

BAB 3

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Kedudukan saya selama kegiatan magang di PT. Indobest Artha Kreasi adalah sebagai *Intern Web Developer* di Divisi Retail. Divisi Retail memiliki dua anggota yang dipimpin oleh Bapak Shendy Harlim sebagai *Product Owner* (PO). Dalam pelaksanaan kerja magang, terutama pada Divisi Retail, koordinasi antar anggota dapat dilakukan secara langsung, melalui aplikasi Telegram, atau Basecamp. Komunikasi secara langsung dan melalui Telegram digunakan untuk membahas topik yang bersifat mendesak dan memerlukan respons cepat. Namun, untuk koordinasi yang berhubungan dengan manajemen proyek, seperti pemberian tugas, *requirement*, jadwal, dan informasi penting lainnya, dilakukan menggunakan Basecamp.

Selama pekerjaan berlangsung, kontrol versi dikelola menggunakan sistem *branch* di GitHub. Pengelolaan repositori dilakukan menggunakan Git melalui CLI (*Command Line Interface*). Selain itu, Divisi Retail mengadakan rapat mingguan untuk berkoordinasi dengan semua anggota tim. Rapat dilakukan dua kali dalam seminggu, yaitu pada hari Senin dan Rabu. Rapat pada hari Senin membahas tugas selama minggu sebelumnya dan rencana tugas untuk minggu berjalan. Sedangkan rapat pada hari Rabu membahas kendala yang dihadapi selama pengerjaan tugas serta *sharing knowledge* yang berkaitan dengan dunia teknologi.

3.2 Tugas yang Dilakukan

Tugas yang dikerjakan selama proyek magang berlangsung meliputi perbaikan *bug* dan pengembangan fitur. Salah satu fitur yang dikerjakan adalah fitur *top-up* deposit *tenant* melalui integrasi dengan Midtrans. Pekerjaan mencakup pembuatan form pembayaran pada sisi *frontend*, penanganan notifikasi pembayaran melalui *callback* pada sisi *backend*, serta implementasi *job* dan *scheduler* otomatis untuk memeriksa dan memperbarui status transaksi secara berkala. Fitur ini memungkinkan saldo *tenant* terisi secara otomatis dan *real-time*.

Fitur kedua adalah penyesuaian sistem harga produk One Page Checkout (OPC) dengan menambahkan pengaturan *markup* harga di panel admin. Harga akan otomatis disesuaikan dan ditampilkan ke pengguna sesuai nilai *markup* yang

ditentukan *tenant*. Pengembangan mencakup penyesuaian di sisi *backend* dan *frontend*, serta pengujian untuk memastikan keakuratan perhitungan.

3.3 Uraian Pelaksanaan Magang

Proses *internship* di PT Indobest Artha Kreasi berlangsung selama 640 jam. Pelaksanaan kerja magang terurai pada Tabel 3.1.

Tabel 3.1. Tugas yang dilakukan setiap minggu selama magang

Minggu Ke-	Pekerjaan yang dilakukan
1	Proses <i>onboarding</i> , instalasi dan konfigurasi proyek Whitelabel agar dapat berjalan secara lokal. Mempelajari struktur proyek whitelabel, memahami alur kerja fitur-fitur utama, serta melakukan perbaikan pada panel admin.
2	Menghapus penggunaan <i>feature flag</i> Growthbook dari seluruh repositori karena ketergantungan pada pihak ketiga yang berpotensi memengaruhi stabilitas dan fleksibilitas proyek Whitelabel. Setelah penghapusan, melakukan pengujian fungsionalitas untuk memastikan tidak ada gangguan pada sistem utama.
3	Melakukan <i>setup</i> awal proyek CMS Whitelabel, serta memulai pengembangan fitur manajemen <i>feature flag tenant</i> untuk keperluan <i>testing</i> fitur-fitur baru.
4	Melakukan perbaikan <i>bug</i> ketika mengecek status pembayaran dan mengembangkan fitur konfigurasi produk pinjaman yang diaktifkan per pengguna.
5	Mengembangkan fitur <i>soft maintenance</i> untuk mencegah transaksi dilakukan oleh <i>tenant</i> yang belum menandatangani Perjanjian Kerja Sama (PKS).
6	Mengembangkan fitur <i>set pending</i> pada CMS Whitelabel untuk memungkinkan admin Whitelabel untuk menandai transaksi prabayar tertentu sebagai <i>pending</i> .
Lanjut pada halaman berikutnya	

Tabel 3.1 Tugas yang dilakukan setiap minggu selama magang (lanjutan)

Minggu Ke-	Pekerjaan yang dilakukan
7	Menampilkan detail informasi <i>affiliator</i> pada <i>dashboard tenant</i> , memperbaiki tampilan struk transaksi PLN dan PDAM, serta menambahkan <i>tooltip</i> saat <i>loading</i> respons dari supplier di halaman daftar harga.
8	Membuat fitur <i>export</i> daftar <i>user</i> ke format xls di panel admin, serta menambahkan fitur <i>sorting</i> produk pada halaman agen produk Kirim Uang.
9	Mengembangkan fitur pengelolaan akses produk <i>E-Money</i> untuk masing-masing <i>user</i> , termasuk membuat fitur menu pengaturan akses di panel admin.
10	Membuat unit test dan dokumentasi fitur pengelolaan akses produk <i>E-Money</i> .
11	Mengembangkan fitur promosi produk Kirim Uang agar dapat ditampilkan kepada semua <i>user</i> , termasuk yang belum terverifikasi, namun dengan validasi tambahan yang memastikan hanya <i>user</i> terverifikasi yang dapat mengakses produk dan melakukan transaksi.
12	Mengembangkan fitur <i>Cutoff Top Up Deposit</i> , termasuk logika <i>backend</i> dan tampilan halaman <i>user</i> agar menampilkan pesan jika melakukan <i>top up</i> di luar waktu yang ditentukan.
13	Membuat kategori baru untuk produk Aktivasi Voucher, baik di panel admin maupun halaman <i>user</i> , termasuk pemisahan logika dan tampilan antarmuka.
14	Memperbaiki <i>bug</i> pada fitur One Page Checkout yang memungkinkan terjadinya intersepsi harga, sehingga transaksi di halaman tersebut dapat dimanipulasi dan berpotensi menyebabkan kerugian finansial.
15	Mengembangkan fitur <i>top-up</i> deposit <i>tenant</i> menggunakan <i>payment gateway</i> Midtrans, mencakup pembuatan formulir pembayaran di sisi <i>frontend</i> , penanganan <i>callback</i> di <i>backend</i> , serta implementasi <i>job scheduler</i> untuk pengecekan dan pembaruan status pembayaran secara otomatis.
Lanjut pada halaman berikutnya	

Tabel 3.1 Tugas yang dilakukan setiap minggu selama magang (lanjutan)

Minggu Ke-	Pekerjaan yang dilakukan
16	Melakukan <i>testing</i> fitur di local dan memperbaiki detail fitur berdasarkan <i>feedback</i> dari Divisi QA.
17	Membuat dokumentasi <i>flow</i> fitur <i>top-up</i> deposit <i>tenant</i> yang baru dan mempelajari kode untuk persiapan proyek di <i>cycle</i> berikutnya.
18	Mengembangkan fitur konfigurasi biaya One Page Checkout pada <i>tenant</i> dengan opsi pembebanan ke <i>user</i> . Melakukan pengujian dan integrasi untuk memastikan fitur berjalan sesuai ekspektasi.
19	Membuat fitur manajemen harga <i>markup</i> One Page Checkout di panel admin dan mengatur tampilan daftar harga pada halaman <i>user</i> .
20	Memperbaiki dan menyempurnakan fitur berdasarkan <i>feedback</i> dari tim QA, serta membuat dokumentasi teknis dan pengguna untuk mendukung penggunaan fitur.

