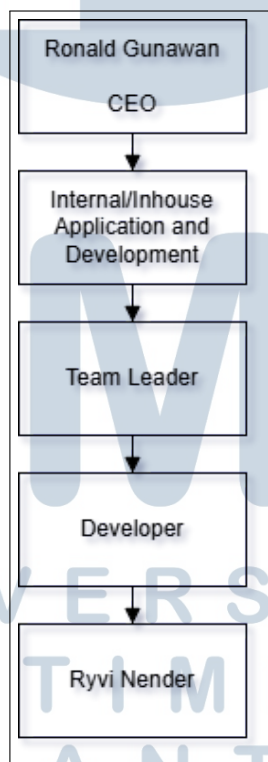


BAB 3

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Pelaksanaan kerja magang dilakukan di PT Vanz Teknologi, tepatnya pada tim pengembangan aplikasi web untuk proyek internal CMS AMS4U. Peran yang dijalankan adalah sebagai Fullstack Engineer, dengan tanggung jawab mencakup pengembangan antarmuka pengguna (*frontend*) menggunakan *framework* React.js, serta pengembangan sisi *backend* menggunakan *framework* NestJS. Selama proses pengembangan, pekerjaan difokuskan pada implementasi fitur, perbaikan *bug*, serta integrasi antara *frontend* dan *backend* agar sistem CMS dapat berjalan secara optimal, Seluruh kegiatan kerja magang ini dilaksanakan di bawah bimbingan Aditia Prasetyo selaku Supervisor, yang memberikan arahan teknis, evaluasi hasil pekerjaan, serta masukan selama proses pengembangan sistem.



Gambar 3.1. Struktur kepemimpinan tim IT

Selama proses magang, bimbingan diberikan oleh selaku supervisor Aditia Prasetyo yang memberikan arahan serta memastikan pekerjaan dilaksanakan sesuai

standar teknis perusahaan dan spesifikasi yang diberikan klien.

3.2 Tugas yang Dilakukan

Selama pelaksanaan kerja magang di PT Vanz Teknologi pada proyek CMS AMS4U, tugas yang dilakukan mencakup pengembangan aplikasi web baik pada sisi *frontend* maupun *backend*. Peran yang dijalankan adalah sebagai *Fullstack Engineer*, dengan uraian tugas sebagai berikut:

A Pengembangan Frontend

- Mengembangkan dan memelihara antarmuka pengguna (UI) pada CMS AMS4U menggunakan *framework* React.js.
- Mengimplementasikan desain antarmuka ke dalam komponen yang *reusable* dan terstruktur.
- Melakukan penyesuaian tampilan (*styling*) agar sesuai dengan standar UI/UX yang ditetapkan.
- Menambahkan fitur baru pada halaman-halaman *website* AMS4U.
- Memperbaiki *bug* pada *frontend* AMS4U.

B Pengembangan Backend

- Melakukan pengembangan dan pemeliharaan sistem *backend* sebagai pendukung operasional aplikasi CMS.
- Menyusun dan menyesuaikan layanan aplikasi untuk memenuhi kebutuhan fungsional sistem.
- Mengelola proses pengolahan data agar mendukung mekanisme penambahan dan pembaruan data.
- Menyediakan layanan data yang dibutuhkan untuk menampilkan informasi pada halaman *dashboard*.
- Melakukan penyesuaian proses pemrosesan data dalam jumlah besar agar berjalan sesuai kebutuhan sistem.

Integrasi Frontend dan Backend

- Melakukan integrasi antara sistem *frontend* dan *backend* agar alur data aplikasi dapat berjalan dengan baik.
- Memastikan data yang dikirim dan diterima antar sistem sesuai dengan kebutuhan fungsional aplikasi.
- Menyesuaikan struktur dan format data agar konsisten antara sisi tampilan dan sisi pemrosesan data.

C Perbaikan Bug dan Optimasi Sistem

- Menelusuri (*tracing*) dan memperbaiki *bug* yang muncul pada CMS, baik di sisi *frontend* maupun *backend*.
- Memperbaiki masalah performa, validasi data, serta *error* yang terjadi saat proses input dan pengolahan data.
- Melakukan *refactoring* kode agar lebih rapi, mudah dibaca, dan dipelihara.

D Koordinasi dan Dokumentasi

- Mengikuti kegiatan rutin seperti *daily standup* dan *meeting* sinkronisasi dengan tim pengembang.
- Mendiskusikan kebutuhan fitur, kendala teknis, dan solusi bersama supervisor dan tim.
- Mendokumentasikan perubahan dan fitur yang dikembangkan sebagai bagian dari proses pengembangan sistem.

3.3 Uraian 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	Pengenalan sistem dan alur kerja perusahaan, mengikuti <i>daily standup</i> , serta memahami struktur proyek CMS dan <i>dashboard</i> yang digunakan.
2	Penyesuaian dan perbaikan navigasi pada CMS, termasuk sinkronisasi breadcrumb, penamaan menu, dan konsistensi struktur halaman.
3	Perbaikan dan penyesuaian tampilan antarmuka (UI) berdasarkan desain UI/UX, termasuk pembaruan ikon dan penataan komponen visual.
4	Pengembangan dan penyempurnaan modul akuntansi, termasuk penambahan daftar COA dan penyesuaian tampilan data transaksi.
5	Perbaikan validasi data dan logika bisnis pada CMS, termasuk penyesuaian <i>field</i> wajib dan <i>fallback handling</i> .
6	Penambahan fitur interaksi pada halaman detail, seperti tombol edit pada modal dan penyempurnaan alur navigasi pengguna.
7	Perbaikan <i>bug</i> pada <i>sidebar</i> , termasuk sinkronisasi menu aktif, izin akses pengguna, dan masalah tampilan submenu.
8	Optimalisasi tampilan dan perhitungan pada modul komisi dan transaksi, termasuk perbaikan tata letak tabel dan komponen hasil.
9	Penyesuaian struktur tabel dan <i>dashboard</i> transaksi, serta peningkatan keterbacaan data pada halaman ringkasan.
10	Pengembangan fitur pendukung <i>dashboard</i> , termasuk penyempurnaan tampilan dan penyesuaian perilaku <i>sidebar</i> .
11	Penambahan fitur filter status dan pengelompokan data untuk meningkatkan kemudahan pencarian dan analisis data.
Lanjut ke halaman berikutnya	

