti pada Gambar 3.17. *Dashboard* ini dibuat untuk dapat melihat status implementasi dengan mudah dan informasi yang diberikan tersimpan jelas dalam sebuah *card* yang dapat menampilkan *Pop Up* dengan mengklik *card* tersebut.



Gambar 3.17. Tampilan *Block Card List* Implementasi dalam setiap tanggal

Tampilan *Pop Up* adalah area yang paling banyak digunakan oleh *admin* SQI untuk melakukan *checklist*, *assign* PIC SQI, penambahan *notes*, dan perubahan status seperti pada Gambar 3.18. Seluruh *changes* yang didaftarkan dapat dilihat seluruhnya dalam *page* ini yang terbagi dalam dua *tab*. *Tab* pertama adalah *Checklist Preparation* dimana berisi *changes* yang akan dicek pada waktu status *Preparation*. Hal ini dibutuhkan oleh *admin* SQI apakah *checklist* yang dibuat oleh *developer* sudah dipersiapkan untuk melakukan implementasi. *Tab* kedua adalah *Checklist Implementation* dimana berisi *changes* yang akan dicek pada saat status *Ready*. Hal ini menandakan kepada *admin* dan *developer* apakah *changes* dalam

Thursday, 09 October 2025

Penurunan Data OMNI-DWH terkait Registrasi Sakuku [ADDITIONAL]

600095812

Edit

BACKLOG

Checklist Preparation

Checklist Implementation

Script DML INFA\_BDM\_MON

(appbigd) upload script server bigdata + run command selain rm

Gitlab Script/datawh

(dwhbigd) Script DML Bigdata

TDV

ELK

Kafka Component (Middleware)

PIC SQL :

Nia Viviwati

Developer

- Rizaldi Suwandi - Lending
- Marcellus Eugene Kaparang - Lending

Notes

Additional Notes

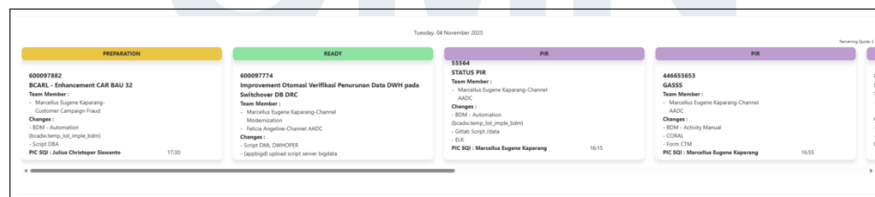
Activity Log

U069025, Submitted at, 2025-10-02 11:14:09

Close

Save

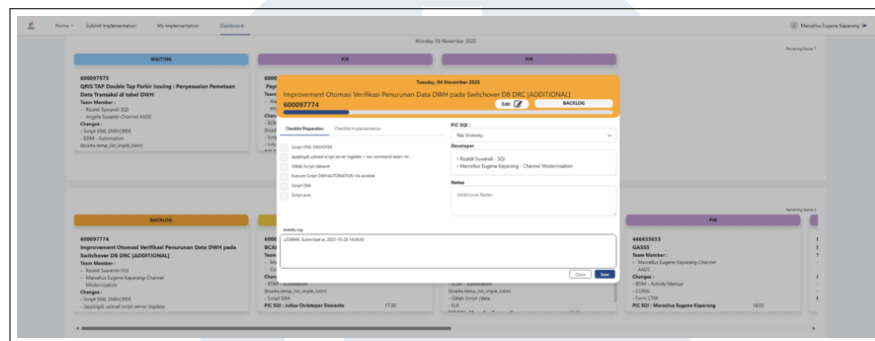
Urutan status implementasi dapat dilihat melalui tampilan *Pop Up* dengan warna khasnya masing-masing. Diawali dengan *emergency, backlog, preparation, ready, pir, waiting*, hingga *closed*. Urutan tersebut ditampilkan menggunakan *sorting* agar status *id* yang lebih prioritas dapat terlihat di bagian paling kiri pada *Dashboard* seperti pada Gambar 3.19.