3.4 Perangkat Penunjang

Selama melaksanakan praktik magang, terdapat berbagai *software* dan *hardware* yang digunakan. Berikut ini adalah beberapa *software* yang digunakan dalam menyelesaikan praktik magang.

- a. Visual Studio Code v1.100.2
- b. PHP Zend Opache v7.4.33
- c. Laravel v5
- d. Composer v1.10.27
- e. Vue.js v2.6.12
- f. Node v12.22.12
- g. NPM v6.14.16
- h. NVM v0.39.1

- i. Yarn v1.22.22
- j. Redis v6.0.16
- k. Openresty v1.27.1.1
- l. MongoDB Compass v1.46.2
- m. HeidiSQL v12.8.0.6908
- n. MySQL v8.0.42
- o. Git v2.34.1
- p. Postman v11.47.2
- q. OpenVPN Connect v3.4.4
- r. Google Chrome v136.0.7103.114
- s. Basecamp 3 v2.3.14
- t. Kaspersky Endpoint Security For Windows v12.8.0.505
- u. Windows Subsystem for Linux v2.5.7.0
- v. Windows 11 Home Single Language v23H2
- w. Ubuntu 22.04.5 LTS

Berikut ini adalah spesifikasi perangkat keras yang digunakan.

- a. Asus Vivobook M1403QA
- b. AMD Ryzen 7 5800H with Radeon Graphics
- c. RAM 16 GB
- d. SSD 512 GB

3.5 Metodologi Pengembangan Perangkat Lunak

Selama pelaksanaan magang, kegiatan pengembangan perangkat lunak dilakukan menggunakan metodologi *Agile*, khususnya pendekatan *Scrum*. Metodologi ini dipilih karena mendukung pengembangan sistem secara iteratif dan adaptif terhadap perubahan kebutuhan. Saya tergabung dalam tim *Small Batch*, yang berfokus pada pengembangan fitur-fitur kecil serta perbaikan *bug* dengan durasi pengerjaan setiap proyek berkisar antara dua hingga tiga minggu. Dalam setiap siklus pengerjaan, dilakukan proses perencanaan tugas di awal minggu, pelaksanaan pengembangan, pengujian mandiri, serta evaluasi dan perbaikan berdasarkan masukan dari *Product Owner* dan tim QA.

Koordinasi dilakukan melalui rapat rutin setiap hari Senin dan Rabu. Rapat hari Senin berfungsi sebagai *sprint planning*, di mana tim menyusun rencana kerja untuk minggu berjalan, sedangkan rapat hari Rabu digunakan sebagai forum *sprint review* untuk membahas progres, kendala yang dihadapi, dan berbagi pengetahuan teknis. Seluruh tugas didistribusikan dan dipantau menggunakan platform manajemen proyek Basecamp, sedangkan komunikasi harian dilakukan melalui Telegram atau diskusi langsung di kantor. Pengembangan dilakukan secara kolaboratif menggunakan Git, dengan pendekatan *branching* untuk setiap fitur, dilanjutkan proses *pull request* dan *code review* sebelum penggabungan ke repositori utama. Dengan metode ini, pengembangan menjadi lebih terstruktur dan efisien, serta memungkinkan adanya umpan balik berkelanjutan dalam setiap siklus kerja.

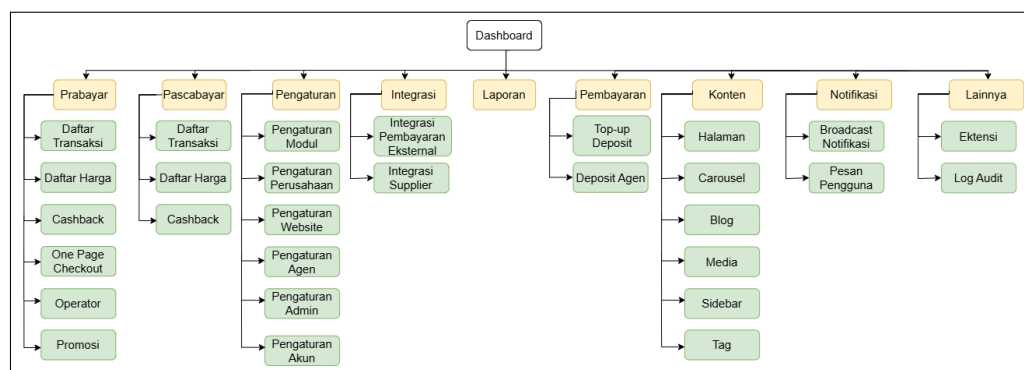
3.6 Proses Pelaksanaan Magang

Terdapat dua proyek utama yang dikembangkan selama masa magang, yang berfokus pada peningkatan efisiensi dan fleksibilitas operasional *tenant* dalam platform Whitelabel. Akses terhadap kedua fitur dilakukan melalui panel admin yang dirancang dengan struktur navigasi sidebar yang sistematis. Setelah berhasil melakukan proses *login*, *tenant* akan diarahkan ke halaman utama “Dashboard” sebagai titik awal pengelolaan sistem.

Navigasi pada panel admin disusun dalam sidebar yang terbagi ke dalam beberapa kategori utama, seperti “Prabayar”, “Pascabayar”, “Pengaturan”, “Integrasi”, “Laporan”, dan “Pembayaran”. Fitur *top-up* deposit tersedia pada menu “Top-Up Deposit” di bawah kategori “Pembayaran”, yang berfokus pada

pengelolaan transaksi keuangan. Sementara itu, fitur pengaturan harga produk dengan skema *markup* dapat diakses melalui submenu “One Page Checkout” (OPC) dalam kategori “Prabayar”, yang berkaitan langsung dengan manajemen harga produk.

Dengan struktur navigasi yang terorganisasi berdasarkan fungsi, akses ke berbagai fitur menjadi lebih intuitif dan efisien. Gambar 3.1 menyajikan *sitemap* yang menggambarkan hierarki menu dan keterhubungan antar fitur dalam panel admin Whitelabel.



Gambar 3.1. *Sitemap* panel admin sistem Whitelabel

Fitur pertama yang dikembangkan adalah sistem *top-up* deposit yang terintegrasi dengan *payment gateway* Midtrans. Fitur ini memungkinkan *tenant* melakukan pengisian saldo secara mandiri melalui mekanisme pembayaran digital yang aman, efisien, dan terotomatisasi.

Fitur kedua adalah pengaturan harga produk pada halaman One Page Checkout (OPC) melalui penerapan nilai *markup*. Pengaturan ini memberikan fleksibilitas dalam menentukan margin keuntungan untuk setiap produk prabayar yang ditawarkan kepada pengguna akhir.

Penjelasan pada bagian berikut mencakup kebutuhan sistem, perancangan alur logika, serta implementasi teknis dari masing-masing fitur tersebut.

3.6.1 Fitur *Top-Up* Deposit *Tenant*

Fitur *top-up* deposit *tenant* merupakan sebuah mekanisme yang memungkinkan *tenant* untuk melakukan pengisian saldo deposit yang digunakan dalam sistem Whitelabel. Fitur ini tersedia melalui panel admin dan terintegrasi dengan *payment gateway* Midtrans untuk memfasilitasi proses pembayaran secara digital menggunakan metode seperti QRIS atau *Virtual Account* (VA). Tujuan dari

fitur ini adalah untuk menyediakan proses pengisian saldo yang efisien, aman, dan terotomatisasi, serta menghindari terjadinya transaksi ganda atau konflik data selama proses berlangsung. Selain memberikan kenyamanan dalam bertransaksi, fitur ini juga berperan penting dalam menjaga integritas data transaksi serta mendukung fleksibilitas *tenant* dalam memilih metode pembayaran yang sesuai kebutuhan.