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

Minggu Ke-	Pekerjaan yang Dilakukan
12	Pengembangan dan pengujian komponen impor data, termasuk dukungan format eksternal dan penyesuaian logika impor.
13	<i>Tracing</i> dan perbaikan <i>bug</i> terkait pemuatan data, validasi impor, dan integrasi antar modul <i>frontend</i> dan <i>backend</i> .
14	Integrasi dan penyesuaian <i>endpoint backend</i> untuk mendukung fitur impor dan parameter akuntansi.
15	Refaktor komponen <i>frontend</i> agar lebih modular dan <i>reusable</i> , khususnya pada komponen <i>ImportData</i> .
16	Penyempurnaan fitur impor data, termasuk validasi <i>header file</i> , penanganan nilai khusus, dan fleksibilitas konfigurasi.
17	Pengembangan fitur ringkasan status order pada <i>dashboard</i> dan integrasi dengan <i>endpoint backend</i> .
18	Optimalisasi performa <i>backend</i> dan <i>frontend</i> , termasuk perbaikan <i>endpoint bulk create</i> dan sinkronisasi data.
19	Pengembangan dan penyempurnaan fitur <i>dashboard</i> visual, termasuk <i>widget</i> , <i>carousel</i> , dan tampilan statistik.
20	Finalisasi fitur, <i>refactor</i> kode, perbaikan <i>bug</i> minor, serta evaluasi dan sinkronisasi akhir sistem sebelum penutupan magang.

3.3.1 Software yang digunakan

Berikut adalah perangkat lunak yang digunakan selama pelaksanaan kerja magang di PT Vanz Teknologi dalam proses pengembangan sistem aplikasi berbasis web:

1. PostgreSQL

PostgreSQL merupakan sistem manajemen basis data relasional berbasis *open-source* yang digunakan untuk menyimpan dan mengelola data aplikasi. PostgreSQL dipilih karena stabilitas, performa tinggi, serta dukungan fitur yang lengkap untuk kebutuhan sistem berskala menengah hingga besar.

PostgreSQL digunakan sebagai basis data utama dalam CMS AMS4U untuk

menyimpan data yang diolah oleh backend, termasuk data sekuritas dan harga sekuritas yang diakses melalui API.

2. **pgAdmin**

pgAdmin adalah alat bantu berbasis antarmuka grafis yang digunakan untuk mengelola dan memonitor basis data PostgreSQL. Perangkat lunak ini mempermudah proses pengelolaan tabel, query, serta inspeksi data selama pengembangan dan pengujian aplikasi.

pgAdmin dimanfaatkan selama pengembangan untuk melakukan pengelolaan struktur tabel, pengecekan data, serta verifikasi hasil proses penyimpanan data pada PostgreSQL.

3. **Docker**

Docker digunakan sebagai platform kontainerisasi untuk menjalankan aplikasi beserta dependensinya dalam lingkungan yang terisolasi. Dengan Docker, proses setup dan deployment aplikasi menjadi lebih konsisten di berbagai lingkungan pengembangan.

Docker digunakan untuk menjalankan backend CMS AMS4U dalam lingkungan terkontainerisasi sehingga konfigurasi aplikasi dapat disesuaikan dengan lingkungan kerja tim.

4. **GitHub**

GitHub digunakan sebagai sistem pengelolaan versi kode sumber (*version control*) dan kolaborasi tim. Melalui GitHub, perubahan kode dapat dilacak dengan baik serta mendukung proses kerja kolaboratif antar developer.

5. **Framework React**

React merupakan *library* JavaScript yang digunakan untuk membangun antarmuka pengguna berbasis komponen. React memungkinkan pengembangan UI yang dinamis, modular, dan mudah untuk dirawat.

React digunakan untuk mengembangkan antarmuka pengguna CMS AMS4U agar data dari backend dapat ditampilkan dan dikelola melalui tampilan web.

6. **Framework NestJS**

NestJS adalah *framework backend* berbasis Node.js yang digunakan untuk membangun aplikasi *server-side* yang terstruktur dan *scalable*. *Framework* ini menerapkan konsep modular dan mendukung arsitektur yang rapi untuk pengembangan API.

NestJS digunakan sebagai framework backend untuk mengimplementasikan endpoint API yang digunakan oleh frontend CMS AMS4U.

3.3.2 Cloning Repository Proyek CMS AMS4U

Pada minggu ketiga pelaksanaan kerja magang, dilakukan proses *cloning repository* dari proyek CMS AMS4U yang digunakan sebagai sistem internal perusahaan. Repository tersebut tersedia pada platform GitHub dan digunakan sebagai dasar pengembangan serta pemeliharaan sistem selama masa magang.

Tujuan dari proses *cloning repository* ini adalah untuk memperoleh salinan *source code* ke lingkungan pengembangan lokal agar dapat dilakukan pengembangan fitur, perbaikan bug, serta proses integrasi antara sisi *frontend* dan *backend* secara lebih optimal.

Adapun *repository* yang digunakan dalam proyek CMS AMS4U dapat diakses melalui tautan berikut:

1. AMS4U Backend : <https://github.com/qbit-tech/ams4u-be>
2. CMS Monorepo: <https://github.com/qbit-tech/cms-monorepo>

Setelah proses *cloning* selesai dilakukan, *source code* dari kedua *repository* tersebut dianalisis secara menyeluruh untuk memahami struktur proyek, arsitektur aplikasi, serta alur kerja sistem. Analisis ini mencakup identifikasi *dependencies* proyek, konfigurasi lingkungan (*environment*), serta keterkaitan antara modul *frontend* dan *backend* dalam mendukung fungsionalitas CMS AMS4U.

