UNIVERSITAS ESA UNGGUL

PROJECT PEMROGRAMAN *MOBILE* SISTEM *MONITORING SALES* BERBASIS *MOBILE* PT CHEMVIRO BUANA INDONESIA

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memenuhi Penilaian Semester Pada Mata Kuliah Pemrograman Mobile

Disusun Oleh:
Ayumi Permana 20220801127

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS ESA UNGGUL
TAHUN 2024/2025

KATA PENGANTAR

Puji Syukur kehadirat Allah SWT yang telah memberikan rahmat, taufik, dan hidayah-Nya, sehingga penulis dapat menyelesaikan laporan tugas proyek ini dengan judul "Project Pemrograman *Mobile* Sistem *Monitoring Sales* Berbasis *Mobile* PT Chemviro Buana Indonesia." Laporan ini disusun sebagai salah satu syarat untuk memenuhi penilaian pada maya kuliah Pemrograman Mobile di Program Studi Teknik Informayika. Penulis menyadari bahwa laporan ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Jefry Sunupurwa Asri, S.Kom., M.Kom selaku Dosen Pengampu Mata Kuliah Pemrograman Mobile.
2. Semua Pihak yang telah berkontribusi dalam mendukung terselesaikannya tugas ini.

Penulis menyadari bahwa laporan ini masih memiliki kekurangan dan jauh dari sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun demi perbaikan di masa depan. Semoga laporan ini dapat memberikan manfaat, baik bagi penulis sendiri maupun pembaca, khususnya dalam pengembangan ilmu di bidang teknologi informasi.

Tangerang, Januari 2025

ABSTRAK

Aplikasi *Mobile* dirancang untuk meningkatkan efisiensi operasional tim *sales* PT Chemviro Buana Indonesia. Dengan fitur utama seperti pencatatan pesanan *real-time*, pelacakan status, dan integrasi dengan sistem CRM perusahaan, aplikasi ini memberikan solusi praktis untuk mengatasi tantangan penjualan konvensional. Proses pengembangan aplikasi melibatkan analisis kebutuhan, desain sistem, dan implementasi fitur utama yang memungkinkan pengguna untuk mencatat pesanan, memantau status pesanan, serta menghasilkan laporan kinerja secara *real-time.* Hasil uji coba menunjukkan peningkatan signifikan dalam efisiensi pencatatan pesanan dan transparansi operasional. Dengan demikian, aplikasi ini tidak hanya meningkatkan produktivitas tim *sales* tetapi juga mendukung pertumbuhan perusahaan melalui layanan yang lebih terstruktur dan responsif.

Kata kunci: Aplikasi *mobile,* efisiensi operasional, pencatatan pesanan, pelacakan status, CRM.

ABSTRACT

The Mobile application is designed to enhance the operational efficiency of the sales team at PT Chemviro Buana Indonesia. With key features such as real-time order recording, status tracking, and integration with the company's CRM system, this application provides practical solutions to address the challenges of conventional sales processes. The application development process involved needs analysis, system design, and the implementation of key features that enable users to record orders, monitor order statuses, and generate real-time performance reports. Trial results show a significant improvement in order recording efficiency and operational transparency. Thus, this application not only enhances the productivity of the sales team but also supports company growth through more structured and responsive services.

Keywords: Mobile application, operational efficiency, order recording, status tracking, CRM.

DAFTAR ISI

KATA PENGANTAR ..... **ii**
ABSTRAK ..... **iii**
DAFTAR ISI ..... **iv**
DAFTAR GAMBAR ..... **vii**
DAFTAR SIMBOL ..... **viii**
BAB 1 PENDAHULUAN **9**
1.1 Latar Belakang 9
1.2 Identifikasi Masalah 11
1.3 Tujuan Tugas Akhir 11
1.4 Manfaat Tugas Akhir 12
1.5 Lingkup Tugas Akhir 12
1.6 Kerangka Berpikir 12
1.7 Sistematika Penulisan Tugas Akhir 13
BAB 2 TINJAUAN PUSTAKA **14**
2.1 Landasan Teori 14
2.1.1 *Sales Order* 14
2.1.2 *Purchase Order* 14
2.1.3 *Cancel Order* 14
2.1.4 *Customer Relathionship Management* 14
2.1.5 *Sales Monitoring* 15
2.1.6 Sistem Berbasis *Mobile* 15
2.1.7 *Framework Flutter* 16
2.1.8 *Framework Laravel* 16
2.1.9 *REST API* 16
2.1.10 *Docker Server* 17
2.1.11 Arsitektur Program 18
2.1.12 Metode *Agile* 18
2.1.13 *Fishbone* 19
BAB 3 METODE PENELITIAN **21**
3.1 Rencana Penelitian 21
3.1.1 Metode *Agile* 21

3.1.2 *Fishbone* 21
3.1.3 Arsitektur Program 22
3.2 Objek Penelitian 22
3.3 Teknik Pengumpulan Data 22
BAB 4 HASIL DAN PEMBAHASAN **24**
4.1 Proses Bisnis 24
4.1.1 Analisis Bisnis 24
4.1.1.1 *Sales* 24
4.1.1.2 *Laboratorium* 25
4.1.1 *Finance* 26
4.1.2 Analisis *Fishbone* 26
4.1.3 *Flowchart* Alur Bisnis 28
4.2 Alur Kerja 30
4.2.1 Kebutuhan Sistem *Mobile Sales Management* 30
4.2.2 Analisis Fungsionalitas Sistem 31
4.2.3 Konfigurasi *Rute API* 31
4.2.4 *Flowchart* Aplikasi 35
4.3 Proses Bisnis 36
4.3.1 Tampilan Awal *(Welcome Screen)* *36*
4.3.2 Tampilan Login *(Login Screen)* 37
4.3.3 Tampilan Utama (Home Screen) 37
4.3.4 Tampilan *Create Order (Create Order Screen)* 38
4.3.5 Tampilan *Create Order Detail (Create Order Screen)* 39
4.3.6 Tampilan *Side Bar* 39
4.3.7 Tampilan Profil *(Profile Screen)* 40
4.3.8 Tampilan *Sales Order (Sales Order Screen)* 41
4.3.9 Tampilan *Sales Order Detail (Sales Order Screen)* 42
4.3.10 Tampilan *Edit Order (Edit Order Screen)* 43
4.3.11 Tampilan *Purchase Order (Purchase Order Screen)* 44
4.3.12 Tampilan *Purchase Order Detail (Purchase Order Screen)* 45
4.3.13 Tampilan *Cancel Order (Cancel Order Screen)* 46
4.3.14 Tampilan *Cancel Order Detail (Cancel Order Screen)* 47
4.3.15 Tampilan *Client (Client Screen)* 48

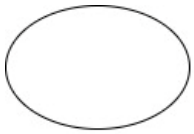
4.3.16 Tampilan *Create Client (Create Client Screen)* 49
4.3.17 Tampilan PDF (SO, PO, CO) 50
BAB 5 KESIMPULAN **52**
DAFTAR REFERENSI **53**

DAFTAR GAMBAR

Gambar 2. 1 *Diagram Fishbone* 19
Gambar 4. 1 *Fishbone Diagram* 28
Gambar 4. 2 Flow *SO to PO Submission* 28
Gambar 4. 3 Keterangan *Flow SO to Po Submission* 29
Gambar 4. 4 *Flow Order Execution* 29
Gambar 4. 5 Keterangan *Flow Order Execution* 30
Gambar 4. 6 Konfigurasi Rute API 32
Gambar 4. 7 Konfigurasi Rute API 32
Gambar 4. 8 Konfigurasi Rute API 33
Gambar 4. 9 Konfigurasi Rute API 33
Gambar 4. 10 Konfigurasi Rute API 34
Gambar 4. 11 Flowchart Aplikasi 35
Gambar 4. 12 Tampilan *Welcome Screen* 36
Gambar 4. 13 Tampilan *Login (Login Screen)* 37
Gambar 4. 14 Tampilan Utama *(Home Screen)* 37
Gambar 4. 15 Tampilan *Create Order (Create Order Screen)* 38
Gambar 4. 16 Tampilan *Create Order (Create Order Screen)* 39
Gambar 4. 17 Tampilan *Side Bar* 39
Gambar 4. 18 Tampilan Profil (*Profile Screen*) 40
Gambar 4. 19 Tampilan *Sales Order (Sales Order Screen)* 41
Gambar 4. 20 Tampilan *Sales Order Detail (Sales Order Detail Screen)* 42
Gambar 4. 21 Tampilan *Sales Order Detail (Sales Order Detail Screen)* 42
Gambar 4. 22 Tampilan *Edit Order Screen* 43
Gambar 4. 23 Tampilan *Edit Order Screen* 44
Gambar 4. 24 Tampilan *Purchase Order Detail* 45
Gambar 4. 25 Tampilan *Purchase Order Screen* 45
Gambar 4. 26 Tampilan *Cancel Order* 46
Gambar 4. 27 Tampilan *Cancel Order Detail* 47
Gambar 4. 28 Tampilan *Cancel Order Detail* 48
Gambar 4. 29 Tampilan *Client Screen* 49
Gambar 4. 30 Tampilan *Create Client Screen* 50
Gambar 4. 31 Tampilan PDF Pesanan Penjualan 51

DAFTAR SIMBOL

1. *Terminator*

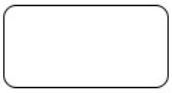


Menandakan awal (*Start*) atau (*End*) dari proses. Oval digunakan untuk menunjukkan titik awal dan akhir alur kerja.

2. *Processing*

Menunjukkan proses atau aktivitas tertentu

3. *Processing*

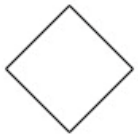


Simbol ini berfungsi untuk menggambarkan sumber informasi atau sistem eksternal yang memberikan kontribusi pada alur kerja dalam

4. *Input/Output*

Menunjukkan proses *input* atau *output*

5. *Decision*

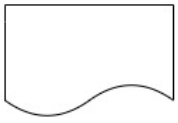


Menandakan proses pengambilan keputusan (*Decision Point*)

6. *Flow Line*

Menghubungkan simbol-simbol dalam *flowchart* untuk menunjukkan alur atau arah proses

7. *Document*



Menandakan dokumen fisik atau digital

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Dalam era digitalisasi yang berkembang pesat saat ini, transformasi digital menjadi sebuah keharusan bagi perusahaan yang ingin tetap kompetitif. Perusahaan dituntut untuk mampu mengelola operasionalnya secara efisien, cepat, dan akurat guna memenuhi kebutuhan pasar yang semakin dinamis. Dalam konteks ini, penggunaan teknologi digital tidak hanya sebagai alat bantu, tetapi juga sebagai solusi strategis untuk meningkatkan efisiensi operasional, pengambilan keputusan yang tepat waktu, serta mengoptimalkan sumber daya perusahaan. Salah satu kebutuhan penting yang muncul adalah adanya sistem terintegrasi yang mampu memonitor aktivitas operasional secara *real-time* dan komprehensif, sehingga perusahaan dapat merespons perubahan dengan lebih cepat dan akurat. Oleh karena itu, merencanakan perancangan sistem *monitoring* berbasis *mobile* menjadi langkah strategis untuk mendukung fleksibilitas dan kemudahan akses informasi, memungkinkan manajemen untuk memantau dan mengelola operasional kapan saja dan di mana saja.

Secara tradisional, banyak perusahaan yang masih mengandalkan sistem manual atau semi-manual, seperti penggunaan aplikasi *spreadsheet* sederhana seperti *Microsoft Excel*, untuk mengelola data operasional dan finansial mereka. Meskipun aplikasi ini bermanfaat dalam jangka pendek, mereka memiliki keterbatasan dalam hal skalabilitas, keakuratan, dan efisiensi, terutama bagi perusahaan dengan kompleksitas operasional yang tinggi. Proses manual seringkali membutuhkan banyak tenaga dan waktu, rentan terhadap kesalahan manusia, dan sulit untuk diintegrasikan dengan sistem lain yang digunakan oleh berbagai departemen. Ketergantungan pada sistem manual semacam ini dapat menghambat pertumbuhan perusahaan dalam jangka panjang, karena tidak mampu menyediakan data *real-time* yang dibutuhkan untuk pengambilan keputusan strategis.

Perancangan merupakan suatu proses yang bertujuan untuk menentukan informasi apa saja yang diperlukan dalam membangun sebuah sistem baru. Dalam tahapan perancangan sistem ini, diperlukan penyusunan gambaran menyeluruh mengenai rancang bangun sistem yang akan digunakan sebagai panduan utama oleh *software engineer*. Panduan ini memiliki peran penting untuk memastikan bahwa perancangan aplikasi dilakukan sesuai dengan komponen-komponen yang telah dirancang dalam sistem yang terkomputerisasi. Oleh karena itu, dalam tahap perancangan ini, hal-hal yang harus dirancang meliputi berbagai aspek seperti *equipment*, program *database*, dan aplikasi, yang semuanya saling mendukung untuk menciptakan sistem yang terintegrasi dan berfungsi secara optimal.

Menurut [1] "Perancangan adalah suatu proses untuk menganalisis, mengevaluasi, meningkatkan, dan mengembangkan suatu sistem, baik fisik maupun non fisik, yang dioptimalkan untuk masa depan dengan menggunakan informasi yang ada". Adapun pengertian perencanaan menurut [2] “Perencanaan bukan hanya sekadar suatu kegiatan teknis, namun suatu proses berkesinambungan dari mengamati, penyesuaian

untuk mengadakan perubahan serta proses belajar, yang mana harus dijaga keseimbangannya".

Sistem merupakan bagian-bagian komponen yang memiliki hubungan satu sama lain baik fisik, maupun non-fisik. yang bersama-sama dalam bekerja demi tujuan yang dituju secara harmonis Klik atau ketuk di sini untuk memasukkan teks.. *Monitoring* atau pengawasan adalah suatu aktivitas yang mencerminkan kesadaran terhadap hal yang ingin diketahui. Kegiatan ini melibatkan pengawasan tingkat tinggi yang memungkinkan dilakukannya pengukuran secara berkala dari waktu ke waktu, sehingga dapat menunjukkan kemajuan menuju target atau sebaliknya, menjauh dari target tersebut. Peneliti sebelumnya [4] mengatakan bahwa "*Monitoring* merupakan salah satu hal yang menjadi dasar yang penting di dalam kegiatan organisasi. dapat dikatakan *monitoring* adalah salah satu tolak ukur untuk menentukan terlaksana dan tercapainya suatu tujuan organisasi. *Monitoring* terdiri dari unsur, komponen, atau variabel yang terorganisir, yang saling berhubungan, dan keterkaitan satu sama lain dan ditujukan untuk mengumpilkan data dari berbagai sumber daya". Sistem *monitoring* adalah suatu aktivitas yang terjadi dalam suatu perangkat. salah satu kegunaan dari sistem *monitoring* adalah dapat mengetahui lebih awal kondisi dari perangkat apabila terjadi suatu masalah[5].

PT Chemviro Buana Indonesia, sebagai perusahaan yang bergerak di bidang laboratorium kalibrasi dan analisis kimia, merupakan salah satu perusahaan yang merasakan tantangan ini secara nyata. Perusahaan ini menghadapi kompleksitas dalam menjalankan operasional sehari-hari, khususnya pada *Departement Sales* yang bertanggung jawab untuk mengelola aktivitas penjualan. Proses yang melibatkan pencatatan order, riwayat order, dan perhitungan komisi penjualan *sales* sering kali dilakukan secara manual atau dengan sistem yang tidak terintegrasi. Hal ini menyebabkan kendala seperti kesalahan *input* data, keterlambatan dalam penyampaian informasi, serta sulitnya manajemen untuk memantau performa penjualan secara keseluruhan.

Ketidakmampuan untuk mengakses data secara *real-time* dan akurat menjadi masalah besar dalam konteks penjualan. Kesalahan dalam pencatatan order atau keterlambatan informasi dapat menyebabkan gangguan dalam alur penjualan, seperti kesalahan dalam pemenuhan pesanan atau keterlambatan dalam penghitungan pendapatan *sales*. Hambatan ini tidak hanya memengaruhi efisiensi operasional, tetapi juga berdampak pada kepuasan pelanggan dan kinerja *Departement Sales* secara keseluruhan.

Untuk mengatasi masalah ini, PT Chemviro Buana Indonesia perlu mengadopsi solusi teknologi yang lebih canggih dan terintegrasi. Salah satu solusi yang diusulkan adalah perancangan sistem *monitoring* berbasis *mobile* yang dirancang khusus untuk *Departement Sales*. Sistem ini memungkinkan tim *sales* untuk melakukan pencatatan order, melihat riwayat order, serta mengakses komisi penjualan secara *real-time*. Dengan fitur-fitur ini, setiap aktivitas penjualan dapat dikelola dengan lebih transparan, akurat, dan efisien.

Aplikasi *mobile* menawarkan berbagai keunggulan, seperti kemudahan penggunaan, akses data yang fleksibel kapan saja dan di mana saja, serta kemampuan untuk memberikan informasi terkini secara langsung kepada pengguna. Sistem *monitoring* berbasis *mobile* ini dirancang untuk memberikan kemudahan dalam mengakses data kapan saja dan di mana saja, sehingga setiap anggota tim *sales* dapat mengambil keputusan yang lebih baik berdasarkan data yang tersedia secara *real-time*. Selain itu, sistem ini juga memungkinkan manajemen untuk memantau performa penjualan secara keseluruhan, mengidentifikasi tren, dan mengambil langkah strategis untuk meningkatkan efektivitas penjualan. Pada akhirnya, adopsi sistem ini diharapkan dapat meningkatkan efisiensi, akurasi, dan kepuasan pelanggan, sekaligus membantu PT Chemviro Buana Indonesia dalam mencapai tujuan strategisnya dan bersaing di pasar yang semakin kompetitif.

1.2 Identifikasi Masalah

Terdapat beberapa permasalahan yang dihadapi oleh PT Chemviro Buana Indonesia terkait proses operasional dan penjualan, antara lain:

1. Proses pencatatan data yang masih dilakukan secara manual, sehingga membutuhkan waktu lebih lama dan rentan terhadap kesalahan.
2. Keterbatasan dalam mengakses data secara *real-time*, yang menghambat kelancaran operasional dan *summary* pendapatan *sales*.
3. Kurangnya integrasi antara departemen seperti Sales, Laboratorium, dan Finance, sehingga menyebabkan alur kerja menjadi tidak sinkron.
4. Kendala dalam memberikan pengalaman pelanggan yang optimal, terutama dalam menyediakan informasi dan layanan yang efisien.