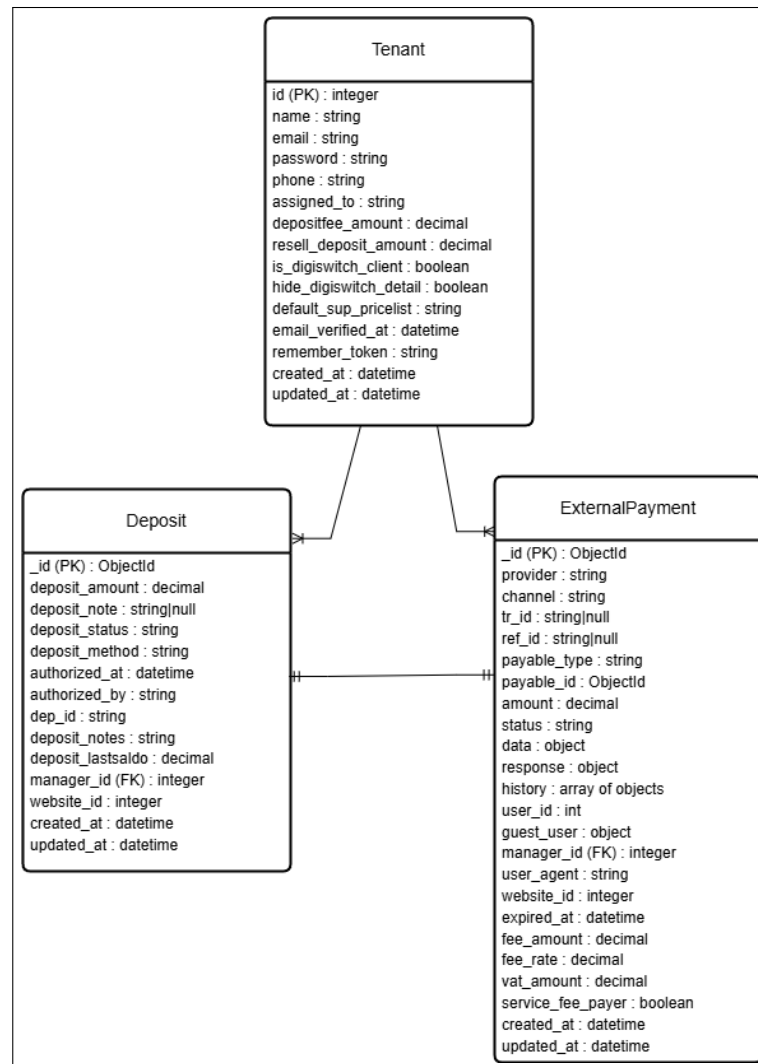
Dalam pengerjaan fitur ini, terdapat beberapa kebutuhan sistem untuk memastikan fungsionalitas berjalan optimal. Proses *top-up* diawali dengan pengisian formulir oleh *tenant* yang mencakup nominal deposit dan pilihan metode pembayaran. Metode yang tersedia adalah *Virtual Account* (VA) dan QRIS yang diakses melalui *payment gateway* Midtrans. Sistem juga menjalankan proses *cron* (penjadwalan tugas otomatis) setiap 10 menit untuk memantau transaksi yang masih berstatus *pending*. Transaksi-transaksi ini akan diproses satu per satu melalui pemanggilan tugas (*job*), yaitu proses otomatis yang dijalankan oleh sistem di latar belakang untuk memeriksa status pembayaran secara otomatis.

Job pengecekan status memiliki peran penting dalam memastikan data pembayaran tetap akurat dan terkini. *Job* tersebut akan mencocokkan data transaksi dalam basis data dengan status terbaru dari Midtrans. Jika terjadi perubahan status seperti pembayaran sukses atau gagal, sistem akan memperbarui informasi tersebut secara otomatis. Mekanisme ini menjaga integritas data dan mengurangi risiko kesalahan akibat intervensi manual.

Fitur *top-up* deposit *tenant* dirancang agar *tenant* dapat mengisi saldo secara mandiri melalui panel admin Whitelabel. Proses ini melibatkan beberapa tahap mulai dari pengecekan transaksi yang belum selesai, pengisian formulir, hingga integrasi dengan Midtrans untuk memverifikasi pembayaran.

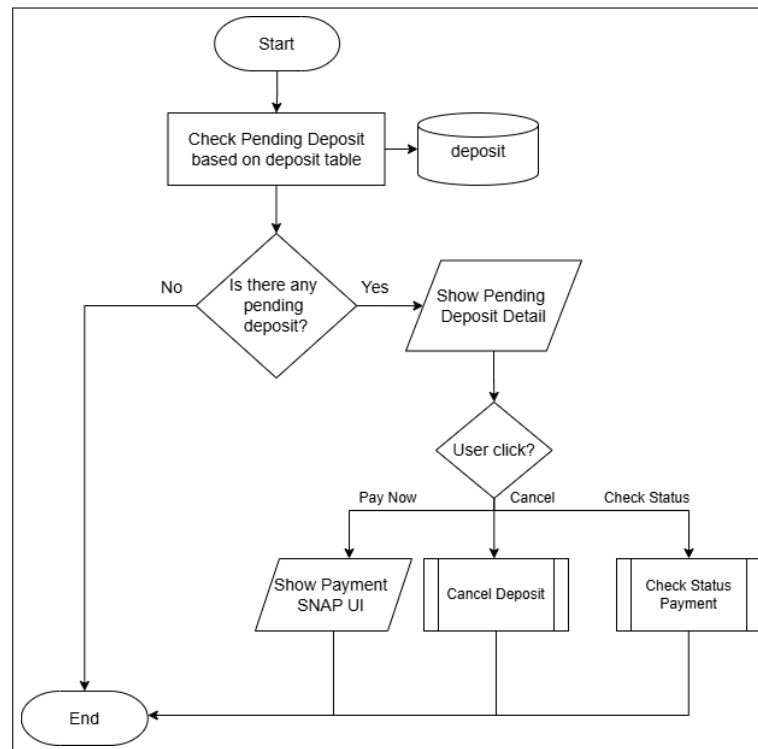
Struktur data yang mendasari fitur ini melibatkan kombinasi basis data relasional dan non-relasional. Dalam hal ini, entitas utama yang berperan adalah tabel *Tenant* yang disimpan dalam basis data SQL, serta tabel *ExternalPayment* dan tabel *Deposit* yang tersimpan di MongoDB. Data tabel *Deposit* mencatat setiap transaksi pengisian saldo yang dilakukan oleh *tenant*, termasuk nominal, status, metode pembayaran, dan referensi ke pengguna terkait. Sementara itu, tabel *ExternalPayment* berfungsi sebagai pencatat detail transaksi yang difasilitasi oleh Midtrans. Relasi antara tabel *Deposit* dan tabel *ExternalPayment* bersifat polimorfik, di mana atribut *payable_type* dan *payable_id* memungkinkan *ExternalPayment* untuk terhubung ke berbagai jenis transaksi, termasuk pengisian deposit. Dengan demikian, integrasi antara sistem pembayaran eksternal dan data

internal dapat dilakukan secara fleksibel. Gambar 3.2 berikut menyajikan diagram *physical ERD* yang menggambarkan hubungan antar entitas dalam fitur ini.



