

## BAB 3

### PELAKSANAAN KERJA MAGANG

#### 3.1 Kedudukan dan Koordinasi

Selama kegiatan magang di PT Surya Digital Indonesia, posisi peserta magang ditempatkan pada *IT Department* yang berada di bawah pengawasan langsung *Supervisor IT*. Kedudukan ini menempatkan peserta magang sebagai bagian dari tim pengembang yang bertanggung jawab dalam proses pembuatan dan pengembangan proyek *website* untuk salah satu klien perusahaan, yaitu toko besi.

Dalam pelaksanaan tugas, peserta magang berperan sebagai pendukung teknis yang membantu tim dalam mengimplementasikan sistem, memperbaiki kode, serta melakukan pengujian fungsional. Setiap kegiatan dilakukan di bawah koordinasi *Supervisor IT* yang berfungsi sebagai penanggung jawab utama proyek dan pengarah kerja seluruh anggota tim.

Hubungan koordinasi antara peserta magang, *developer*, dan supervisor bersifat hierarkis dan fungsional, di mana setiap keputusan teknis maupun strategis tetap mengacu pada arahan *Supervisor IT*. Komunikasi kerja dilaksanakan secara rutin melalui rapat koordinasi internal maupun laporan kemajuan proyek yang disampaikan secara berkala.

##### 3.1.1 Alur Kerja dan Koordinasi Magang

###### 1. Perencanaan pembagian tugas

Pada tahap awal, Supervisor IT mengadakan *meeting* internal untuk menjelaskan tujuan, ruang lingkup, dan target dari proyek *website* toko besi. Dalam sesi ini ditetapkan pembagian tugas antara anggota tim, termasuk penugasan khusus kepada peserta magang sesuai kemampuan dan bidang yang dikuasai.

###### 2. Perancangan dan Desain

Setelah perencanaan selesai, tim melakukan perancangan struktur dan desain *website* yang meliputi antarmuka pengguna (*user interface*) serta alur navigasi. Peserta magang bekerja sama dengan developer dalam mengimplementasikan desain dasar dan struktur halaman menggunakan teknologi yang telah ditentukan, seperti *HTML*, *CSS*, dan *JavaScript*.

*Supervisor IT* memberikan arahan teknis dan melakukan peninjauan terhadap hasil rancangan sebelum tahap pengembangan dimulai.

### 3. Pengembangan dan Implementasi

Pada tahap ini, peserta magang berperan aktif dalam proses pembuatan fitur website berdasarkan rancangan yang telah disetujui. Setiap progres pekerjaan dilaporkan secara rutin kepada *developer* dan *Supervisor IT* untuk mendapatkan evaluasi serta umpan balik. Koordinasi dilakukan melalui pertemuan harian atau laporan mingguan agar seluruh tim memiliki pemahaman yang sama terhadap perkembangan proyek.

### 4. Pengujian dan Revisi

Setelah pengembangan selesai, dilakukan pengujian (*testing*) untuk memastikan seluruh fitur berjalan dengan baik dan bebas dari kesalahan. Peserta magang turut membantu dalam proses *debugging* dan dokumentasi hasil pengujian. Apabila ditemukan kendala teknis, *Supervisor IT* memberikan instruksi untuk melakukan perbaikan hingga sistem mencapai stabilitas yang diinginkan.

### 5. Finalisasi dan Dokumentasi

Pada tahap akhir, tim melakukan finalisasi terhadap proyek website toko besi, termasuk penyusunan laporan hasil kerja dan dokumentasi teknis sistem. Peserta magang membantu dalam penulisan dokumentasi kode, panduan penggunaan, serta catatan perbaikan sistem. Seluruh hasil kerja dikompilasi oleh *developer* dan diverifikasi oleh *Supervisor IT* sebelum diserahkan kepada pihak klien.

Alur kerja dan koordinasi yang terstruktur ini mencerminkan profesionalisme Divisi IT PT Surya Digital Indonesia dalam menjalankan setiap proyek. Dengan adanya bimbingan langsung dari *Supervisor IT*, kegiatan magang berjalan efektif serta memberikan pengalaman kerja nyata dalam proses pengembangan produk digital yang berbasis proyek klien.

## 3.2 Uraian Pelaksanaan Magang

Selama periode magang, seluruh aktivitas yang dilakukan didokumentasikan melalui jurnal kerja mingguan. Pencatatan ini berfungsi untuk merekam setiap tahapan kegiatan, mulai dari penerimaan tugas, pelaksanaan

pekerjaan teknis, hingga penyelesaian kendala yang muncul. Jurnal tersebut menjadi bukti pelaksanaan kegiatan magang sekaligus catatan kontribusi terhadap proyek-proyek yang dikerjakan. Ringkasan dari jurnal kerja mingguan tersebut dapat dilihat pada Tabel 3.1 berikut.

Tabel 3.1. Uraian Pelaksanaan Magang

Minggu	Kegiatan
1	<p>Hari pertama magang: briefing dan pembagian tugas sebagai <i>Fullstack Developer</i> di Divisi IT.</p> <p>Adaptasi dengan tim proyek serta memahami struktur dan sumber kode (<i>source code</i>) proyek website toko besi.</p> <p>Penanganan <i>bug</i> pada bagian frontend, termasuk perbaikan responsivitas dan tata letak antarmuka.</p>
2	<p>Pengujian fungsionalitas dan <i>user flow</i> pada website toko besi.</p> <p>Pembagian tugas untuk pengembangan <i>backend</i> dan <i>API</i> antara anggota tim.</p> <p>Implementasi <i>database</i> produk dengan sistem kategori dan SKU.</p>
3	<p>Pembuatan dan pengujian <i>API call</i> untuk integrasi <i>database</i>.</p> <p>Penggunaan <i>Docker</i> untuk menginisialisasi dan mengelola <i>backend server Next.js</i>.</p> <p>Pengembangan halaman <i>homepage</i> dan <i>product page</i> untuk menampilkan daftar produk.</p> <p>Pembuatan <i>API endpoint</i> untuk <i>keranjang belanja</i> dan sistem pesanan pengguna.</p>
4	<p>Pengujian dan perbaikan <i>endpoint API</i> untuk fitur pemesanan dan keranjang belanja.</p> <p>Implementasi sistem <i>authentication</i> (login dan register) menggunakan <i>JWT</i>.</p> <p>Penyelesaian tampilan halaman <i>login</i> dan <i>register</i> di sisi frontend.</p>
5	<p>Pengerjaan <i>backend</i> untuk proses autentikasi dan validasi akun pengguna.</p> <p>Pengujian dan debugging pada proses registrasi dan login pengguna.</p> <p>Perbaikan <i>bug</i> pada sistem order agar harga dan berat sesuai dengan data produk.</p>

Tabel 3.1. Uraian Pelaksanaan Magang (Lanjutan)

Minggu	Kegiatan
6	<p>Pengerjaan sistem <i>checkout</i> serta revisi <i>dashboard admin</i>.</p> <p>Penambahan fitur manajemen produk dan kategori pada <i>admin dashboard</i>.</p> <p>Pengembangan fitur pemilihan <i>driver</i> untuk pengantaran pesanan pelanggan.</p>
7	<p>Pembuatan halaman <i>quotation</i> dan sistem ekspor data menjadi file PDF.</p> <p>Implementasi proses konversi <i>quotation</i> menjadi <i>invoice</i> dengan perhitungan pajak.</p> <p>Penambahan sistem pelacakan (<i>tracking</i>) pesanan melalui halaman transaksi.</p>
8	<p>Implementasi sistem impor data produk melalui <i>Excel/Spreadsheet</i>.</p> <p>Penambahan fitur <i>edit</i> produk dan <i>order</i> di <i>admin dashboard</i>.</p> <p>Pengembangan sistem <i>bidding</i> antara pelanggan dan admin.</p>
9	<p>Integrasi notifikasi WhatsApp menggunakan <i>Mekari Qontak API</i>.</p> <p>Penyesuaian backend untuk mendukung pelanggan tamu dan terdaftar.</p> <p>Penyelesaian proses <i>deployment</i> website toko besi ke server <i>VPS</i> menggunakan <i>Docker</i>.</p>
10	<p>Pengembangan proyek tambahan untuk aplikasi <i>Pickbooth</i> (software photobooth).</p> <p>Pembuatan <i>UI design</i> menu dan <i>settings</i> pada software photobooth.</p> <p>Integrasi backend dan autentikasi pengguna pada aplikasi <i>Pickbooth</i>.</p>
11	<p>Pembuatan sistem <i>license key</i> dan panel pengaturan (<i>settings panel</i>).</p> <p>Implementasi pembayaran menggunakan <i>Midtrans</i> dan <i>QRIS</i>.</p> <p>Pembuatan fitur untuk <i>printing</i>, <i>frame design</i>, serta pengunggahan foto ke <i>cloud storage</i>.</p>



Tabel 3.1. Uraian Pelaksanaan Magang (Lanjutan)

Minggu	Kegiatan
12	<p>Penyempurnaan fitur <i>capture</i>, <i>filter</i>, dan <i>live photo video</i>.</p> <p>Pembuatan fitur pemilihan hasil foto untuk dicetak dan sistem <i>auto-cut</i> ukuran 4x6.</p> <p>Pengujian dan perbaikan <i>bug</i> pada proses cetak dan pengunggahan foto ke <i>cloud</i>.</p>
13	<p>Pembuatan halaman <i>driver page</i> pada aplikasi Android untuk menerima dan memproses order pelanggan.</p> <p>Penyempurnaan sistem pengiriman dan penugasan <i>driver</i>.</p>