1.3 Tujuan Tugas Akhir

Tujuan dari tugas akhir ini adalah untuk memberikan solusi terhadap berbagai permasalahan yang dihadapi PT Chemviro Buana Indonesia dalam proses operasionalnya. Beberapa tujuan yang ingin dicapai dalam tugas akhir ini meliputi:

1. Mengembangkan sistem *monitoring* berbasis *mobile* yang dirancang untuk menggantikan proses manual yang selama ini menggunakan Excel agar dapat terintegrasi secara *online*.
2. Mengoptimalkan proses operasional pada pemesanan, perubahan pesanan, dan *summary pendapatan* secara *online* yang langsung terintegrasi ke sistem perusahaan.
3. Mengurangi potensi kesalahan *input* data dan meminimalkan kendala komunikasi antar-departemen dengan menyediakan alur kerja yang terstruktur dan otomatis.
4. Meningkatkan pengalaman pengguna melalui fitur-fitur yang mempermudah akses dan pengelolaan data.

1.4 Manfaat Tugas Akhir

Manfaat yang diharapkan dari adanya sistem *monitoring* untuk PT Chemviro Buana Indonesia ini yaitu:

1. Meningkatkan efisiensi operasional perusahaan dengan mengurangi proses manual yang memakan waktu.
2. Mengoptimalkan proses kerja melalui integrasi data dan alur yang lebih sistematis.

3. Meningkatkan transparansi dan akurasi data sehingga mendukung pengambilan keputusan yang lebih tepat.
4. Meminimalkan kesalahan operasional sekaligus memperbaiki komunikasi antar-departemen.
5. Memberikan pengalaman pengguna yang lebih baik melalui antarmuka yang intuitif dan fitur yang relevan dengan kebutuhan.

1.5 Lingkup Tugas Akhir

Adapun lingkup dari Tugas Akhir yang dikerjakan adalah sebagai berikut:

1. Aplikasi yang dikembangkan berbasis *mobile* untuk mendukung aktivitas *sales*.
2. Pembangunan Aplikasi terintegrasi dengan sistem CRM perusahaan untuk efisiensi alur kerja.
3. Memfasilitasi pencatatan dan konversi pesanan dengan validasi otomatis.
4. Mendukung pelaporan performa *sales* secara *real-time* dan notifikasi otomatis.

1.6 Kerangka Berpikir

Kerangka berpikir dalam penelitian ini dirancang untuk memberikan alur pemikiran yang sistematis dan logis, mulai dari identifikasi masalah hingga penyusunan laporan penelitian. Proses penelitian ini melibatkan beberapa tahapan utama, yaitu:

1. Identifikasi Masalah
   Tahap ini dimulai dengan pengamatan terhadap proses kerja di PT Chemviro Buana Indonesia, khususnya pada aktivitas penjualan. Permasalahan yang diidentifikasi adalah adanya keterbatasan pada proses manual yang digunakan, seperti pencatatan data yang tidak terintegrasi, lambatnya akses informasi, dan inefisiensi operasional.
2. Studi Literatur
   Tahap ini melibatkan pengumpulan referensi terkait, baik dari jurnal, buku, maupun dokumen lain yang relevan dengan perancangan sistem penjualan berbasis *mobile*. Studi literatur membantu dalam memahami solusi yang telah diterapkan pada penelitian sebelumnya.
3. Pengumpulan Data
   Data dikumpulkan melalui observasi, dan studi dokumen untuk mengidentifikasi kebutuhan pengguna (*user requirement*) serta memahami proses bisnis yang sedang berjalan.
4. Analisis Data
   Data yang telah dikumpulkan dianalisis untuk menentukan permasalahan utama yang harus diatasi dan merancang solusi teknologi yang sesuai. Analisis ini menghasilkan gambaran kebutuhan sistem serta spesifikasi awal untuk sistem yang akan dikembangkan.
5. Perancangan Sistem
   Berdasarkan analisis data, sistem dirancang dengan mempertimbangkan integrasi antar-departemen, kemudahan penggunaan, serta kemampuan untuk meningkatkan efisiensi proses penjualan. Pada tahap ini, *prototype* sistem mulai dikembangkan.
6. Penyusunan Laporan

Tahapan terakhir adalah mendokumentasikan seluruh proses penelitian, mulai dari identifikasi masalah, analisis, hingga hasil akhir. Laporan ini disusun sebagai bentuk tanggung jawab akademik sekaligus panduan implementasi sistem di masa depan.

1.7 Sistematika Penulisan Tugas Akhir

Laporan tugas akhir ini dibuat untuk mendokumentasikan seluruh proses penelitian secara sistematis. Sistematika penulisan adalah sebagai berikut:

BAB 1 PENDAHULUAN

Bab ini berisi latar belakang, identifikasi masalah, tujuan penelitian, manfaat penelitian, ruang lingkup, kerangka berpikir, dan sistematika penulisan.

BAB 2 TINJAUAN PUSTAKA

Bab ini membahas teori-teori yang relevan, hasil penelitian terdahulu, dan landasan konseptual yang mendukung penelitian ini.

BAB 3 METODE PENELITIAN

Bab ini menjelaskan rencana penelitian, obyek penelitian, dan teknik pengumpulan data.

BAB 4 HASIL DAN PEMBAHASAN

Bab ini memaparkan hasil dari penelitian yang dilakukan, termasuk evaluasi sistem yang telah dirancang, serta pembahasan mendalam mengenai hasil tersebut.

BAB 5 KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dari penelitian, pencapaian tujuan, serta saran untuk pengembangan lebih lanjut.

BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 *Sales Order*

Sales Order adalah dokumen yang mencatat rincian produk atau layanan yang diminta oleh klien. Dokumen ini mencakup informasi penting seperti identitas klien, rincian produk, harga, dan syarat pembayaran. *Sales Order* berfungsi sebagai titik awal dalam proses pemantauan, di mana informasi mengenai permintaan klien dicatat secara sistematis [6]. *Sales Order* pada pesanan mengandung transkripsi asli yang berisi sumber data yang paling dapat diandalkan tentang rencana pemasaran dan efektivitas tenaga penjualan perusahaan [7] . Dengan adanya *Sales Order*, sistem *monitoring* dapat melacak dan mengelola setiap tahap dari pemenuhan pesanan, mulai dari pengajuan hingga eksekusi.

2.1.2 *Purchase Order*

Purchase Order melibatkan penyusunan rincian kebutuhan layanan atau produk yang diminta oleh klien, termasuk parameter-parameter teknis yang relevan. Jika diperlukan, dilakukan negosiasi terkait harga atau detail layanan hingga tercapai kesepakatan. Dokumen *Purchase Order* ini kemudian dikirimkan kepada klien untuk dilakukan peninjauan dan persetujuan. Setelah klien memberikan persetujuan, tahapan selanjutnya adalah eksekusi pesanan sesuai dengan dokumen PO yang telah disetujui. Laporan ini kemudian digunakan oleh divisi keuangan sebagai dasar untuk penyusunan faktur dan penyelesaian pembayaran dari klien [8]. Dokumen ini berfungsi sebagai kontrak formal antara pembeli dan penjual, mencakup rincian pesanan, syarat dan ketentuan, serta tanggal pengiriman. PO memastikan bahwa semua informasi terkait pesanan terintegrasi dan dapat diakses oleh semua departemen yang terlibat, sehingga memudahkan koordinasi dan pengawasan terhadap proses pemenuhan pesanan.

2.1.3 *Cancel Order*

Cancel digunakan untuk membatalkan pesanan yang sudah ada dan menggantinya dengan pesanan yang serupa. Pembatalan pesanan menghilangkan kemungkinan pengisian ganda pada pesanan yang sudah diubah [9]. *Sales Order (SO)* dapat dibatalkan oleh penjual jika pelanggan memutuskan untuk tidak melanjutkan transaksi sebelum dokumen *Purchase Order (PO)* dibuat. Pembatalan ini biasanya terjadi ketika klien mengubah keputusan mereka karena berbagai alasan, seperti kebutuhan yang berubah, ketidakcocokan dengan harga atau syarat, atau kendala lain yang menghalangi proses lebih lanjut.

2.1.4 *Customer Relathionship Management*

Customer Relationship Management (CRM) merupakan sebuah pendekatan yang digunakan untuk mengelola hubungan antara perusahaan dan pelanggan dengan tujuan meningkatkan kepuasan pelanggan dan mendukung pertumbuhan bisnis melalui pengumpulan, analisis dan penggunaan data pelanggan. *Customer Relationship Management* (CRM) memiliki tiga komponen utama yaitu *Operational* CRM yang berfokus pada proses bisnis seperti pemasaran, penjualan dan layanan kepada pelanggan, *Analytical* CRM berfokus pada pengumpulan dan analisis data pelanggan

guna mendukung pengambilan keputusan, dan *Collaborative* CRM yang berfokus untuk memfasilitasi komunikasi antara perusahaan dengan pelanggan melalui berbagai saluran seperti sosial media, email atau telepon [10].

Dengan menawarkan taktik yang berguna untuk mengelola interaksi pelanggan di semua ukuran, mulai dari perusahaan rintisan hingga konglomerat multinasional, penerapan CRM juga secara langsung memengaruhi efektivitas operasional perusahaan. CRM memainkan peran penting dalam meningkatkan profitabilitas bisnis selain meningkatkan pengalaman pelanggan. CRM memungkinkan bisnis untuk sepenuhnya memahami keinginan klien dan menawarkan solusi yang sesuai melalui proses yang terorganisir, membina hubungan pelanggan dengan perusahaan yang bertahan lama [11].

2.1.5 *Sales Monitoring*

Sistem *monitoring* adalah suatu mekanisme atau alat yang digunakan untuk mengawasi, mengelola, dan menganalisis aktivitas operasional dalam suatu organisasi. Tujuan utama dari sistem ini adalah untuk memastikan bahwa mampu menentukan kondisi perangkat sebelum terjadi masalah agar proses bisnis berjalan dengan efisien, efektif, dan sesuai dengan standar yang telah ditetapkan [5]. Untuk dengan cepat mendapatkan perspektif keseluruhan pada kebanyakan metrik kinerja penjualan, banyak perusahaan telah menggunakan *dashboard* penjualan yang memungkinkan *sales* untuk memonitoring atau menilai status penjualan sehari-hari dan hasil kinerja keuangan serta gambar ringkasan *up-to-date* pada kinerja penjualan maasing-masing individu *sales*. Manajer *sales* dan *salespeople* mereka bergantung pada analisis yang disediakan oleh *sales dashboard* dari berbagai jenis untuk memantau operasi penjualan sehari-hari serta aktivitas jangka panjang. Informasi yang ditampilkan pada *sales dashboard* mungkin berbeda secara luas tergantung pada industri atau perusahaan[12].

2.1.6 Sistem Berbasis Mobile

Perangkat *mobile* terdiri dari beberapa bagian, seperti prosesor, RAM, kartu grafis, dan berbagai sensor. Tampilan layar sentuh pada perangkat seluler menawarkan pengalaman pengguna yang lebih baik dalam hal *input* [13]. Perangkat seluler berfungsi sebagai alat komunikasi dan menawarkan pengalaman pengguna yang menarik. Teknologi ini memungkinkan pengguna bekerja lebih efektif, akurat, dan cerdas, dan meningkatkan komunikasi dan kerja tim. Kemajuan teknologi *mobile* telah memungkinkan aplikasi yang semakin memenuhi kebutuhan akan data dan akses aplikasi yang cepat. Pengguna saat ini terbiasa dengan perangkat canggih seperti ponsel pintar dan tablet, yang menawarkan mobilitas dan fleksibilitas [14]. Menjalankan aplikasi *mobile* merupakan salah satu tugas terpenting sistem operasi *mobile*. Sistem operasi *mobile* harus mengendalikan antarmuka pengguna dan komponen *middleware* agar dapat menjalankan aplikasi seluler pada perangkat keras. Tiga tingkatan membentuk struktur sebagian besar aplikasi seluler: tampilan, aplikasi, dan data. Antarmuka pengguna merupakan fokus utama dari tingkatan presentasi. Antarmuka pengguna yang dirancang dengan baik memudahkan pengguna untuk mengirimkan permintaan dan menerima hasil yang bermakna. Kalkulasi dan pengambilan keputusan yang logis merupakan tanggung jawab tingkatan aplikasi. Selain itu, ia menganalisis dan mentransfer data antara tingkatan data dan presentasi. Inti dari aplikasi adalah

tingkatan aplikasi. Lapisan data bertugas menyediakan alat dasar untuk memanipulasi dan menyimpan data [13].

2.1.7 *Framework Flutter*

Flutter merupakan kerangka kerja dengan basis kode tunggal yang memudahkan pembuatan aplikasi seluler yang berjalan di berbagai platform, seperti Android dan *iOS*. *Flutter* memungkinkan pengembang untuk berkonsentrasi pada persyaratan bisnis, seperti manajemen *input*, konektivitas internet, dan aplikasi secara *online*. *Google* menciptakan *Flutter* dan bahasa pemrograman *Dart* yang menjadi dasar. Keduanya bersifat *open source* dengan dukungan kuat dari *Google* [15]. Tersedia sejumlah *widget* praktis dan mudah digunakan di *Flutter*, yang memudahkan pembuatan aplikasi seluler. *Widget* ini mendukung animasi dan gestur. Jika *widget* diubah, perbandingan antara status sebelumnya dan saat ini dilakukan sebagai bagian dari pemrograman reaktif. Pemrograman reaktif adalah pondasi untuk membuat aplikasi seluler dengan *Flutter* [16]. Salah satu kelebihan *flutter* yang paling signifikan adalah arsitekturnya yang didesain sangat modular dan fleksibel, sehingga memudahkan *developer* untuk melakukan kustomisasi dan membangun aplikasi dengan cepat. Arsitektur *flutter* juga memudahkan untuk membangun aplikasi lintas *platform* karena didesain untuk bekerja di berbagai *platform* [17].

2.1.8 *Framework laravel*

Laravel adalah kerangka kerja pengembangan *web open-source* yang didasarkan pada pola arsitektur MVC *(Model, View, Controller)*. Ditulis dalam PHP, *Laravel* dirancang oleh Taylor Otwell pada tahun 2011. Konvensi lebih penting daripada konfigurasi sebagai prinsip utama yang digunakan *Laravel* dalam membangun platformnya. Pengkodean terkadang bisa menjadi proses yang tidak efisien karena pengembang akhirnya menggunakan solusi yang kurang ideal. *Laravel* menghilangkan dugaan dalam persamaan dengan membuat estimasi tentang tujuan akhir secara *real-time. Laravel* dapat membantu membuat kode yang lebih elegan dan efisien, kompatibilitas dengan berbagai basis data dan aplikasi [18]. *Laravel* mempunyai alasan yang kuat untuk membuat kemudahan para *developers. Laravel* adalah *framework* pengembangan aplikasi yang cepat dan berfokus pada kurva pembelajaran yang mudah. Komponen dalam *Laravel* menyediakan API yang konsisten dan struktur yang dapat diprediksi di seluruh *framework. Laravel* juga menyediakan seluruh ekosistem alat untuk membangun dan meluncurkan aplikasi. Selanjurnya, *Laravel* berfokus pada "konvensi daripada konfigurasi" - yang berarti bahwa jika ingin menggunakan *default Laravel*, harus melakukan lebih sedikit pekerjaan dibandingkan dengan *framework* lain yang mengharuskan untuk mendeklarasikan semua pengaturan dan konfigurasi yang direkomendasikan. Proyek yang dibangun di *Laravel* membutuhkan waktu lebih sedikit daripada yang dibangun di sebagian besar *framework* PHP lainnya [19].

2.1.9 *REST API*

Application Programming Interface (API) adalah memaparkan sekumpulan data dan fungsi untuk memfasilitasi interaksi antar program komputer. *Representational State Transfer* (REST) dijelaskan sebagai derivasi gaya arsitektur *web* oleh Roy Fielding. Sedangkan, REST API adalah antarmuka layanan *web* yang sesuai dengan gaya arsitektur *web*. Secara umum, API memaparkan serangkaian data dan fungsi untuk

memfasilitasi interaksi antara program komputer dan memungkinkan untuk bertukar informasi [20]. *Mobile apps* perlu terhubung ke *server* di *internet* agar dapat digunakan sepenuhnya atau setidaknya dapat digunakan secara maksimal, sebagian logika bisnis pada aplikasi dan logika pemrosesan tugas berat pada server di *cloud.* Data yang diambil pada perangkat *mobile* dikirim ke *server* melalui panggilan *API,* yang diteruskan ke *services* dan kemudian ke basis data. Data yang dikirimkan oleh API harus ringan, memastikan API dapat digunakan oleh perangkat dengan daya pemrosesan terbatas [21]. Pengembangan layanan *custom* REST API yang dirancang untuk memenuhi kebutuhan spesifik aplikasi. Proses implementasi meliputi pembuatan *endpoint* khusus, penyesuaian struktur data, serta pengaturan logika *backend* yang relevan. Integrasi REST API pada aplikasi *mobile* bertujuan untuk memastikan komunikasi data antara *frontend* dan *backend* agar dapat berjalan secara optimal. REST API memungkinkan aplikasi *mobile* untuk mengambil, mengolah, dan mengirim data dalam format JSON, yang mempermudah pemrosesan dan *transfer* data. Selain itu, mekanisme otentikasi untuk memastikan keamanan data selama proses integrasi berlangsung. Penerapan standar dalam pengembangan *custom* REST API sangat penting untuk memastikan *interoperabilitas* dan kemudahan penggunaan. Pengembanagan mengikuti prinsip RESTful, seperti penggunaan metode HTTP (GET, POST, PUT, DELETE) yang tepat, struktur URL yang logis, serta implementasi status kode HTTP yang sesuai. Selain itu, untuk meningkatkan keamanan, *token* API diterapkan sebagai mekanisme otentikasi untuk membatasi akses *endpoint* REST API. Setiap *token* dapat dikonfigurasi untuk memberikan hak akses tertentu berdasarkan pengguna atau aplikasi. Proses ini menggunakan pendekatan modern seperti OAuth atau JSON Web Token (JWT) untuk menjaga integritas dan keamanan data. Implementasi ini memastikan bahwa hanya pengguna yang berwenang dapat mengakses data sensitif melalui REST API [22].