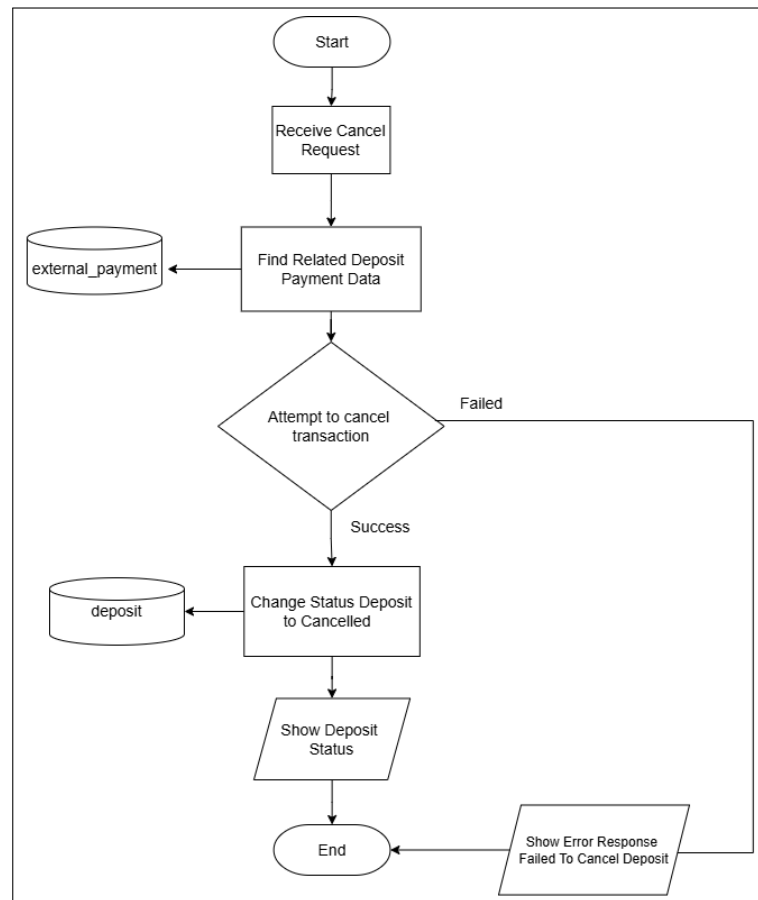
Gambar 3.2. Diagram *physical ERD* untuk fitur *top-up* deposit *tenant*

Proses dimulai dengan sistem yang secara otomatis memeriksa apakah terdapat transaksi *top-up* yang masih berstatus *pending*. Jika tidak ditemukan, *tenant* dapat langsung melanjutkan ke pengisian formulir. Jika terdapat transaksi yang belum selesai, sistem akan menampilkan detail transaksi tersebut dan memberikan opsi kepada *tenant* untuk melanjutkan pembayaran, membatalkan transaksi, atau memeriksa status terkini. Gambar 3.3 menggambarkan alur pengecekan transaksi yang masih tertunda.



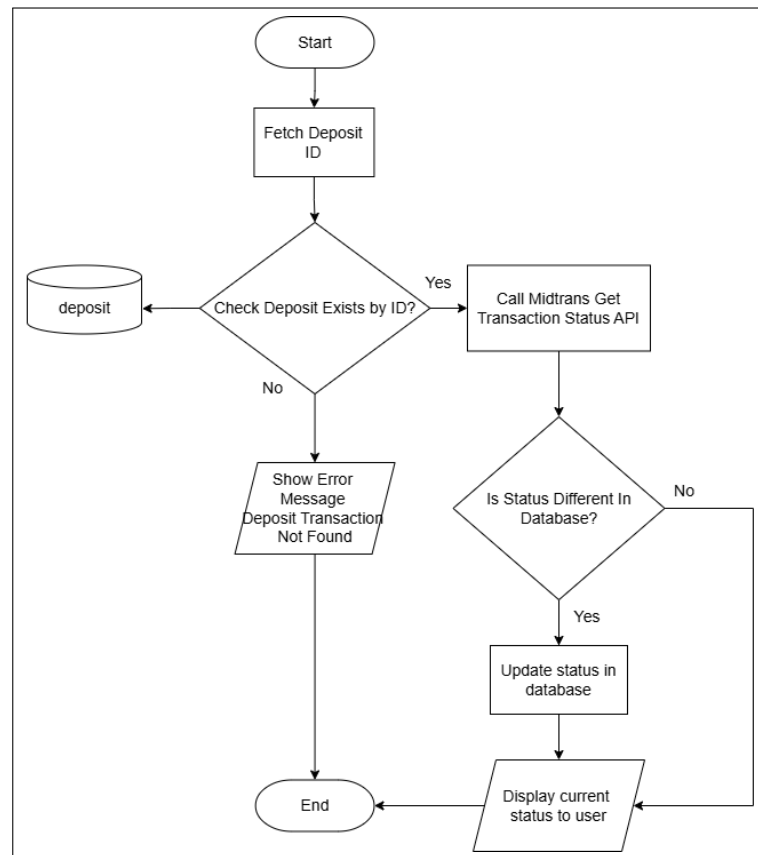
Gambar 3.3. Diagram alur pengecekan deposit *pending*

Jika *tenant* memilih untuk membatalkan transaksi, sistem akan mengirimkan permintaan pembatalan ke *payment gateway* Midtrans dan mengubah status transaksi di basis data menjadi *cancelled*, seperti ditunjukkan pada Gambar 3.4.



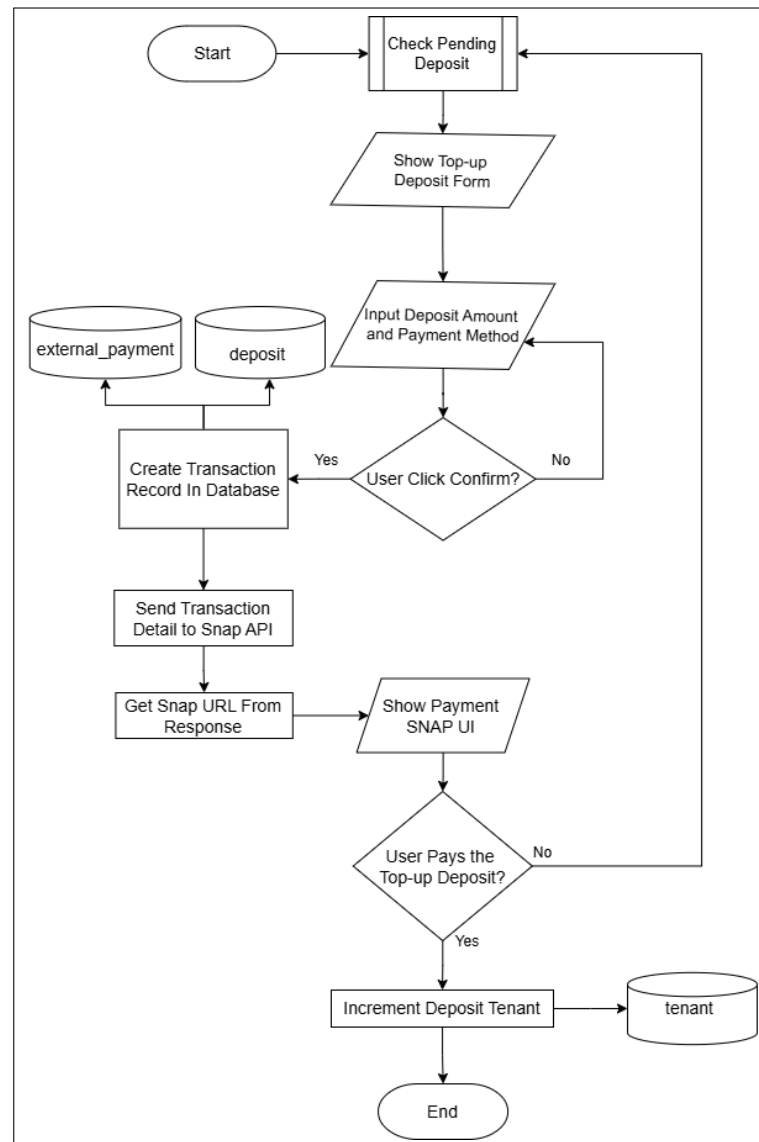
Gambar 3.4. Diagram pembatalan deposit

Jika *tenant* memilih untuk memeriksa status, maka sistem akan mengakses status terbaru dari *payment gateway* Midtrans dan memperbaruinya seperti dalam Gambar 3.5.