Kegiatan ini bertujuan untuk memastikan proses pengembangan dan pemeliharaan sistem dapat dilakukan secara efisien, terstruktur, serta sesuai dengan standar teknis yang telah diterapkan oleh perusahaan.

3.3.3 Pengembangan Fitur Sistem

Pengembangan fitur mencakup pembaruan pada mekanisme sistem di latar belakang dan penyempurnaan elemen visual. Berikut adalah daftar perbaikan antar muka serta *backend* yang dilakukan:

A Backend

1. Dilakukan pembaruan pada endpoint Securities dan Securities Price dengan menambahkan mekanisme *upsert* pada proses *bulk create*.

Pada endpoint Securities, sistem melakukan pengecekan keberadaan data berdasarkan secCode sebelum menentukan apakah data akan dibuat sebagai entri baru atau diperbarui. Proses ini melibatkan validasi terhadap data referensi seperti *instrument type*, *market*, *sector*, dan *currency* untuk memastikan relasi data sesuai sebelum disimpan ke basis data.

Pada endpoint Securities Price, dilakukan penyesuaian logika untuk memproses data harga sekuritas berdasarkan kombinasi secId dan closingdate. Sistem mengelompokkan data input, melakukan pencarian data harga yang berpotensi sudah ada, kemudian menentukan apakah setiap data akan disimpan sebagai data baru atau diperbarui. Seluruh proses dijalankan dalam satu transaksi *database* untuk menjaga konsistensi selama proses *bulk create*.

Tabel 3.2. Hasil Pengujian Endpoint Securities dan Securities Price

Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
API Create Bulk Securities	Data sekuritas baru berhasil ditambahkan apabila kode sekuritas belum terdaftar di basis data.	Sesuai dengan yang diharapkan.	Sesuai
API Update Bulk Securities	Data sekuritas yang memiliki kode sama berhasil diperbarui tanpa membuat data duplikat.	Sesuai dengan yang diharapkan.	Sesuai
API Validasi Referensi Securities	Data sekuritas hanya dapat diproses apabila referensi seperti <i>instrument type</i> , <i>market</i> , <i>sector</i> , dan <i>currency valid</i> .	Sesuai dengan yang diharapkan.	Sesuai
Lanjut ke halaman berikutnya			

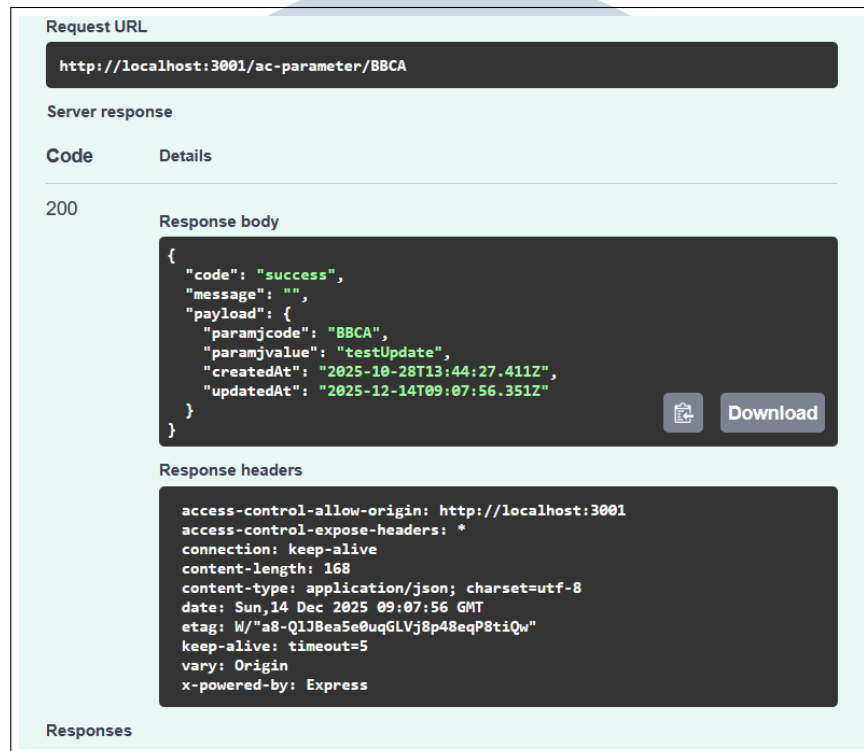
Tabel 3.2 Hasil Pengujian Endpoint Securities dan Securities Price (lanjutan)

Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
API Create Bulk Securities Price	Data harga sekuritas baru berhasil ditambahkan apabila kombinasi sekuritas dan tanggal penutupan belum tersedia.	Sesuai dengan yang diharapkan.	Sesuai
API Update Securities Price	Data harga sekuritas diperbarui apabila data dengan sekuritas dan tanggal penutupan yang sama telah ada.	Sesuai dengan yang diharapkan.	Sesuai
API Bulk Create dengan Data Campuran	Data baru ditambahkan dan data lama diperbarui dalam satu proses bulk sesuai kondisi masing-masing data.	Sesuai dengan yang diharapkan.	Sesuai

- Menambahkan fitur Accounting Parameter pada sisi *backend* yang berfungsi sebagai konfigurasi pendukung modul akuntansi. Fitur ini dikembangkan melalui beberapa *endpoint* utama, yaitu Init untuk inisialisasi data awal, Find dan FindAll untuk pengambilan data parameter, serta Update untuk memperbarui nilai parameter sesuai kebutuhan sistem. Pengembangan fitur ini bertujuan untuk memberikan fleksibilitas dalam pengaturan parameter akuntansi tanpa perlu melakukan perubahan langsung pada kode program, serta memastikan data akuntansi dapat dikelola secara terstruktur dan terpusat.