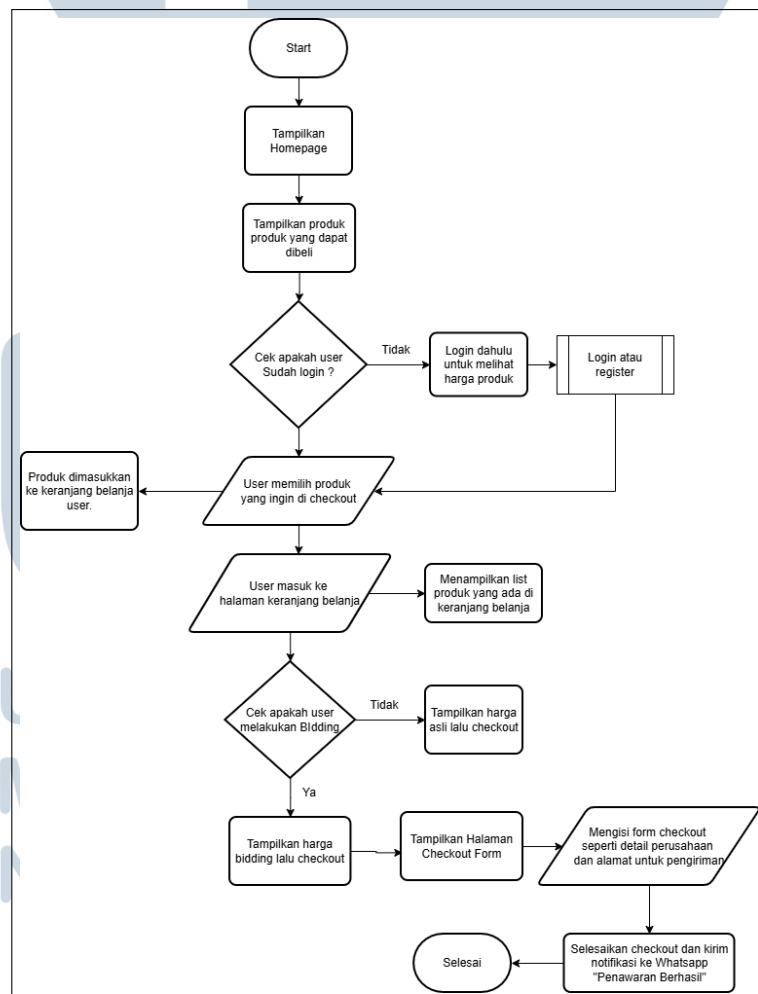


### 3.3 Perancangan Website

#### 3.3.1 Flowchart

##### A Flowchart sistem Checkout untuk User

Alur pada *Flowchart* gambar 3.1 menjelaskan proses lengkap yang dilalui pengguna saat melakukan *checkout* di *website* e-commerce toko besi. Proses dimulai dari tampilan beranda dan daftar produk, kemudian sistem memverifikasi apakah pengguna sudah melakukan login. Jika belum, pengguna diarahkan untuk *login* atau registrasi terlebih dahulu agar dapat melihat harga produk. Setelah berhasil masuk, pengguna memilih produk yang ingin di beli dan sistem memasukkan produk tersebut ke dalam keranjang belanja. Pada tahap ini, pengguna diarahkan menuju halaman keranjang untuk melihat daftar barang yang akan dibeli.

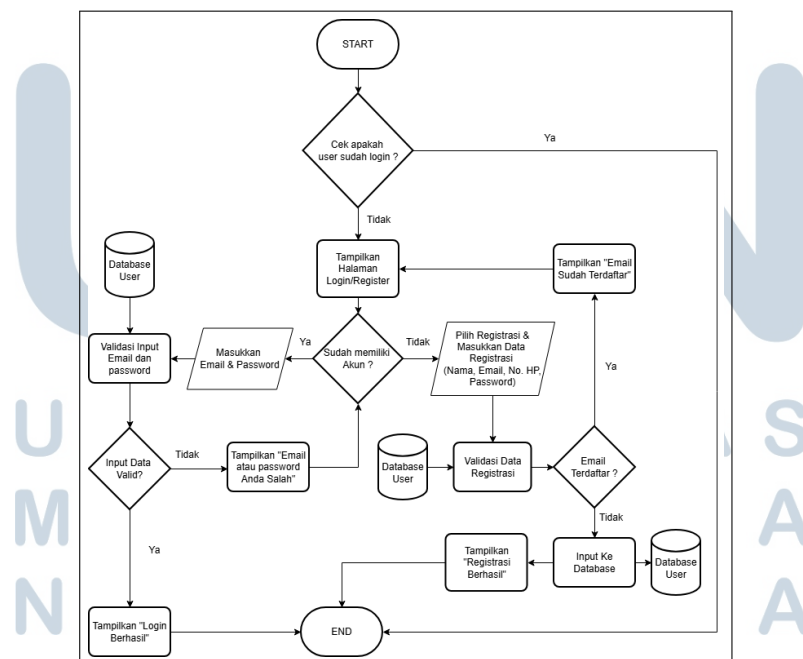


Gambar 3.1. *Flowchart* sistem *Checkout* untuk *User*

Setelah berada pada halaman keranjang, sistem melakukan pengecekan apakah pengguna melakukan *bidding*. Jika tidak, harga asli produk ditampilkan; namun jika ya, harga *bidding* akan digunakan saat proses *checkout*. Selanjutnya, pengguna diarahkan ke halaman formulir *checkout* untuk mengisi informasi penting seperti data perusahaan, alamat pengiriman, dan preferensi pengiriman atau pengambilan barang. Ketika formulir telah selesai diisi, proses *checkout* diselesaikan, dan sistem secara otomatis mengirimkan notifikasi melalui *WhatsApp* kepada admin dengan pesan “Penawaran Berhasil,” menandakan bahwa pesanan telah berhasil dibuat.

## B Flowchart autentikasi untuk Login atau Register

Flowchart autentikasi pada Gambar 3.2 dimulai dari proses pengecekan sesi melalui token *NextAuth*. Jika token masih valid, user langsung dianggap sudah login dan proses berhenti. Namun ketika token tidak ditemukan atau telah kedaluwarsa, sistem menampilkan halaman login atau registrasi. Pada mode login, user memasukkan email dan password yang kemudian divalidasi oleh *NextAuth Credentials Provider*. Jika validasi sukses, sistem menghasilkan JWT baru sebagai session token dan menyimpannya pada cookie sehingga user resmi terautentikasi.



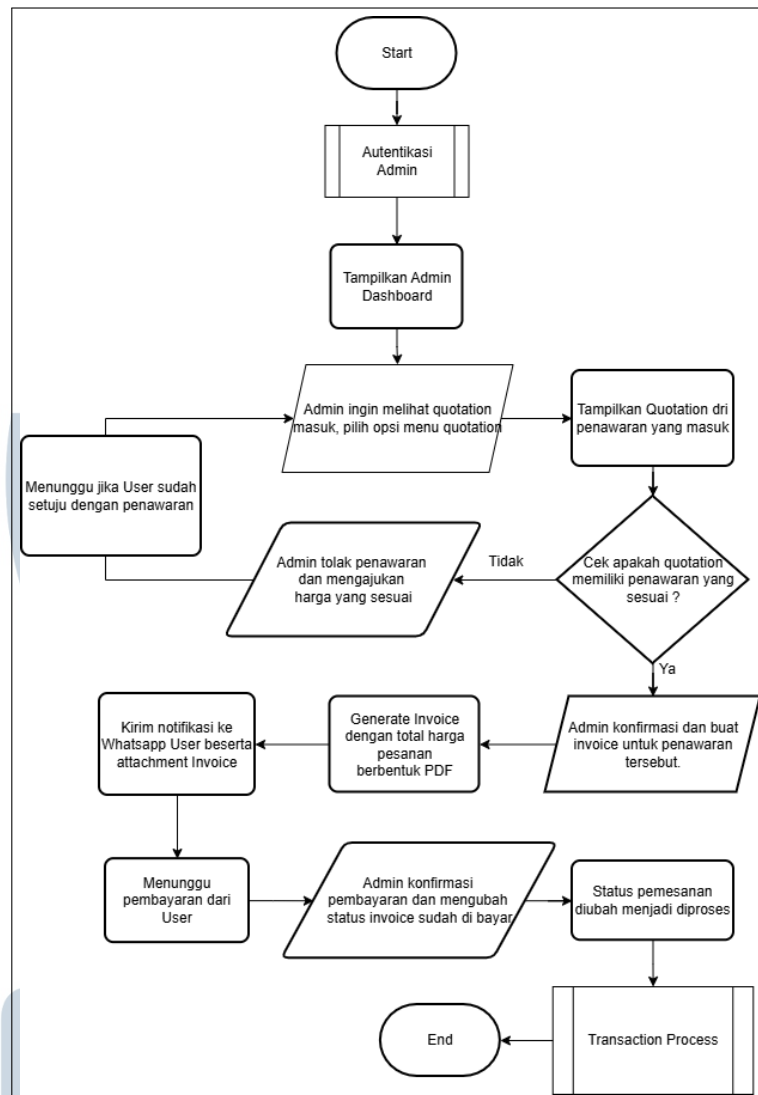
Gambar 3.2. Flowchart autentikasi untuk User Login atau Register

Pada sisi registrasi, user mengisi data seperti nama, email, nomor telepon, dan password. Backend terlebih dahulu memeriksa apakah email sudah terdaftar. Jika sudah ada, sistem menampilkan notifikasi bahwa email telah digunakan. Jika belum, data disimpan ke database dan proses registrasi dinyatakan berhasil. Setelah itu user dapat melakukan login, dan sama seperti jalur sebelumnya, *NextAuth* akan membuat JWT session token yang kemudian dipakai untuk autentikasi di request berikutnya. Dengan demikian seluruh alur autentikasi pada Gambar 3.2 berpusat pada verifikasi kredensial dan validitas token sesi.

### **C Flowchart untuk proses admin approval Quotation dan generate Invoice**

Pada Gambar 3.3 ditunjukkan alur proses kerja pada halaman *Admin Dashboard* yang digunakan untuk mengelola permintaan *quotation* dari pengguna. Proses dimulai dari autentikasi *admin*, kemudian sistem menampilkan *dashboard* sebagai pusat kendali. *Admin* dapat mengakses menu quotation untuk melihat daftar permintaan yang masuk, kemudian melakukan pengecekan terhadap kecocokan penawaran yang diajukan oleh pengguna. Jika harga yang diajukan tidak sesuai, *admin* dapat memberikan revisi harga dan menunggu persetujuan dari pengguna sebelum proses dilanjutkan. Tahapan ini memastikan bahwa setiap penawaran telah melalui proses verifikasi sehingga harga yang disepakati benar- benar sesuai dengan kebijakan perusahaan.

Setelah pengguna menyetujui penawaran, admin akan membuat *invoice* berdasarkan total biaya yang telah disepakati. Sistem kemudian menghasilkan *invoice* dalam format PDF dan mengirimkannya kepada pengguna melalui WhatsApp sebagai lampiran. *Admin* selanjutnya menunggu konfirmasi pembayaran dari pengguna dan melakukan pembaruan status *invoice* menjadi “sudah dibayar” ketika pembayaran diterima. Pada tahap tersebut, status pesanan otomatis berubah menjadi “diproses”, sehingga pesanan dapat masuk ke tahap *Transaction Process*.

