BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Kedudukan pada kegiatan magang di PT Cranium Royal Aditama batch 8 sebagai salah satu Full Stack Developer yang terdiri dari tujuh peserta. Selama masa pelatihan, materi dan bimbingan diberikan oleh Bapak Sugito, VP Engineering di PT Cranium Royal Aditama. Selain itu banyak bantuan yang diberikan oleh mentor dari batch yang ada sebelumnya. Peserta magang diberikan tugas untuk mengembangkan fitur pada sistem Enterprise Resource Planning milik Cranium. Pengembangan fitur ini dikoordinasikan melalui platform Discord dan juga WhatsApp untuk mempermudah komunikasi antar individu. Perkembangan progress dan pemberian tugas dapat diakses melalui figma. Selain itu peserta magang juga berkesempatan menjadi Quality Assurance.

3.2 Tugas yang Dilakukan

Tugas yang diterima selama masa magang sebagai Full Stack Developer di PT Cranium Royal Aditama mencakup back-end dan juga front-end. Bahasa pemrograman yang digunakan adalah JavaScript, framework Java Spring-boot, TypeScript, HTML, dan CSS. Software yang digunakan pada pengembangan kali ini adalah IntelliJ, PGAdmin, GitHub Desktop, dan Postman. Pengembangan sistem ERP yang dilakukan adalah penambahan fitur pada sub-modul Area dan UOM Conversion serta bug fixing di modul Master. Proses pengembangan ini mencakup penyempurnaan fitur CRUD (Create, Read, Update, dan Delete). Proses dilakukan mulai dari pengembangan back-end, unit-test, hingga tampilan front-end. Selain itu, tugas lain yang diterima selama masa magang di Cranium adalah Quality Assurance Analyst. Pengerjaan dilakukan pada project WinGas dan juga BCA. Tugas ini dilakukan untuk memastikan bahwa semua requirement yang di minta oleh client dapat dipenuhi dengan baik.

3.3 Uraian Pelaksanaan Magang

Berikut adalah uraian pelaksanaan kerja magang selama berada di PT Cranium Royal Aditama mulai dari tabel pelaksanaan kerja magang, *sitemap*,

flowchart, hingga tugas yang dilakukan.

3.3.1 Pelaksanaan Kerja Magang

Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Pekerjaan yang dilakukan setiap minggu selama pelaksanaan magang

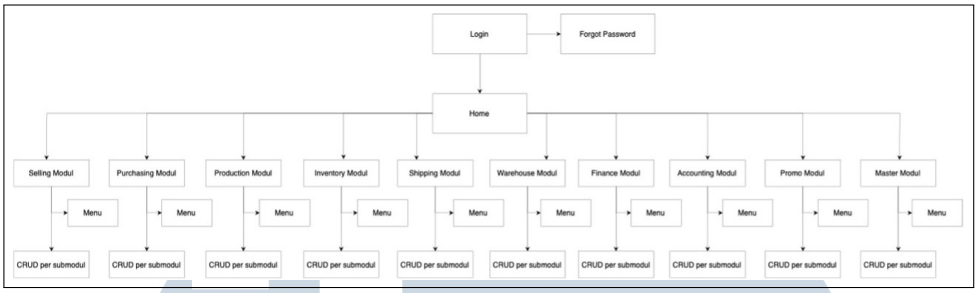
Minggu Ke -	Pekerjaan yang dilakukan
1	Pengenalan perusahaan (onboarding), instalas software penunjang, tools, library, set up Java Springboot, NextJS, PostgreSQL, serta alur kerja di perusahaan.
2	Training Springboot, konsep clean code, dan berfokus pada Controller, Service, Repository, Data Transfer Object, serta Validation.
3	Melanjutkan training backend pada ERP yang berfokus pada struktur kode, message, serta authorization.
4	Melanjutkan training backend proyek ERP yang berfokus pada konsep tesu suites (contract test dan unit test).
5	Review materi training backend serta persiapan pembelajaran frontend.
6	Memulai trianing frontend Cranium ERP.
7	Melanjutkan training frontend Cranium ERP, berfokus pada struktur halaman CRUD dan integrasi dengan backend melalui endpoint yang sudah dibuat.
8	Melanjutkan training frontend yang berfokus pada pembuatan halaman CRUD dan unit test.
9	Review training frontend dan backend.
10	Onboarding serta setup pada proyek utama Cranium ERP.
11	Mengerjakan tugas pada modul Master. Bagian atau submodul Area. Mengerjakan fitur auto generate untuk Area kode pada Backend.
12	Mengerjakan frontend untuk field code serta revisi pada backend.
Lanjut di halaman berikutnya	

Tabel 3.1 – Pekerjaan yang dilakukan setiap minggu selama pelaksanaan magang (lanjutan)

Minggu Ke -	Pekerjaan yang dilakukan
13	Mengerjakan tugas pada modul Master submodul UOM Conversion untuk UOM Conversion Code pada backend.
14	Mengerjakan frontend untuk field code serta revisi pada unit test dan backend.
15	Onboarding Quality Assurance pada proyek WinGass WPI. Pembuatan scenario test dan testing app Driver.
16	Quality Assurance pada proyek BCA LMS. Pembuatan test scenarios & testing modul Libraries.
17	Requirement review & pembuatan test scenarios pada modul Items Quality Assurance BCA modul
18	BCA LMS Quality Assurance testing modul Items & review requirement modul References
19	Pembuatan Test Scenarios modul References pada BCA LMS bagian Item Types, Kode OJK, Kode AR, Penyelenggara, Course Group, Category.
20	Pembuatan Test Scenarios modul References pada BCA LMS bagian Withdrawal Reason, Completion Status, Events & Public Holiday, Physical Resources, Mapping TTF Content & Add Instructor ID, Contributor Type, Mapping Contributor, Rate Apresiasi Contributor, dan Rate Insentif BLI.

3.3.2 Sitemap

Sitemap merupakan navigasi dalam pengembangan sebuah *website* agar lebih mudah diakses oleh pengunjung, membantu mereka menemukan informasi yang dibutuhkan dengan cepat dan juga efisien. Pada pengembangan sistem ERP, sitemap dirancang seperti pada gambar 3.1 agar halaman *login* menjadi akses utama yang dapat diakses pengguna. Setelah berhasil masuk, pengguna akan diarahkan kedalam halaman tertentu sesuai yang dibutuhkan oleh pengguna.