2.1.10 *Docker Server*

Docker merupakan sebuah platform yang digunakan untuk mempermudah perancangan, pengiriman dan menjalankan aplikasi dengan menggunakan teknologi *container. Container* memungkinkan pengembang untuk mengemas aplikasi beserta dependensinya ke dalam suatu unit yang dapat dijalankan secara konsisten. Dalam arsitektur *Docker*, terdapat komponen utama yang disebut dengan *Docker Engine. Docker Engine* terdiri atas dua bagian utama yaitu :

1. *Docker Server,*
   Merupakan sebuah layanan pada *host* dan bertanggung jawab untuk membangun, menjalankan dan mengelola *container Docker. Docker Server* akan menjalankan permintaan API *Docker* dan mengelola objek *Docker* seperti *image*, *container*, *network* dan *volume.*
2. *Docker Client,*
   Merupakan sebuah layanan yang memungkinkan interaksi dengan *Docker Server*. Pengguna dapat menggunakan perintah *Docker* melalui *Command Line Interface* (CLI) untuk berkomunikasi dengan *Docker Server* yang kemudian akan menjalankan perintah tersebut [23].

Container Docker berjalan di dalam lingkungan yang terisolasi. Artinya aplikasi yang berada di satu *container* tidak akan saling mempengaruhi satu dengan yang lainnya. Hal tersebut sangat menguntungkan untuk pengguna yang memiliki beberapa aplikasi yang perlu dijalankan dalam waktu yang bersamaan namun tetap terpisah untuk konfigurasi dan dependensinya [24].

2.1.11 Arsitektur Program

Layered Architecture adalah pola arsitektur perangkat lunak yang memisahkan aplikasi ke dalam lapisan-lapisan yang berbeda berdasarkan tanggung jawabnya, seperti *Presentation Layer* untuk UI, *Domain Layer* untuk logika bisnis, dan *Data Layer* untuk komunikasi dengan *database* atau API. Pendekatan ini bertujuan untuk meningkatkan keterpisahan tanggung jawab (S*eparation of Concerns*) dan kemudahan pemeliharaan. Setiap lapisan hanya berkomunikasi dengan lapisan di atas atau di bawahnya, sehingga mengurangi ketergantungan antar komponen. Dengan demikian, *Layered Architecture* menjadi salah satu solusi efektif untuk membangun aplikasi yang *modular* dan *scalable* [25].

2.1.12 Metode *Agile*

Metode *agile* adalah metodologi perancangan sistem yang didasarkan pada proses yang dilakukan berulang dimana, aturan dan solusi disepakati dengan kolaborasi antar tiap tim secara terorganisir dan terstruktur. Secara umum manfaat dari metode *agile* adalah terlibatnya *user* secara langsung dalam pengembangan sistem di mana sistem akan langsung berfokus pada fitur yang dibutuhkan oleh pengguna dan lebih terorganisir. Tujuan metode pengembangan *agile* meliputi terjaganya kualitas perangkat lunak yang dikembangkan, anggaran efisiensi dan sumber daya karena pembangunan berfokus pada fitur yang dibutuhkan oleh pengguna, lebih terorganisir, dan kesempatan untuk berkolaborasi antar pemangku kepentingan [26].

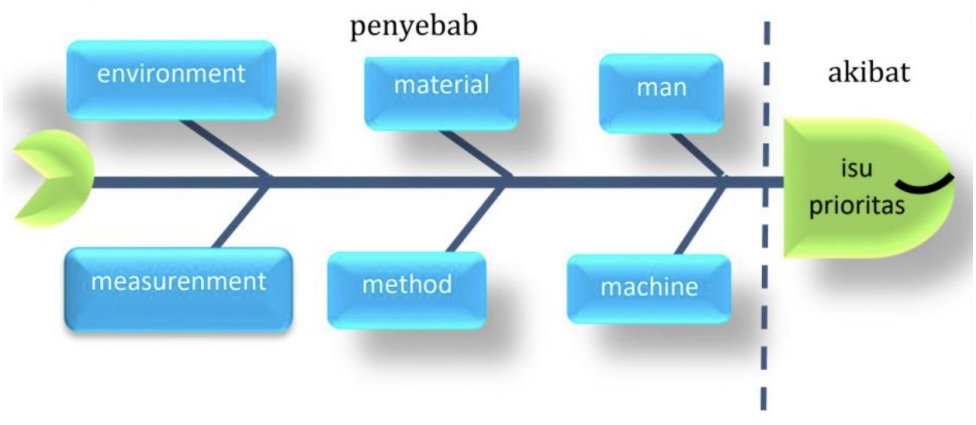
Prinsip-prinsip *agile*, seperti tercantum dalam manifesto *agile*, menekankan nilai-nilai seperti individu dan interaksi lebih dari proses dan alat, serta respon terhadap perubahan lebih dari mengikuti rencana yang telah ditetapkan. Berikut adalah dua belas prinsip *agile* yang dijabarkan dalam manifesto :

1. Memuaskan Pelanggan: prioritas utama adalah memuaskan pelanggan melalui pengiriman awal dan berkelanjutan perangkat lunak yang bernilai.
2. Mengakomodasi Perubahan: selamat datang perubahan persyaratan, bahkan di akhir pengembangan. Proses *agile* memanfaatkan perubahan untuk keunggulan kompetitif pelanggan.
3. Pengiriman yang Sering: Mengirim perangkat lunak yang berfungsi secara sering, dengan preferensi untuk periode waktu yang lebih pendek.
4. Kerja sama Harian: bisnis dan pengembang harus bekerja sama setiap hari sepanjang proyek.
5. Proyek Didorong oleh Individu Termotivasi: bangun proyek di sekitar individu yang termotivasi. Beri mereka lingkungan dan dukungan yang mereka butuhkan, dan percayalah mereka untuk menyelesaikan pekerjaan.
6. Komunikasi Langsung: metode paling efisien dan efektif untuk menyampaikan informasi ke dan di dalam tim pengembangan adalah percakapan tatap muka.

7. Perangkat Lunak yang Berfungsi sebagai Ukuran Kemajuan: perangkat lunak yang berfungsi adalah ukuran utama kemajuan.
8. Kecepatan Berkelanjutan: proses *agile* mempromosikan pengembangan berkelanjutan. Sponsor, pengembang, dan pengguna harus dapat mempertahankan kecepatan yang konstan tanpa batas.
9. Keunggulan Teknis dan Desain yang Baik: perhatian terus-menerus terhadap keunggulan teknis dan desain yang baik meningkatkan kelincahan.
10. Kesederhanaan: seni memaksimalkan jumlah pekerjaan yang tidak dilakukan adalah penting.
11. Tim *Self-Organizing*: arsitektur terbaik, persyaratan, dan desain muncul dari tim yang mengorganisasi sendiri.
12. Refleksi dan Penyesuaian Berkala: secara berkala, tim merefleksikan bagaimana menjadi lebih efektif, kemudian menyetel dan menyesuaikan perilaku mereka [27].

2.1.13 *Fishbone*

Konsep pengambilan keputusan melalui analisis *Fishbone*, juga dikenal sebagai diagram ishikawa atau diagram tulang ikan. Analisis *Fishbone* adalah teknik pengambilan keputusan yang digunakan untuk memeriksa faktor-faktor apa saja yang menyebabkan suatu masalah atau situasi terjadi dan bagaimana hal itu berdampak pada mereka. Teknik ini pertama kali dikembangkan oleh Kaoru Ishikawa pada tahun 1960. Pada dasarnya, analisis *Fishbone* digambarkan sebagai diagram tulang ikan dengan sumbu utama (tulang ikan) dan cabang yang menunjukkan faktor-faktor yang menyebabkan masalah yang sedang dihadapi. Setiap bagian dari diagram kemudian dapat dipecah menjadi faktor-faktor yang lebih khusus untuk membantu para pengambil keputusan memahami faktor-faktor yang memengaruhi situasi saat ini [28]. Diagram tulang ikan pada gambar 2.1 biasanya digunakan sebagai metode untuk mengidentifikasi semua elemen penyebab yang dianggap bertanggung jawab atas masalah tertentu. Kepala ikan menunjukkan masalah yang terjadi, dan tulang menunjukkan sumber masalah tersebut. Terlebih dahulu, harus memetakan penyebab masalah dan menganalisisnya menggunakan diagram tulang ikan untuk mengidentifikasi penyebab utamanya. Setelah itu, menawarkan ide atau inovasi terbaik untuk menyelesaikan masalah isu prioritas [29] .



Gambar 2. 1 *Diagram Fishbone*

Diagram *Fishbone* dapat dikatakan juga sebagai diagram sebab akibat, karena dapat digunakan untuk mengidentifikasi dan menganalisis berbagai faktor penyebab suatu masalah. Berikut merupakan detail penjelasan komponen komponen dalam diagram *Fishbone* yaitu :

1. Masalah utama atau *Effect*

Pada diagram *Fishbone*, masalah utama akan ditempatkan pada kepala, hal tersebut dilakukan menggambarkan topik yang sedang di analisis.

2. Kategori penyebab utama atau *Major Categories*

Pada diagram *Fishbone*, kategori penyebab utama akan digambarkan sebagai tulang besar yang bercabang dari tulang punggung utama. Dimana setiap cabang akan mewakili setiap kategori yang dapat mempengaruhi masalah utama. Kategori ini menggunakan pendekatan 6M, berikut merupakan detail penjelasannya :

1. *Man* atau Manusia
   Berisikan faktor yang berkaitan dengan tenaga kerja, misal keterampilan, pengalaman atau kondisi kerja.
2. *Machine* atau Mesin
   Berisikan faktor yang berkaitan dengan peralatan atau teknologi yang digunakan dalam proses.
3. *Method* atau Metode
   Berisikan faktor yang berhubungan dengan prosedur, proses atau cara kerja.
4. Material
   Berisikan faktor yang berkaitan dengan bahan baku atau material yang akan digunakan.
5. *Measurenment* atau Pengukuran
   Berisikan faktor yang berhubungan dengan alat pengukur, data dan keakuratan pengukuran.
6. *Environment* atau Lingkungan
   Berisikan faktor lingkungan kerja atau kondisi pada eksternal yang dapat mempengaruhi proses [30].

BAB 3
METODE PENELITIAN

3.1 Rencana Penelitian

Rencana penelitian ini bertujuan untuk mengembangkan sistem pemantauan penjualan *mobile* di PT Chemviro Buana Indonesia dengan menggunakan metode *Agile.* Pendekatan ini dipilih karena dapat mengembangkan sistem secara bertahap dan lebih fleksibel dalam merespon kompleksitas kebutuhan operasional perusahaan. Analisis utama dalam perusahaan menggunakan pendekatan *FishBone* Diagram. *Fishbone* Diagram digunakan sebagai metode penelitian untuk menganalisis penyebab utama suatu masalah dengan mencari solusi yang tepat untuk mengembangkan aplikasi *mobile* untuk *sales*. Selain itu, arsitektur program *Feature-First* akan diterapkan dalam perancangan aplikasi. Arsitektur ini memungkinkan perancangan sistem modular dengan mengalokasikan setiap fitur sebagai unit perancangan yang berbeda. Metode ini dimaksudkan untuk meningkatkan efisiensi perancangan dengan mengurangi ketergantungan antar fitur dan membuat proses *debugging* dan pembaruan fitur di masa mendatang lebih mudah. Diharapkan bahwa kombinasi metode *Agile* dengan arsitektur *Feature-First* akan memungkinkan fleksibilitas tinggi untuk menyesuaikan diri dengan perubahan kebutuhan bisnis.

3.1.1 Metode *Agile*

Penerapan metode *Agile* dalam perancangan aplikasi *mobile sales* ini bertujuan untuk memastikan setiap tahap perancangan dilakukan secara iteratif dan adaptif. Dengan pendekatan *Agile*, tim pengembang dapat berkolaborasi secara intensif dengan pemangku kepentingan untuk menetapkan prioritas fitur, menyusun solusi, dan menyempurnakan sistem berdasarkan umpan balik yang diperoleh secara berkala. Hal ini sejalan dengan tujuan utama, yaitu menggantikan proses manual berbasis *Excel* menjadi sistem *monitoring online* yang terintegrasi, serta mengoptimalkan operasional pemesanan, perubahan pesanan, dan pencatatan pendapatan. Melalui iterasi berkelanjutan, potensi kesalahan *input* data dapat diminimalkan, alur kerja antar-departemen menjadi lebih terstruktur, dan pengalaman pengguna ditingkatkan dengan fitur-fitur yang mendukung pengelolaan data secara efisien dan mudah diakses.

3.1.2 *Fishbone*

Saat mengembangkan aplikasi *mobile sales*, metode analisis Fishbone (Ishikawa) digunakan untuk menentukan dan mengklasifikasikan penyebab utama permasalahan. Sebagai isu utama dalam diagram *fishbone*, “Proses operasional dan pencatatan penjualan yang kurang efisien atau dilakukan secara manual” digunakan untuk mengkategorikan elemen penyebab menjadi enam kelompok: pengukuran *(measurenment)* , manusia *(man)*, metode *(method)*, material, mesin *(machine)*, dan lingkungan *(environment)*. Penyebab-penyebab spesifik diperiksa pada setiap kategori, seperti kurangnya pelatihan teknis pada faktor manusia, sistem manual yang tidak terintegrasi pada faktor metode, hingga ketidaktepatan data pada faktor pengukuran. Dengan menggunakan metode ini, *Fishbone* membantu mengidentifikasi secara sistematis penyebab-penyebab yang mendasarinya, menyebabkan perbaikan yang disarankan seperti validasi *input* data, desain antarmuka yang mudah digunakan, dan

mengimpor semua data historis ke dalam sistem aplikasi secara terstruktur. Semua itu dapat dikembangkan secara tepat untuk memfasilitasi penerapan aplikasi *mobile* yang efektif.

3.1.3 Arsitektur *Program*

Pada Proyek *Flutter* ini menggunakan *Layered Architecture*, yang memisahkan aplikasi menjadi beberapa lapisan dengan tanggung jawab yang jelas. *Folder pages* merepresentasikan *Presentation Layer*, bertanggung jawab untuk menampilkan antarmuka pengguna (UI) dan mengelola interaksi, seperti *file po_page.dart* yang menampilkan data *Purchase Order* (PO). Folder models berfungsi sebagai *Domain Layer*, tempat mendefinisikan model data seperti *product.dart* untuk merepresentasikan struktur produk. Selanjutnya, folder *services* menangani *Data Layer*, dengan file seperti *api_service.dart* yang digunakan untuk komunikasi dengan API eksternal dan pengambilan data dari server. Folder *routes* berfungsi sebagai *Routing Layer*, mendefinisikan navigasi antar halaman aplikasi, seperti menghubungkan halaman utama ke detail PO. Dengan pemisahan ini, aplikasi menjadi lebih modular dan mudah dipelihara, karena perubahan pada satu lapisan tidak memengaruhi lapisan lainnya. Pendekatan ini sesuai dengan prinsip *Separation of Concerns* dan *Single Responsibility Principle*, sehingga memastikan aplikasi tetap *scalable* dan dapat dikembangkan dengan efisien.

3.2 Objek Penelitian

Penelitian ini dilakukan oleh PT Chemviro Buana Indonesia dengan tujuan mengembangkan sistem *monitoring online* yang menggantikan proses manual menjadi berbasis *online*. Sistem ini dirancang untuk memastikan setiap pengguna hanya memiliki akses ke data dan fitur yang relevan dengan tugas dan tanggung jawab masing-masing. Selain itu, sistem ini bertujuan mempermudah akses informasi penting dalam perusahaan untuk meningkatkan efisiensi operasional.

Penelitian ini berfokus pada perancangan sistem monitoring untuk departemen *sales* melalui aplikasi *mobile*. Aplikasi ini dirancang untuk membantu tim *sales* dalam mengelola pesanan, mengedit pesanan, menambahkan klien baru, serta memantau penghasilan komisi dari setiap pesanan yang berhasil. Dengan fitur-fitur ini, diharapkan sistem dapat meningkatkan produktivitas tim *sales* dan memberikan kemudahan dalam pelaksanaan tugas mereka.

3.3 Teknik Pengumpulan Data

Dalam penelitian ini, teknik pengumpulan data dilakukan untuk memperoleh informasi yang relevan dan mendalam terkait permasalahan, kebutuhan, dan spesifikasi sistem yang akan dirancang. Adapun teknik yang digunakan meliputi:

1. Observasi

   Observasi dilakukan dengan cara mengamati langsung proses kerja dan alur operasional di PT Chemviro Buana Indonesia, khususnya pada aktivitas penjualan, laboratorium, dan keuangan. Tujuan dari observasi ini adalah untuk mengidentifikasi masalah yang terjadi serta memahami kebutuhan operasional di setiap departemen.
2. Studi Dokumen

Studi dokumen dilakukan dengan memeriksa laporan-laporan, dokumen penjualan, dan dokumen lain yang relevan. Hal ini dilakukan untuk mendapatkan data historis dan memahami pola operasional yang sudah berjalan di perusahaan.

Teknik pengumpulan data ini dilakukan secara sistematis untuk memastikan bahwa data yang diperoleh mencerminkan kebutuhan aktual dan dapat digunakan sebagai dasar dalam analisis dan perancangan sistem.

BAB 4
HASIL DAN PEMBAHASAN

4.1 Proses Bisnis

4.1.1 *Analisis Bisnis*

4.1.1.1 *Sales*

Sales adalah bagian dari perusahaan yang bertugas untuk menawarkan produk atau layanan kepada calon klien dengan tujuan menciptakan penjualan. Peran *sales* tidak hanya menjual, tetapi juga membangun hubungan dengan klien, memahami kebutuhan klien, serta memberikan solusi yang sesuai melalui produk atau layanan yang ditawarkan. Dalam konteks PT Chemviro Buana Indonesia, *sales* berfungsi sebagai penghubung antara perusahaan dan klien, memastikan bahwa jasa konsultan, jasa kalibrasi, jasa analisis kimia, perdagangan alat laboratorium serta keselamatan kerja yang ditawarkan oleh PT Chemviro Buana Indonesia dapat memenuhi kebutuhan klien secara optimal.