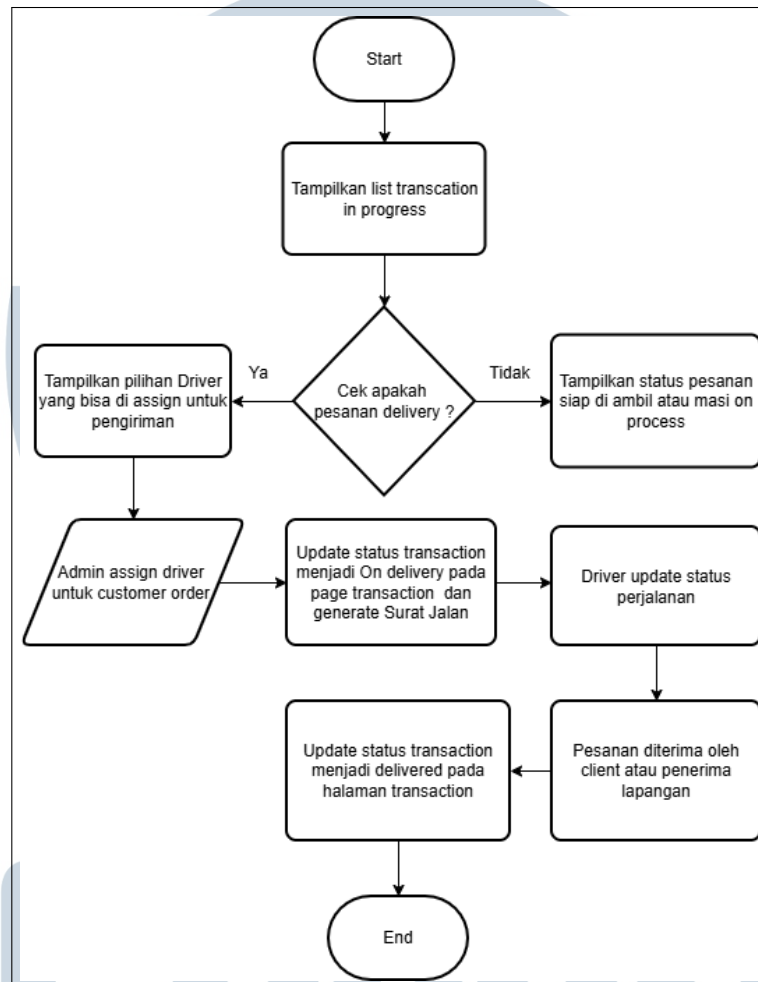


Gambar 3.3. Flowchart untuk proses *admin* menangani *customer* di *admin dashboard*

#### D Flowchart untuk memperbarui status *Transaction*

Flowchart pada Gambar 3.4 menggambarkan alur pembaruan status transaksi yang sedang berlangsung. Proses dimulai dengan menampilkan daftar transaksi *in progress*, kemudian sistem memeriksa apakah pesanan tersebut membutuhkan pengiriman. Jika termasuk pesanan *delivery*, admin memilih dan melakukan *assign* driver untuk *order* tersebut, lalu sistem memperbarui status menjadi *On Delivery* serta menghasilkan Surat Jalan. Setelah *driver* memulai perjalanan dan memperbarui statusnya, pesanan akan diterima oleh *client* atau penerima lapangan, dan status transaksi diperbarui menjadi *Delivered*. Sementara

itu, jika pesanan bukan *delivery*, sistem hanya menampilkan status bahwa pesanan siap diambil atau masih diproses tanpa melalui tahapan pengiriman.



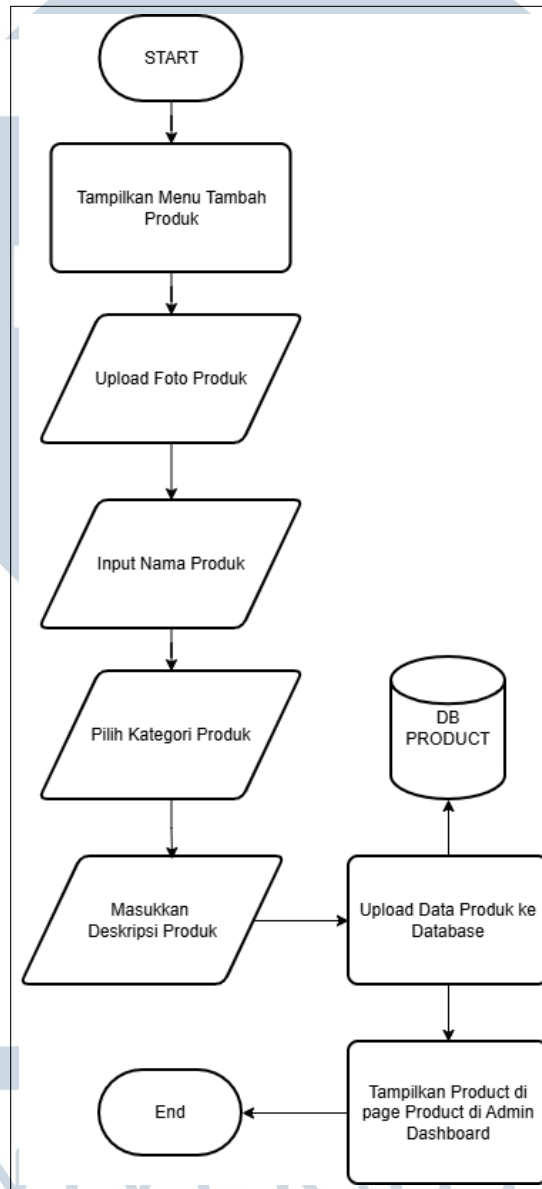
Gambar 3.4. Flowchart untuk *update status Transaction Process*

#### E Flowchart untuk upload produk melalui admin dashboard

Berdasarkan flowchart pada Fig. 3.5, proses penambahan produk dimulai ketika admin mengakses menu tambah produk pada sistem. Admin kemudian melakukan unggah foto produk sebagai representasi visual, dilanjutkan dengan pengisian nama produk untuk identifikasi. Setelah itu, admin memilih kategori produk yang sesuai guna mempermudah pengelompokan data. Selanjutnya, admin memasukkan deskripsi produk sebagai informasi pendukung yang menjelaskan detail produk. Setelah seluruh data diinput, sistem akan memproses dan mengunggah data produk ke dalam basis data (*database product*). Apabila proses



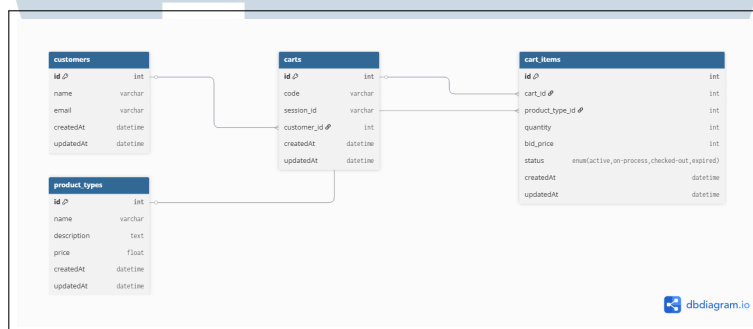
penyimpanan berhasil, produk yang telah ditambahkan akan ditampilkan pada halaman daftar produk di dashboard admin, menandakan bahwa proses penambahan produk telah selesai.



Gambar 3.5. Flowchart untuk upload produk di admin dashboard

### 3.3.2 Diagram Database

Pada gambar 3.6 terlihat diagram database untuk *keranjang belanja system* yang menggambarkan alur relasi antara pelanggan, keranjang belanja, dan produk yang ditambahkan. Pada sistem ini, tabel *customers* berperan sebagai entitas utama yang mewakili pengguna terdaftar. Setiap pelanggan dapat memiliki satu atau beberapa *carts* (keranjang belanja), yang masing-masing diidentifikasi dengan kode unik *code* dan *session\_id* untuk membedakan sesi pembelian. Relasi satu ke banyak (*one-to-many*) antara *customers* dan *carts* menunjukkan bahwa satu pelanggan bisa membuat beberapa transaksi dalam waktu berbeda. Tabel *carts* juga menyimpan informasi waktu pembuatan dan pembaruan *createdAt* dan *updatedAt* yang berguna untuk melacak aktivitas pengguna dalam sistem.

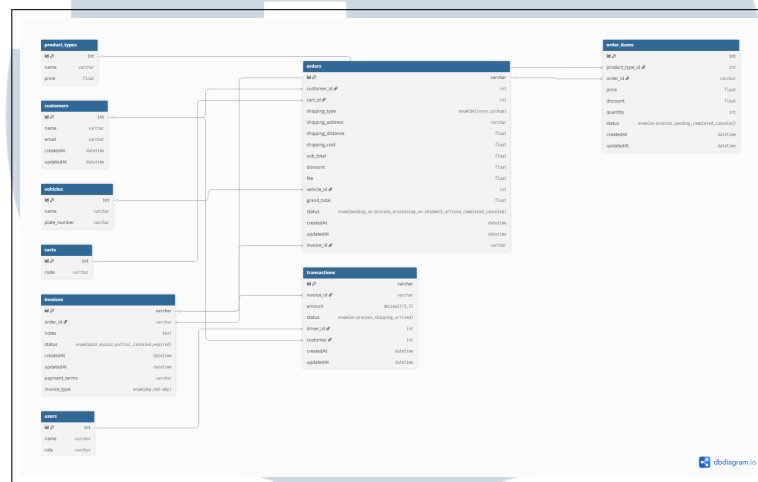


Gambar 3.6. Diagram Database Sistem Keranjang Belanja

Selanjutnya, setiap keranjang belanja dapat memiliki beberapa item yang tersimpan dalam tabel *cart\_items*. Tabel ini menghubungkan *cart\_id* dari *carts* dan *product\_type\_id* dari *product\_types*, membentuk relasi *many-to-many* antara keranjang dan produk. *Field* seperti *quantity*, *bid\_price*, dan status berfungsi mencatat jumlah barang, harga penawaran dari pengguna (jika ada), serta status item (aktif, diproses, atau sudah *checkout*). Struktur ini memungkinkan sistem *e-commerce* untuk mengelola daftar produk yang dimasukkan ke keranjang secara dinamis sebelum diproses menjadi pesanan. Dengan demikian, diagram ini menunjukkan fondasi utama dari modul *Add to Cart*, yang menjadi tahap awal dalam proses transaksi pembelian.