Gambar 3.1. Sitemap

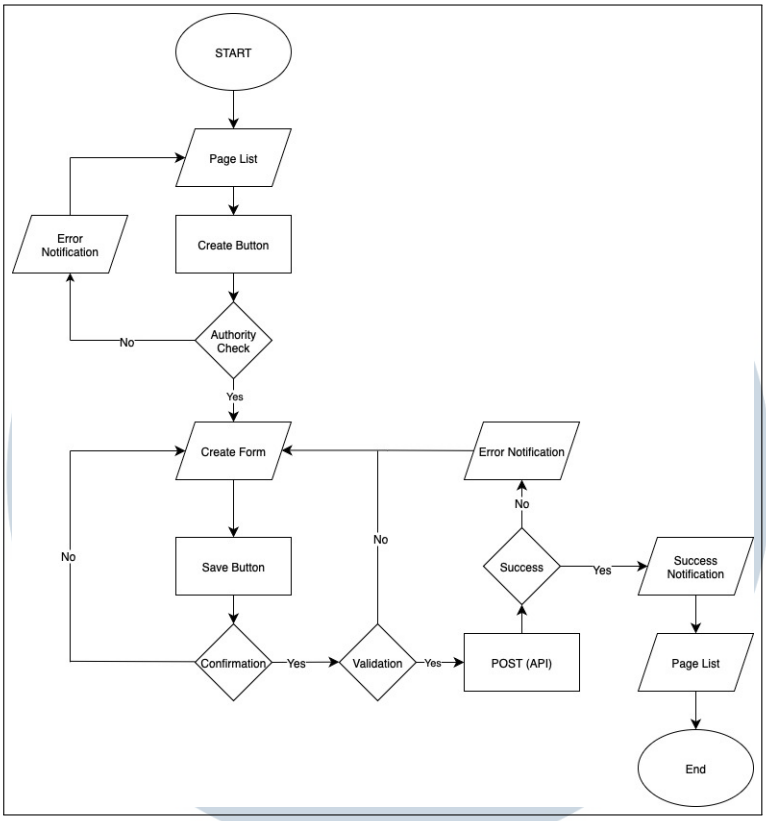
Sumber: [11]

3.3.3 Flowchart

Flowchart digunakan sebagai acuan untuk memahi proses kerja dari sistem ERP yang akan dikembangkan. Halaman yang dikembangkan dan disempurnakan mencakup fitur *create*, *read*, *update*, dan *delete*. Seluruh proses yang akan dilewati oleh pengguna akan digambarkan melalui *flowchart* yang ada.

A Create

Pada proses pembuatan data (*create*) akan diawali dari halaman *list* masing-masing submodul seperti pada flowchart 3.2. User akan menekan tombol *create* dan diarahkan pada halaman *create*. Pada halaman *create* user akan disuguhi dengan *form create* yang berisi kolom-kolom (*fields*) dari submodul tertentu. Sebelum dapat mengakses halaman *create*, sistem akan melakukan *checking* terhadap *authority* yang dimiliki oleh pengguna. Jika user tidak memiliki *authority* yang *valid*, maka akan muncul notifikasi *error*. Setelah memiliki *authority* untuk *create* maka pengguna dapat mengakses halaman *create form*. Pengguna perlu mengisi *form* dan melakukan *save*. Muncul *pop-up* modal konfirmasi sebelum data akan di validasi, jika data madatory sudah terisi maka barulah data akan *hit* API (POST). Data akan di validasi untuk pengecekan duplikat/*invalid* data, jika gagal maka akan muncul notifikasi *error* dan mengembalikan pengguna ke halaman *create.* Jika berhasil maka akan muncul notifikasi sukses dan mengembalikan pengguna ke halaman *list*.

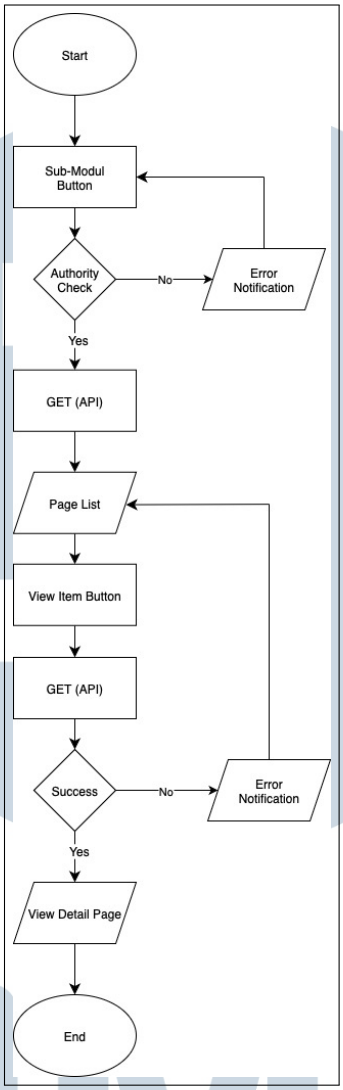


Gambar 3.2. *Flowchart create*

Gambar 3.2 menggambarkan bagaimana alur kerja *create* pada sistem ERP yang disempurnakan. Alur kerja create digunakan dan berlaku pada semua submodul yang ada.

B Read

Pada proses untuk melihat data (read) akan diawali dengan pengguna memilih submodul melalui button sidebar. Jika pengguna memiliki authority maka pengguna akan hit API (read) untuk menampilkan data dari submodul yang dipilih, namun jika tidak memiliki akses maka pop-up notifikasi error akan muncul seperti pada flowchart 3.3. Setelah pengguna mendapatkan tampilan list data dari submodul yang dipilih maka pengguna juga dapat mengakses data untuk masing-masing id dengan view button. Ketika pengguna menekan view button maka halaman view akan terbuka dan akan menampilkan data detail dari setiap id.



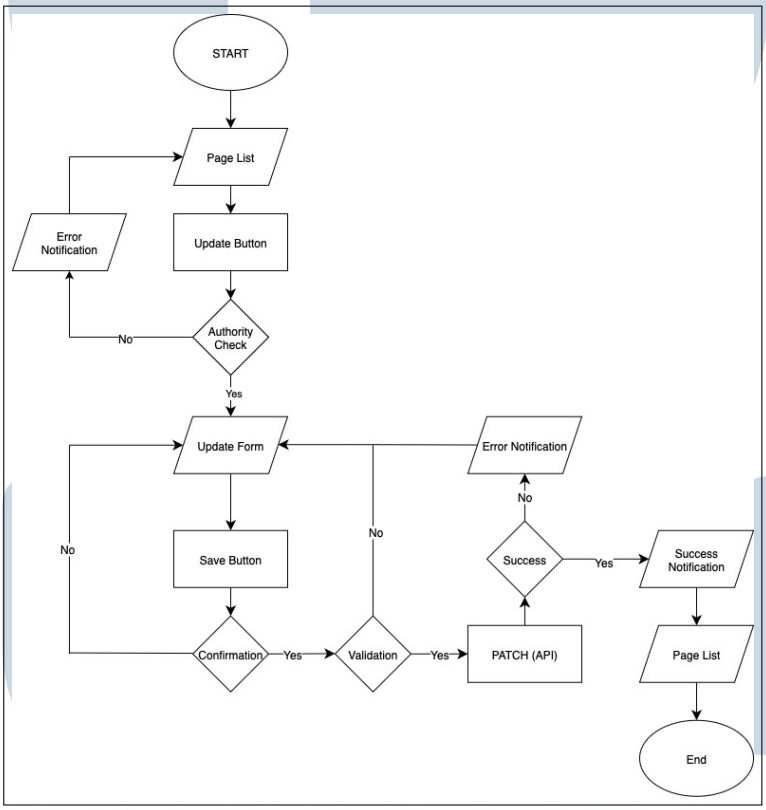
Gambar 3.3. *Flowchart read*

Gambar 3.3 menggambarkan bagaimana alur kerja *read* pada sistem ERP yang disempurnakan. Alur kerja *read*digunakan dan berlaku pada semua submodul yang ada.