Departemen *Sales* di PT Chemviro Buana Indonesia memainkan peran penting dalam mengelola hubungan dengan klien dan memastikan layanan perusahaan yang bergerak di bidang konsultan, kalibrasi, analisis kimia, perdagangan alat laboratorium serta keselamatan kerja dapat tersampaikan dengan baik. Proses bisnis di departemen ini mencakup tiga tahap utama yaitu, *Sales Order* (SO), *Purchase Order* (PO), dan *Cancel* (Pembatalan). Berikut analisis detail datri setiap tahap:

1. *Sales Order (SO)*

   Proses bisnis dalam penjualan (*sales*) yang berkaitan dengan pembuatan *Sales Order* (SO) melibatkan beberapa tahapan penting yang dirancang untuk memastikan kebutuhan klien terpenuhi dengan baik. Tahapan ini mencakup pencarian klien dan negosiasi. *Sales* kemudian mengidentifikasi kebutuhan awal calon klien melalui pengamatan dan komunikasi terbuka. *Sales* memperkenalkan layanan atau produk yang dimiliki oleh perusahaan, menjelaskan manfaat yang dapat diperoleh klien, dan menanyakan lebih dalam tentang kebutuhan spesifik klien. Tujuan utama dari langkah ini adalah membangun kepercayaan, menciptakan hubungan positif, dan membuka peluang untuk melanjutkan ke diskusi yang lebih mendalam. Jika ada klien yang menunjukkan minat untuk menggunakan jasa PT Chemviro Buana Indonsia, *salesmen* akan menerima permintaan dari klien dan mulai menyusun dokumen *Sales Order (*SO). Dokumen ini mencakup detail produk dan layanan yang ditawarkan, termasuk jenis jasa yang diminati, spesifikasi teknis, harga, serta potensi diskon. Setelah kesepakatan tercapai, dokumen SO akan di-*approve* oleh *salesmen* untuk kemudian diubah statusnya menjadi *Purchase Order* (PO).

2. *Purchase Order (PO*)

   Proses pembuatan *Purchase Order* (PO) diawali dengan pengajuan *Sales* Order (SO) oleh pihak *salesman* melalui sistem *Integrated Marketing System* (CRM). Tahap ini melibatkan penyusunan rincian kebutuhan layanan atau produk yang diminta oleh klien, termasuk parameter-parameter teknis yang relevan. SO berisi

rincian kebutuhan layanan atau produk yang diminta oleh klien. Jika diperlukan, dilakukan negosiasi terkait harga atau detail layanan hingga tercapai kesepakatan. Setelah SO disetujui, sistem secara otomatis menghasilkan *draft* PO berdasarkan data yang telah diinput. *Draft* PO ini kemudian dikirimkan kepada klien untuk dilakukan peninjauan dan persetujuan. Setelah klien memberikan persetujuan, tahapan selanjutnya adalah eksekusi pesanan sesuai dengan dokumen PO yang telah disetujui. Dokumen ini diteruskan kepada laboratorium untuk melaksanakan layanan atau pengadaan barang yang diminta. Dalam situasi tertentu, jika terjadi kendala seperti keterlambatan atau perubahan teknis, laporan rekomendasi akan disusun dan diteruskan kepada pihak pihak terkait untuk evaluasi dan tindak lanjut. Setelah pekerjaan selesai, laporan hasil pelaksanaan dari laboratorium dikirimkan ke sistem CRM untuk diverifikasi dan disahkan. Laporan ini kemudian digunakan oleh divisi keuangan sebagai dasar untuk penyusunan faktur dan penyelesaian pembayaran dari klien

3. *Cancel*

   Dalam beberapa situasi, *Sales Order* (SO) dapat dibatalkan oleh penjual jika pelanggan memutuskan untuk tidak melanjutkan transaksi sebelum dokumen *Purchase Order* (PO) dibuat. Pembatalan ini biasanya terjadi ketika klien mengubah keputusan mereka karena berbagai alasan, seperti kebutuhan yang berubah, ketidakcocokan dengan harga atau syarat, atau kendala lain yang menghalangi proses lebih lanjut. Dalam situasi ini, *sales* harus memastikan bahwa semua komunikasi dengan klien terdokumentasi dengan baik untuk menghindari kesalahpahaman. Jika pelanggan menolak *Sales Order* (SO), maka order tersebut akan dibatalkan. Sebaliknya, jika pelanggan melakukan persetujuan, *Sales Order* (SO) akan dilanjutkan ke tahap pemrosesan sesuai prosedur yang telah ditetapkan. Proses pembatalan hanya bisa dilakukan selama status order masih berada pada tahap *Sales Order* (SO). Setelah itu, sistem harus memperbarui status SO menjadi “Dibatalkan” apabila pelanggan menolak order agar seluruh tim yang terlibat mengetahui perubahan tersebut dan tidak melanjutkan proses berikutnya.

4.1.1.2 *Laboratorium*

Laboratorium merupakan tempat pelaksana teknis yang memiliki peran strategis dalam mendukung kelancaran proses bisnis, khususnya dalam eksekusi pekerjaan sesuai dengan detail yang tercantum pada dokumen *Purchase Order* (PO). Proses pelaksanaan kegiatan laboratorium sepenuhnya bergantung pada persetujuan dokumen dari pihak klien melalui sistem *Customer Relationship Management* (CRM). Dengan mekanisme ini, laboratorium memastikan bahwa setiap pekerjaan yang dilakukan telah terverifikasi dan sesuai dengan permintaan yang disepakati, sehingga meminimalkan potensi kesalahan dalam eksekusi. Pekerjaan yang dilakukan oleh laboratorium mengacu pada parameter-parameter spesifik yang tercantum dalam PO, seperti jenis analisis atau pengujian yang diminta. Oleh sebab itu, ketelitian dalam proses penyusunan dokumen oleh bagian penjualan (*salesman*) menjadi aspek krusial untuk menjamin akurasi dan kelengkapan informasi sebelum proses eksekusi dimulai. Sistem CRM berperan penting dalam mengintegrasikan data dan informasi antar

departemen, sehingga laboratorium dapat mengakses dokumen dengan efisien dan menjalankan tugas secara optimal. Hasil akhir yang dihasilkan oleh laboratorium berupa Laporan Hasil Uji (LHU) menjadi komponen penting yang akan diteruskan kepada bagian keuangan untuk persetujuan penerbitan faktur (*invoice*). Laporan ini tidak hanya berfungsi sebagai dokumentasi hasil pekerjaan tetapi juga sebagai bukti bahwa laboratorium telah menyelesaikan tugas sesuai dengan permintaan klien. Apabila terjadi keterlambatan dalam proses pelaksanaan, laboratorium juga bertanggung jawab memberikan laporan yang memuat penyebab keterlambatan serta rekomendasi langkah langkah solusi yang dapat diambil.

4.1.1.3 *Finance*

Pada PT Chemviro Buana Indonesia, semua transaksi keuangan perusahaan dikelola secara terpusat oleh departemen keuangan (*finance*). Departemen ini memegang peran penting dalam memastikan kelancaran operasional perusahaan melalui pengelolaan keuangan yang terstruktur. Salah satu tugas utama departemen keuangan adalah memeriksa *invoice* masuk dari pihak eksternal, terutama untuk memastikan bahwa order telah selesai dikerjakan oleh laboratorium. Proses ini dilakukan dengan sangat teliti, mengingat laporan hasil uji (LHU) yang dikirimkan oleh laboratorium merupakan komponen krusial untuk menyetujui penerbitan faktur (*invoice*). Setelah LHU diverifikasi dan dinyatakan sesuai, departemen keuangan akan melanjutkan proses penerbitan *invoice* kepada pelanggan. *Invoice* yang dikirimkan mencakup detail pembayaran, seperti jumlah yang harus dibayarkan, tenggat waktu pembayaran, dan informasi rekening tujuan. Departemen keuangan memastikan bahwa dokumen ini jelas dan akurat agar memudahkan pelanggan dalam menyelesaikan kewajiban pembayaran mereka. Tidak hanya itu, departemen keuangan juga bertanggung jawab untuk memperbarui status pesanan dalam sistem internal perusahaan. Setiap pesanan yang telah diterbitkan *invoice*-nya akan diubah statusnya menjadi "*waiting payment*". Jika pembayaran dari pelanggan telah diterima, status pesanan akan diperbarui lagi menjadi "*PAID*", sehingga memberikan transparansi dan memudahkan pengelolaan laporan keuangan PT Chemviro Buana Indonesia.

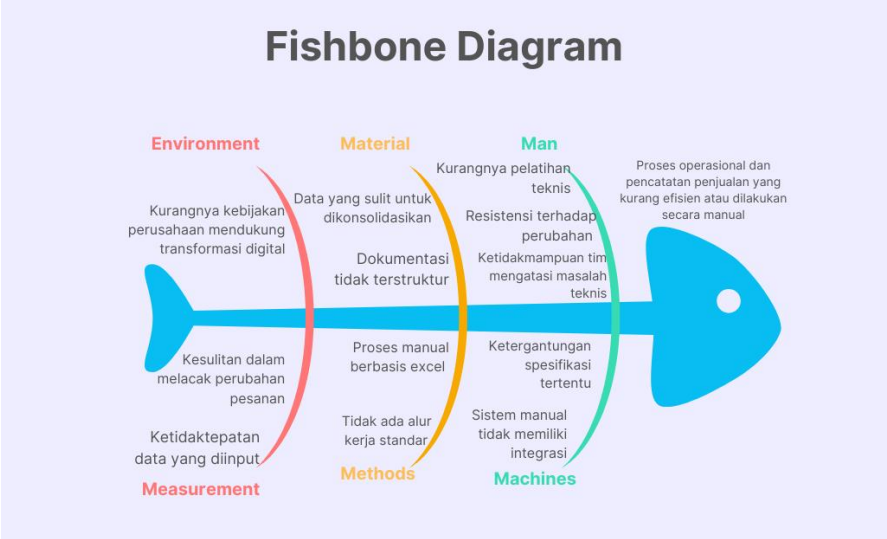
4.1.2 Analisis *Fishbone*

Dalam upaya mengembangkan aplikasi *mobile sales* yang efektif dan terintegrasi, diperlukan metode analisis yang mampu mengidentifikasi akar masalah secara sistematis maka dari itu digunakan diagram *fishbone* agar dapat mengidentifikasi masalah. Sehingga mendapatkan hasil bahwa ada satu masalah utama dengan enam penyebab masalah seperti yang digambarkan pada gambar diagram xx. Berikut merupakan penjelasan penerapan *Fishbone* untuk analisis berdasarkan kategori penyebab masalah utama yaitu:

1. Masalah Utama atau *Effect*
   "Proses operasional dan pencatatan penjualan yang kurang efisien atau dilakukan secara manual"
2. Faktor *Man* atau Manusia
   Salah satu tantangan utama dalam proses digitalisasi yaitu kurangnya kesiapan sumber daya manusia. Dalam penelitian ini sumber daya yang dimaksud adalah

sales, tim *sales* sering kali tidak mendapatkan pelatihan teknis secara maksimal untuk menggunakan aplikasi digital, karena hal tersebut tim *sales* cenderung menggunakan metode manual. Selain itu, perubahan dari metode manual ke sistem berbasis *mobile* juga menjadi hambatan yang sangat signifikan, karena tim *sales* merasa tidak mampu untuk menangani masalah teknis yang muncul saat sedang digunakan.

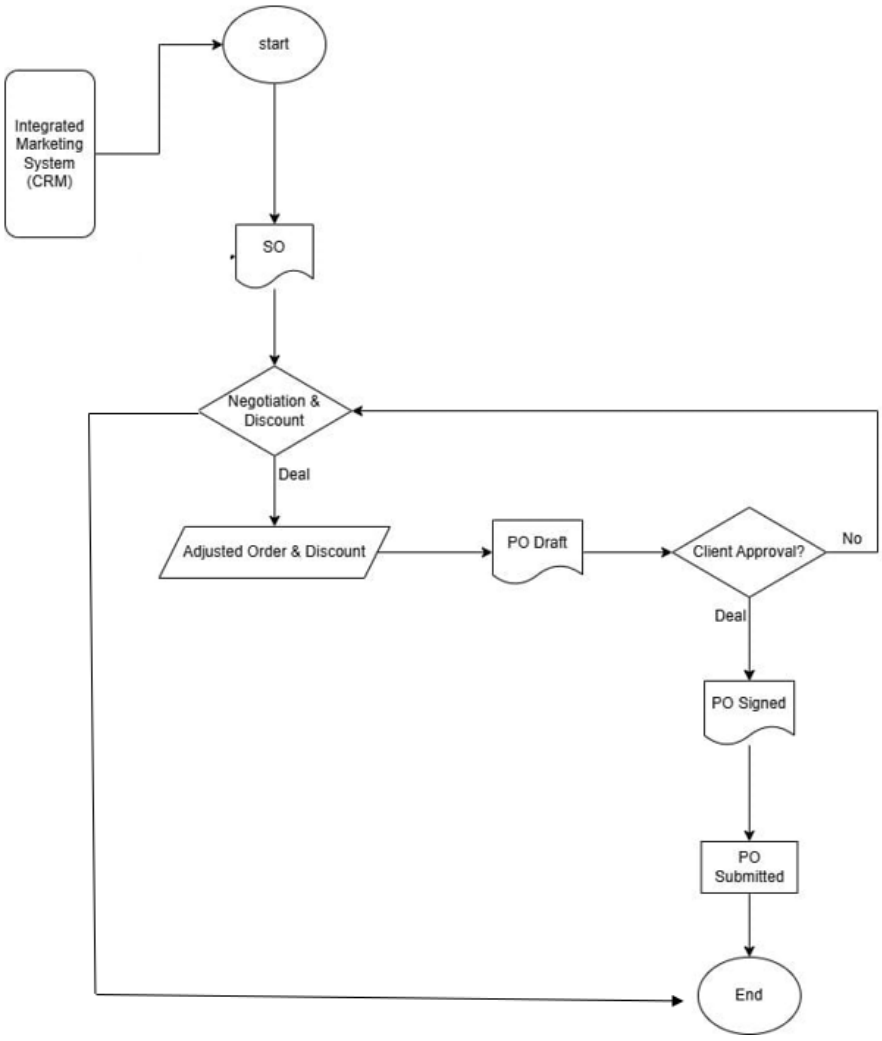
3. Faktor *Machine* atau Mesin
   Pada Faktor mesin, sistem manual yang tidak memiliki integritas internet mengakibatkan keterlambatan informasi karena data harus diperbarui secara manual dan tidak dapat akses secara *real-time*. Lebih jauh lagi, ketergantungan pada perangkat dengan spesifikasi yang telah ditetapkan sebelumnya membatasi fleksibilitas dan aksesibilitas pengguna, yang menghambat pengambilan keputusan yang lebih cepat.
4. Faktor *Method* atau Metode
   Metode manual berbasis *Excel* yang digunakan saat ini sangat rentan terhadap kesalahan *input* data yang dapat mengakibatkan ketidakakuratan laporan dan keputusan yang salah. Selain itu, tidak adanya alur kerja standar untuk pemesanan, perubahan pesanan dan pelaporan menambah kompleksitas proses.
5. Faktor *Material*
   Ketidakmampuan untuk mengakses data yang terorganisir dan dokumen yang tidak terstruktur, sehingga sulit untuk mengambil keputusan yang cepat dan tepat. Melacak informasi seperti bahan baku, stok atau operasi yang diperlukan merupakan hal yang sulit karena data tersebar di berbagai sumber dan pencatatan manusia tidak standar. Hal ini dapat membuat proses penjualan menjadi kurang akurat dan efisien.
6. Faktor *Measurenment* atau Pengukuran
   Data manual yang tidak akurat menurunkan akurasi pelaporan, dan sulit untuk memantau performa *sales* jika tidak ada alat pemantauan *real-time.* Tidak adanya sistem sistem pelacakan terintegrasi juga membuat sulit untuk memantau perubahan pendapatan atau pesanan.
7. Faktor *Environment* atau Lingkungan
   Alur kerja menjadi tidak efektif dan rawan kesalahan jika mengandalkan prosedur manual dan teknologi konvensional, seperti penggunaan *Excel* dan mengelola data penjualan. Pengambilan keputusan terhambat oleh ketidakmampuan tim *sales* dan departemen lain untuk berkomunikasi dan berbagi informasi secara *real-time* tanpa sistem yang terintegrasi.