Pada gambar 3.7 terlihat diagram database untuk *checkout system* yang menggambarkan hubungan antara pesanan, produk yang dipesan, faktur, dan transaksi pembayaran. Tabel *orders* menjadi entitas utama yang menyimpan informasi pesanan pelanggan, seperti alamat pengiriman, biaya ongkir, subtotal, diskon, serta total keseluruhan transaksi. Setiap *order* terhubung dengan tabel

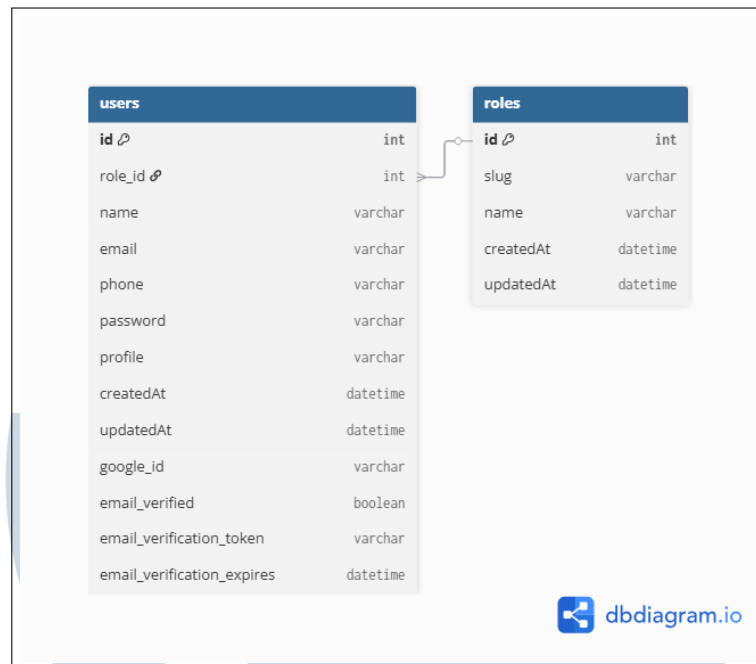
*customers* melalui atribut *customer\_id* untuk mencatat siapa pelanggan yang melakukan pembelian, dan juga dapat dikaitkan dengan tabel *carts* sebagai sumber asal keranjang belanja sebelum dilakukan proses *checkout*. Relasi ini bersifat satu ke banyak (*one-to-many*), di mana satu pelanggan dapat memiliki beberapa pesanan. Selain itu, setiap *order* juga memiliki keterkaitan dengan *vehicles* dan *users* (pengemudi) yang mengatur pengiriman pesanan, sehingga mencakup keseluruhan proses dari pemesanan hingga distribusi barang.



Gambar 3.7. Diagram Database Sistem Pemesanan

Selanjutnya, tabel *order\_items* berfungsi sebagai penghubung antara pesanan dan produk yang dipesan melalui atribut *order\_id* dan *product\_type\_id*. Struktur ini memungkinkan sistem untuk mencatat detail setiap produk dalam pesanan, termasuk harga, jumlah, dan status proses. Tabel *invoices* berelasi dengan *orders* untuk mencatat faktur penjualan serta status pembayaran seperti *paid* atau *unpaid*. Sedangkan tabel *transactions* menjadi tahap akhir dalam proses *checkout*, mencatat setiap transaksi keuangan yang dilakukan berdasarkan faktur tertentu melalui atribut *invoice\_id*. Dengan adanya relasi antar tabel tersebut, sistem *checkout* mampu mengelola keseluruhan alur transaksi mulai dari pesanan pelanggan hingga penyelesaian pembayaran secara terstruktur dan terintegrasi.

Pada Gambar 3.8 ditampilkan struktur relasi antara tabel *users* dan *roles* yang digunakan untuk mengatur sistem autentikasi serta pembagian hak akses pada aplikasi. Tabel *roles* berfungsi sebagai daftar definisi peran, yang terdiri dari lima jenis hak akses berbeda, yaitu *user*, *guest*, *super-admin*, *driver*, dan *admin*. Setiap peran memiliki tingkat otoritas masing-masing, mulai dari *guest* yang hanya dapat mengakses layanan tanpa login, hingga *super-admin* yang memiliki kontrol penuh.

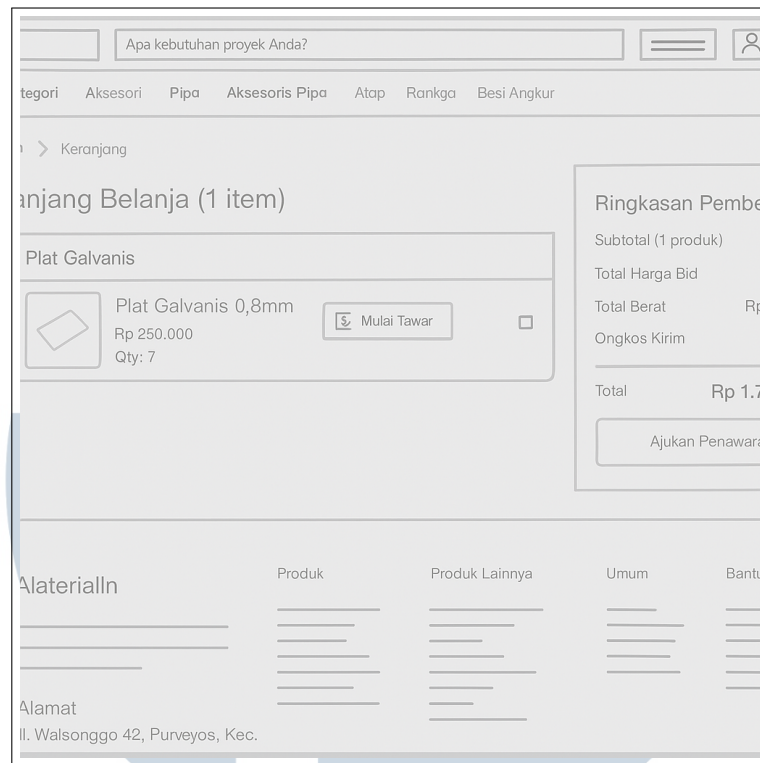


Gambar 3.8. Diagram Database User dan Roles

Sementara itu, tabel *users* menyimpan data akun pengguna seperti nama, email, nomor telepon, kredensial login, serta status verifikasi. Kolom *role\_id* menjadi kunci penting untuk menghubungkan setiap akun ke salah satu peran yang didefinisikan pada tabel *roles*. Dengan adanya relasi ini, sistem dapat menentukan kemampuan dan batasan setiap pengguna berdasarkan peran yang dimilikinya, sehingga pengelolaan akses dapat dilakukan dengan lebih terstruktur dan aman.

### 3.3.3 Wireframe Tampilan

Fitur sistem *bidding* pada gambar 3.9 dirancang agar pengguna dapat melakukan penawaran harga produk secara fleksibel dan interaktif. Setiap penawaran dapat dilakukan dengan dua cara, yaitu menggunakan sistem *increment* sebesar Rp500 untuk menaikkan atau menurunkan harga secara bertahap, atau dengan mengetik langsung nominal harga yang diinginkan pada kolom input. Mekanisme ini memberikan kebebasan bagi pengguna dalam menentukan strategi tawar-menawar sesuai kebutuhan, sekaligus menjaga agar proses negosiasi tetap efisien dan terkontrol. Sistem kemudian akan menyimpan nilai penawaran tersebut dalam basis data melalui atribut *bid\_price*, dan memperbarui total harga serta status item secara otomatis, sehingga menciptakan pengalaman belanja yang lebih dinamis dan realistis layaknya transaksi langsung antara pembeli dan penjual.



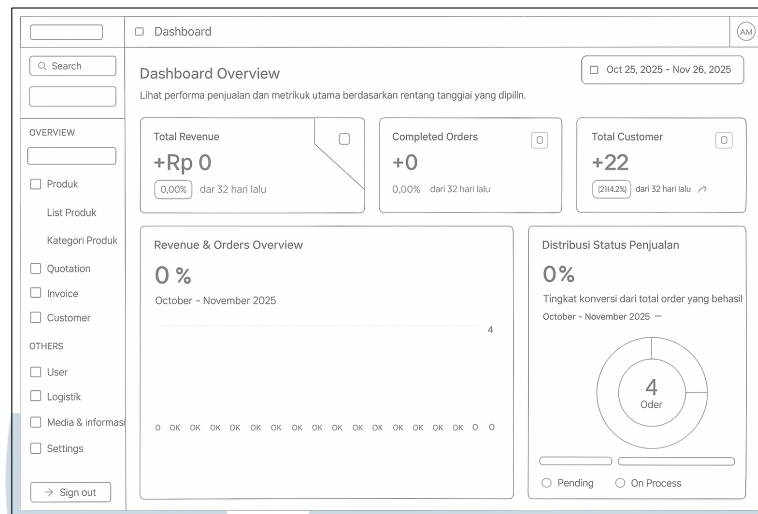
Gambar 3.9. Wireframe Sistem Keranjang Belanja

Halaman untuk sistem *checkout* pada gambar 3.10 berfungsi untuk mengumpulkan informasi penting dari pengguna sebelum proses pengiriman dan pembayaran dilakukan. Formulir ini mencakup data seperti nama perusahaan, nama lengkap, alamat pengiriman, serta opsi untuk mencentang apabila penerima barang berbeda dengan pembeli (penerima lapangan). Selain itu, pengguna dapat menentukan lokasi pengiriman secara akurat melalui fitur *pinpoint* lokasi menggunakan GPS atau melalui input manual.

Di sisi kanan halaman, terdapat bagian rincian pengiriman yang menampilkan metode pengiriman (*delivery* atau *pickup*), estimasi ongkos kirim, serta ringkasan total harga pesanan. Struktur ini dirancang agar proses *checkout* berjalan efisien dan transparan, serta meminimalkan potensi kesalahan baik dalam penentuan alamat pengiriman maupun perhitungan biaya keseluruhan.