C Update

Pada proses untuk melakukan *update* pada sebuah data, akan diawali dari halaman *list* masing-masing *submodul* seperti pada flowchart 3.4. Pengguna akan menekan tombol *update* dan akan diarahkan ke halaman *update*. Pengguna akan mendapatkan *updateform* yang merupakan kolom-kolom (*fields*) yang dapat diisi untuk memperbarui/ mengganti sebuah data. Sebelum pengguna dapat mengakses

halaman *update* maka pengguna harus memiliki *authority* terlebih dahulu. Sistem akan melakukan *authority checking* pada pengguna, jika *valid* maka pengguna dapat mengakses dan melakukan update, jika tidak maka akan muncul *notifikasi error*. Pengguna dapat mengisi *form* dan melakukan *save*. Ketika pengguna melakukan *save* maka akan muncul *pop-up modal* konfirmasi sebelum pengguna melakukan *update* data, jika data berhasil di validasi maka sistem akan *hit* API (PATCH). Data akan di validasi untuk pengecekan duplikat/invalid data, jika gagal maka akan muncul *notifikasi error* dan mengembalikan pengguna ke halaman *update form*. Namun jika berhasil maka akan muncul *notifikasi* sukses dan mengembalikan pengguna ke halaman *list*.

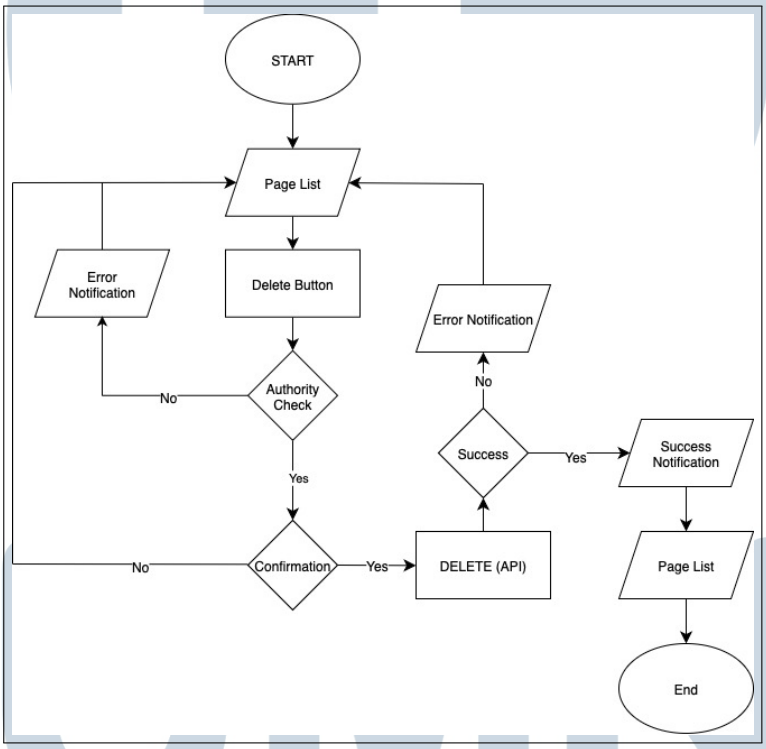


Gambar 3.4. *Flowchart update*

Gambar 3.4 menggambarkan bagaimana alur kerja *update* pada sistem ERP yang disempurnakan. Alur kerja *update*digunakan dan berlaku pada semua submodul yang ada.

D Delete

Pada proses delete sebuah data, akan diawali dari halaman list masing-masing submodul seperti pada flowchart 3.5. Pengguna akan menekan tombol delete pada id yang dipilih, pop-up modal confirmation akan muncul. Jika pengguna memilih yes maka sistem akan hit API (DELETE) dan menghapus data tersebut. Jika ditemukan error maka akan muncul notifikasi error kepada pengguna, jika sukses maka pengguna akan menerima notifikasi sukses, data akan terhapus dan pengguna akan diarahkan kembali ke halaman list.



Gambar 3.5. *Flowchart delete*

Gambar 3.5 menggambarkan bagaimana alur kerja *delete* pada sistem ERP yang disempurnakan. Alur kerja *delete* digunakan dan berlaku pada semua submodul yang ada.

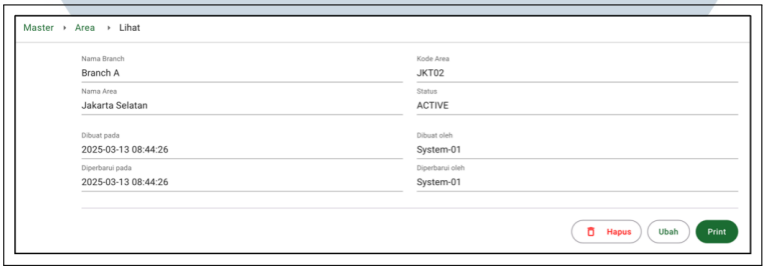
3.3.4 Pengembangan dan Konsep

Pengembangan dan juga perbaikan fitur yang dilakukan selama magang di PT Cranium Royal Aditama serta konsep yang dgunakan untuk pengembangan.

A Pengembangan

Pengembangan dan juga penyempurnaan di yang dilakukan selama berada di proyek ERP Cranium berfokus pada perbaikan *bugs*. Perbaikan *bugs* difokuskan pada modul master, dimana setelah dilakukan internal testing temukan beberapa *bug* yang menggangu sistem ERP, yang sudah dibuat sebelumnya. Perbaikan *bug* ini mencakup frontend dan juga backend. Jenis *bug* yang ditemukan bervariatif dan memiliki penyelesaian masalah yang bervariatif juga. Beberapa *bugs* yang ditemukan:

- Perbaikan fitur pada submodul *Area*. Terdapat beberapa perbaikan pada submodul area. *Perbaikan* fitur *area code* yang mana seharusnya tidak dapat diupdate setelah di *create* seperti yang ada pada gambar 3.6 dengan mengubah kode seperti pada kode 3.1.