Gambar 4. 1 *Fishbone Diagram*

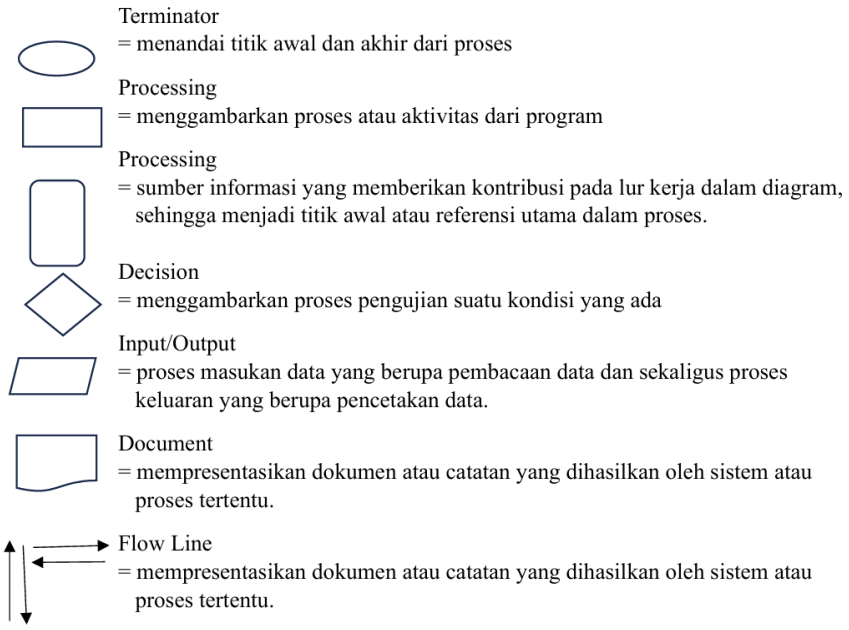
4.1.3 *Flowchart* Alur Bisnis

1. *SO to PO Submission*



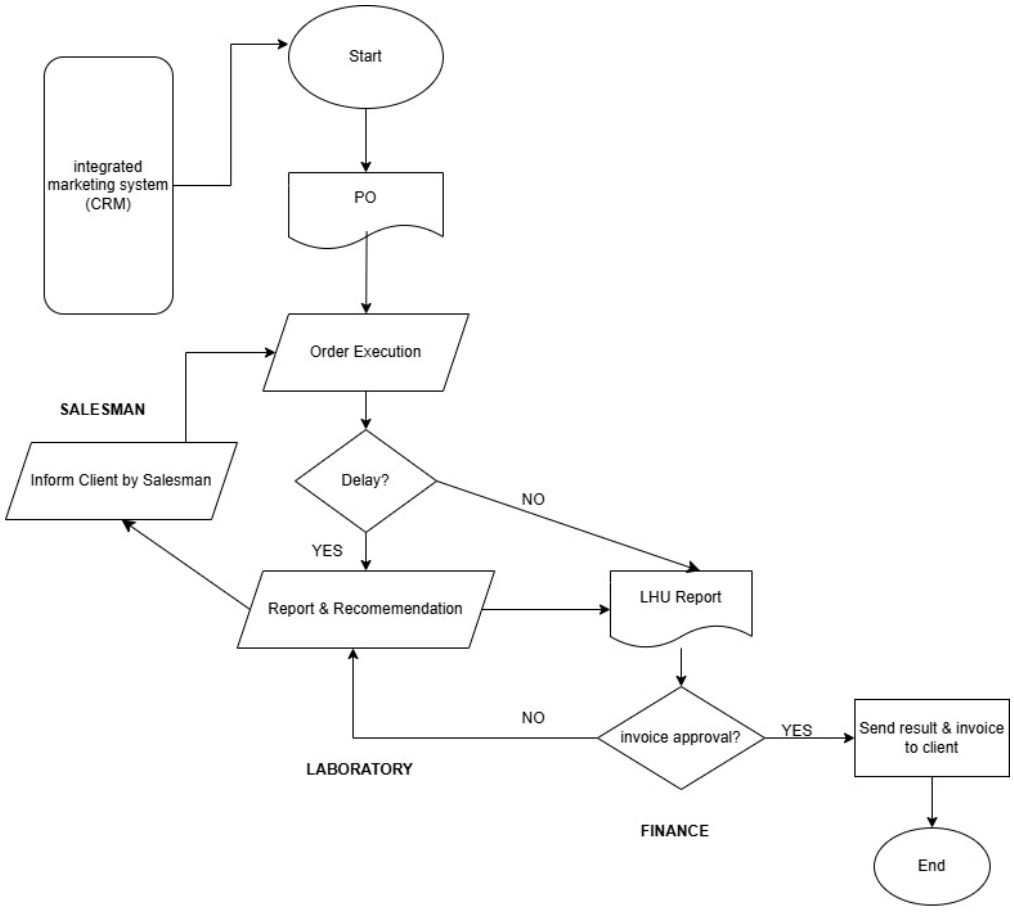
Gambar 4. 2 *Flow SO to PO Submission*

Keterangan:



Gambar 4. 3 Keterangan *Flow SO to Po Submission*

2. *Order Execution*



Gambar 4. **4** *Flow Order Execution*

Keterangan :

Gambar 4. **5** Keterangan *Flow Order Execution*

4.2 Alur Kerja

4.2.1 Kebutuhan Sistem *Mobile Sales Management*

Analisis kebutuhan sistem mengacu pada BRD yang dirancang untuk mendukung aktivitas *sales* secara efisien dan terorganisir. Fitur utama sistem meliputi:

1. Autentikasi dan Keamanan Akses

   Fungsi ini memastikan hanya pengguna resmi yang dapat mengakses aplikasi melalui proses *login*. Validasi kredensial menjaga keamanan data perusahaan, sementara fungsi *logout* menutup sesi dengan aman untuk melindungi informasi pengguna.
2. Daftar Layanan

   Dengan fitur ini *sales* dapat mengetahui jenis layanan apa saja yang tersedia, seperti analisis air dan analisis lingkungan beserta detail kategori produk dan harga.
3. Pengelolaan Data Pesanan

   Dengan fitur ini dapat memudahkan *sales* untuk membuat pesanan klien dengan menginput *status order*, nama klien, *branch company*, produk dan diskon. *Detail order* akan muncul setelah *create order* dan dapat diakses di SO *page*. Jika terjadi perubahan pada detail pesanan, *sales* dapat dengan mudah mengedit pesanan sesuai permintaan klien. Serta jika klien setuju melanjutkan proses order, status SO (*Sales Order*) akan berubah menjadi PO (*Purchase Order*). Namun, jika klien tidak setuju, maka status SO akan menjadi CO (*Cancel Order*).
4. Pemantauan dan Pengelolaan Status Pesanan

Fungsi ini dirancang untuk memberikan akses *real-time* ke seluruh pesanan yang tercatat. *Sales* dapat memantau status pesanan (SO, PO, atau CO) dan mengambil tindakan sesuai kebutuhan. Fungsi ini meningkatkan transparansi dan memudahkan koordinasi dengan tim lain dalam proses operasional.

5. Informasi Profil Pengguna
   Fitur ini menampilkan informasi *personal sales*, termasuk nama, email, cabang perusahaan, nomor telepon, jumlah pesanan dan klien. Profil ini dapat diakses melalui menu Profil dan otomatis diperbarui berdasarkan aktivitas pengguna di aplikasi.
6. Mengunduh *Detail Order* Sebagai PDF
   Fitur ini dibuat untuk memudahkan *sales* mengunduh informasi pesanan dalam format PDF sebagai bentuk dokumentasi fisik dan juga dapat sebagai dokumen persetujuan klien ketika melakukan penawaran dengan *sales.*
7. Laporan Pendapatan *Sales*
   Dengan adanya fitur ini memudahkan *sales* untuk melihat informasi mengenai total pendapatan yang dihitung berdasarkan total pesanan yang telah dibuat dan telah berstatus *Purchase Order* (PO).

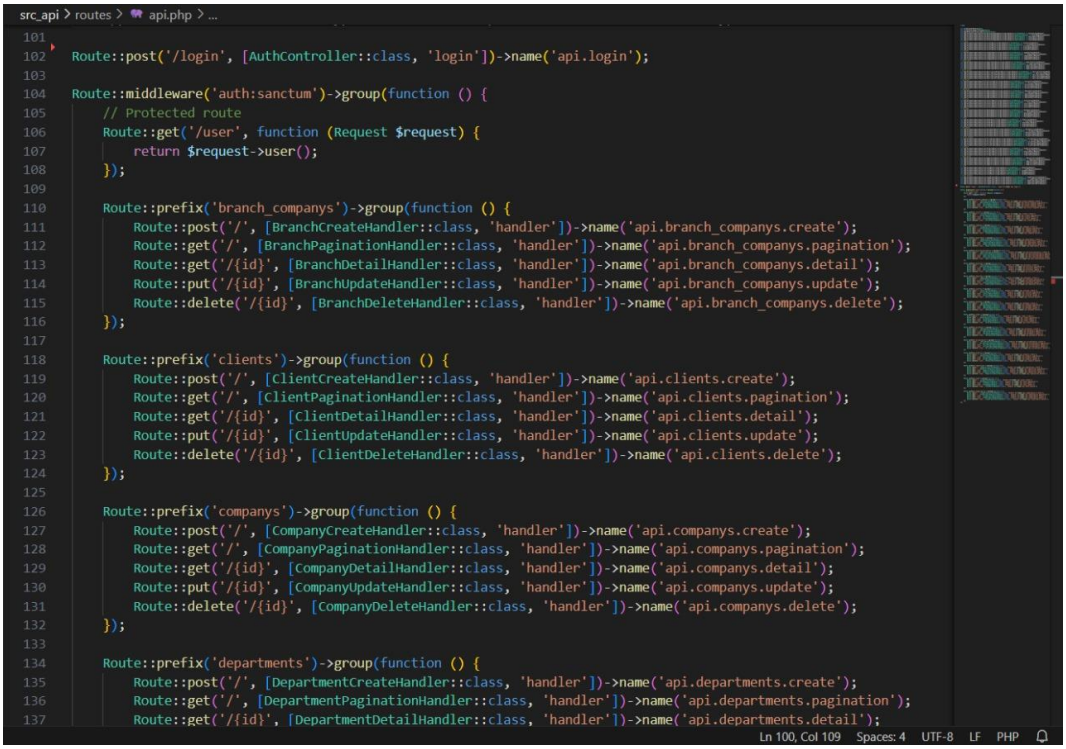
4.2.2 Analisis *Fungsionalitas Sistem*

Fungsi-fungsi strategis yang diidentifikasi mencakup:

1. Meningkatkan efisiensi operasional, dengan memanfaatkan aplikasi ini, proses penjualan menjadi lebih terorganisir sehingga akan meminimalisir adanya kesalahan dalam pencatatan dimana hal ini dapat mendukung efisiensi operasional perusahaan.
2. Manajemen produk order yang terorganisir dengan fitur order yang terdapat dalam aplikasi, dapat membantu pengguna untuk melihat orderan yang telah diproses kemudian orderan yang harus dikirim dan melakukan edit pada order.
3. Meningkatkan produktivitas *sales*, aplikasi dapat membantu *sales* bekerja lebih baik dengan adanya fitur *input edit* dan pelacakan *history order*.
4. Meningkatkan kepuasan pelanggan dengan kemampuan untuk melacak status *order* secara *real-time* dan memastikan keakuratan informasi, pelanggan akan mendapatkan layanan yang baik dan profesional.
5. Mengoptimalkan jadwal dan aktivitas *sales*, *sales* dapat berfokus untuk menjangkau klien karena proses administrasi yang cepat dan mudah.

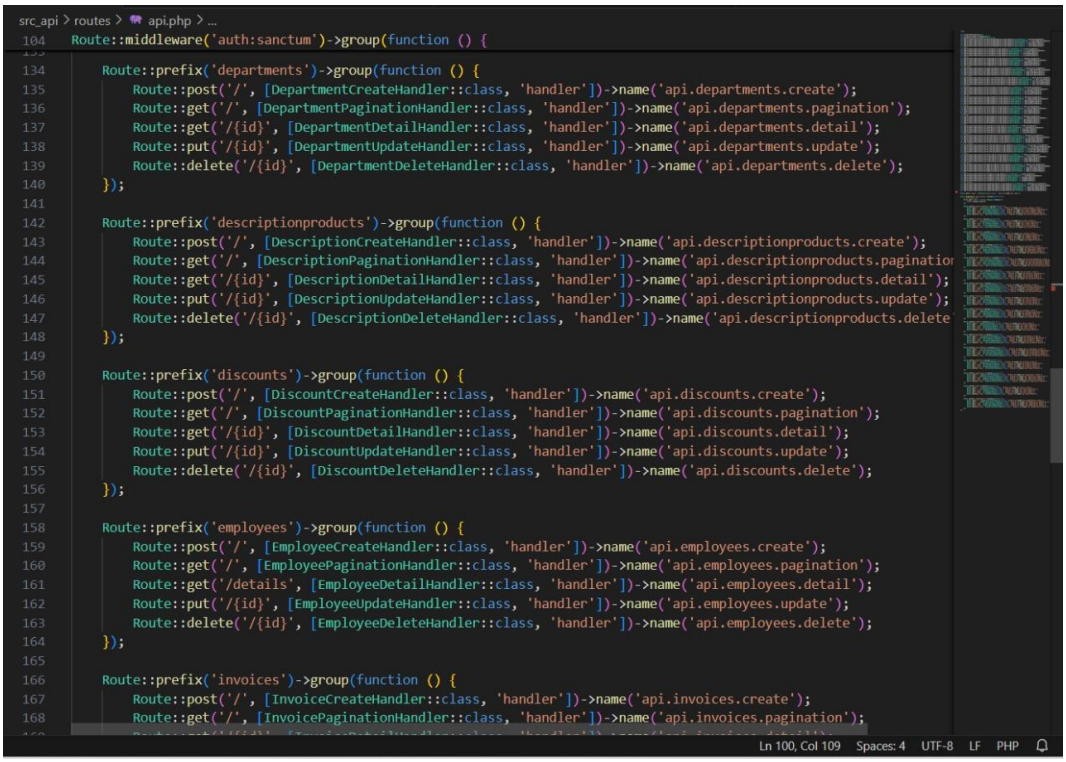
4.2.3 Konfigurasi *Rute API*

Berikut merupakan struktur dan pengelolaan *endpoint* API yang dirancang untuk memenuhi kebutuhan integrasi dan pengolahan data :



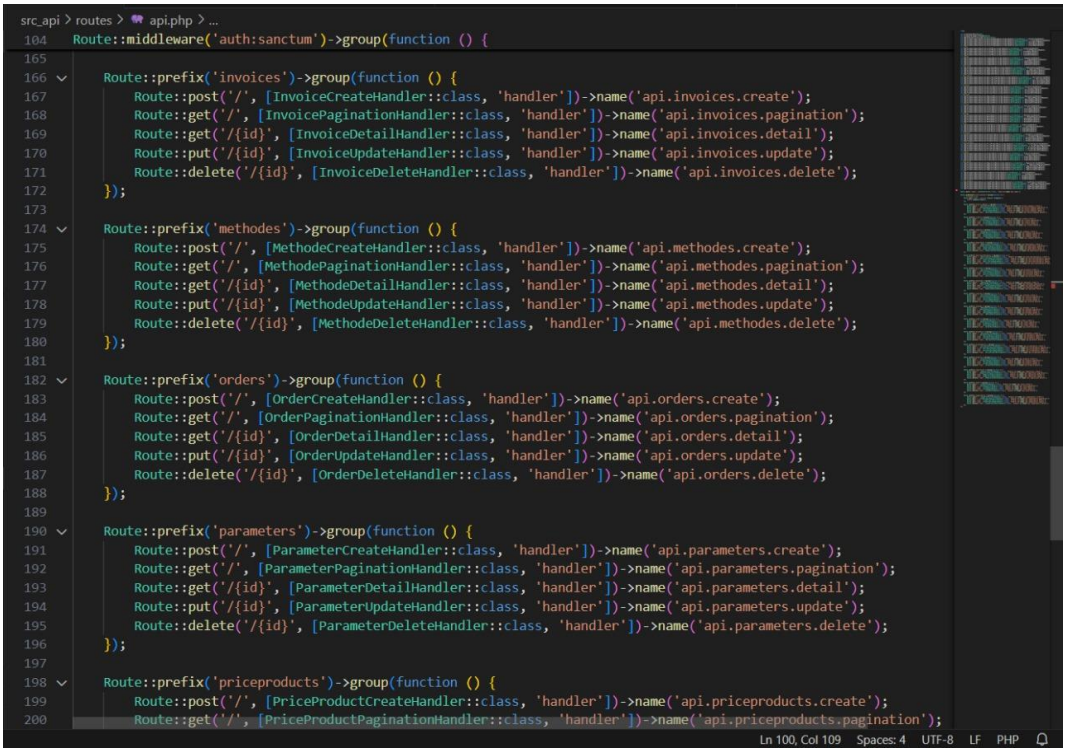
```
src_api > routes > api.php > ...
101
102 Route::post('/login', [AuthController::class, 'login'])->name('api.login');
103
104 Route::middleware('auth:sanctum')->group(function () {
105     // Protected route
106     Route::get('/user', function (Request $request) {
107         return $request->user();
108     });
109
110     Route::prefix('branch_companys')->group(function () {
111         Route::post('/', [BranchCreateHandler::class, 'handler'])->name('api.branch_companys.create');
112         Route::get('/', [BranchPaginationHandler::class, 'handler'])->name('api.branch_companys.pagination');
113         Route::get('/{id}', [BranchDetailHandler::class, 'handler'])->name('api.branch_companys.detail');
114         Route::put('/{id}', [BranchUpdateHandler::class, 'handler'])->name('api.branch_companys.update');
115         Route::delete('/{id}', [BranchDeleteHandler::class, 'handler'])->name('api.branch_companys.delete');
116     });
117
118     Route::prefix('clients')->group(function () {
119         Route::post('/', [ClientCreateHandler::class, 'handler'])->name('api.clients.create');
120         Route::get('/', [ClientPaginationHandler::class, 'handler'])->name('api.clients.pagination');
121         Route::get('/{id}', [ClientDetailHandler::class, 'handler'])->name('api.clients.detail');
122         Route::put('/{id}', [ClientUpdateHandler::class, 'handler'])->name('api.clients.update');
123         Route::delete('/{id}', [ClientDeleteHandler::class, 'handler'])->name('api.clients.delete');
124     });
125
126     Route::prefix('companys')->group(function () {
127         Route::post('/', [CompanyCreateHandler::class, 'handler'])->name('api.companys.create');
128         Route::get('/', [CompanyPaginationHandler::class, 'handler'])->name('api.companys.pagination');
129         Route::get('/{id}', [CompanyDetailHandler::class, 'handler'])->name('api.companys.detail');
130         Route::put('/{id}', [CompanyUpdateHandler::class, 'handler'])->name('api.companys.update');
131         Route::delete('/{id}', [CompanyDeleteHandler::class, 'handler'])->name('api.companys.delete');
132     });
133
134     Route::prefix('departments')->group(function () {
135         Route::post('/', [DepartmentCreateHandler::class, 'handler'])->name('api.departments.create');
136         Route::get('/', [DepartmentPaginationHandler::class, 'handler'])->name('api.departments.pagination');
137         Route::get('/{id}', [DepartmentDetailHandler::class, 'handler'])->name('api.departments.detail');
Ln 100, Col 109  Spaces: 4  UTF-8  LF  PHP
```