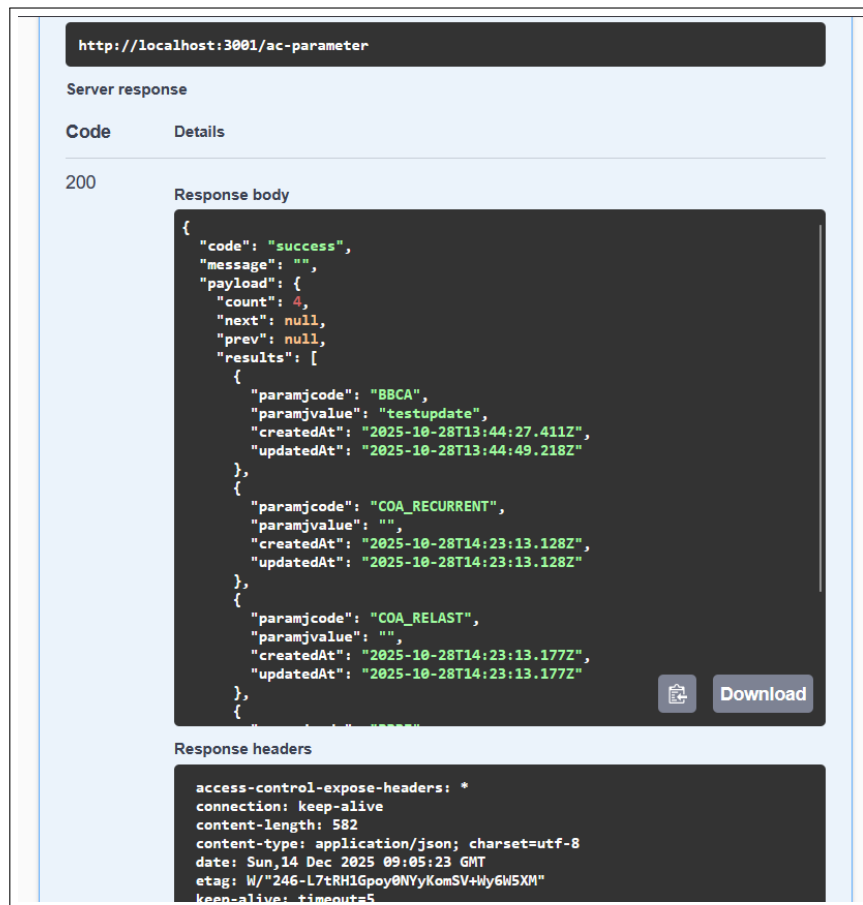
Gambar 3.2 menampilkan hasil pemanggilan *endpoint* ac-parameter/id menggunakan metode PATCH. *Endpoint* ini digunakan untuk melakukan pembaruan data parameter tertentu berdasarkan nilai id yang diberikan. Pada contoh tersebut, server mengembalikan kode status 200, yang menandakan bahwa proses pembaruan data berhasil dilakukan. Respons body berisi informasi parameter yang telah diperbarui, termasuk nilai parameter serta

waktu pembuatan dan pembaruan data. *Response header* juga ditampilkan sebagai bagian dari hasil pengujian *endpoint*.



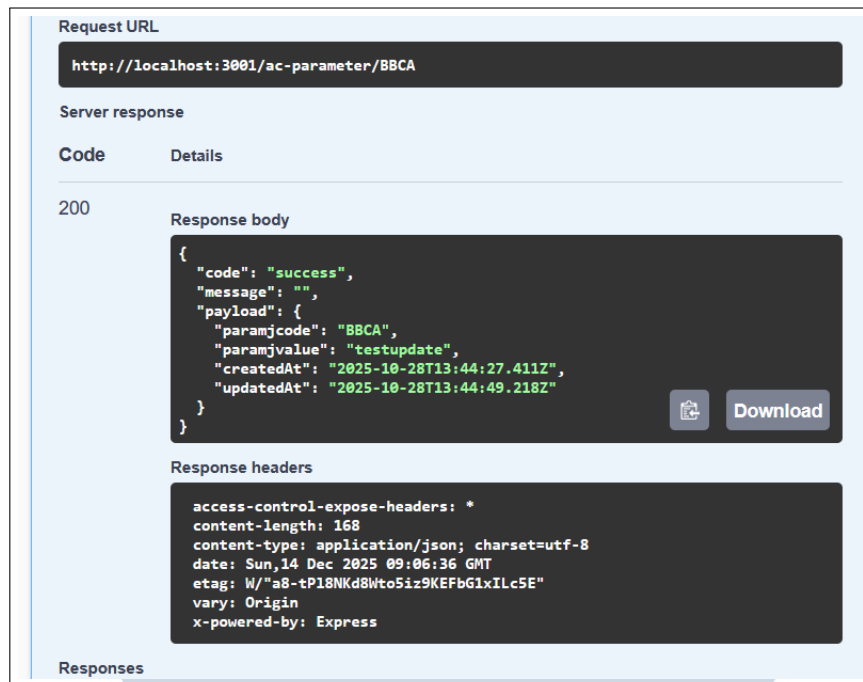
Gambar 3.2. *Response endpoint* PATCH `ac-parameter/[id]`

Gambar 3.3 menunjukkan hasil pemanggilan *endpoint* `ac-parameter` dengan metode GET untuk mengambil seluruh data parameter yang tersedia. Respons yang dikembalikan oleh server memiliki kode status 200, yang menandakan permintaan berhasil diproses. Data ditampilkan dalam bentuk daftar (*list*) parameter yang mencakup kode parameter, nilai parameter, serta informasi waktu pembuatan dan pembaruan. *Endpoint* ini mendukung pengambilan data secara kolektif dan digunakan untuk menampilkan daftar parameter pada sistem.