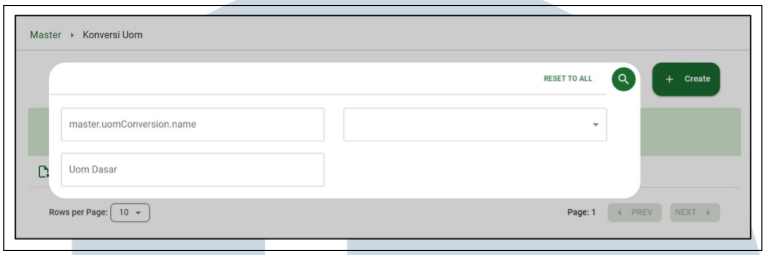
Gambar 3.6. Tampilan submodul area

```
<TextField
            defaultValue={area.areaCode}
            fullWidth
            id="areaCode"
            label={t('master.area.areaCode', { ns: 'master'
   })}
            error={Boolean(errors['areaCode'])}
            helperText={errors['areaCode']?.message ?? ''}
            {...register('areaCode')}
            disabled
          />
```

Kode 3.1: *Disable area code*

- Perbaikan fitur pada submodul *UOM Conversion*. Terdapat beberapa perbaikan pada submodul *UOM Conversion*, seperti translate yang salah

seperti pada gambar 3.7 dengan menggunakan kode seperti pada kode 3.2, dan juga *path* yang salah. Serta penambahan *field code*.

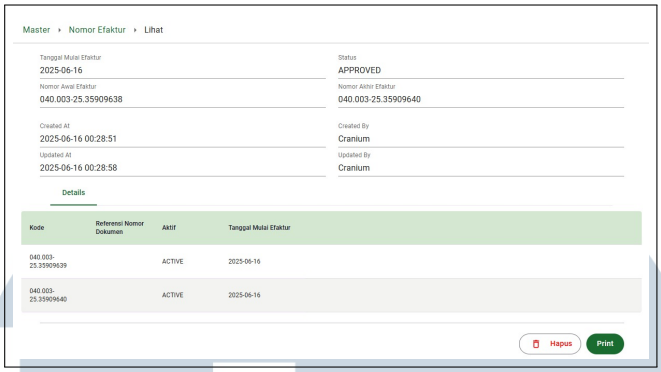


Gambar 3.7. Tampilan submodul UOM *Conversion*

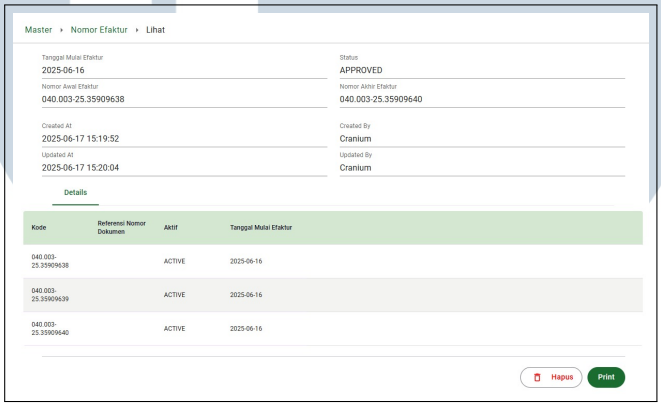
```
"master.uomConversion.uomConversionName": "Uom Conversion
    Name",

"master.uomConversion.name": "Uom Conversion Name",

  "master.uomConversion.baseUomName": "Base Uom",
  "master.uomConversion.baseQuantity": "Base Quantity",
  "master.uomConversion.targetUomName": "Target Uom",
  "master.uomConversion.targetQuantity": "Target Quantity",
  "master.breadcrumb.uomConversion": "Uom Conversion",
  "master.uomConversion.created": "Uom Conversion {{
    uomConversion.uomConversionName}} Created",
  "master.uomConversion.updated": "Uom Conversion  {{
    uomConversion.uomConversionName}}  Updated",
  "master.uomConversion.deleted": "Uom Conversion  {{
    uomConversion.uomConversionName}}  Deleted",
  "master.uomConversion.uomConversionDetail": "Conversion
    Detail",
```

Kode 3.2: UOM *Conversion Translation*

- Perbaikan fitur pada submodul *Efaktur Number*. Terdapat perbaikan pada submodul *Efaktur Number* menjadi seperti pada gambar 3.9 yang mana awalnya nomor awal *efaktur* tidak termasuk pada nomor *efaktur* seperti pada gambar 3.8 dengan menggunakan kode seperti pada 3.3.



Gambar 3.8. Tampilan submodul Efaktur Number



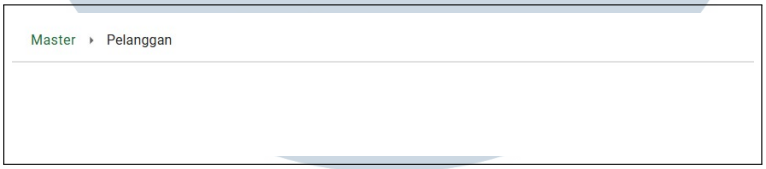
Gambar 3.9. Tampilan submodul Efaktur Number sesudah

```
1   private void generateEfakturNumberDetails(EfakturNumber
    efakturNumber) {
2       String startNumber = efakturNumber.getStartingNumber
    ();
3       String endNumber = efakturNumber.getEndingNumber();
4
5       String prefix = startNumber.substring(0, startNumber.
    lastIndexOf('.') + 1);
6       int startNumeric = Integer.parseInt(startNumber.
    substring(startNumber.lastIndexOf('.') + 1));
7       int endNumeric = Integer.parseInt(endNumber.substring
    (endNumber.lastIndexOf('.') + 1));
8
9       for (int i = startNumeric; i <= endNumeric; i++) {
10          String code = String.format("%s%08d", prefix, i);
11
12          EfakturNumberDetail detail = new
    EfakturNumberDetail();
```

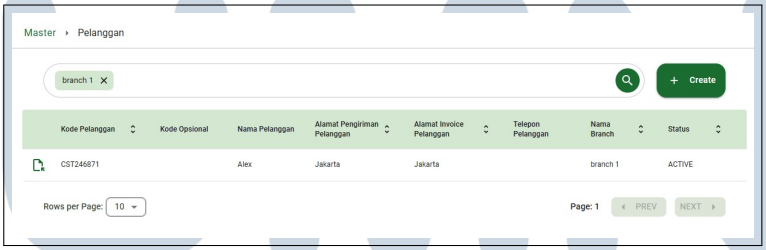
```
13              detail.setEfakturNumber(efakturNumber);
14              detail.setCode(code);
15              detail.setReferencesDocumentNo("");
16              detail.setIsActive((short)
    EfakturNumberDetailStatus.ACTIVE.getValue());
17
18              efakturNumberDetailRepository.save(detail);
19          }
20      }
```

Kode 3.3: Efaktur Number *Code*

- Perbaikan fitur pada submodul Customer. Terdapat perbaikan pada submodul customer, dimana fitur search branch tidak dapat dilakukan seperti pada gambar 3.10 dengan menggunakan kode 3.4 sehingga search branch dapat dilakukan seperti pada gambar 3.11.