Gambar 4. 6 Konfigurasi Rute API



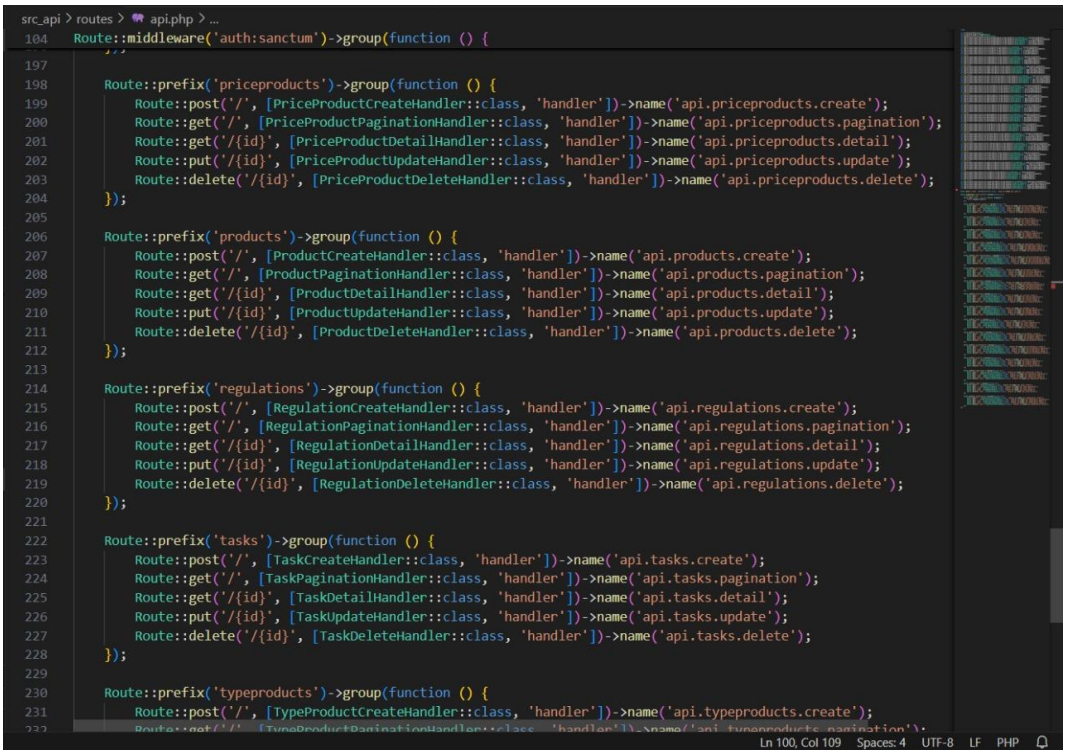
```
src_api > routes > api.php > ...
104 Route::middleware('auth:sanctum')->group(function () {
134     Route::prefix('departments')->group(function () {
135         Route::post('/', [DepartmentCreateHandler::class, 'handler'])->name('api.departments.create');
136         Route::get('/', [DepartmentPaginationHandler::class, 'handler'])->name('api.departments.pagination');
137         Route::get('/{id}', [DepartmentDetailHandler::class, 'handler'])->name('api.departments.detail');
138         Route::put('/{id}', [DepartmentUpdateHandler::class, 'handler'])->name('api.departments.update');
139         Route::delete('/{id}', [DepartmentDeleteHandler::class, 'handler'])->name('api.departments.delete');
140     });
141
142     Route::prefix('descriptionproducts')->group(function () {
143         Route::post('/', [DescriptionCreateHandler::class, 'handler'])->name('api.descriptionproducts.create');
144         Route::get('/', [DescriptionPaginationHandler::class, 'handler'])->name('api.descriptionproducts.paginatio
145         Route::get('/{id}', [DescriptionDetailHandler::class, 'handler'])->name('api.descriptionproducts.detail');
146         Route::put('/{id}', [DescriptionUpdateHandler::class, 'handler'])->name('api.descriptionproducts.update');
147         Route::delete('/{id}', [DescriptionDeleteHandler::class, 'handler'])->name('api.descriptionproducts.delete
148     });
149
150     Route::prefix('discounts')->group(function () {
151         Route::post('/', [DiscountCreateHandler::class, 'handler'])->name('api.discounts.create');
152         Route::get('/', [DiscountPaginationHandler::class, 'handler'])->name('api.discounts.pagination');
153         Route::get('/{id}', [DiscountDetailHandler::class, 'handler'])->name('api.discounts.detail');
154         Route::put('/{id}', [DiscountUpdateHandler::class, 'handler'])->name('api.discounts.update');
155         Route::delete('/{id}', [DiscountDeleteHandler::class, 'handler'])->name('api.discounts.delete');
156     });
157
158     Route::prefix('employees')->group(function () {
159         Route::post('/', [EmployeeCreateHandler::class, 'handler'])->name('api.employees.create');
160         Route::get('/', [EmployeePaginationHandler::class, 'handler'])->name('api.employees.pagination');
161         Route::get('/details', [EmployeeDetailHandler::class, 'handler'])->name('api.employees.detail');
162         Route::put('/{id}', [EmployeeUpdateHandler::class, 'handler'])->name('api.employees.update');
163         Route::delete('/{id}', [EmployeeDeleteHandler::class, 'handler'])->name('api.employees.delete');
164     });
165
166     Route::prefix('invoices')->group(function () {
167         Route::post('/', [InvoiceCreateHandler::class, 'handler'])->name('api.invoices.create');
168         Route::get('/', [InvoicePaginationHandler::class, 'handler'])->name('api.invoices.pagination');
Ln 100, Col 109  Spaces: 4  UTF-8  LF  PHP
```

Gambar 4. **7** Konfigurasi Rute API



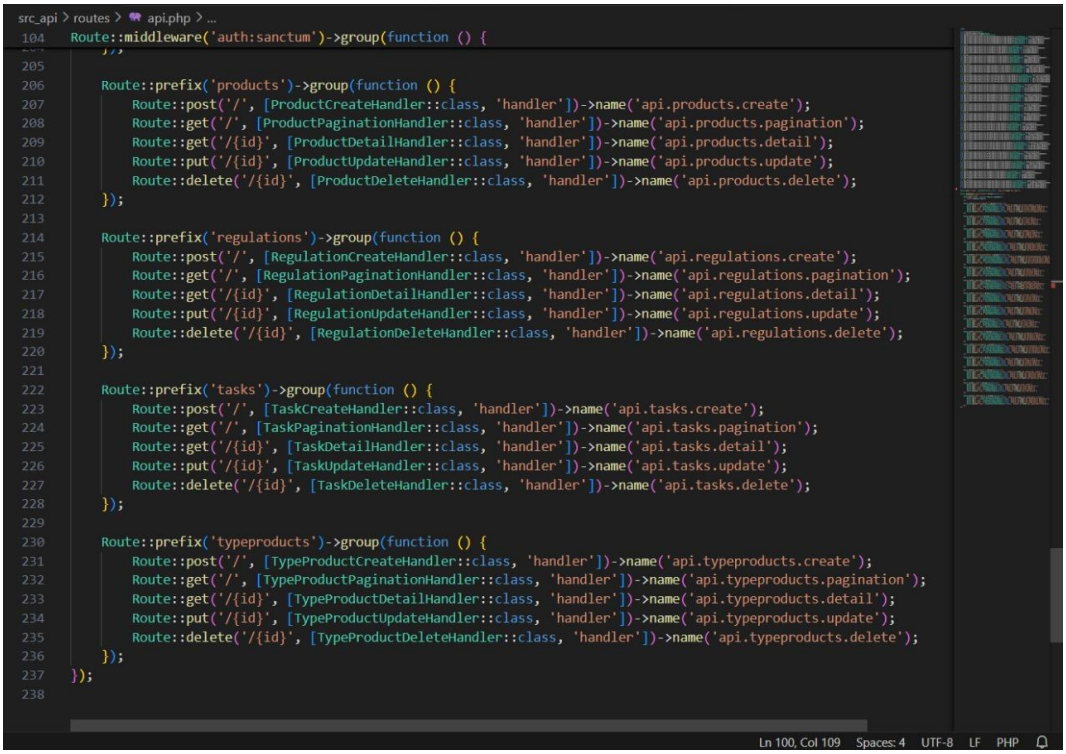
```
src_api > routes > api.php > ...
104  Route::middleware('auth:sanctum')->group(function () {
165
166      Route::prefix('invoices')->group(function () {
167          Route::post('/', [InvoiceCreateHandler::class, 'handler'])->name('api.invoices.create');
168          Route::get('/', [InvoicePaginationHandler::class, 'handler'])->name('api.invoices.pagination');
169          Route::get('/{id}', [InvoiceDetailHandler::class, 'handler'])->name('api.invoices.detail');
170          Route::put('/{id}', [InvoiceUpdateHandler::class, 'handler'])->name('api.invoices.update');
171          Route::delete('/{id}', [InvoiceDeleteHandler::class, 'handler'])->name('api.invoices.delete');
172      });
173
174      Route::prefix('methodes')->group(function () {
175          Route::post('/', [MethodeCreateHandler::class, 'handler'])->name('api.methodes.create');
176          Route::get('/', [MethodePaginationHandler::class, 'handler'])->name('api.methodes.pagination');
177          Route::get('/{id}', [MethodeDetailHandler::class, 'handler'])->name('api.methodes.detail');
178          Route::put('/{id}', [MethodeUpdateHandler::class, 'handler'])->name('api.methodes.update');
179          Route::delete('/{id}', [MethodeDeleteHandler::class, 'handler'])->name('api.methodes.delete');
180      });
181
182      Route::prefix('orders')->group(function () {
183          Route::post('/', [OrderCreateHandler::class, 'handler'])->name('api.orders.create');
184          Route::get('/', [OrderPaginationHandler::class, 'handler'])->name('api.orders.pagination');
185          Route::get('/{id}', [OrderDetailHandler::class, 'handler'])->name('api.orders.detail');
186          Route::put('/{id}', [OrderUpdateHandler::class, 'handler'])->name('api.orders.update');
187          Route::delete('/{id}', [OrderDeleteHandler::class, 'handler'])->name('api.orders.delete');
188      });
189
190      Route::prefix('parameters')->group(function () {
191          Route::post('/', [ParameterCreateHandler::class, 'handler'])->name('api.parameters.create');
192          Route::get('/', [ParameterPaginationHandler::class, 'handler'])->name('api.parameters.pagination');
193          Route::get('/{id}', [ParameterDetailHandler::class, 'handler'])->name('api.parameters.detail');
194          Route::put('/{id}', [ParameterUpdateHandler::class, 'handler'])->name('api.parameters.update');
195          Route::delete('/{id}', [ParameterDeleteHandler::class, 'handler'])->name('api.parameters.delete');
196      });
197
198      Route::prefix('priceproducts')->group(function () {
199          Route::post('/', [PriceProductCreateHandler::class, 'handler'])->name('api.priceproducts.create');
200          Route::get('/', [PriceProductPaginationHandler::class, 'handler'])->name('api.priceproducts.pagination');
                                                                    Ln 100, Col 109  Spaces: 4  UTF-8  LF  PHP
```

Gambar 4. **8** Konfigurasi Rute API



```
src_api > routes > api.php > ...
104  Route::middleware('auth:sanctum')->group(function () {
197
198      Route::prefix('priceproducts')->group(function () {
199          Route::post('/', [PriceProductCreateHandler::class, 'handler'])->name('api.priceproducts.create');
200          Route::get('/', [PriceProductPaginationHandler::class, 'handler'])->name('api.priceproducts.pagination');
201          Route::get('/{id}', [PriceProductDetailHandler::class, 'handler'])->name('api.priceproducts.detail');
202          Route::put('/{id}', [PriceProductUpdateHandler::class, 'handler'])->name('api.priceproducts.update');
203          Route::delete('/{id}', [PriceProductDeleteHandler::class, 'handler'])->name('api.priceproducts.delete');
204      });
205
206      Route::prefix('products')->group(function () {
207          Route::post('/', [ProductCreateHandler::class, 'handler'])->name('api.products.create');
208          Route::get('/', [ProductPaginationHandler::class, 'handler'])->name('api.products.pagination');
209          Route::get('/{id}', [ProductDetailHandler::class, 'handler'])->name('api.products.detail');
210          Route::put('/{id}', [ProductUpdateHandler::class, 'handler'])->name('api.products.update');
211          Route::delete('/{id}', [ProductDeleteHandler::class, 'handler'])->name('api.products.delete');
212      });
213
214      Route::prefix('regulations')->group(function () {
215          Route::post('/', [RegulationCreateHandler::class, 'handler'])->name('api.regulations.create');
216          Route::get('/', [RegulationPaginationHandler::class, 'handler'])->name('api.regulations.pagination');
217          Route::get('/{id}', [RegulationDetailHandler::class, 'handler'])->name('api.regulations.detail');
218          Route::put('/{id}', [RegulationUpdateHandler::class, 'handler'])->name('api.regulations.update');
219          Route::delete('/{id}', [RegulationDeleteHandler::class, 'handler'])->name('api.regulations.delete');
220      });
221
222      Route::prefix('tasks')->group(function () {
223          Route::post('/', [TaskCreateHandler::class, 'handler'])->name('api.tasks.create');
224          Route::get('/', [TaskPaginationHandler::class, 'handler'])->name('api.tasks.pagination');
225          Route::get('/{id}', [TaskDetailHandler::class, 'handler'])->name('api.tasks.detail');
226          Route::put('/{id}', [TaskUpdateHandler::class, 'handler'])->name('api.tasks.update');
227          Route::delete('/{id}', [TaskDeleteHandler::class, 'handler'])->name('api.tasks.delete');
228      });
229
230      Route::prefix('typeproducts')->group(function () {
231          Route::post('/', [TypeProductCreateHandler::class, 'handler'])->name('api.typeproducts.create');
                                                                    Ln 100, Col 109  Spaces: 4  UTF-8  LF  PHP
```

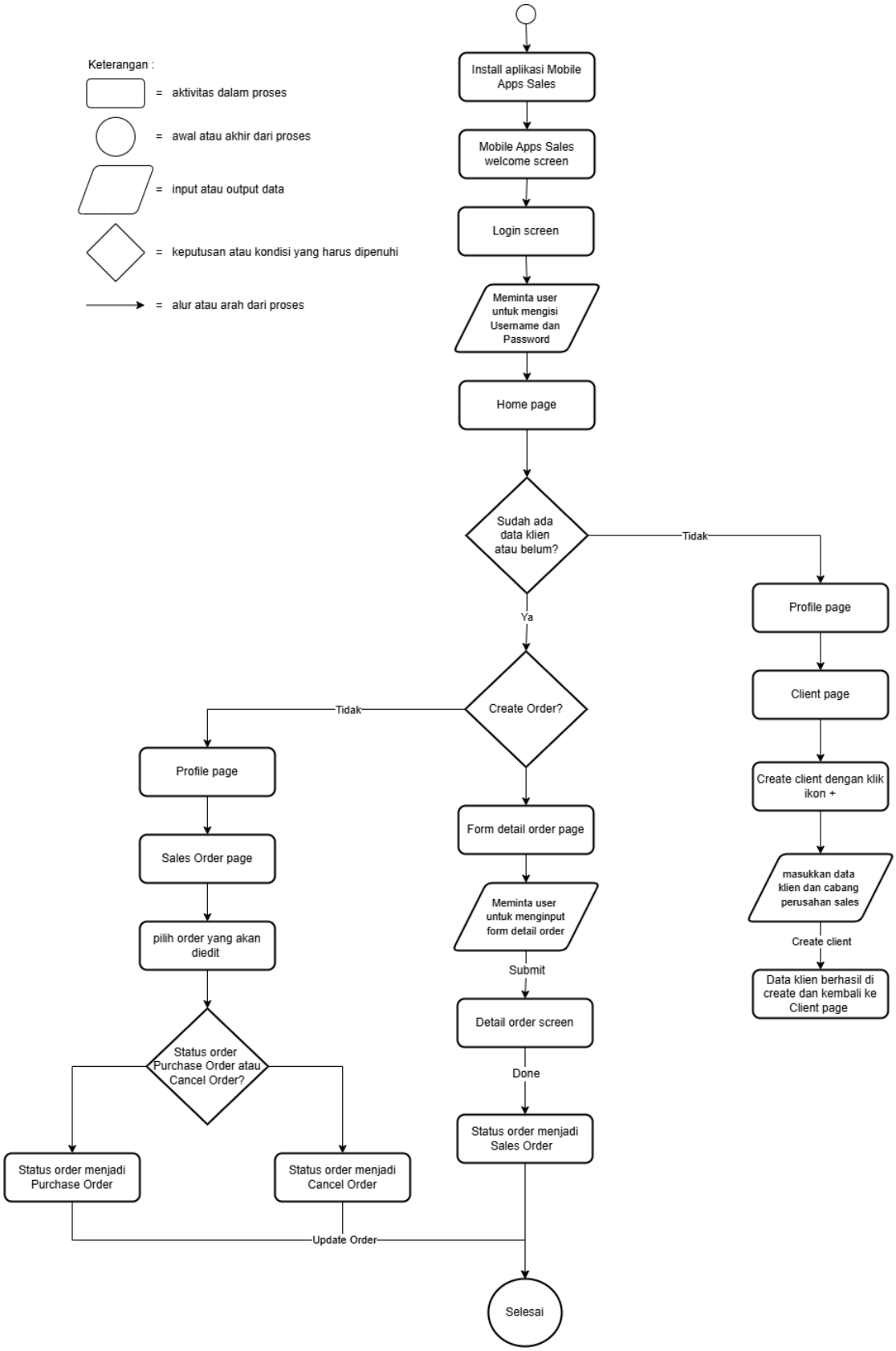
Gambar 4. 9 Konfigurasi Rute API



Gambar 4. **10** Konfigurasi Rute API

Gambar-gambar di atas menunjukkan struktur rute API yang digunakan dalam sistem. Setiap rute API dikelompokkan berdasarkan fungsi utamanya, seperti pengelolaan produk, regulasi, tugas, dan tipe produk. Pendekatan ini bertujuan untuk memastikan bahwa setiap API memiliki fungsi yang terorganisir dengan baik dan mudah diakses oleh pengembang.

4.2.4 *Flowchart* Aplikasi



Gambar 4. **11** Flowchart Aplikasi

Proses dimulai dengan **menginstal aplikasi *Mobile Apps Sales*** untuk mendukung pembuatan dan pengelolaan order. Setelah instalasi selesai, pengguna

diarahkan ke halaman *welcome screen* dan kemudian *login* dengan memasukkan *username* dan *password*. Hal ini dilakukan untuk memastikan keamanan dan validasi akses ke dalam sistem aplikasi.