Gambar 3.5. Diagram pengecekan status deposit

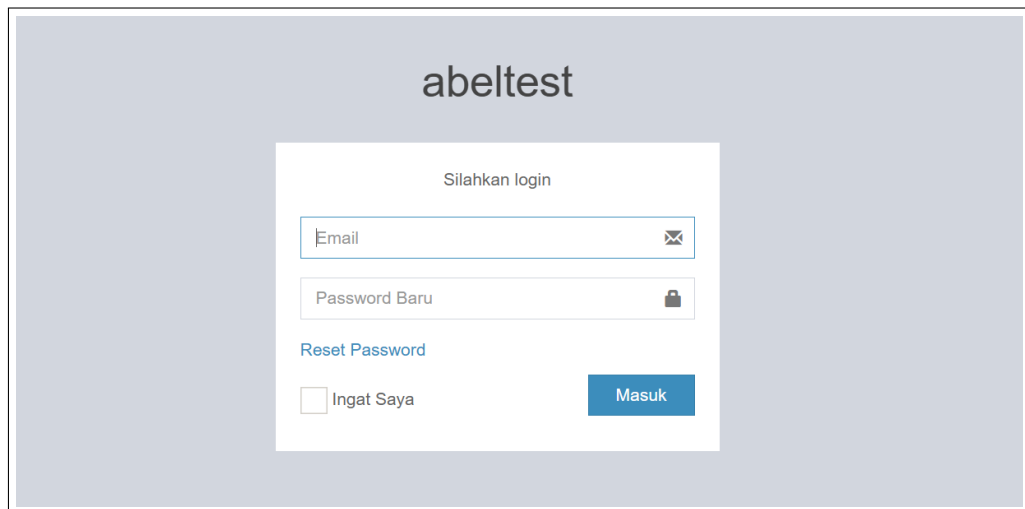
Jika tidak ada transaksi yang tertunda, *tenant* akan diarahkan untuk mengisi formulir *top-up*, sebagaimana ditunjukkan pada Gambar 3.6. *Tenant* diminta memasukkan nominal dan memilih metode pembayaran, kemudian sistem akan mencatat permintaan ini dalam basis data dan mengirimkan data transaksi ke *payment gateway* Midtrans menggunakan Snap API.



Gambar 3.6. Diagram alur *top-up* deposit *tenant*

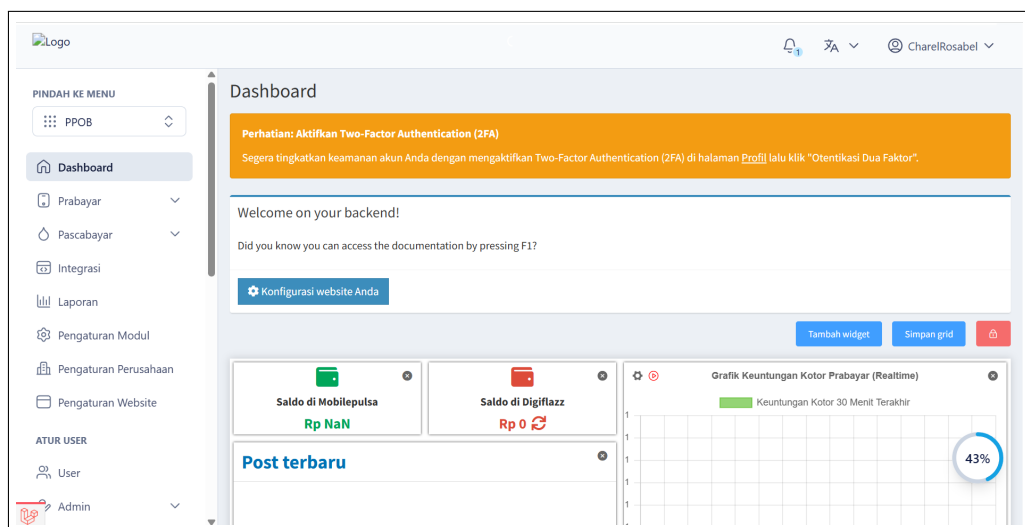
Midtrans akan membalas dengan tautan Snap yang ditampilkan kepada *tenant* agar dapat menyelesaikan pembayaran. Jika pembayaran berhasil, sistem akan menerima notifikasi melalui *webhook* dan memperbarui status transaksi. Proses ini memastikan transaksi selalu tercatat secara *real-time* dan aman. Sistem juga tetap menjalankan pengecekan status secara otomatis melalui *cron job* untuk memastikan transaksi yang tidak segera dibayar tetap terpantau.

Tampilan implementasi fitur ini dapat dilihat melalui beberapa antarmuka pada panel admin. Gambar 3.7 menunjukkan tampilan login, sementara Gambar 3.8 memperlihatkan halaman utama setelah login. Dari halaman ini, *tenant* dapat mengakses fitur *top-up* deposit seperti tampak pada Gambar 3.9.



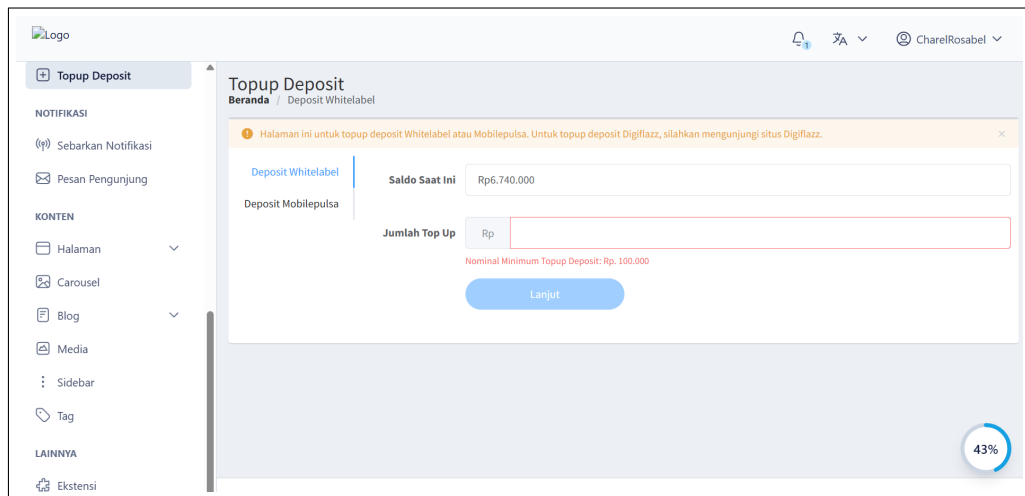
Gambar 3.7. Halaman *login* panel admin

Gambar 3.7 menunjukkan tampilan awal saat *tenant* mengakses panel admin. Untuk dapat mengakses seluruh fitur yang tersedia, *tenant* wajib melakukan proses *login*. Jika belum memiliki akun, pengguna harus mendaftar terlebih dahulu. *Tenant* yang tidak memiliki akun tidak dapat melanjutkan ke panel admin.