Gambar 3.10. Tampilan submodul *Customer*



Gambar 3.11. Tampilan submodul *Customer* sesudah

```
1           if (!Objects.isNull(customerRequestDto.getBranchId()))
    {
2           spec = Specification.where(spec).and(
    specSearchCriteria.getSpecification(new SearchCriteria("
    branchId", customerRequestDto.getBranchId(), SearchOperation.
    EQUAL)));
3       }
```

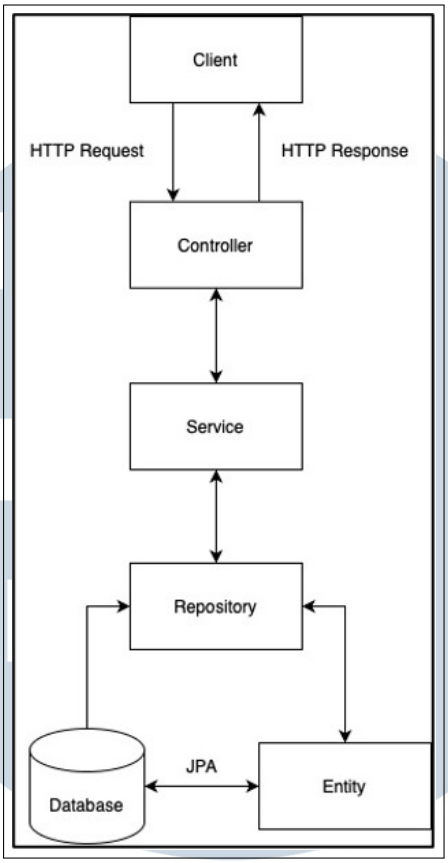
Kode 3.4: *Customer branch id search*

B Konsep

Beberapa konsep yang digunakan dalam pengembangan ERP Cranium antara lain adalah sebagai berikut:

1. *Data Transfer Object, Entity,* dan *Mapper Data Transfer Object* (DTO) adalah objek sederhana yang digunakan untuk membawa data antar proses atau antar *layer* dalam aplikasi. DTO tidak mengandung logika bisnis, melainkan hanya berisi data yang diperlukan dan metode untuk mengakses data tersebut. Penggunaan DTO bertujuan untuk membatasi data yang dikirimkan, sehingga hanya data yang relevan yang diteruskan ke client. Dengan demikian, data sensitif yang tidak perlu tidak akan terekspos. *Entity* adalah objek yang merepresentasikan struktur data pada *database*. Entity memetakan tabel dalam database ke dalam bentuk objek yang dapat digunakan dalam aplikasi. *Entity* juga mendefinisikan kolom dan relasi antar tabel dalam database agar dapat diolah melalui sistem. *Mapper* adalah komponen yang berfungsi untuk mengubah data dari satu bentuk ke bentuk lain. Dalam pengembangan ERP Cranium, mapper digunakan untuk mengonversi DTO menjadi *entity*, maupun sebaliknya, agar data dapat diproses dengan baik di setiap layer yang berbeda.

2. *Controller, Service*, dan *Repository* ERP Cranium menerapkan arsitektur bertingkat (*layered architecture*) yang memisahkan tanggung jawab setiap komponen menjadi tiga bagian utama: Controller, Service, dan Repository. Controller berperan sebagai lapisan presentasi yang menerima permintaan (request) dari client dan mengembalikannya dalam bentuk respon (response). Controller tidak mengolah logika bisnis, tetapi hanya meneruskan data yang diterima ke *service*. Validasi sederhana dapat dilakukan pada *controller* untuk memastikan data yang masuk layak diproses. *Service* bertanggung jawab untuk mengelola logika bisnis dalam aplikasi. *Service* memproses data yang diterima dari controller, mengelola alur bisnis, dan mengatur transformasi data, baik dari DTO ke *entity* maupun sebaliknya. *Service* menjadi penghubung antara *controller* dan *repository*. *Repository* merupakan lapisan yang berinteraksi langsung dengan database. *Repository* mengelola operasi CRUD (*Create, Read, Update, Delete*) dan bertugas menyimpan atau mengambil data dari *database* sesuai kebutuhan *service*.



Gambar 3.12. Hubungan *Controller-Service-Repository*

Proses dimulai ketika client mengirimkan HTTP *request* melalui API seperti pada gambar 3.12. *Controller* menerima *request* tersebut dan meneruskannya ke *service*. *Service* kemudian memproses data dan berinteraksi dengan *repository* untuk melakukan operasi database. Data hasil pemrosesan dikembalikan dari *repository* ke *service*, kemudian diteruskan ke *controller* untuk dikirimkan kembali ke client dalam bentuk *response*.

3. *Unit Test* dan *Contract Test* Pengujian merupakan bagian penting dalam pengembangan ERP Cranium untuk memastikan kualitas dan stabilitas sistem. Unit Test dilakukan untuk menguji bagian terkecil dari aplikasi, seperti metode atau *class*, secara terpisah. Pengujian ini bertujuan untuk memverifikasi bahwa setiap unit berfungsi sesuai dengan yang diharapkan. Pada *backend*, *unit test* digunakan untuk menguji metode-metode pada *service*, sedangkan pada *frontend*, *unit test* digunakan untuk memastikan komponen *UI* berfungsi dengan benar. *Contract Test* bertujuan untuk memastikan bahwa komunikasi antar bagian dalam aplikasi berjalan sesuai

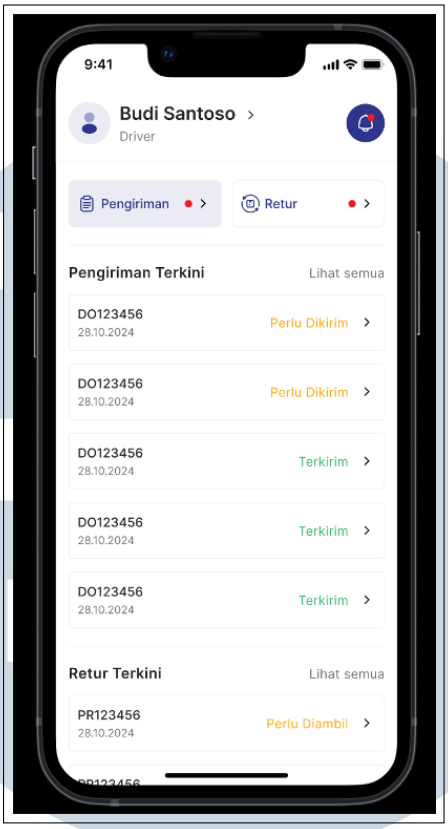
dengan kesepakatan atau kontrak yang telah ditetapkan, terutama pada API yang digunakan oleh *client*. *Contract test* memastikan bahwa endpoint yang disediakan sudah sesuai dengan format *request* dan *response* yang diharapkan. Dalam pengembangan ERP Cranium, unit test dan *contract test* diterapkan baik pada *backend* maupun *frontend* untuk memaksimalkan keandalan aplikasi.