Gambar 3.3. *Response endpoint GET ac-parameter*

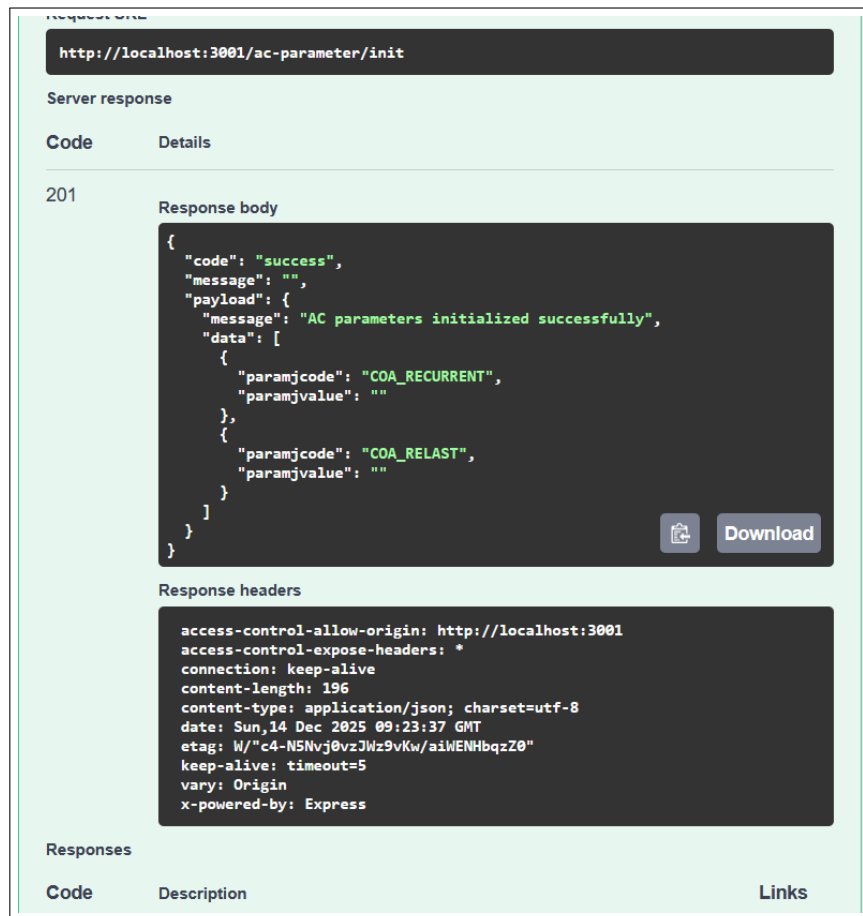
Gambar 3.4 merupakan contoh hasil pemanggilan *endpoint* ac-parameter/id menggunakan metode GET. *Endpoint* ini berfungsi untuk mengambil detail data parameter tertentu berdasarkan id yang diberikan. Respons server menunjukkan kode status 200, yang menandakan data berhasil ditemukan dan dikembalikan. Informasi yang ditampilkan meliputi kode parameter, nilai parameter, serta metadata waktu pembuatan dan pembaruan data. *Endpoint* ini digunakan untuk kebutuhan pengambilan data parameter secara spesifik.



Gambar 3.4. *Response endpoint* GET ac-parameter/[id]

Gambar 3.5 merupakan contoh hasil pemanggilan *endpoint* ac-parameter/init menggunakan metode POST. *Endpoint* ini berfungsi untuk melakukan inisialisasi parameter pada sistem. Respons server menunjukkan kode status 201, yang menandakan bahwa proses inisialisasi berhasil dilakukan dan data baru telah dibuat. Informasi yang ditampilkan dalam *response body* meliputi status keberhasilan, pesan konfirmasi "AC parameters initialized successfully", serta daftar data parameter yang mencakup paramjcode dan paramjvalue. *Endpoint* ini digunakan untuk kebutuhan penyiapan data parameter awal dalam aplikasi.

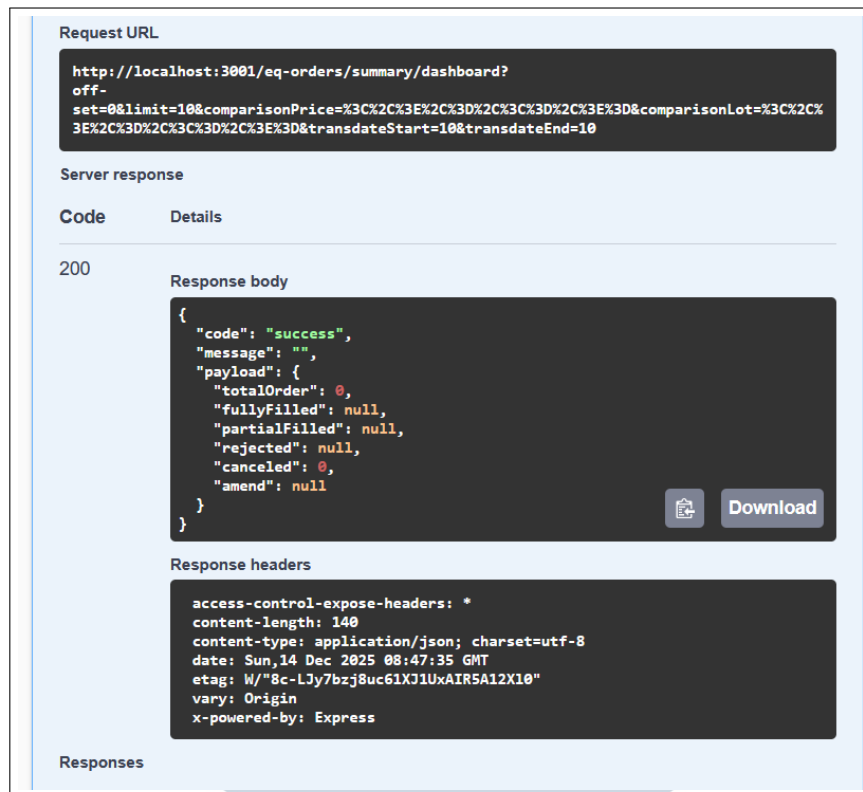
UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.5. Response endpoint POST ac-parameter/init

- Menambahkan endpoint summary/dashboard pada modul Equity Order untuk menyediakan data berupa jumlah *total order* (hari ini), *Fully Filled*, *Partial Filled*, *Rejected*, *Canceled*, dan *Amend*.