Setelah berhasil *login*, pengguna masuk ke **halaman utama (h*ome page*)** aplikasi. Pada halaman ini, pengguna dapat memulai proses pembuatan order dengan memilih tombol **"*Create Order*"**. Pilihan ini akan membawa pengguna ke halaman *form detail order*, di mana pengguna diminta untuk mengisi informasi terkait *order* seperti nama produk, jumlah, harga, dan data lainnya yang relevan.

Setelah semua data terisi, pengguna menekan tombol "*Submit*" untuk menyimpan data tersebut. Selanjutnya, data order yang telah tersimpan akan ditampilkan pada halaman detail order untuk ditinjau ulang. Aplikasi secara otomatis menghasilkan file PDF berisi data *sales order*, yang kemudian dikirimkan kepada klien untuk proses persetujuan.

Pada tahap berikutnya, klien diminta untuk memberikan keputusan terkait order. Jika klien menyetujui, maka status order akan diperbarui menjadi *Purchase Order*. Namun, jika klien tidak menyetujui, status order akan berubah menjadi *Cancel Order*. Setelah keputusan klien diterima dan status order diperbarui, proses dianggap selesai.

4.3 Tampilan *Mobile*

Berikut penjelasan secara rinci mengenai tampilan *mobile*, termasuk desain antarmuka, kegunaan dalam mendukung pengalaman pengguna, serta penjabaran fitur-fitur utama yang tersedia pada *Mobile Apps Sales.*

4.3.1 Tampilan Awal *(Welcome Screen)*

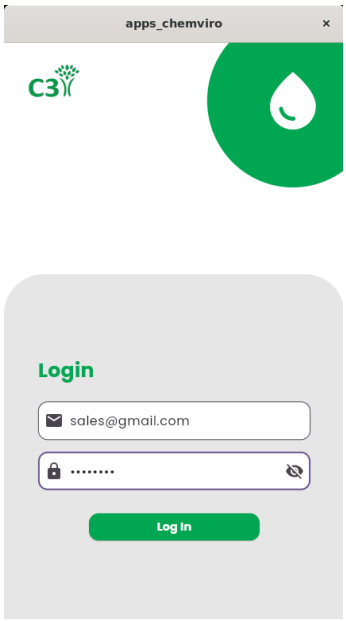
Berikut adalah tampilan awal atau *welcome screen* dari *Mobile Apps Sales*:

Gambar 4. **12** Tampilan *Welcome Screen*

Tampilan ini muncul saat aplikasi pertama kali dijalankan. Fungsi utamanya adalah menyambut pengguna dengan logo aplikasi dan memperkuat *branding.* Latar belakang hijau dengan logo aplikasi "C3" yang mencerminkan profesionalisme dan ramah lingkungan.

4.3.2 Tampilan *Login (Login Screen)*

Berikut adalah tampilan *login* dari *Mobile Apps Sales*:



Gambar 4. **13** Tampilan *Login (Login Screen)*

Tampilan *Login* ini menampilkan *form login* berfungsi sebagai tempat pengguna memasukkan email dan kata sandi untuk mengakses aplikasi, dilengkapi dengan ikon untuk menampilkan atau menyembunyikan kata sandi demi meningkatkan kenyamanan dan keamanan pengguna, serta validasi *input* untuk memastikan data *login* yang dimasukkan benar sebelum masuk ke sistem.

4.3.3 Tampilan Utama *(Home Screen)*

Berikut adalah tampilan utama dari *Mobile Apps Sales*:

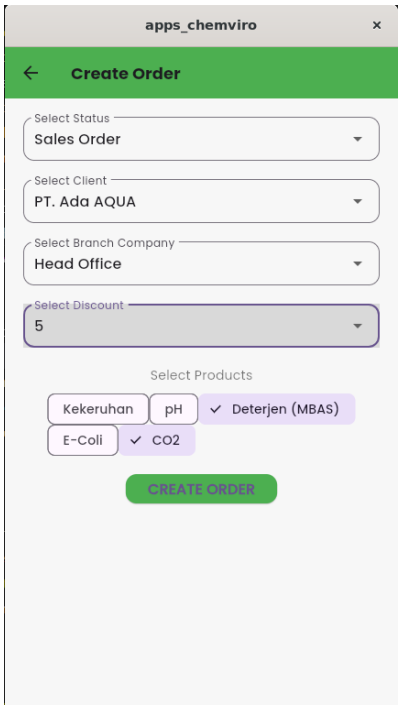
Gambar 4. 14 Tampilan Utama *(Home Screen)*

Tampilan utama (home) menampilkan daftar layanan Analisis Air dan Analisis Lingkungan. Pada layanan "Analisis Air," pengguna dapat menemukan informasi layanan seperti Kekeruhan dengan kategori "air *hygiene sanitasis*" dan harga Rp50.000,00, serta pH dengan kategori "air limbah IPAL" dan harga Rp75.000,00. Sementara itu, layanan "Analisis Lingkungan" memuat informasi layanan seperti Deterjen (MBAS), E-Coli, dan CO2, yang masing-masing memiliki kategori "analisis udara 24 jam" dengan harga bervariasi mulai dari Rp30.000,00 hingga Rp120.000,00. Setiap layanan dilengkapi dengan ikon panah (^) yang memungkinkan pengguna untuk memperluas atau menyembunyikan daftar layanan, memberikan pengalaman interaksi yang dinamis dan intuitif.

Di bagian bawah layar, terdapat tombol "*Create Order*" yang ditampilkan dengan desain tombol hijau besar dan dilengkapi ikon plus (+). Tombol ini dirancang untuk memudahkan pengguna membuat pesanan baru dengan cepat. Secara keseluruhan, desain aplikasi ini mengedepankan prinsip minimalis dan fungsional. Latar belakang putih menciptakan kesan bersih, sementara warna hijau pada elemen penting mempertegas identitas aplikasi yang ramah lingkungan. Informasi yang disusun secara terstruktur dan tata letak yang intuitif memastikan pengguna dapat bernavigasi dan menggunakan aplikasi dengan nyaman.

4.3.4 Tampilan *Create Order (Create Order Screen)*

Berikut adalah tampilan *create order* dari *Mobile Apps Sales:*



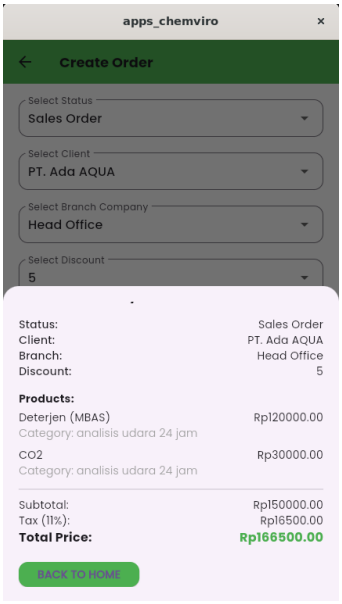
Gambar 4. **15** Tampilan *Create Order (Create Order Screen)*

Sistem ini berfungsi untuk mencatat detail pesanan, seperti nama klien, produk, diskon, dan cabang perusahaan, dilengkapi dengan fitur *dropdown* untuk memilih klien dan produk serta validasi otomatis guna memastikan data lengkap sebelum pesanan dibuat,

sehingga meminimalkan kesalahan pencatatan dan memastikan dokumentasi informasi pesanan yang akurat.

4.3.5 Tampilan *Create Order Detail (Create Order Screen)*

Berikut adalah tampilan *create order* dari *Mobile Apps Sales:*

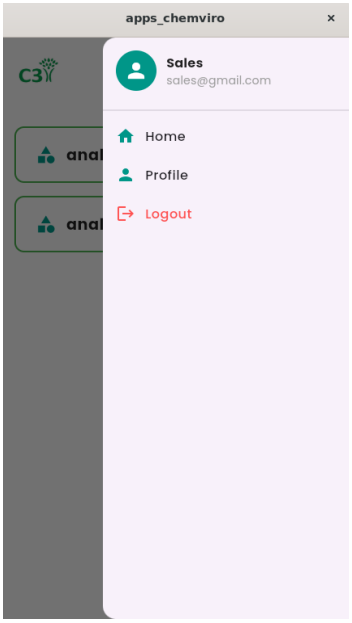


Gambar 4. **16** Tampilan *Create Order (Create Order Screen)*

Sistem ini menyediakan daftar pesanan dengan rincian seperti nomor pesanan, nama klien, produk yang dipesan, dan total harga, dilengkapi dengan ikon navigasi untuk melihat detail atau memperbarui status pesanan, sehingga memudahkan pengguna dalam melacak dan mengelola semua pesanan.

4.3.6 Tampilan *Side Bar*

Berikut adalah tampilan *side bar* dari *Mobile Apps Sales:*



Gambar 4. **17** Tampilan *Side Bar*

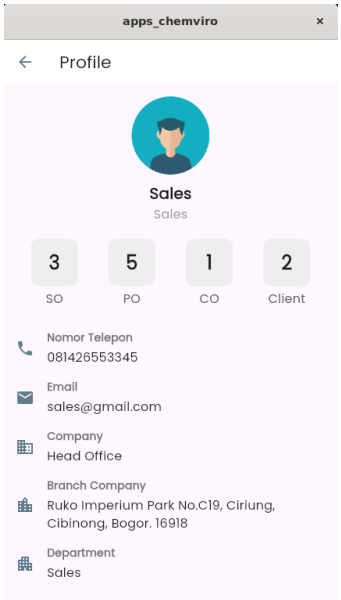
Tampilan *Side Bar* ini dirancang untuk memberikan kemudahan navigasi bagi pengguna. Pada bagian atas *side* bar, terdapat *header* yang menampilkan informasi akun pengguna berupa nama pengguna, yaitu "*Sales*," dan email yang terhubung, *sales@gmail.com*. Di sebelah kiri nama, terdapat ikon berbentuk lingkaran dengan simbol orang sebagai representasi visual profil pengguna, yang membantu mengidentifikasi akun yang sedang aktif.

Bagian utama *side* bar berisi tiga menu navigasi yang disusun secara vertikal untuk memudahkan pengguna. Menu pertama adalah *Home*, dilengkapi dengan ikon rumah kecil berwarna hijau, yang berfungsi untuk membawa pengguna kembali ke halaman utama atau *dashboard* aplikasi. Menu kedua adalah *Profile*, dengan ikon orang berwarna hijau, yang mengarahkan pengguna ke halaman profil untuk mengelola informasi pribadi atau akun mereka. Menu ketiga adalah *Logout*, yang ditandai dengan ikon panah keluar berwarna merah. Menu ini berfungsi untuk keluar dari akun pengguna, menutup sesi *login*, dan memastikan keamanan data. Warna merah pada menu *Logout* memberikan perhatian khusus, mengingat fungsinya yang penting dan tidak dapat diulang tanpa *login* kembali.

Desain *side* bar menggunakan latar belakang putih dengan teks dan ikon yang memiliki kontras jelas, memastikan kemudahan membaca dan aksesibilitas. Efek latar transparan di luar area *side* bar menonjolkan tampilan *side* bar sekaligus menunjukkan bahwa elemen lainnya tidak dapat diakses sementara *side* bar terbuka. Susunan menu yang rapi dengan jarak yang cukup antar item meningkatkan kenyamanan pengguna dalam memilih opsi yang diinginkan.

4.3.7 Tampilan Profil (Profile Screen)

Berikut adalah tampilan profil dari *Mobile Apps Sales*:



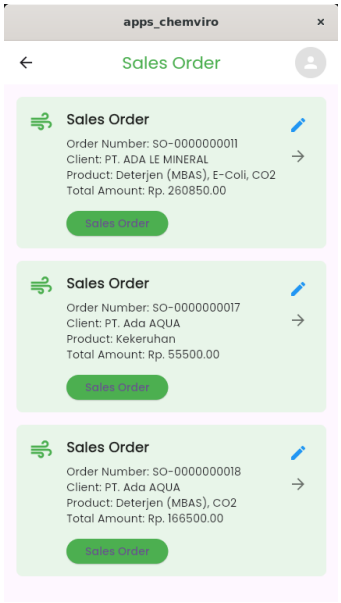
Gambar 4. **18** Tampilan Profil (*Profile Screen*)

Tampilan *Profile* ini dirancang untuk memberikan informasi mendetail tentang akun pengguna. Bagian atas halaman menampilkan *header* dengan judul "*Profile*" yang diikuti dengan ikon panah kembali di sisi kiri untuk navigasi kembali ke halaman

sebelumnya. Di bawahnya terdapat ikon *avatar* berbentuk lingkaran dengan ilustrasi pengguna, disertai nama pengguna "*Sales*" yang ditampilkan dengan teks tebal, serta jabatan atau peran pengguna, juga dituliskan sebagai "*Sales*" dalam teks lebih kecil. Tepat di bawah informasi ini, terdapat empat kartu (*card*) statistik yang memberikan ringkasan aktivitas pengguna, yaitu jumlah SO/SALES ORDER (3), PO/PURCHASE ORDER (5), CO/CANCEL ORDER (1), dan *Client* (2), masing-masing ditampilkan dalam kotak abu-abu muda yang rapi dan mudah dibaca. Bagian berikutnya menyajikan informasi kontak pengguna, dimulai dengan Nomor Telepon (081426553345) yang ditandai dengan ikon telepon, diikuti dengan Email (sales@gmail.com) yang memiliki ikon amplop sebagai indikasi visual. Selanjutnya, terdapat informasi perusahaan, seperti Company yang menunjukkan "*Head Office*," dan Branch Company yang mencantumkan alamat lengkap "Ruko Imperium Park No.C19, Cirung, Cibinong, Bogor, 16918," serta Department yang menyatakan "*Sales*." Semua informasi ini disusun secara rapi dalam tata letak vertikal dengan ikon pendukung, teks yang mudah dibaca, dan warna latar putih serta elemen berwarna pastel untuk memberikan kesan profesional namun tetap ramah pengguna.

4.3.8 Tampilan *Sales Order (Sales Order Screen)*

Berikut adalah tampilan *sales order* dari *Mobile Apps Sales:*



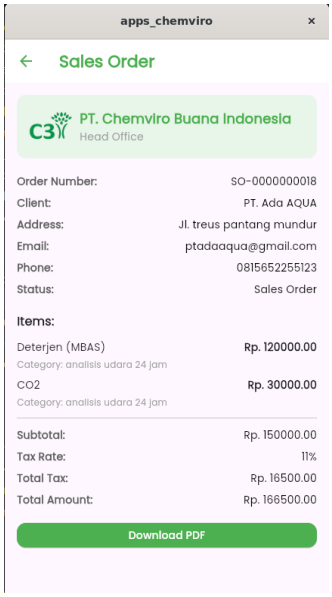
Gambar 4. **19** Tampilan *Sales Order (Sales Order Screen)*

Tampilan *Sales Order* ini dirancang untuk menampilkan daftar pesanan penjualan secara terstruktur dan informatif. Di bagian atas layar terdapat *header* dengan judul "*Sales Order,"* yang menggunakan warna hijau mencolok untuk menonjolkan halaman ini. Di sisi kiri *header*, terdapat ikon panah untuk kembali ke halaman sebelumnya, sementara di sisi kanan terdapat ikon profil berbentuk lingkaran untuk navigasi cepat ke informasi pengguna. Setiap kartu Sales Order menampilkan informasi detail pesanan, seperti Order *Number*, nama *Client*, daftar *Product* yang dipesan, dan Total *Amount* dalam format yang jelas dan terorganisasi. Contohnya, salah satu kartu mencantumkan "Order *Number*: SO-0000000011," dengan *client* "PT. ADA LE

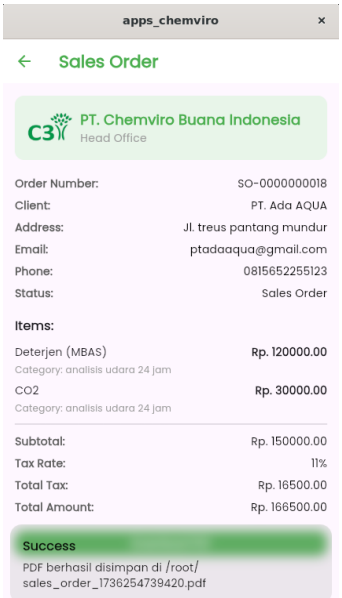
MINERAL," produk seperti "Deterjen (MBAS), E-Coli, CO2," dan total biaya "Rp260.850,00." Kartu-kartu ini dilengkapi dengan ikon pensil di pojok kanan atas untuk mengedit pesanan dan ikon panah di sebelah kanan untuk melihat rincian lebih lanjut. Setiap kartu juga memiliki tombol hijau besar bertuliskan "*Sales Order*," yang menonjol sebagai aksi utama untuk memproses atau mengelola pesanan. Warna latar kartu menggunakan hijau pastel yang berbeda untuk memberikan kesan visual yang bersih dan profesional, memudahkan pengguna membedakan setiap pesanan.

4.3.9 Tampilan *Sales Order Detail (Sales Order Detail Screen)*

Berikut adalah tampilan *sales order* detail dari *Mobile Apps Sales:*



Gambar 4. **20** Tampilan *Sales Order Detail (Sales Order Detail Screen)*



Gambar 4. **21** Tampilan *Sales Order Detail (Sales Order Detail Screen)*

Tampilan *Sales Order Detail* ini dirancang untuk memberikan informasi lengkap mengenai pesanan tertentu secara terstruktur. Di bagian atas terdapat *header*

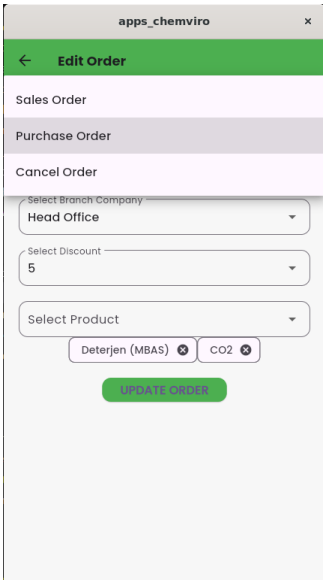
dengan judul "*Sales Order*" yang ditampilkan dalam warna hijau, dengan ikon panah kembali di sisi kiri untuk navigasi ke halaman sebelumnya. Bagian pertama menampilkan informasi perusahaan dengan logo C3 di sebelah kiri dan nama "PT. Chemviro Buana Indonesia" beserta deskripsi "*Head Office*," yang disorot dalam latar hijau pastel untuk memberikan identitas yang jelas. Di bawahnya, detail pesanan disusun secara rapi, mencakup *Order Number* ("SO-0000000018"), *Client* ("PT. Ada AQUA"), Address ("Jl. treus pantang mundur"), Email ("ptadaaqua@gmail.com"), *Phone* ("0815652255123"), dan Status pesanan ("*Sales Order*").