3.3.5 Assurance Analyst

Selama kegiatan magang di Cranium, project yang sedang berlangsung cukup bervariatif dan jumlah yang signifikan. Pada kegiatan magang terdapat juga project WinGas, dalam project ini terfokus ke dalam tiga bagian, *website* CMS, aplikasi *driver*, dan juga aplikasi *sales*. Terdapat juga *project* BCA yang berfokus pada sistem LMS.

A WinGas

Proyek WinGas (kode proyek: WPI) merupakan proyek pengembangan yang terdiri dari tiga komponen utama, yaitu *Website* CMS (*Content Management System*), Aplikasi *Driver*, dan Aplikasi *Sales*. Setiap komponen memiliki peran penting dalam menunjang efisiensi, koordinasi, dan produktivitas dari sistem yang dibuat. Pada gambar 3.13 merupakan tampilan dari Aplikasi *Driver* WinGas.



Gambar 3.13. Tampilan aplikasi *driver* WinGas

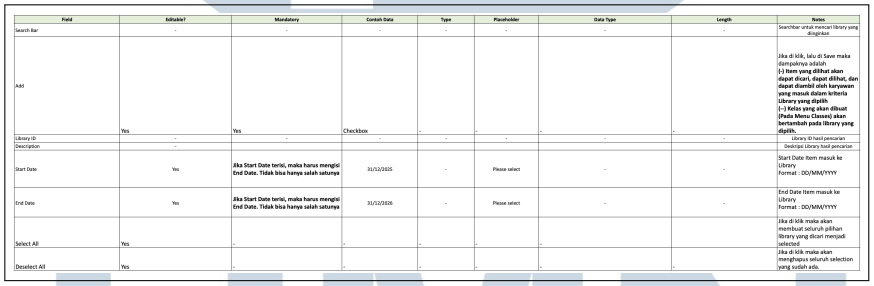
Untuk memastikan sistem berjalan sesuai kebutuhan pengguna dan standar kualitas yang ditetapkan, dilakukan proses *Quality Assurance (QA)* secara menyeluruh. QA bertujuan menjaga stabilitas, keamanan, dan performa setiap aplikasi dalam proyek ini. Salah satu tahapan penting dalam QA adalah penyusunan t*est case* yang mengacu pada dokumen requirement dari tim analis dan *stakeholder*. *Test case* ini menjadi panduan pengujian untuk memastikan fitur berjalan sesuai fungsinya. Setelah *test case* disusun dan divalidasi, tim QA melakukan pengujian secara manual maupun otomatis, disesuaikan dengan kompleksitas fitur yang diuji. Hasil pengujian dianalisis untuk mengidentifikasi *bug*, error, *dan* potensi risiko. Dengan proses QA yang terstruktur dan terdokumentasi, proyek WinGas memastikan seluruh modul telah memenuhi standar kualitas sebelum digunakan dalam lingkungan operasional.

B BCA LMS

Proyek BCA LMS merupakan proyek pengembangan sistem yang berfokus pada pembuatan *Learning Management System* (LMS). Sistem ini dirancang

untuk memfasilitasi proses pembelajaran daring yang terstruktur, efisien, dan mudah diakses oleh pengguna. Dalam pengembangan BCA LMS, tim berupaya membangun platform yang mampu mengelola materi pembelajaran, mengatur jadwal pelatihan, memantau progres pengguna, serta menyediakan fitur evaluasi secara terintegrasi. Tujuan utama dari proyek ini adalah menciptakan lingkungan belajar digital yang dapat meningkatkan efektivitas pelatihan bagi pengguna internal. Untuk memastikan sistem LMS berjalan sesuai kebutuhan pengguna dan memenuhi standar kualitas yang ditetapkan, dilakukan proses *Quality Assurance (QA)* secara menyeluruh. Proses QA pada proyek BCA LMS mengikuti alur berikut:

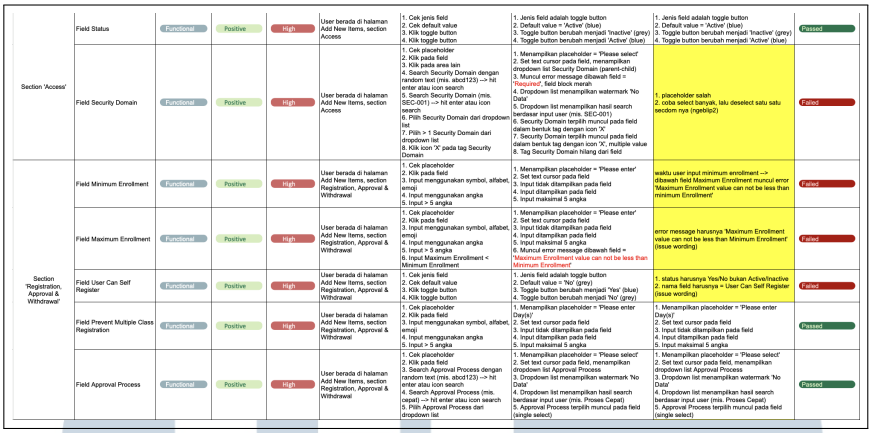
1. *Update Requirement* dari User Proses QA dimulai setelah tim pengembang menerima pembaruan kebutuhan atau requirement dari pengguna. *Requirement* ini menjadi dasar bagi tim QA dalam menyusun skenario pengujian yang relevan. Requirement dari *user* diberikan dalam bentuk tabel *spreadsheet* seperti pada gambar 3.14.



Field	Editable?	Mandatory	Contoh Data	Type	Placeholder	Data Type	Length	Notes
Search Bar	-	-	-	-	-	-	-	Searchbar untuk mencari library yang diinginkan
Add	Yes	Yes	Checkbox	-	-	-	-	Jika di klik, lalu di Save maka dampaknya adalah (-) Item yang dilihat akan dapat dicari, dapat dilihat, dan dapat diambil oleh karyawan yang masuk dalam kriteria Library yang dipilih (--) Kelas yang akan dibuat (Pada Menu Classes) akan bertambah pada library yang dipilih.
Library ID	-	-	-	-	-	-	-	Library ID hasil pencarian
Description	-	-	-	-	-	-	-	Deskripsi Library hasil pencarian
Start Date	Yes	Jika Start Date terisi, maka harus mengisi End Date. Tidak bisa hanya salah satunya	31/12/2025	-	Please select	-	-	Start Date Item masuk ke Library Format : DD/MM/YYYY
End Date	Yes	Jika Start Date terisi, maka harus mengisi End Date. Tidak bisa hanya salah satunya	31/12/2026	-	Please select	-	-	End Date Item masuk ke Library Format : DD/MM/YYYY
Select All	Yes	-	-	-	-	-	-	Jika di klik maka akan membuat seluruh pilihan library yang dicari menjadi selected
Deselect All	Yes	-	-	-	-	-	-	Jika di klik maka akan menghapus seluruh selection yang sudah ada.