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

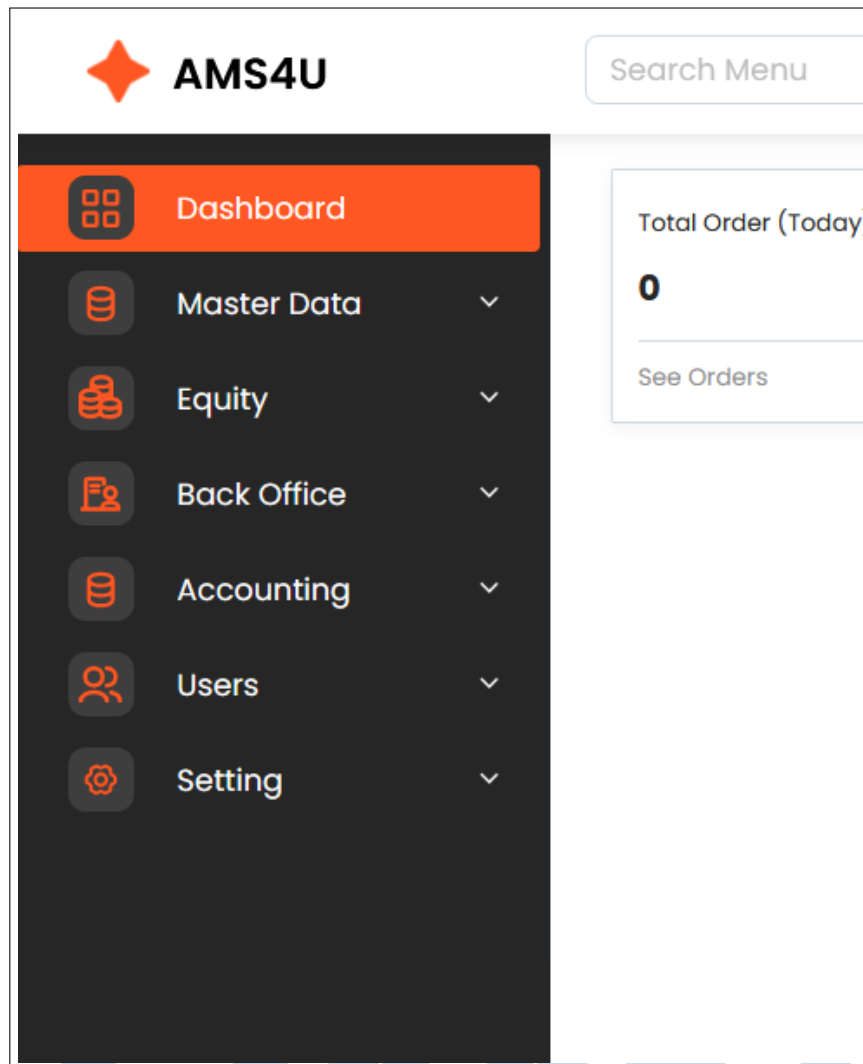


Gambar 3.6. *Response endpoint GET eq-orders/summary/dashboard*

B Antar Muka

Guna meningkatkan fungsionalitas dan pengalaman pengguna, dilakukan serangkaian optimalisasi pada elemen visual sistem. Berikut adalah detail perbaikan antar muka yang telah diimplementasikan:

- (a) Memperbaiki berbagai permasalahan pada *sidebar*, meliputi :
 - i. *Sidebar* tidak dapat mengingat menu yang terakhir dibuka ketika browser dimuat ulang.
 - ii. *Sidebar* tidak tersinkronisasi dengan izin akses (role-based access) dari akun pengguna.
 - iii. Ketidaksesuaian tampilan warna latar belakang submenu *popup* dengan warna *sidebar* utama.

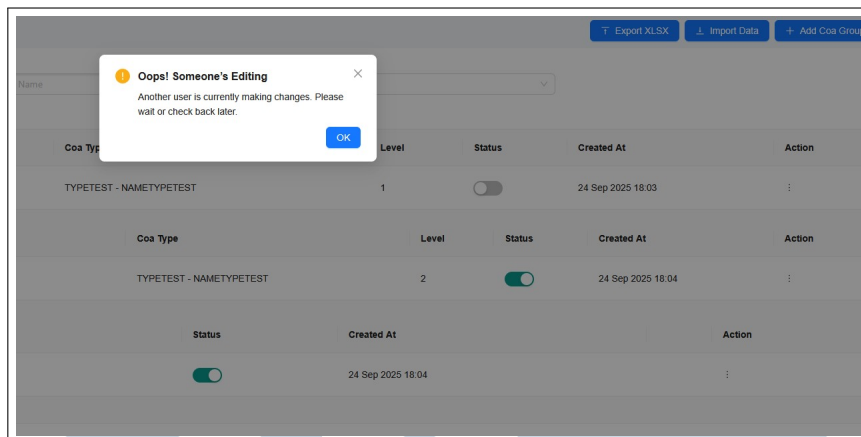


Gambar 3.7. Tampilan *sidebar* AMS4U

(b) Melakukan perbaikan pada modul *Chart of Account* (COA) yang mengalami kendala saat proses pengeditan data. Permasalahan yang diperbaiki meliputi:

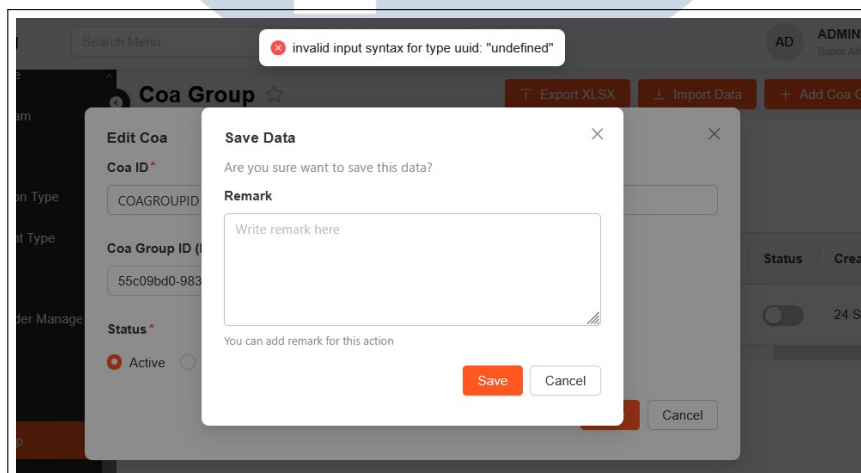
- i. Kondisi ketika pengguna pertama kali melakukan pengeditan lalu membatalkannya, yang menyebabkan COA terkunci selama beberapa menit yang membuat pengalaman *user* menjadi tidak nyaman.