Setiap perubahan dan aktivitas dalam *Pop Up* akan tercatat dalam kolom *Activity Log*. Pada setiap aktivitas berisi informasi terkait *user domain* dari pengguna dan waktu saat melakukan perubahan, seperti *submitted*, *updated*, dan *assign PIC SQI*. Pada suatu *case* tertentu, implementasi yang dilakukan dapat terjadi penambahan *script* atau bahkan perubahan tanggal. Untuk mengatasi hal tersebut,

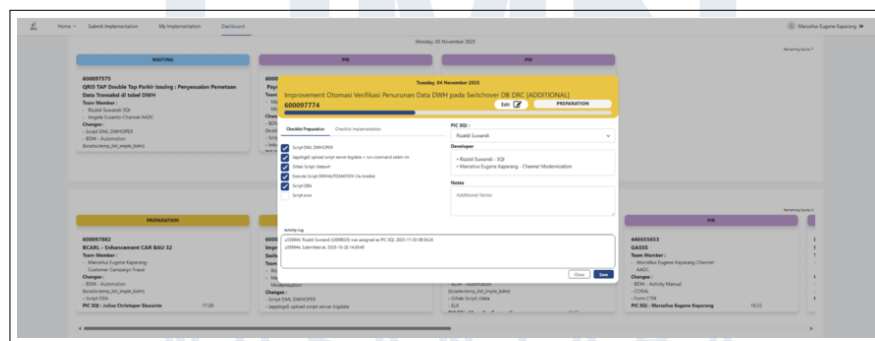
diberikan *button Edit* dengan *icon pencil* bagi *developer* maupun *admin*. Hanya saja *developer* tidak bisa melakukan *edit* apabila statusnya bukan *Backlog* atau *Preparation*. Hal ini dibatasi bagi *developer* melalui penjagaan *Enabled Button*.

Setiap implementasi diawali dari status *backlog* seperti pada Gambar 3.20. *Admin SQI* bertugas untuk melakukan *assign PIC* bagi setiap implementasi yang berada pada *backlog* dengan memilih nama pada *list dropdown PIC SQI*.



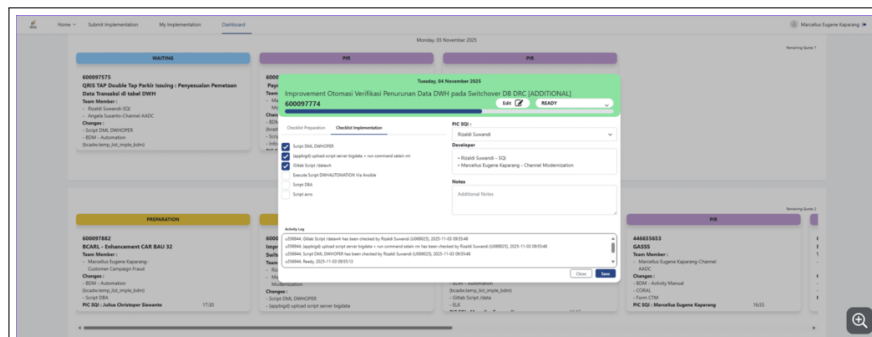
Gambar 3.20. Status *Backlog*

Status *backlog* dapat otomatis berubah menjadi *preparation* setelah *PIC SQI* sudah dipilih. Pada Gambar 3.21, persiapan *changes* dapat dilakukan untuk mengecek kelengkapan data sesuai *checklist* yang diberikan *developer*.



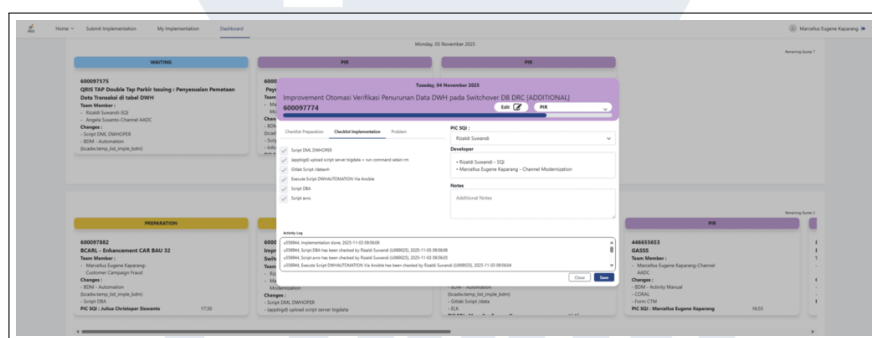
Gambar 3.21. Status *Preparation*

Setelah seluruh *checklist* sudah siap, status akan otomatis berubah ke *ready* seperti pada Gambar 3.22.