Gambar 3.14. *Requirement items*

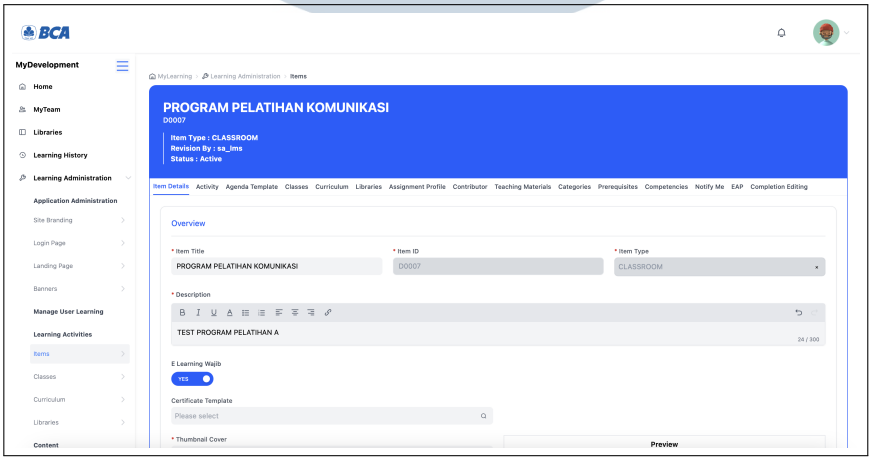
2. *Create Scenario* Sambil menunggu proses pengembangan dari tim developer, tim QA mulai menyusun skenario pengujian (*test scenario*) dan *test case* berdasarkan *requirement* yang telah diperbarui. Penyusunan skenario seperti pada gambar 3.15 dilakukan secara paralel untuk memastikan kesiapan pengujian saat pengembangan selesai.



Section 'Access'	Field Status	Functional	Positive	High	User berada di halaman Add New Items, section Access	1. Cek jenis field 2. Cek default value 3. Klik toggle button 4. Klik toggle button	1. Jenis field adalah toggle button 2. Default value = 'Active' (blue) 3. Toggle button berubah menjadi 'Inactive' (grey) 4. Toggle button berubah menjadi 'Active' (blue)	1. Jenis field adalah toggle button 2. Default value = 'Active' (blue) 3. Toggle button berubah menjadi 'Inactive' (grey) 4. Toggle button berubah menjadi 'Active' (blue)	Passed
	Field Security Domain	Functional	Positive	High	User berada di halaman Add New Items, section Access	1. Cek placeholder 2. Klik pada field 3. Klik pada area lain 4. Search Security Domain dengan random text (mis. abcd123) --> hit enter atau icon search 5. Search Security Domain (mis. SEC-001) --> hit enter atau icon search 6. Pilih Security Domain dari dropdown list 7. Pilih > 1 Security Domain dari dropdown list 8. Klik icon 'X' pada tag Security Domain	1. Menampilkan placeholder = 'Please select' 2. Set text cursor pada field, menampilkan dropdown list Security Domain (parent-child) 3. Muncul error message dibawah field = 'Required', field block merah 4. Dropdown list menampilkan watermark 'No Data' 5. Dropdown list menampilkan hasil search berdasar input user (mis. SEC-001) 6. Security Domain terpilih muncul pada field dalam bentuk tag dengan icon 'X' 7. Security Domain terpilih muncul pada field dalam bentuk tag dengan icon 'X', multiple value 8. Tag Security Domain hilang dari field	1. placeholder salah 2. coba select banyak, lalu deselect satu satu secdom nya (ngeblip2)	Failed
Section 'Registration, Approval & Withdrawal'	Field Minimum Enrollment	Functional	Positive	High	User berada di halaman Add New Items, section Registration, Approval & Withdrawal	1. Cek placeholder 2. Klik pada field 3. Input menggunakan symbol, alfabet, emoji 4. Input menggunakan angka 5. Input > 5 angka	1. Menampilkan placeholder = 'Please enter' 2. Set text cursor pada field 3. Input tidak ditampilkan pada field 4. Input ditampilkan pada field 5. Input maksimal 5 angka	waktu user input minimum enrollment --> dibawah field Maximum Enrollment muncul error 'Maximum Enrollment value can not be less than minimum Enrollment'	Failed
	Field Maximum Enrollment	Functional	Positive	High	User berada di halaman Add New Items, section Registration, Approval & Withdrawal	1. Cek placeholder 2. Klik pada field 3. Input menggunakan symbol, alfabet, emoji 4. Input menggunakan angka 5. Input > 5 angka 6. Input Maximum Enrollment < Minimum Enrollment	1. Menampilkan placeholder = 'Please enter' 2. Set text cursor pada field 3. Input tidak ditampilkan pada field 4. Input ditampilkan pada field 5. Input maksimal 5 angka 6. Muncul error message dibawah field = 'Maximum Enrollment value can not be less than Minimum Enrollment'	error message harusnya 'Maximum Enrollment value can not be less than Minimum Enrollment' (issue wording)	Failed
	Field User Can Self Register	Functional	Positive	High	User berada di halaman Add New Items, section Registration, Approval & Withdrawal	1. Cek jenis field 2. Cek default value 3. Klik toggle button 4. Klik toggle button	1. Jenis field adalah toggle button 2. Default value = 'No' (grey) 3. Toggle button berubah menjadi 'Yes' (blue) 4. Toggle button berubah menjadi 'No' (grey)	1. status harusnya Yes/No bukan Active/Inactive 2. nama field harusnya = User Can Self Register (issue wording)	Failed
	Field Prevent Multiple Class Registration	Functional	Positive	High	User berada di halaman Add New Items, section Registration, Approval & Withdrawal	1. Cek placeholder 2. Klik pada field 3. Input menggunakan symbol, alfabet, emoji 4. Input menggunakan angka 5. Input > 5 angka	1. Menampilkan placeholder = 'Please enter Daryia' 2. Set text cursor pada field 3. Input tidak ditampilkan pada field 4. Input ditampilkan pada field 5. Input maksimal 5 angka	1. Menampilkan placeholder = 'Please enter Daryia' 2. Set text cursor pada field 3. Input tidak ditampilkan pada field 4. Input ditampilkan pada field 5. Input maksimal 5 angka	Passed
	Field Approval Process	Functional	Positive	High	User berada di halaman Add New Items, section Registration, Approval & Withdrawal	1. Cek placeholder 2. Klik pada field 3. Search Approval Process dengan random text (mis. abcd123) --> hit enter atau icon search 4. Search Approval Process (mis. cepat) --> hit enter atau icon search 5. Pilih Approval Process dari dropdown list	1. Menampilkan placeholder = 'Please select' 2. Set text cursor pada field, menampilkan dropdown list Approval Process 3. Dropdown list menampilkan watermark 'No Data' 4. Dropdown list menampilkan hasil search berdasar input user (mis. Proses Cepat) 5. Approval Process terpilih muncul pada field (single select)	1. Menampilkan placeholder = 'Please select' 2. Set text cursor pada field, menampilkan dropdown list Approval Process 3. Dropdown list menampilkan watermark 'No Data' 4. Dropdown list menampilkan hasil search berdasar input user (mis. Proses Cepat) 5. Approval Process terpilih muncul pada field (single select)	Passed