Berikut pada gambar 3.8 merupakan tampilan halaman COA yang menampilkan pesan *error* dalam bentuk modal.



Gambar 3.8. Tampilan halaman COA *error* saat mengedit

- ii. munculnya *error userId undefined* saat menyimpan perubahan data. Berikut pada gambar 3.9 merupakan komponen modal detail yang berisi informasi COA dalam kondisi *user* akan menyimpan data, namun ada *error* "invalid input syntax for type uuid: 'undefined'".



Gambar 3.9. Tampilan halaman COA *error* menyimpan perubahan

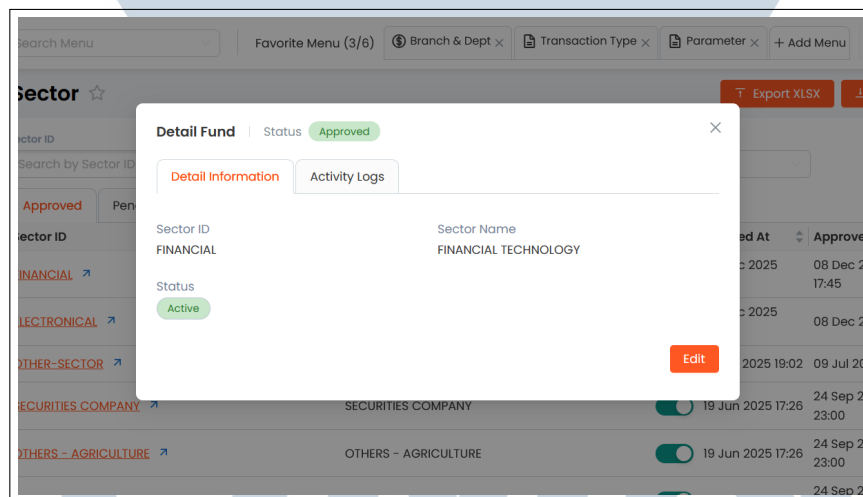
C Implementasi Fitur Sistem

Implementasi fitur sistem ini dilakukan secara menyeluruh, mencakup pembaruan pada sisi antarmuka untuk interaksi pengguna serta penyediaan logika di sisi *backend*. Sinergi antara elemen visual dan pemrosesan data di latar belakang memastikan bahwa setiap aksi yang dilakukan pengguna melalui antarmuka dapat diproses secara valid oleh server. Berikut adalah

detail implementasi yang mencakup tampilan antarmuka dan mekanisme *endpoint* API yang digunakan.

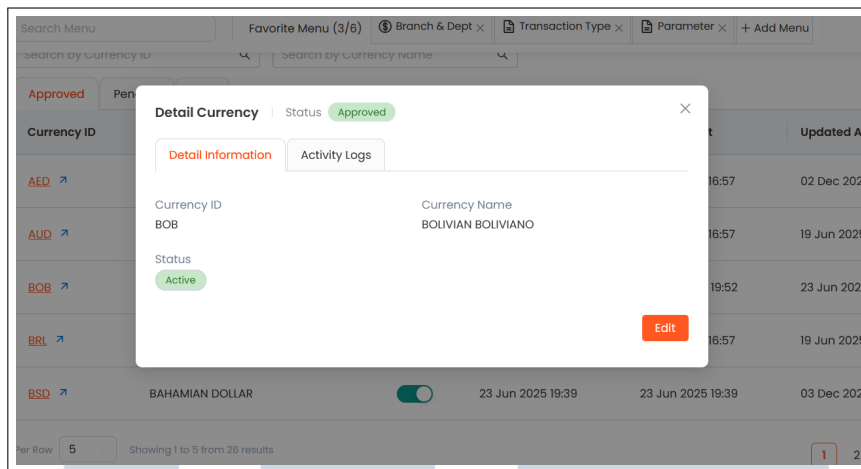
D Antar Muka

- (a) Menambahkan tombol Edit pada komponen modal Detail yang dapat dikonfigurasi menggunakan fungsi khusus (*custom function*). Dengan adanya tombol ini, alur pengeditan data menjadi lebih fleksibel karena dapat disesuaikan dengan kebutuhan masing-masing modul tanpa harus membuat komponen baru. Fitur ini meningkatkan kemudahan pengguna dalam mengakses proses pengeditan langsung dari tampilan detail data. Gambar 3.10 merupakan tampilan modal saat dipakai untuk menampilkan informasi dari data *fund*.



Gambar 3.10. Tampilan komponen modal detail ketika dipakai untuk melihat data *fund*

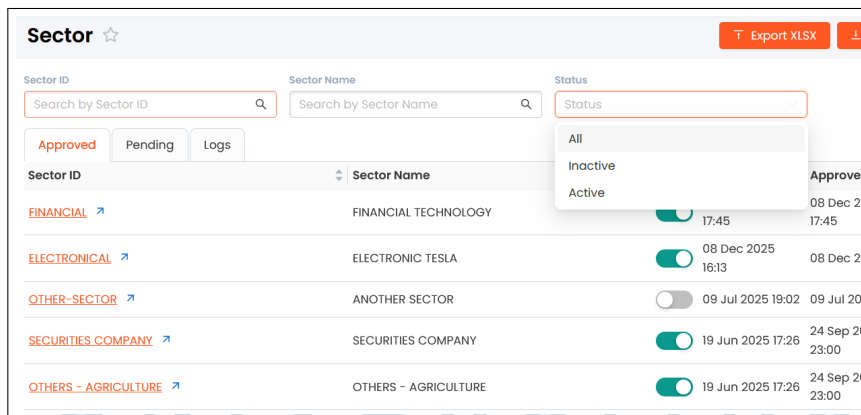
Gambar 3.11 merupakan tampilan modal saat dipakai untuk menampilkan informasi dari data *currency*.