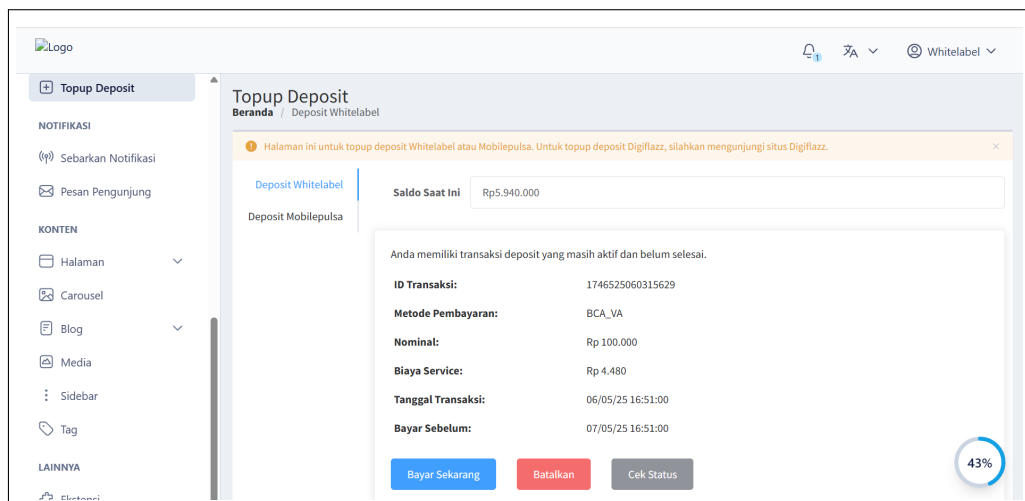
Gambar 3.8. *Dashboard* panel admin

Setelah berhasil masuk, *tenant* akan diarahkan ke halaman *dashboard* seperti pada Gambar 3.8. Pada halaman ini, *tenant* dapat mengakses berbagai fitur yang tersedia, salah satunya adalah halaman *top-up* deposit.



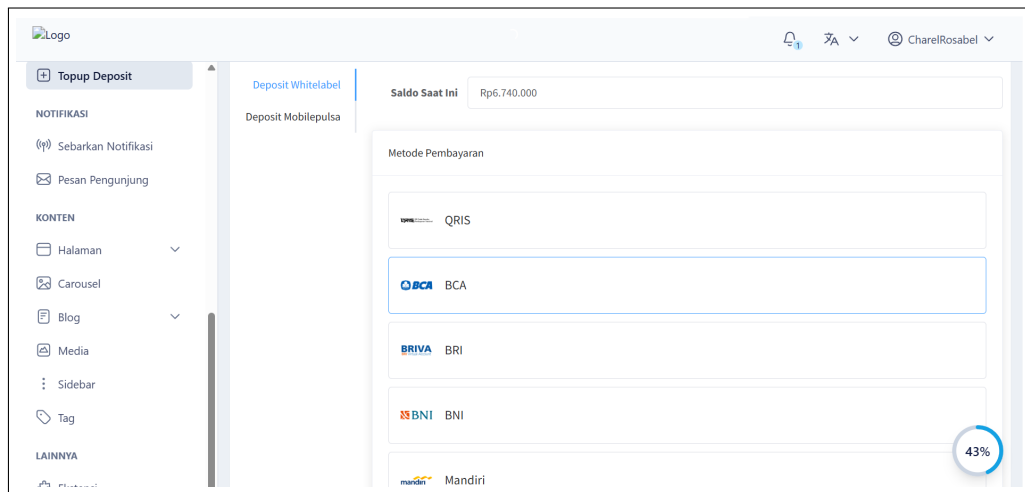
Gambar 3.9. Halaman *top-up* deposit

Pada halaman *topup deposit* seperti pada Gambar 3.9, *tenant* dapat mengisi saldo deposit. Jika tidak ada transaksi yang sedang berstatus *pending*, maka formulir *top-up* akan tersedia dan proses pengisian saldo dapat dilakukan.



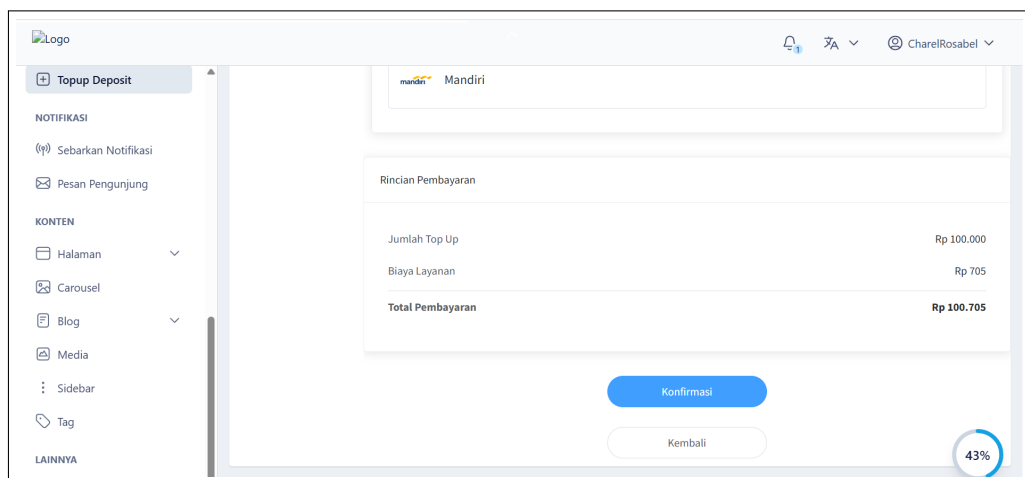
Gambar 3.10. *Processing state* pada halaman *top-up* deposit

Namun, jika terdapat transaksi sebelumnya yang masih berstatus *pending*, sistem akan menampilkan detail transaksi tersebut seperti ditunjukkan pada Gambar 3.10. Formulir *top-up* akan dinonaktifkan hingga status transaksi tersebut berubah.



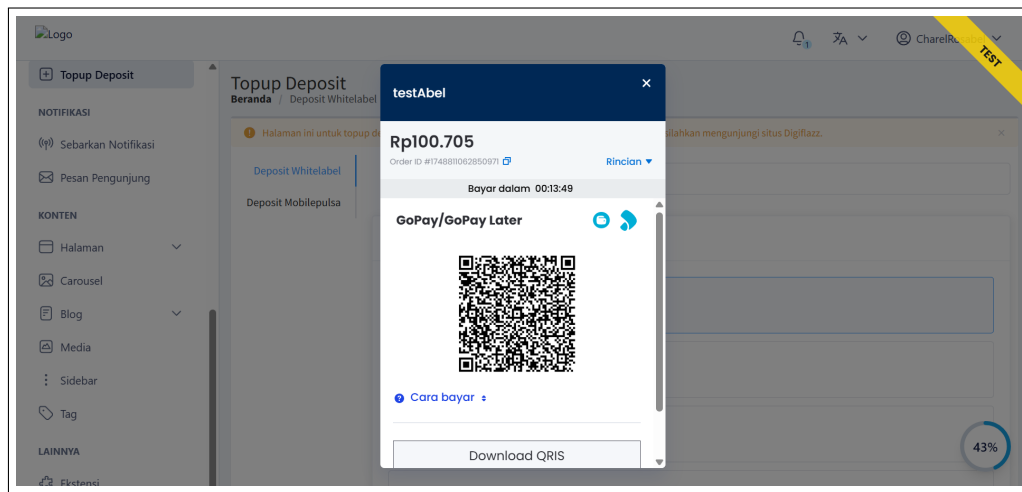
Gambar 3.11. Pemilihan metode pembayaran pada halaman *top-up* deposit

Untuk melakukan *top-up*, *tenant* dapat mengisi nominal deposit yang diinginkan, kemudian memilih metode pembayaran yang tersedia seperti QRIS atau *Virtual Account* dari berbagai bank. Ilustrasi pemilihan metode pembayaran ditampilkan pada Gambar 3.11.