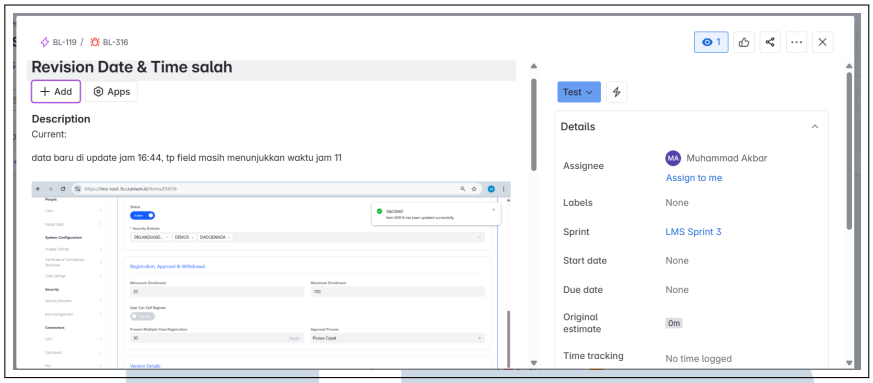
Gambar 3.15. *Scenario Items*

3. *Testing* Setelah fitur selesai dikembangkan, tim QA melakukan pengujian berdasarkan skenario yang telah dibuat. Pengujian dilakukan secara manual maupun otomatis sesuai kebutuhan. Pada tahap ini, QA memverifikasi apakah fungsi berjalan sesuai dengan kebutuhan pengguna. Pengujian dilakukan melalui staging seperti pada gambar 3.16.



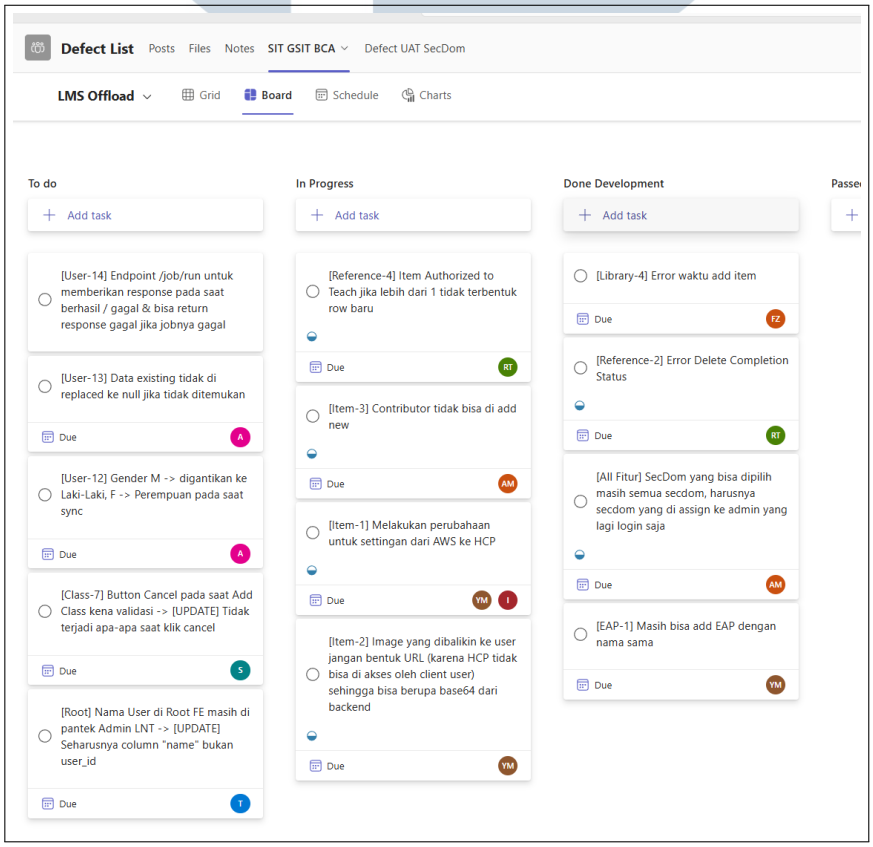
Gambar 3.16. Tampilan *Staging* BCA LMS

4. *Bug Report* Jika ditemukan ketidaksesuaian atau error, tim QA akan mendokumentasikan temuan tersebut dalam sistem *ticketing* seperti pada gambar 3.17. Laporan bug (*bug report*) disusun secara detail, meliputi langkah replikasi, deskripsi masalah, serta dokumentasi pendukung seperti tangkapan layar.



Gambar 3.17. Jira *ticket*

5. *Retest Bug* Setelah bug diperbaiki oleh tim *developer*, QA akan melakukan pengujian ulang (*retest*) untuk memastikan perbaikan berjalan dengan baik dan tidak menimbulkan *error* baru. Jika ditemukan *bug tambahan*, QA akan mengulang proses pelaporan pada jira seperti pada gambar 3.18 hingga seluruh temuan terselesaikan.



Gambar 3.18. *Defect list board*

6. *Deliver to User* Setelah semua fitur teruji dan dinyatakan valid, sistem akan diserahkan kepada pengguna untuk dilakukan *user acceptance testing* (*UAT*) dan dapat dilanjutkan ke tahap produksi atau *deployment*. Dengan alur QA yang terstruktur dan terdokumentasi ini, proyek BCA LMS diharapkan dapat selesai dan memiliki sistem yang stabil, sesuai kebutuhan dan permintaan pengguna, serta memiliki performa yang optimal saat digunakan.

3.4 Kendala dan Solusi yang Ditemukan

Kendala yang dialami selama masa magang:

- Komponen yang memiliki kompleksitas tinggi membutuhkan waktu yang lama dan belum ditemukan *best pratice*nya sehingga membuang waktu.
- *Resources* (*memory, ram,* dan *processor*) terpakai dengan berat sehingga membuat *laptop*/device kewalahan dalam menjalankan proyeknya. Hal ini membuang waktu dan sangat merugikan.
- Banyak perubahan yang dilakukan secara mendadak.

Solusi yang ditemukan:

- Memperbanyak komunikasi dengan tim untuk mempercepat pengerjaan.
- Mempelajari referensi yang ada.
- Memastikan semua requirement dari user jelas sehingga tidak menjadi masalah dikemudian hari