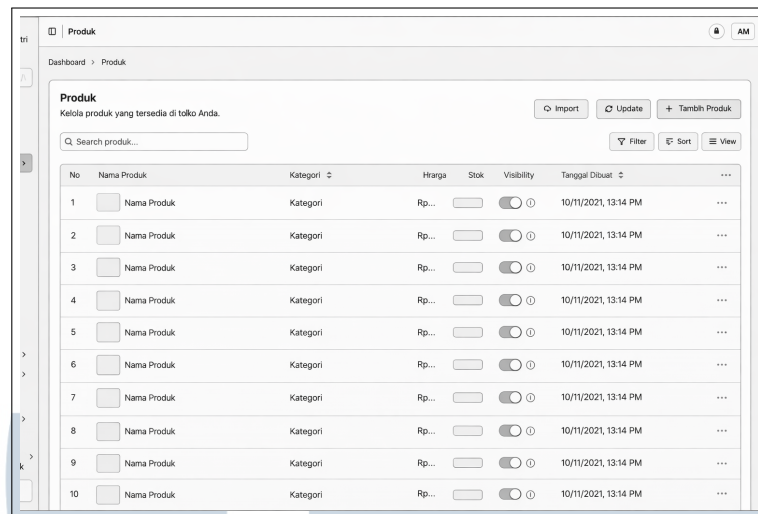




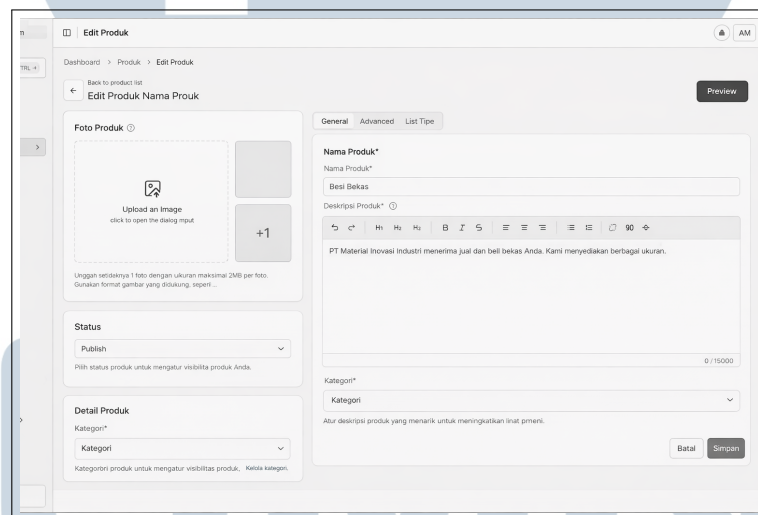
Gambar 3.11. Wireframe Admin Dashboard

Wireframe halaman *List Produk* pada gambar 3.12 dirancang sebagai pusat pengelolaan data produk pada dashboard admin. Pada halaman ini, admin dapat melihat seluruh daftar produk yang tersimpan dalam sistem secara terstruktur dalam bentuk tabel yang memuat informasi utama seperti nama produk, kategori, harga, stok, status visibilitas, tanggal pembuatan, dan aktivitas terakhir. Selain itu, halaman ini dilengkapi dengan fitur pencarian, penyaringan (*filter*), pengurutan (*sort*), serta pengaturan tampilan untuk memudahkan admin dalam menemukan dan mengelola produk dengan jumlah data yang besar. Tersedia pula tombol aksi seperti *Import*, *Update*, dan *Tambah Produk* yang memungkinkan admin melakukan manajemen produk secara efisien dan terpusat.

Wireframe halaman *Edit Produk* pada gambar 3.13 berfungsi sebagai antarmuka untuk memperbarui informasi detail suatu produk yang telah tersimpan di dalam sistem. Halaman ini menyediakan area unggah foto produk, pengaturan status publikasi, serta formulir input yang mencakup nama produk, deskripsi produk, dan kategori produk. Struktur halaman dibagi secara jelas antara pengelolaan media dan pengisian informasi teks guna meningkatkan kenyamanan dan fokus admin saat melakukan pengeditan data. Selain itu, tersedia tombol *Preview* untuk meninjau tampilan produk sebelum disimpan, serta tombol *Simpan* dan *Batal* sebagai kontrol akhir dalam proses perubahan data produk agar tetap terkontrol dan minim kesalahan.



Gambar 3.12. Wireframe Halaman Produk Admin Dashboard



Gambar 3.13. Wireframe Halaman Edit Produk Admin Dashboard

Halaman *Quotation* pada gambar 3.14 merupakan bagian dari *admin dashboard* yang dirancang untuk memudahkan admin dalam memantau dan mengelola permintaan penawaran harga dari pengguna. Pada *wireframe* tersebut, tampilan disusun dalam bentuk tabel berukuran besar agar informasi seperti jenis *order*, status pesanan, tanggal permintaan, serta nama pemesan dapat dibaca dengan jelas dan cepat.

Tabel ini dirancang bersifat dinamis, sehingga setiap data baru yang masuk akan otomatis diperbarui tanpa memerlukan interaksi manual dari admin. Pendekatan ini memastikan bahwa proses pemantauan *quotation* berlangsung

efisien dan real-time. Selain itu, struktur tabel yang rapi membantu admin melakukan pengecekan dan tindak lanjut dengan lebih akurat, sehingga alur pengelolaan pesanan dapat berjalan lebih teratur dan responsif.

#	Quote ID	Received	Customer	Phone	Quotation Status	Shipping	Status Harga	Items	Subtotal	Origin
1	G08JUMQJ			—		Pending	Delivery			...
2	GRD-104UR4			—		On Process	Delivery			...
3	100UMAW			—		On Process	Shipping			...
4	GRD-F0MQJ			—		Delivery	Pickup			...
5	ONPR0TORI					Delivery	Delivery			...
6	GRBBLUR7					Pickup	Delivery			...
7	GRD101HQJ					Delivery	Pickup			...
8	GRB787BRP					Pending	Pickup			...
4	GRB800SKFP			—		Delivery	Large Ad			...

Showing 1 - 10 of 14 results

Rows per page: 10 Page 1 of 2

Gambar 3.14. Wireframe Quotation Admin Dashboard

### 3.3.4 Tampilan Website

Halaman *login* yang terdapat pada gambar 3.15 ini berfungsi sebagai *interface* untuk autentikasi *user*. Apabila *user* sudah memiliki akun, *user* dapat memasukkan *email* dan *password* untuk masuk. Tersedia juga opsi untuk *login* dengan *google* untuk memudahkan agar *user* dapat langsung diautentikasi dari *google account* yang mereka miliki. Apabila *user* belum memiliki akun, maka *user* dapat klik pada bagian "Daftar sekarang" untuk diarahkan ke halaman pendaftaran akun.

UNIVERSITAS  
MULTIMEDIA  
NUSANTARA

**Masuk**

Masukkan email dan password Anda untuk melanjutkan

**Email**

nama@email.com

**Password**

Masukkan password

☐ Ingat saya [Lupa password?](#)

**Masuk**

ATAU MASUK DENGAN

Google

Belum punya akun? [Daftar sekarang](#)

Gambar 3.15. Tampilan halaman *Login* untuk User

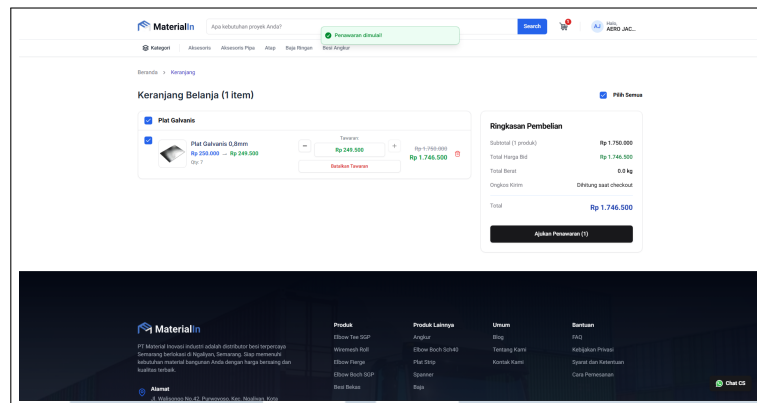
Halaman pendaftaran pada gambar 3.16 ditujukan untuk *user* yang ingin melakukan pendaftaran akun dengan memasukkan data seperti *email*, nomor telepon, dan *password* yang ingin digunakan. Setelah klik *daftar sekarang*, *user* akan menerima email konfirmasi untuk melakukan verifikasi email. Setelah itu *user* dapat kembali ke halaman login untuk masuk dengan akun yang baru dibuat.

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

The image shows a user registration form titled "Daftar Akun" (Register Account). Below the title is a subtitle "Buat akun baru untuk memulai" (Create a new account to get started). The form contains several input fields: "Nama Lengkap" (Full Name) with a placeholder "Masukkan nama lengkap"; "Email" with a placeholder "nama@email.com" and a red error message "Email tidak valid" below it; "Nomor Telepon \*" (Phone Number) with a country code dropdown set to "+62" and a placeholder "812xxxxxxx"; "Password" with a placeholder "Minimal 8 karakter (huruf besar, kecil, angka)" and a red error message "Password minimal 2 karakter." below it; and "Konfirmasi Password" with a placeholder "Ulangi password". A blue button labeled "Daftar Sekarang" (Register Now) is positioned below the password fields. Below this button is a horizontal line with the text "ATAU DAFTAR DENGAN" (OR REGISTER WITH). Underneath is a button with the Google logo and the text "Google". At the bottom of the form is a link that says "Sudah punya akun? Masuk Sekarang" (Already have an account? Log in Now).

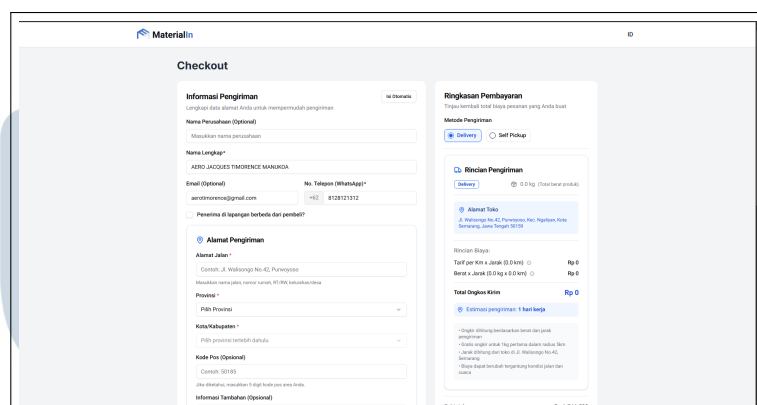
Gambar 3.16. Tampilan halaman pendaftaran untuk User

Halaman keranjang belanja pada gambar 3.17 ini dapat diakses melalui ikon keranjang terletak di samping *profile user*. Halaman keranjang belanja ini berfungsi untuk *user* melihat list produk yang sudah mereka tambahkan ke dalam keranjang belanja. *User* dapat melihat total, kuantitas, dan juga *user* dapat melakukan *bidding* (penawaran) harga pada setiap produk.



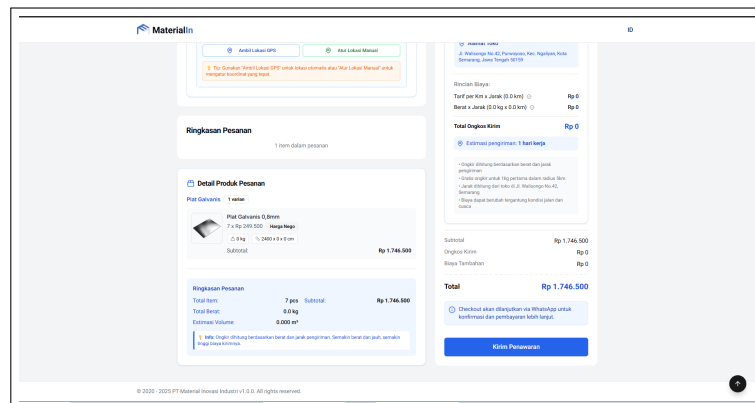
Gambar 3.17. Tampilan keranjang belanja untuk User

Setelah user sudah selesai menambahkan produk ke keranjang belanja dan ingin melakukan *checkout* untuk mengajukan penawaran, maka user akan langsung diarahkan ke *page form checkout* yang terlihat pada gambar 3.18 dan 3.19 untuk mengisi data-data yang diperlukan untuk pengiriman. Di halaman *checkout* ini juga, *user* dapat melihat kalkulasi ongkir berdasarkan berat dan jarak pengiriman. Setelah semua data-data diisi, user dapat kirim penawaran dan akan menerima detail pesanan juga melalui Whatsapp.