Gambar 3.12. Total nominal *top-up* deposit yang akan dibayar

Setelah itu, sistem akan menampilkan rincian total nominal yang harus dibayar sesuai pilihan metode pembayaran, seperti terlihat pada Gambar 3.12. Informasi ini berguna sebagai konfirmasi sebelum pengguna melanjutkan proses.



Gambar 3.13. Halaman pembayaran *top-up* deposit

Langkah berikutnya, sistem mengirim permintaan ke *payment gateway* Midtrans Snap API dan menerima respons berupa tautan SNAP. Halaman tersebut menampilkan kode QR dan instruksi pembayaran seperti pada Gambar 3.13. Tenant kemudian dapat melakukan pembayaran sesuai metode yang dipilih.

Setelah permintaan pembayaran dikirimkan, sistem mencatat transaksi sebagai *pending* pada basis data. Ini berfungsi sebagai penanda bahwa proses *top-up* sedang berlangsung dan mencegah transaksi ganda dalam waktu yang bersamaan.

Sistem akan menerima *callback* dari Midtrans ketika transaksi berhasil (status *settlement*) atau gagal (status *expired*). Status transaksi kemudian diperbarui di database secara otomatis. Jika berhasil, maka saldo tenant akan langsung ditambahkan sesuai dengan jumlah pembayaran. Sementara itu, pada sisi *frontend*, tampilan transaksi *pending* akan hilang, dan *tenant* dapat melakukan *top-up* ulang jika dibutuhkan.

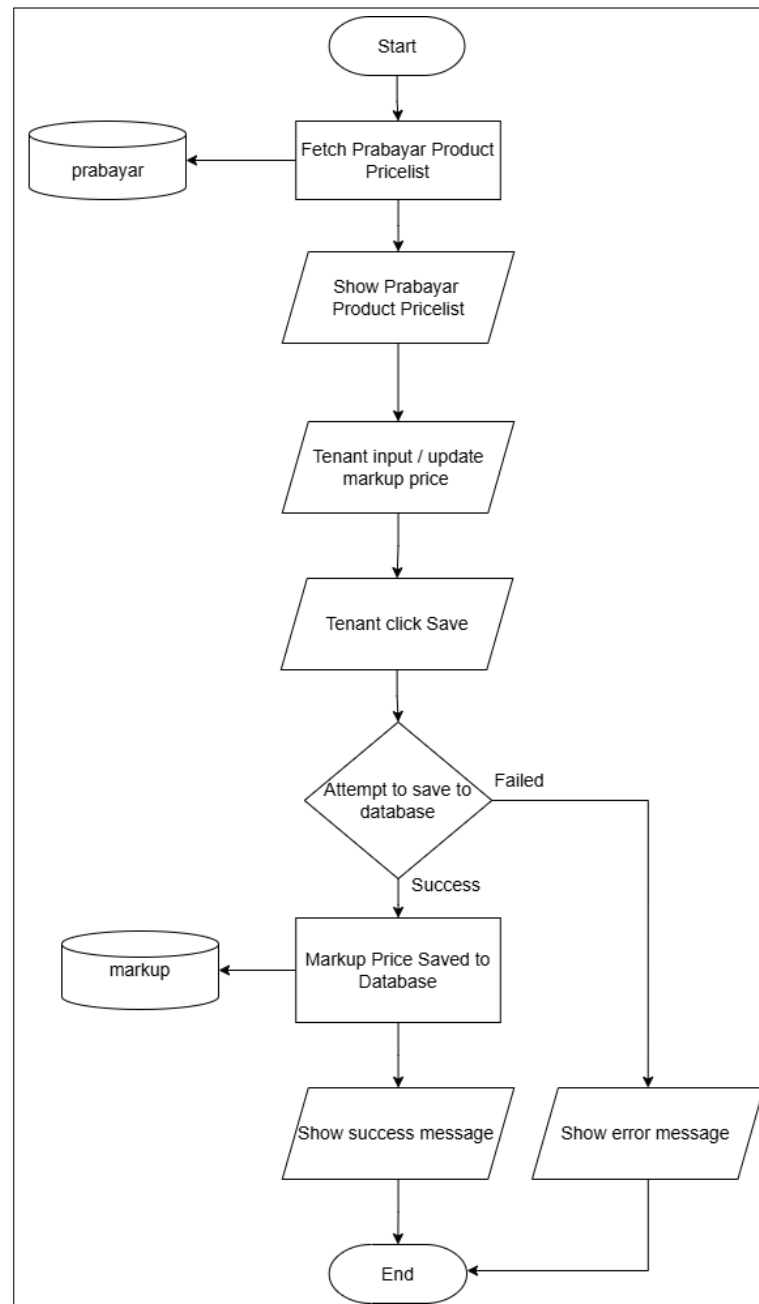
3.6.2 Fitur harga *markup* pada One Page Checkout

Fitur harga *markup* pada One Page Checkout (OPC) memungkinkan admin *tenant* untuk menetapkan harga jual produk prabayar kepada pengguna akhir dengan menambahkan nilai *markup* tertentu di luar harga dasar produk. Fitur ini dirancang agar setiap halaman One Page Checkout hanya menampilkan satu produk spesifik, dan harga akhir yang ditampilkan kepada pengguna telah mencerminkan harga dasar ditambah *markup*. Dengan adanya pengaturan *markup* ini, *tenant* memiliki fleksibilitas dalam menentukan margin keuntungan dan menyesuaikan harga jual sesuai strategi bisnis masing-masing.

Beberapa kebutuhan sistem untuk fitur ini mencakup dua hal utama. Pertama, *tenant* perlu halaman untuk mengatur nilai *markup*. Halaman ini dapat diakses melalui menu prabayar di panel admin, menampilkan daftar produk yang telah diaktifkan. Setiap produk memiliki kolom *markup* yang bisa diedit secara manual maupun secara massal. Nilai-nilai ini akan disimpan dalam tabel *markup* di basis data.