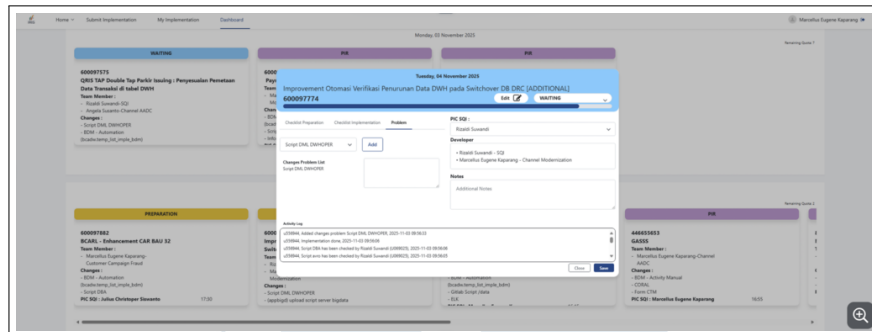
Gambar 3.22. Status Ready

Implementasi dapat dilaksanakan setelah status telah *ready*. Apabila telah selesai dilaksanakan, status akan otomatis berubah menjadi PIR (*Post-Implementation Review*) seperti pada Gambar 3.23.



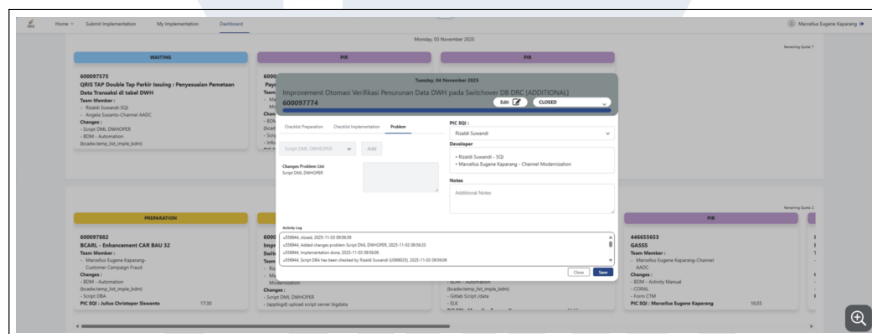
Gambar 3.23. Status PIR

Dalam suatu implementasi, dapat terjadi *problem* atau temuan dimana hasilnya belum sesuai atau terdapat *script* yang bermasalah. Admin SQI dapat mengubah status menjadi *waiting* dengan menambahkan *problem* dan keterangan kendala yang terjadi. Status akan otomatis berubah setelah *problem* ditambahkan seperti pada Gambar 3.24.



Gambar 3.24. Status *Waiting*

Setelah seluruh implementasi berjalan dengan baik dan telah diverifikasi, status dapat diubah menjadi *closed* oleh *admin* SQI dengan mengubah status *dropdown* seperti pada Gambar 3.25.



Gambar 3.25. Status *Closed*

### 3.4 Kendala dan Solusi yang Ditemukan

Beberapa kendala yang terjadi.

1. Pengenalan terhadap aplikasi *lowcode* *Outsystems* yang membutuhkan adaptasi penggunaan *logic* dan fitur-fitur yang disediakan. Aplikasi ini sesuai dengan namanya lebih berfokus pada *flowchart* untuk membuat logika atau *action* dalam pengembangan aplikasi.
2. *Error* yang terjadi ketika melakukan *testing*. Hal ini sering terjadi ketika *action* yang dipakai tidak berfungsi dengan baik meskipun *logic* yang dipakai seharusnya sudah benar.

Solusi yang dilakukan atas kendala tersebut.

1. Mempelajari terlebih dahulu secara mendalam untuk konsep dasar melalui dokumentasi pada fase pertama pengembangan. Hal ini didapatkan juga melalui transfer ilmu dari teman magang sebelumnya selama dua minggu. Selain itu, dibutuhkan latihan secara langsung agar dapat memahami penggunaan dan fitur aplikasi. Pada saat awal, dilakukan perbaikan koneksi data setelah melakukan migrasi secara mandiri.
2. Jika terjadi *error*, proses *debugging* dilakukan dengan memberikan *breakpoint* pada sebuah *logic* atau *action* yang akan dicek. Setiap satu langkah dilakukan pengecekan apakah *output* nya sudah sesuai dengan yang diharapkan. Apabila *error* yang terjadi tidak bisa dilakukan *debug*, maka perlu menambahkan *button* percobaan untuk menyimpan sebuah *action* dan dilakukan *testing logic On Click*.