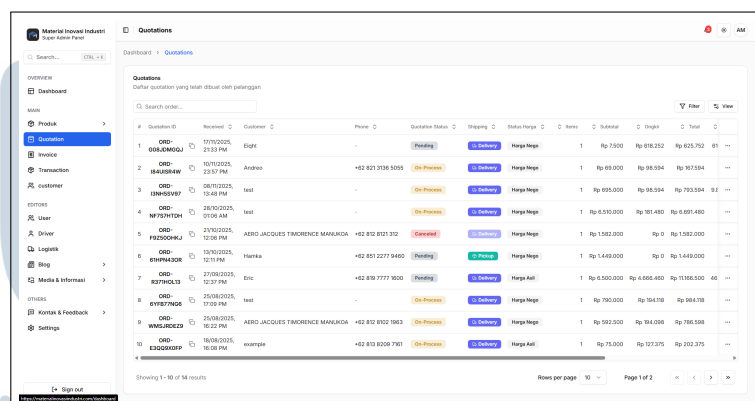
Gambar 3.18. Tampilan *Checkout* untuk User



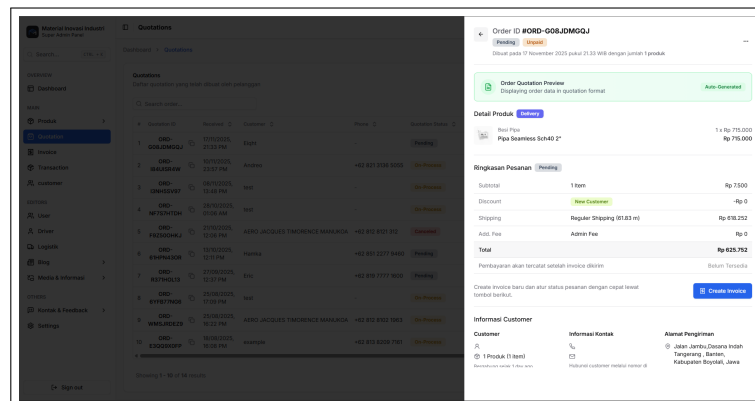


Gambar 3.19. Tampilan *Checkout* untuk User

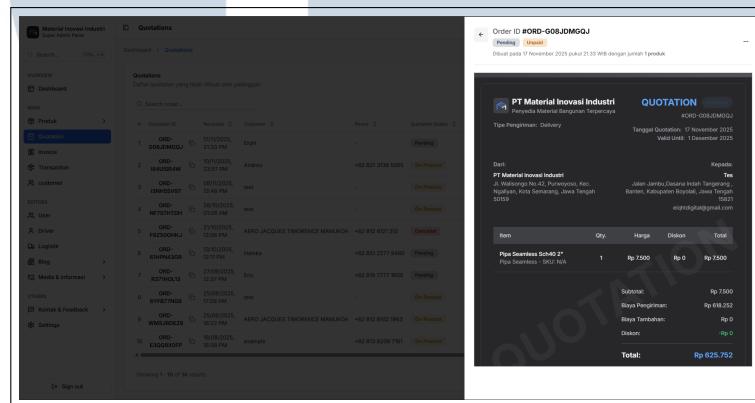
Pada halaman *Quotation* di gambar 3.20, admin dapat melihat list *Quotation* penawaran yang masuk dari user. Terdapat *Quotation* status apakah masih pending atau on-process, kemudian status harga nego atau harga awal, lalu sub-total, ongkir, dan ID *Quotation*. Untuk melihat detail penawaran, admin bisa klik pada ID *Quotation* dan akan memunculkan side panel seperti di gambar 3.21. Kemudian terdapat juga preview PDF untuk *Quotation* seperti pada gambar 3.22. Admin bisa mengkonfirmasi penawaran yang cocok dengan klik *Confirm* seperti pada gambar 3.23.



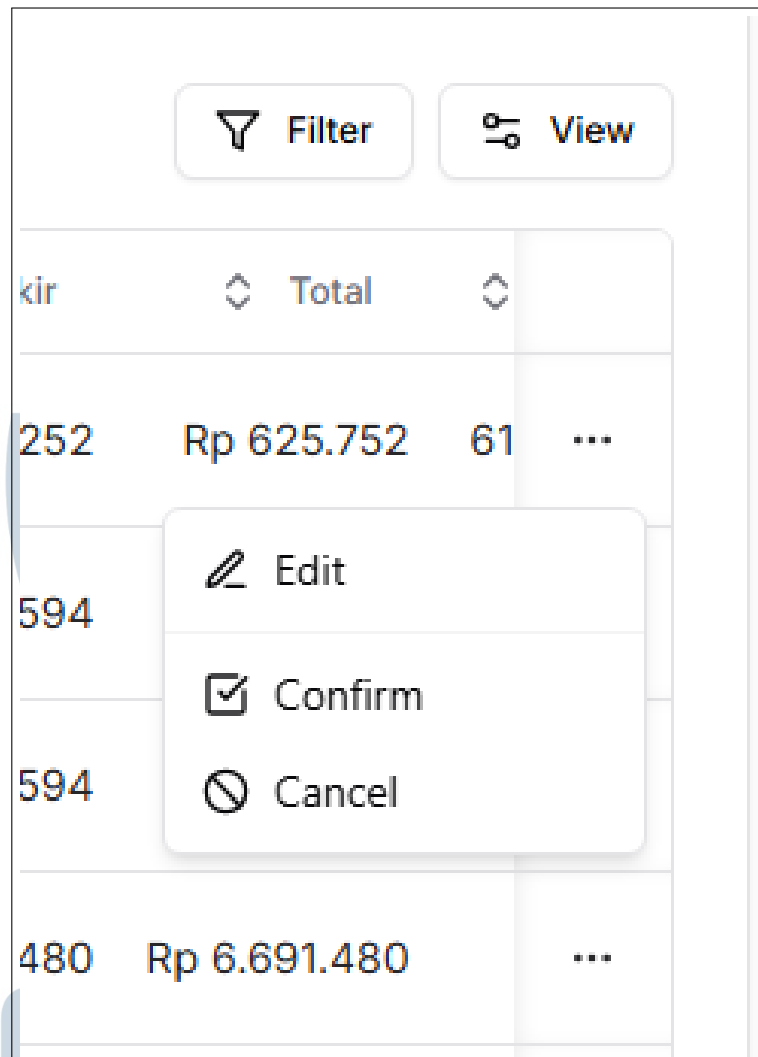
Gambar 3.20. Tampilan halaman *Quotation* untuk Admin