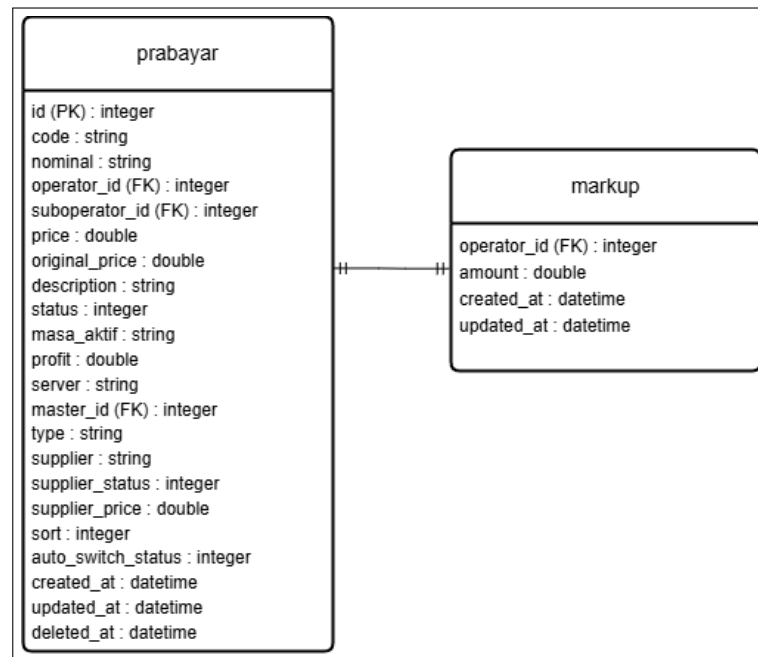
Kedua, sistem perlu menerapkan nilai *markup* yang telah disimpan tersebut ke halaman produk One Page Checkout. Saat pengguna membuka halaman One Page Checkout, sistem akan menjumlahkan harga dasar dengan nilai *markup* yang sesuai, berdasarkan kombinasi produk dan *markup*, lalu menampilkan harga akhirnya ke pengguna.

Untuk lebih jelasnya, Gambar 3.14 menunjukkan alur proses pengaturan *markup* oleh *tenant*. Proses dimulai ketika *tenant* mengakses halaman pengaturan, lalu mengisi nilai *markup* untuk setiap produk yang tersedia. Setelah nilai disimpan, sistem akan menyimpan informasi tersebut di basis data dan menggunakannya dalam perhitungan harga akhir.



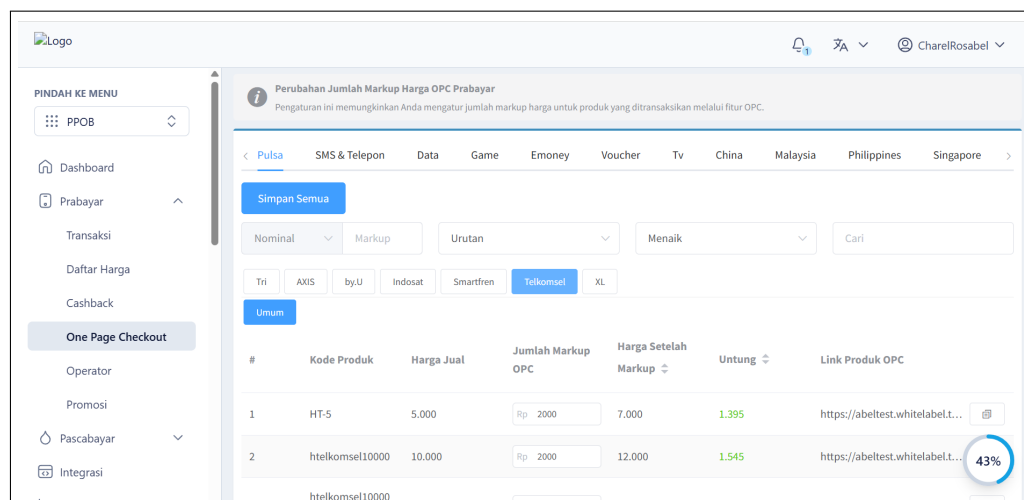
Gambar 3.14. Diagram akses dan pengaturan harga *markup* oleh *tenant*

Struktur penyimpanan data terkait produk prabayar dan harga markup dapat dilihat pada Gambar 3.15. Diagram ini menunjukkan hubungan antara tabel-tabel seperti prabayar dan *markup* yang digunakan dalam mekanisme penentuan harga akhir produk pada halaman One Page Checkout.



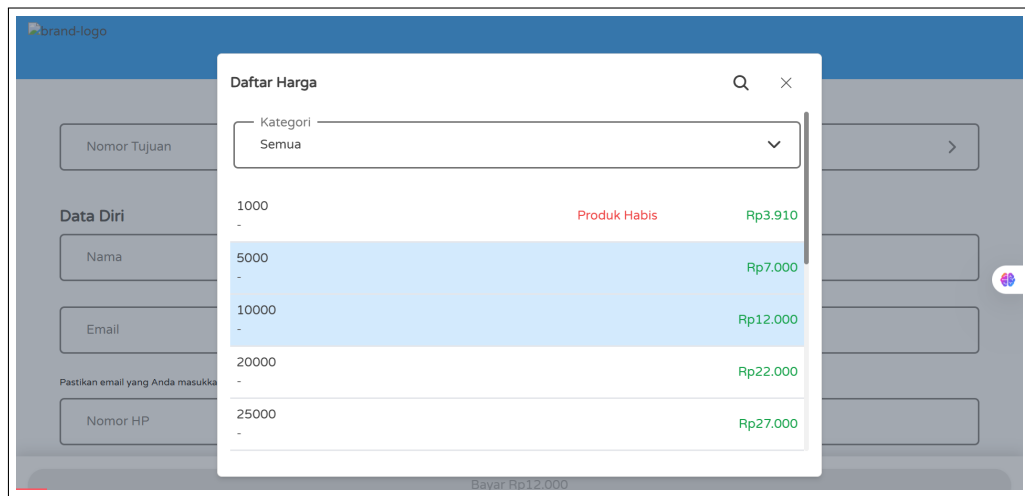
Gambar 3.15. Diagram *physical ERD* relasi produk prabayar dan *markup*

Pada sisi antarmuka, *tenant* dapat mengakses halaman pengaturan *markup* melalui panel admin. Di halaman ini, *tenant* dapat mengedit nilai *markup* untuk masing-masing produk secara langsung ataupun secara massal, seperti yang ditampilkan pada Gambar 3.16.



Gambar 3.16. Tampilan halaman pengaturan *markup* di panel admin

Setelah nilai *markup* ditetapkan, sistem akan menerapkannya pada tampilan produk di halaman One Page Checkout. Gambar 3.17 memperlihatkan bagaimana produk ditampilkan dari sisi *tenant*, sedangkan gambar berikutnya akan menunjukkan tampilan harga akhir dari sisi pengguna.



Gambar 3.17. Tampilan produk dari sisi *tenant* pada halaman One Page Checkout

Tampilan harga akhir produk yang telah dikenakan *markup*, sebagaimana dilihat oleh pengguna akhir, diperlihatkan pada Gambar 3.18. Pengguna tidak melihat rincian nilai *markup* maupun harga dasar, hanya harga final yang telah dihitung oleh sistem.



Gambar 3.18. Harga akhir produk setelah dikenakan *markup* di fitur One Page Checkout

Dengan demikian, pengguna disuguhkan tampilan harga yang sederhana dan jelas, sementara *tenant* tetap memiliki keleluasaan dalam mengatur strategi harga secara internal.

3.7 Kendala yang Ditemukan

Selama praktik magang, kendala utama yang ditemukan adalah:

- Memerlukan waktu untuk memahami arsitektur program berbasis *multi-tenancy*.
- Mengoptimasi performa aplikasi agar tetap responsif saat menangani data dalam jumlah besar.

3.8 Solusi terhadap Kendala yang Ditemukan

Beberapa langkah yang dilakukan untuk mengatasi kendala tersebut antara lain:

- a. Mempelajari dokumentasi, berdiskusi, dan meminta arahan dari senior serta rekan kerja.
- b. Menerapkan teknik optimasi kode dan *debugging* yang efektif agar aplikasi berjalan lebih stabil dan responsif.