Gambar 3.11. Tampilan komponen modal detail ketika dipakai untuk melihat data *Currency*

- (b) Menambahkan fitur filter berdasarkan status pada halaman *Sector*, yang mencakup opsi *Active*, *Inactive*, dan *All*. Fitur ini memungkinkan pengguna untuk melakukan penyaringan data secara lebih spesifik sesuai dengan status yang diinginkan, sehingga mempermudah proses pencarian dan pengelolaan data sektor dalam sistem.

Gambar 3.12 merupakan halaman *Sector* yang menampilkan daftar *sector* yang ada.

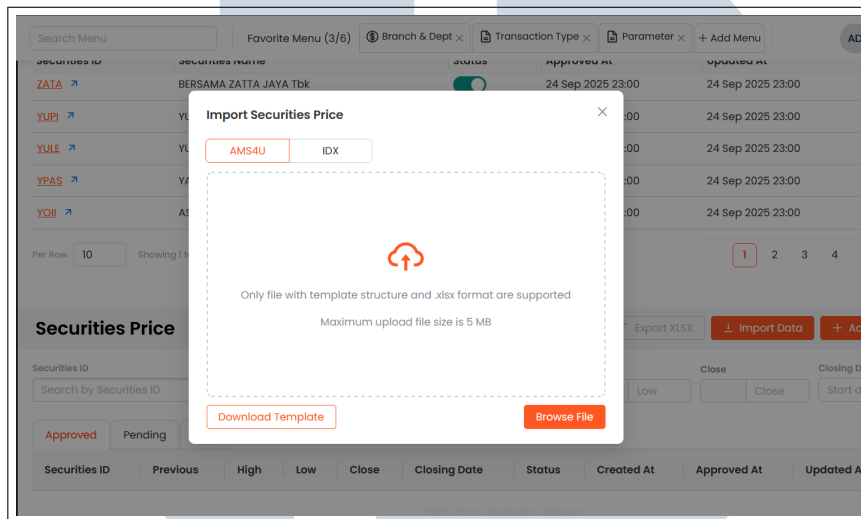


Gambar 3.12. Tampilan halaman *sector*

- (c) Mengembangkan komponen *ImportData* agar mendukung proses impor *file* *xlsx* dengan struktur kolom yang bervariasi. Pengembangan ini mencakup penyesuaian perilaku tombol, judul, serta logika pengolahan tanggal untuk setiap jenis impor. Selain itu, ditambahkan mekanisme validasi untuk mendeteksi keberadaan data *Key header* pada *file* sebelum

proses impor dijalankan, sehingga dapat mencegah kesalahan format dan meningkatkan keakuratan data yang diimpor ke dalam sistem.

Gambar 3.13 merupakan tampilan komponen ImportData yang berbentuk modal.



Gambar 3.13. Tampilan komponen ImportData

Gambar 3.14 merupakan tampilan komponen ImportData yang menampilkan *preview* dari hasil import data dengan format AMS4U.



Securities ID	Previous	High	Low	Close	Closing Date
WTON	97	98	97	97	12 December 2025
YELO	150	Not Set	Not Set	150	12 December 2025
YOII	95	99	95	96	12 December 2025
YPAS	735	775	710	710	12 December 2025
YULE	2700	Not Set	Not Set	2700	12 December 2025
YUPI	1605	1620	1575	1610	12 December 2025
ZATA	67	67	62	64	12 December 2025
ZBRA	50	Not Set	Not Set	50	12 December 2025

Gambar 3.14. Tampilan komponen ImportData *preview*

- (d) Menambahkan fitur ringkasan status order pada halaman Dashboard yang menampilkan informasi jumlah *Total Order* (hari ini), *Fully Filled*, *Partial Filled*, *Rejected*, *Canceled*, dan *Amend*. Fitur ini bertujuan untuk memberikan gambaran cepat mengenai aktivitas transaksi yang sedang berlangsung, serta membantu pengguna dalam memantau performa sistem secara keseluruhan melalui tampilan *dashboard* yang informatif. Gambar 3.15 merupakan tampilan halaman *dashboard* yang menampilkan informasi ringkas yang berhubungan dengan eq-Order.

Status	Count	Action
Total Order (Today)	0	See Orders
Fully Filled	0	See Orders
Partial Filled	0	See Orders
Rejected	0	See Orders
Canceled	0	See Orders
Amend	0	See Orders

Gambar 3.15. Tampilan halaman *dashboard*

3.3.4 Kendala dan Solusi yang Ditemukan

Selama pelaksanaan kegiatan magang, terdapat beberapa kendala yang dihadapi. Berikut ini adalah beberapa permasalahan yang ditemukan beserta solusi yang dilakukan untuk mengatasinya:

1. Salah satu kendala yang ditemukan selama pelaksanaan kerja magang adalah konfigurasi *environment* pengembangan yang belum sepenuhnya lengkap dan seragam. Perbedaan konfigurasi tersebut menyebabkan beberapa fitur tidak dapat berjalan optimal pada tahap awal pengembangan.

Solusi: yang dilakukan adalah dengan mengadakan diskusi, *meeting*, dan konsultasi secara lebih intensif bersama supervisor dan tim pengembang untuk menyesuaikan serta menyamakan konfigurasi *environment* yang digunakan, sehingga proses pengembangan dapat berjalan dengan lebih lancar.