Gambar 3.21. Tampilan halaman *Quotation* untuk Admin

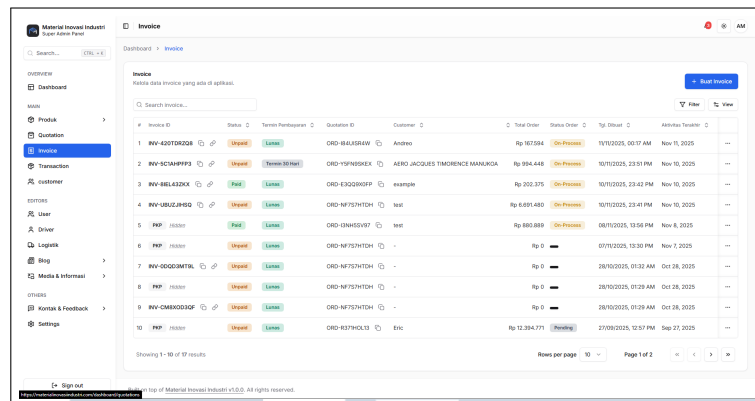


Gambar 3.22. Tampilan halaman *Quotation* untuk Admin

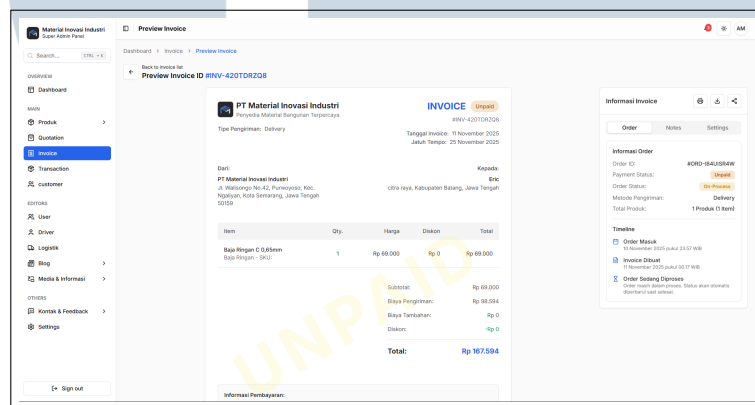


Gambar 3.23. Tampilan halaman *Quotation* untuk Admin

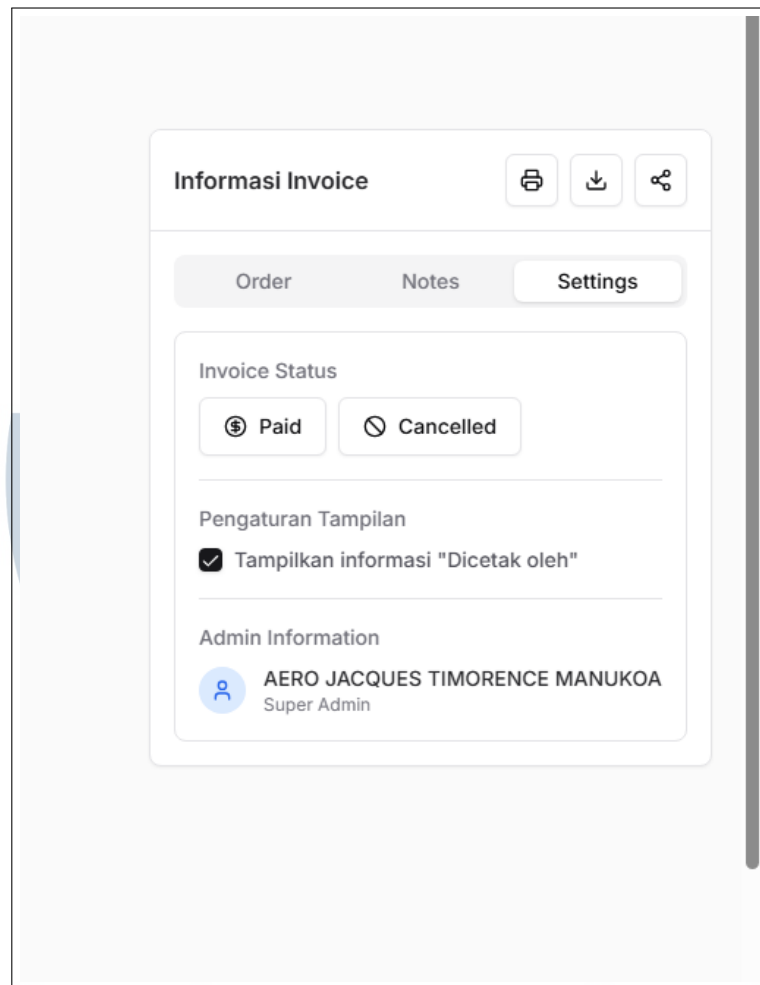
Tampilan halaman pada gambar 3.24 ini menampilkan data-data *invoice* yang sudah digenerate dari quotation yang sudah dikonfirmasi oleh *admin*. Dapat dilihat pada gambar 3.25 merupakan contoh *PDF Invoice* yang sudah di generate. *PDF* ini juga dikirimkan melalui Whatsapp sebagai notifikasi untuk *user*. Di tampilan *preview invoice* ini juga terdapat opsi untuk mengubah status pembayaran *invoice* apakah lunas atau tidak jadi (*paid* atau *cancelled*) terlihat pada gambar 3.26. Kemudian jika ingin membuat *invoice* baru yang perlu diupdate, admin bisa membuat *invoice* dengan *ORDER-ID* yang tersedia dan mengisi catatan serta memilih termin pembayaran jika diperlukan seperti pada gambar 3.27.



Gambar 3.24. Tampilan halaman *Invoice* untuk Admin

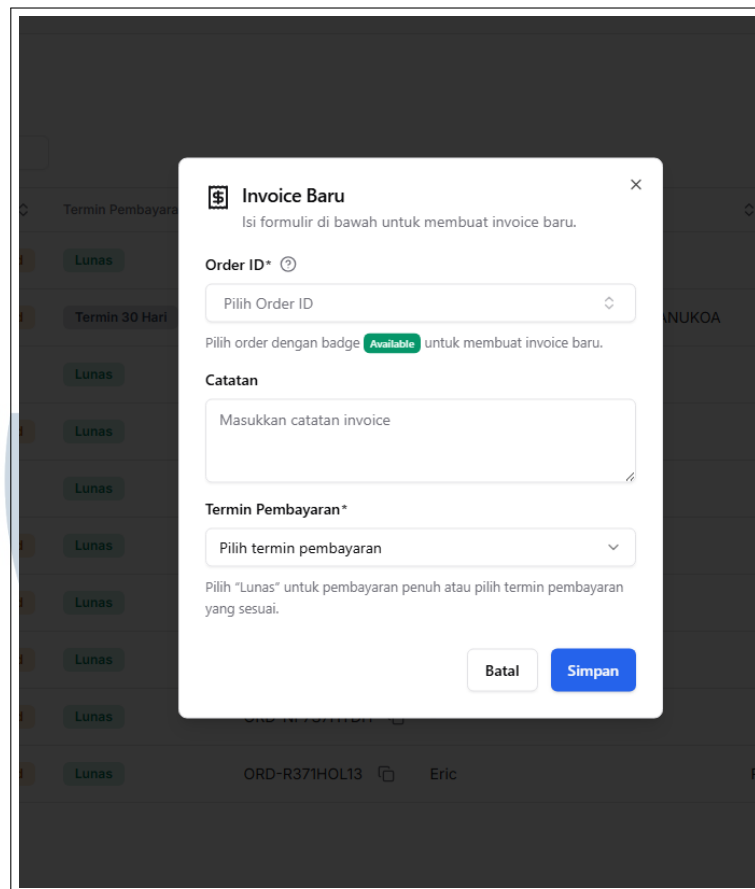


Gambar 3.25. Tampilan halaman *Invoice* untuk Admin



Gambar 3.26. Tampilan halaman *Invoice* untuk *Admin*

UMIN  
UNIVERSITAS  
MULTIMEDIA  
NUSANTARA



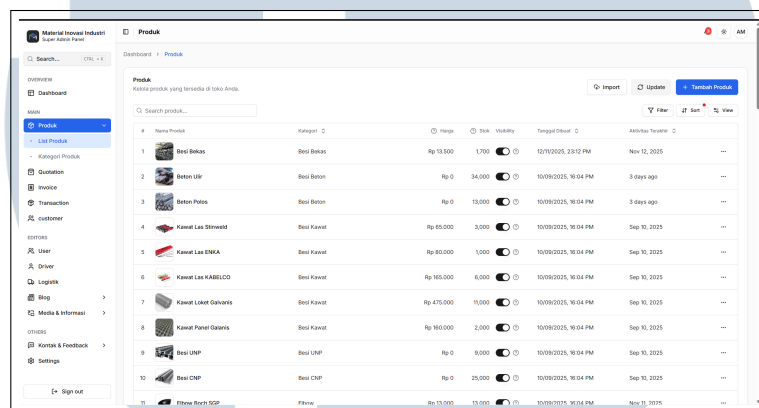
Gambar 3.27. Tampilan halaman *Invoice* untuk *Admin*

Halaman pada gambar 3.28 merupakan bagian dari sistem manajemen produk yang dirancang dengan antarmuka sidebar untuk navigasi yang terstruktur. Pengguna dapat mengakses berbagai modul seperti Dashboard, Product, Quotation, Invoice, dan lainnya melalui menu yang terorganisir dalam kategori *OVERVIEW*, *EDITORS*, dan *OTHERS*. Pada bagian utama, ditampilkan daftar produk dalam format tabel yang memuat informasi seperti nama produk, kategori, harga, stok, visibilitas, tanggal dibuat, dan riwayat perubahan. Pengguna dapat mencari produk tertentu melalui fitur pencarian di bagian atas, serta melihat produk berdasarkan kategori yang telah dikelompokkan. Tampilan ini memungkinkan pengguna untuk mengelola inventaris produk dengan mudah dan cepat melalui panel administrasi yang terpusat.

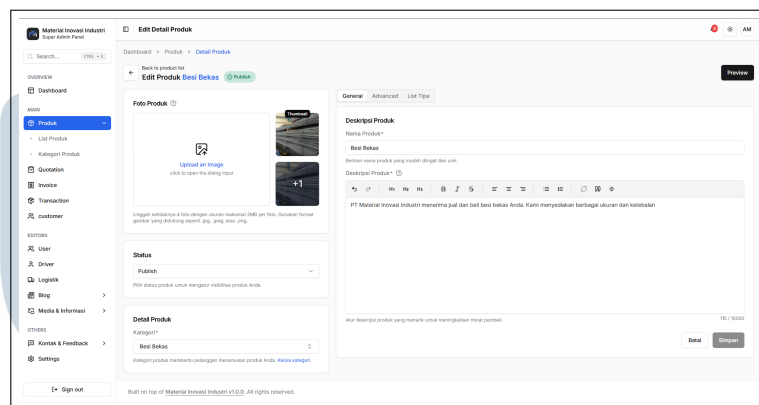
Kemudian pada gambar 3.29, fitur edit produk pada halaman ini memungkinkan pengguna untuk mengubah data produk secara lengkap, termasuk gambar, nama, stok, kategori, dan deskripsi. Pengguna dapat mengakses fitur ini melalui tombol atau opsi edit yang tersedia pada setiap baris produk di tabel.



Setelah diklik, akan muncul formulir atau modal yang berisi data produk yang dapat diperbarui, seperti mengganti gambar utama, memperbarui harga dan jumlah stok, mengubah kategori produk, serta menyunting deskripsi untuk informasi yang lebih akurat. Perubahan yang disimpan akan langsung diperbarui dalam sistem dan tercatat dalam kolom "Akhir Diubah" untuk pelacakan riwayat edit. Fitur ini dirancang untuk memudahkan pengelolaan produk tanpa perlu menghapus dan menambahkan data baru secara manual.



Gambar 3.28. Tampilan halaman *List Product* untuk Admin



Gambar 3.29. Tampilan halaman *Edit Product* untuk Admin

### 3.4 Tugas yang Dilakukan

#### 3.4.1 Pembuatan Formulir Data Pelanggan

Mengembangkan formulir data pelanggan (*customer form*) untuk mengumpulkan informasi penting dari pengguna, seperti nama lengkap, alamat pengiriman, nomor telepon, serta metode pembayaran. Formulir ini dilengkapi

dengan validasi input untuk memastikan keakuratan dan kelengkapan data sebelum dikirim ke server.

#### **3.4.2 Implementasi Sistem Autentikasi Pengguna**

Dalam aspek keamanan sistem, ada fitur yang diimplementasikan untuk autentikasi pengguna yang mencakup proses *registrasi* dan *login*. Sistem ini menggunakan metode *JSON Web Token* (JWT) untuk otorisasi pengguna dan pengelolaan sesi secara aman. Fitur autentikasi ini bertujuan membatasi akses terhadap halaman atau fungsi tertentu, seperti transaksi dan pengelolaan data akun.

#### **3.4.3 Pengembangan Dashboard Admin untuk Quotation dan Invoice**

Pada sisi administrator, dikembangkan sebuah modul *dashboard admin* yang berfungsi sebagai pusat pengelolaan data pesanan. Modul ini menampilkan daftar *quotation* (penawaran harga) serta *invoice* (tagihan) yang dihasilkan dari proses transaksi pengguna. Melalui fitur ini, admin dapat memantau seluruh aktivitas pemesanan, melakukan verifikasi, serta mencetak dokumen transaksi secara otomatis.

#### **3.4.4 Implementasi Sistem Bidding Harga Barang**

Mengembangkan sistem *bidding* atau penawaran harga yang memungkinkan pengguna melakukan penawaran terhadap produk tertentu dengan kenaikan minimum sebesar Rp500 setiap kali penawaran dilakukan. Sistem ini dikembangkan secara *real-time* menggunakan mekanisme *server-side rendering* (SSR) dari *Next.js 13*, sehingga setiap perubahan harga dapat langsung terlihat oleh seluruh pengguna yang berpartisipasi dalam proses *bidding*. Fitur ini diharapkan dapat meningkatkan interaksi pengguna sekaligus menambah nilai kompetitif dalam proses jual beli produk.

#### **3.4.5 Pengembangan Sistem Impor Produk melalui File Excel**

Selain fitur transaksi dan autentikasi, dikembangkan juga sistem untuk melakukan *import* data produk secara massal menggunakan file Excel (.xlsx). Fitur ini memungkinkan pihak admin untuk menambahkan atau memperbarui daftar produk dengan cara mengunggah file Excel yang berisi informasi produk seperti

nama barang, kode produk, harga, stok, dan deskripsi. Sistem ini memanfaatkan pustaka *xlsx* pada *Node.js* untuk membaca dan memproses data dari file, kemudian menyimpannya secara otomatis ke dalam basis data. Dengan adanya fitur ini, proses manajemen inventori menjadi lebih efisien, mengurangi potensi kesalahan input manual, serta mempercepat pembaruan data produk dalam jumlah besar.

### 3.5 Perangkat dan Software yang Digunakan

#### 3.5.1 Perangkat Keras (Hardware)

Perangkat keras yang digunakan selama proses pengembangan memiliki spesifikasi yang mampu menunjang aktivitas pemrograman dan pengujian sistem berbasis web. Adapun spesifikasi perangkat keras yang digunakan adalah sebagai berikut:

- Unit Komputer/Laptop dengan spesifikasi:
  - Prosesor: Intel Core i5 (generasi ke-10) atau setara
  - Memori (RAM): 8 GB
  - Penyimpanan: Solid State Drive (SSD) 512 GB
  - Sistem Operasi: Windows 11 (64-bit)
- Perangkat jaringan berupa koneksi internet berkecepatan stabil untuk mendukung proses instalasi dependensi, pembaruan library, serta pengujian aplikasi secara daring.

#### 3.5.2 Perangkat Lunak (Software)

Perangkat lunak yang digunakan berperan penting dalam menunjang seluruh tahapan pengembangan sistem, mulai dari perancangan antarmuka, pengkodean, manajemen versi, hingga penyusunan lingkungan produksi. Adapun perangkat lunak yang digunakan adalah sebagai berikut:

- Visual Studio Code, sebagai *Integrated Development Environment* (IDE) utama dalam proses penulisan dan pengelolaan kode sumber aplikasi.
- Node.js, digunakan sebagai *runtime environment* yang memungkinkan eksekusi bahasa pemrograman JavaScript pada sisi server.

- Next.js 13, berperan sebagai *framework* utama pengembangan aplikasi berbasis React, dengan dukungan *server-side rendering* (SSR) dan *static site generation* (SSG) untuk meningkatkan performa serta optimasi mesin pencari.
- Docker, dimanfaatkan sebagai platform untuk melakukan kontainerisasi aplikasi, sehingga lingkungan pengembangan dan produksi dapat berjalan secara konsisten dan terisolasi.
- Git, digunakan sebagai sistem kontrol versi untuk mencatat perubahan kode dan mendukung kolaborasi dalam pengembangan perangkat lunak.
- Google Chrome, digunakan sebagai peramban utama untuk melakukan pengujian, *debugging*, serta evaluasi antarmuka pengguna (UI) dari website yang dikembangkan.

### 3.5.3 Integrasi Sistem

Integrasi antara *Next.js 13*, *Node.js*, dan *Docker* memberikan fleksibilitas tinggi dalam pengembangan aplikasi berbasis web. Kombinasi ini memungkinkan proses pengembangan yang modular, efisien, serta memudahkan penerapan prinsip *continuous integration and deployment* (CI/CD). Dengan demikian, sistem yang dihasilkan dapat diimplementasikan secara konsisten pada berbagai lingkungan tanpa mengalami kendala kompatibilitas.

## 3.6 Metode Pengujian Sistem

Dalam proses pengembangan website e-commerce toko besi, pengujian dilakukan menggunakan dua pendekatan utama, yaitu *blackbox testing* dan *whitebox testing*. Kedua metode ini diterapkan untuk memastikan bahwa sistem berjalan sesuai kebutuhan pengguna, memiliki fungsionalitas yang stabil, serta memiliki struktur logika internal yang benar.

### 3.6.1 Blackbox Testing

*Blackbox testing* digunakan untuk menguji fungsionalitas sistem tanpa melihat struktur kode program. Fokus pengujian berada pada hubungan antara input

dan output, dengan tujuan memastikan bahwa setiap fitur menghasilkan keluaran sesuai dengan yang diharapkan.

Pada proyek website e-commerce ini, *blackbox testing* dilakukan pada beberapa fitur berikut:

- Registrasi dan Login Pengguna: Pengujian dilakukan dengan mencoba berbagai kombinasi input valid dan invalid untuk memastikan sistem memberikan respons yang sesuai.
- Proses Checkout dan Transaksi: Pengujian mencakup penambahan produk ke keranjang, perhitungan total harga, pemilihan metode pembayaran, hingga proses pengiriman pesanan.
- Manajemen Produk pada Admin Dashboard: Memastikan fitur tambah, edit, dan hapus produk berjalan sesuai skenario.
- Pencarian dan Filter Produk: Menguji apakah hasil pencarian dan filter menampilkan produk yang relevan dan sesuai kriteria.

Melalui pengujian ini, sistem dapat divalidasi dari sudut pandang pengguna akhir sehingga seluruh fungsi utama dapat dipastikan berjalan sesuai kebutuhan.

### 3.6.2 Whitebox Testing

*Whitebox testing* diterapkan untuk memeriksa logika internal sistem, terutama pada bagian backend yang menangani proses autentikasi, transaksi, serta pengelolaan data pengguna dan produk. Pendekatan ini memastikan bahwa alur kontrol dan struktur kode berjalan sesuai desain.

Pengujian dilakukan pada beberapa aspek berikut:

- Autentikasi Pengguna: Memastikan proses validasi input, verifikasi token, serta middleware proteksi halaman berjalan dengan benar.
- Unit Test pada Fungsi Perhitungan Transaksi: Menguji kalkulasi total harga, ongkos kirim, dan pajak pada berbagai kondisi input.
- Pengujian Alur Kontrol pada API Endpoint: Memverifikasi bahwa setiap *endpoint* mengeksekusi cabang logika (*branch*) secara benar, termasuk validasi data, penanganan error, dan respons yang dikembalikan.

Dengan adanya *whitebox testing*, stabilitas dan keamanan sistem dapat dipastikan tidak hanya pada tingkat tampilan, tetapi juga pada struktur kode program.

### 3.7 Kendala dan Solusi yang Ditemukan

#### 1. Ketiadaan Dokumentasi dari Developer Sebelumnya

Salah satu kendala utama yang dihadapi adalah tidak tersedianya dokumentasi proyek dari pengembang sebelumnya. Hal ini menyebabkan kesulitan dalam memahami struktur kode, alur kerja sistem, serta dependensi yang digunakan pada proyek. Solusi: Untuk mengatasi permasalahan ini, dilakukan proses *code review* secara menyeluruh guna mempelajari fungsi-fungsi penting dan relasi antar modul dalam sistem. Selain itu, dokumentasi internal mulai disusun menggunakan format *README* dan *Notion workspace* agar dapat dijadikan acuan bagi tim pengembang berikutnya.

#### 2. Penambahan Fitur yang Memerlukan Perombakan Sistem

Fitur-fitur baru yang ditambahkan ke dalam proyek, seperti sistem pemesanan dan pengelolaan *inventory*, membutuhkan perubahan besar baik di sisi *backend* maupun *frontend*. Hal ini menimbulkan kendala kompatibilitas dengan struktur kode yang sudah ada sebelumnya. Solusi: Untuk mengatasi hal tersebut, dilakukan analisis struktur data dan *API endpoint* yang sudah ada sebelum implementasi fitur baru. Proses refaktorisasi dilakukan secara bertahap agar tidak mengganggu modul lain. Setiap pembaruan juga diuji melalui *staging environment* sebelum diimplementasikan ke sistem utama.

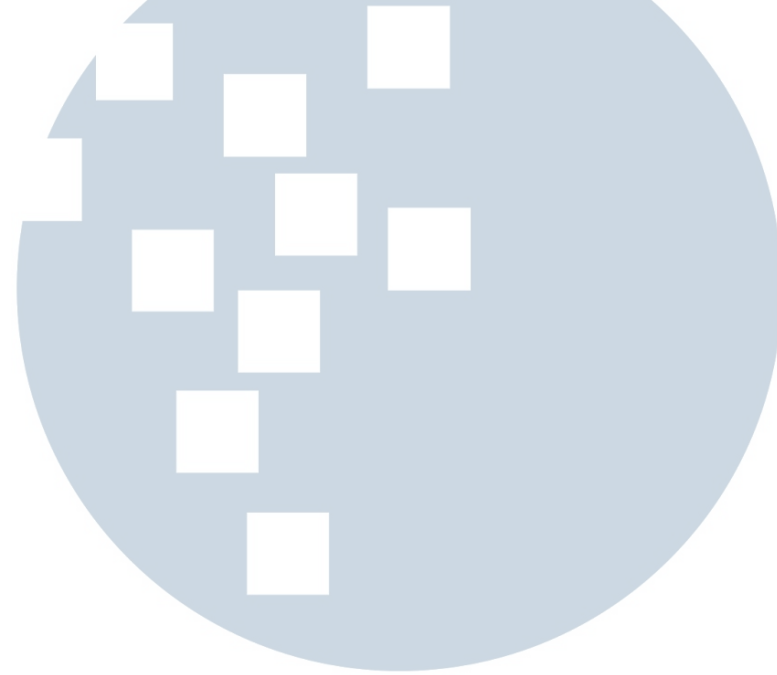
#### 3. Kesulitan dalam Penggunaan Lingkungan *Docker*

Lingkungan pengembangan proyek menggunakan *Docker*, yang pada awalnya menimbulkan kendala karena kompleksitas konfigurasi dan dependensi antar *container*. Hal ini menyebabkan keterlambatan dalam proses inisialisasi *server* dan migrasi basis data. Solusi: Untuk mengatasi kendala tersebut, dilakukan pembelajaran mandiri dan bimbingan langsung dari Supervisor IT mengenai penggunaan *Docker*. Selain itu, disusun beberapa skrip otomatis (*Docker command scripts*) untuk mempermudah proses *build*, *restart server*, serta migrasi basis data. Dengan cara ini, efisiensi kerja tim meningkat dan risiko kesalahan konfigurasi dapat diminimalkan.

Secara keseluruhan, kendala-kendala yang muncul selama masa magang



berhasil diselesaikan melalui komunikasi aktif dengan tim pengembang, pendampingan dari Supervisor IT, serta inisiatif dalam melakukan riset dan dokumentasi mandiri. Proses ini memberikan pengalaman berharga dalam pengelolaan proyek berbasis kolaboratif serta pengembangan sistem dengan standar industri.



UMN  
UNIVERSITAS  
MULTIMEDIA  
NUSANTARA