Bagian berikutnya memuat daftar *Items* yang dipesan, termasuk nama item seperti "Deterjen (MBAS)" dengan harga Rp120.000,00 dan "*CO2*" dengan harga Rp30.000,00, serta kategori layanan ("analisis udara 24 jam") untuk masing-masing item. Di bagian bawah, terdapat rangkuman biaya, meliputi *Subtotal* (Rp150.000,00), *Tax Rate* (11%), Total *Tax* (Rp16.500,00), dan Total *Amount* (Rp166.500,00), yang disusun dengan format yang mudah dibaca. Di bagian paling bawah, terdapat tombol hijau besar bertuliskan "*Download* PDF," yang memungkinkan pengguna mengunduh rincian pesanan dalam format PDF untuk kebutuhan dokumentasi atau pencetakan. Tampilan ini secara keseluruhan mengedepankan kejelasan, aksesibilitas informasi, dan fungsi yang efisien, memudahkan pengguna dalam memahami dan mengelola detail pesanan secara menyeluruh.

Pada tampilan *Sales Order Detail*, ketika tombol "*Download* PDF" diklik, muncul pesan konfirmasi di bagian bawah halaman yang menunjukkan bahwa proses pengunduhan telah berhasil dilakukan. Pesan tersebut ditampilkan dalam kotak hijau dengan label "*Success*", memberikan indikasi visual yang jelas bahwa tindakan telah berhasil. Di dalam kotak, terdapat teks yang menjelaskan bahwa file PDF berhasil disimpan dengan lokasi penyimpanan spesifik, seperti pada contoh "/root/sales_order_1736254739420.pdf."

4.3.10 Tampilan *Edit Order (Edit Order Screen)*

Berikut adalah tampilan *edit order* dari *Mobile Apps Sales:*

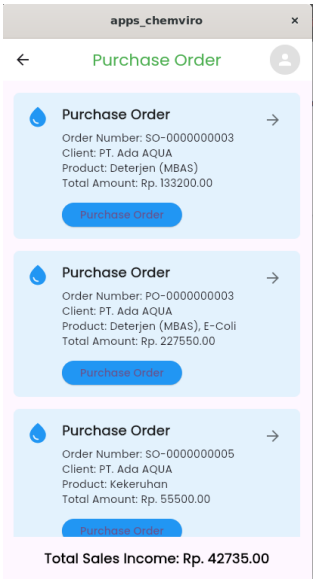


Gambar 4. **22** Tampilan *Edit Order Screen*

Tampilan *Edit Order* ini dirancang untuk memungkinkan pengguna melakukan pembaruan terhadap detail pesanan dengan antarmuka yang sederhana namun fungsional. Bagian atas layar menampilkan *header* hijau dengan judul "*Edit Order*" yang diikuti ikon panah kembali di sisi kiri untuk navigasi ke halaman sebelumnya. Di bawahnya terdapat menu *drop-down* yang memungkinkan pengguna untuk memilih jenis pesanan yang sedang diedit, seperti *Sales Order*, *Purchase Order*, atau *Cancel Order*, yang memberikan fleksibilitas dalam pengelolaan pesanan. Bagian utama formulir mencakup beberapa *input* yang dapat disesuaikan, seperti *Select Branch Company* dengan opsi yang terdaftar, dalam contoh ini *"Head Office,"* serta *Select Discount*, yang memungkinkan pengguna memilih persentase diskon, misalnya "5." Pengguna juga dapat mengelola produk dengan memilih dari daftar di bagian *Select Product*, di mana produk yang telah dipilih seperti "Deterjen (MBAS)" dan "CO2" ditampilkan dalam bentuk *tag* yang dapat dihapus dengan ikon silang jika diperlukan. Di bagian bawah, terdapat tombol hijau besar bertuliskan *"UPDATE ORDER*", yang berfungsi untuk menyimpan perubahan yang telah dilakukan pada pesanan.

4.3.11 Tampilan *Purchase Order (Purchase Order Screen)*

Berikut adalah tampilan *purchase order* dari *Mobile Apps Sales*:



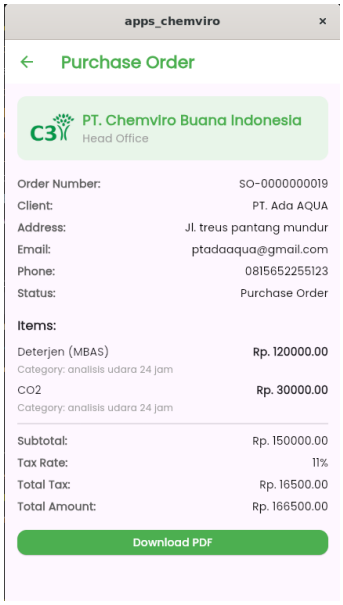
Gambar 4. 23 Tampilan *Edit Order Screen*

Tampilan *Purchase Order* ini dirancang untuk menampilkan daftar pesanan pembelian secara jelas dan informatif. Bagian atas layar memiliki *header* dengan judul "*Purchase Order*" yang menggunakan warna hijau sebagai penekanan, diikuti oleh ikon panah di sisi kiri untuk kembali ke halaman sebelumnya dan ikon profil di sisi kanan untuk akses cepat ke informasi akun pengguna. Setiap kartu pesanan mencakup informasi detail, seperti Order *Number*, nama *Client*, daftar *Product* yang dipesan, dan Total *Amount* yang tertera dalam format yang mudah dipahami. Contohnya, salah satu kartu mencantumkan "*Order Number*: SO-0000000011," *client* "PT. Ada AQUA," produk "Kekeruhan," dengan total biaya sebesar Rp555.00,00. Setiap kartu dilengkapi dengan ikon tetesan air biru di sisi kiri sebagai elemen visual yang merepresentasikan

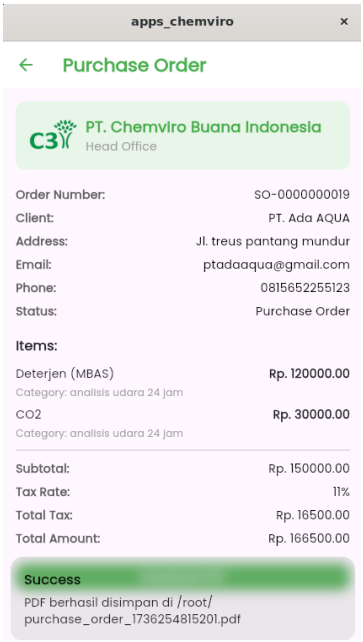
kategori pembelian dan ikon panah di sisi kanan untuk melihat rincian lebih lanjut dari pesanan tersebut. Tombol biru besar bertuliskan "*Purchase Order*" di bagian bawah setiap kartu menonjol sebagai aksi utama yang memungkinkan pengguna untuk mengelola pesanan dengan mudah. Latar belakang biru muda pada setiap kartu memberikan kesan profesional dan memudahkan pengguna untuk memfokuskan perhatian pada detail pesanan. Di bagian bawah layar, terdapat informasi Total *Sales Income* yang ditampilkan jelas untuk memudahkan pengguna melacak total pendapatan penjualan, seperti "Rp. 42735.00," mendukung pengalaman pengguna dengan navigasi intuitif dan akses langsung ke detail penting.

4.3.12 Tampilan *Purchase Order Detail (Purchase Order Detail Screen)*

Berikut adalah tampilan *purchase order* detail dari *Mobile Apps Sales*:



Gambar 4. **24** Tampilan *Purchase Order Detail*

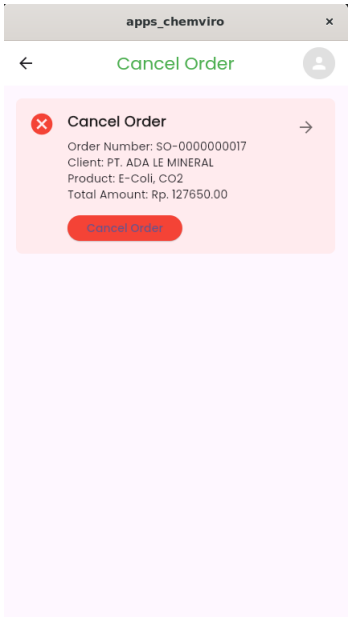


Gambar 4. **25** Tampilan *Purchase Order Screen*

Tampilan *Purchase Order* Detail ini memberikan informasi lengkap mengenai detail pesanan pembelian dengan tata letak yang rapi dan profesional. Bagian atas halaman menampilkan *header* berwarna hijau dengan judul "*Purchase Order*," diikuti oleh ikon panah di sisi kiri untuk kembali ke halaman sebelumnya. Di bawahnya, terdapat bagian yang menampilkan identitas perusahaan, termasuk logo C3, nama "PT. Chemviro Buana Indonesia," dan deskripsi "*Head Office*," yang diletakkan di dalam kotak dengan latar hijau pastel. Informasi detail pesanan disusun secara vertikal, meliputi Order *Number* ("SO-0000000019"), nama *Client* ("PT. Ada AQUA"), *Address* ("Jl. treus pantang mundur"), Email ("ptadaaqua@gmail.com"), *Phone* ("0815652255123"), dan Status pesanan ("*Purchase Order*"). Bagian *Items* menampilkan daftar produk yang dipesan, seperti "Deterjen (MBAS)" dengan harga Rp120.000,00 dan "CO2" dengan harga Rp30.000,00, disertai kategori layanan masing-masing ("analisis udara 24 jam"). Di bagian bawah, terdapat rincian biaya, termasuk *Subtotal* (Rp150.000,00), *Tax Rate* (11%), Total *Tax* (Rp16.500,00), dan Total *Amount* (Rp166.500,00). Tombol hijau besar bertuliskan "*Download* PDF" berada di bagian bawah untuk memudahkan pengguna mengunduh rincian pesanan. Ketika tombol ini diklik, muncul notifikasi hijau dengan label *"Success"* di bagian bawah layar, yang mengonfirmasi bahwa file PDF berhasil disimpan, dengan lokasi penyimpanan yang tercantum, seperti "/root/purchase_order_1736254815201.pdf." Penambahan notifikasi ini memberikan kejelasan kepada pengguna mengenai hasil tindakan yang dilakukan, memastikan pengalaman pengguna yang transparan dan efisien. Tampilan ini secara keseluruhan memadukan desain intuitif dengan aksesibilitas tinggi untuk memudahkan pengelolaan dan dokumentasi pesanan.

4.3.13 Tampilan *Cancel Order (Cancel Order Screen)*

Berikut adalah tampilan *cancel order* dari *Mobile Apps Sales*:

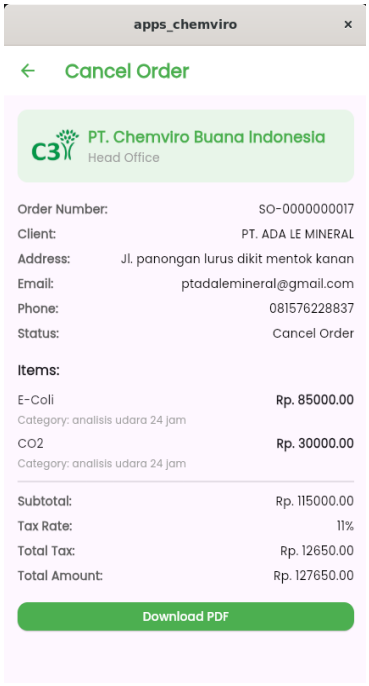


Gambar 4. **26** Tampilan *Cancel Order*

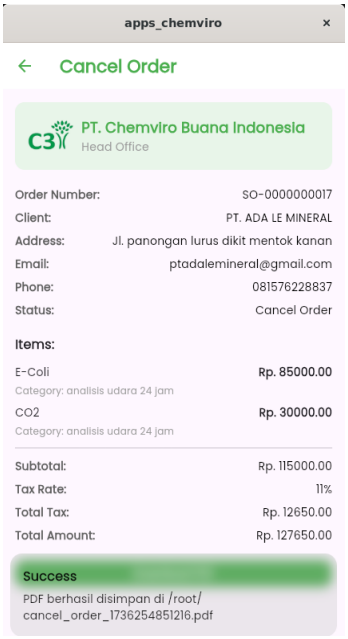
Tampilan *Cancel* Order ini dirancang untuk memberikan informasi terkait pesanan yang dibatalkan dengan visual yang jelas dan terstruktur. Bagian atas layar menampilkan *header* hijau dengan judul "*Cancel Order*," yang diikuti oleh ikon panah di sisi kiri untuk kembali ke halaman sebelumnya, serta ikon profil di sisi kanan untuk akses ke informasi pengguna. Di bagian utama, terdapat kartu berwarna merah muda yang menampilkan informasi pesanan secara detail. Informasi ini mencakup *Order Number* ("SO-0000000017"), nama Client ("PT. ADA LE MINERAL"), daftar *Product* yang dibatalkan ("E-Coli, CO2"), dan Total *Amount* sebesar Rp127.650,00. Di sisi kiri atas kartu terdapat ikon lingkaran merah dengan tanda silang sebagai simbol visual yang merepresentasikan status pembatalan pesanan, sementara di sisi kanan terdapat ikon panah untuk navigasi ke detail pesanan lebih lanjut jika diperlukan. Di bagian bawah kartu, terdapat tombol aksi berwarna merah bertuliskan "*Cancel Order*," yang memberikan akses langsung untuk mengonfirmasi atau menindaklanjuti pembatalan pesanan. Desain ini secara keseluruhan mengedepankan kejelasan dengan elemen warna yang mencolok untuk memfokuskan perhatian pengguna pada status pembatalan, sekaligus memberikan navigasi yang mudah dan intuitif untuk mengelola pesanan yang dibatalkan.

4.3.14 Tampilan *Cancel Order Detail (Cancel Order Detail Screen)*

Berikut adalah tampilan *cancel order* detail dari *Mobile Apps Sales*:



Gambar 4. **27** Tampilan *Cancel Order Detail*



Gambar 4. **28** Tampilan *Cancel Order Detail*

Tampilan *Cancel Order* Detail memberikan informasi rinci terkait pesanan yang dibatalkan dengan desain yang bersih dan terstruktur. Bagian atas layar menampilkan header hijau dengan judul "*Cancel Order,"* diikuti oleh ikon panah di sisi kiri untuk kembali ke halaman sebelumnya. Di bawahnya, terdapat informasi identitas perusahaan, termasuk logo C3, nama "PT. Chemviro Buana Indonesia," dan deskripsi "*Head Office*," dengan latar hijau pastel yang menonjol. Informasi detail pesanan disusun secara vertikal, mencakup Order Number ("SO-0000000017"), nama *Client* ("PT. ADA LE MINERAL"), Address ("Jl. panongan lurus dikit mentok kanan"), Email ("ptadalemineral@gmail.com"), Phone ("081576228837"), dan Status pesanan ("Cancel Order"). Bagian Items merinci produk yang dibatalkan, seperti "E-Coli" dengan harga Rp85.000,00 dan "CO2" dengan harga Rp30.000,00, serta kategori layanan masing-masing ("analisis udara 24 jam"). Di bagian bawah, terdapat rincian biaya, termasuk *Subtotal* (Rp115.000,00), *Tax Rate* (11%), *Total Tax* (Rp12.650,00), dan *Total Amount* (Rp127.650,00). Tombol hijau besar bertuliskan "*Download* PDF" memungkinkan pengguna untuk mengunduh detail pembatalan pesanan dalam format PDF. Ketika tombol ini diklik, muncul notifikasi hijau di bagian bawah layar dengan label *"Success"* yang mengonfirmasi bahwa file PDF berhasil disimpan di lokasi tertentu, seperti "/root/cancel_order_1736254851216.pdf."

4.3.15 Tampilan *Client (Client Screen)*

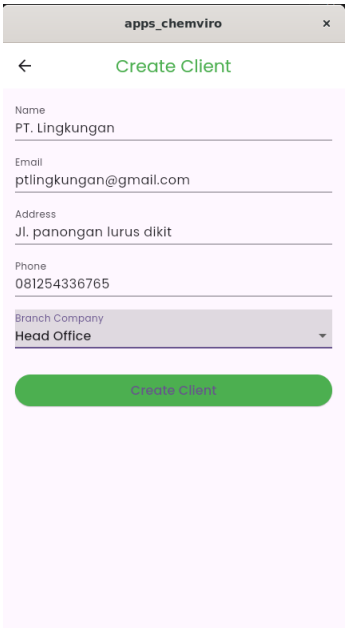
Berikut adalah tampilan client dari Mobile Apps Sales:

Gambar 4. **29** Tampilan *Client Screen*

Tampilan *Client* ini dirancang untuk menampilkan daftar klien dengan informasi yang lengkap dan terstruktur. Bagian atas layar menampilkan *header* hijau dengan judul "*Client,*" dilengkapi dengan ikon panah di sisi kiri untuk kembali ke halaman sebelumnya dan ikon profil di sisi kanan untuk akses informasi pengguna. Setiap klien ditampilkan dalam kartu dengan latar hijau muda, memberikan visual yang jelas dan membedakan setiap entri. Kartu tersebut mencakup nama klien, seperti "PT. Ada AQUA" dan "PT. ADA LE MINERAL," yang ditampilkan dengan huruf tebal untuk penekanan. Di bawah nama, terdapat informasi detail seperti Email (contoh: "ptadaaqua@gmail.com"), *Address* (contoh: "Jl. treus pantang mundur"), dan *Phone* (contoh: "0815652255123"), yang disusun secara vertikal untuk memudahkan pembacaan. Di sisi kiri atas setiap kartu, terdapat ikon orang berwarna hijau sebagai simbol representasi klien, sementara di sisi kanan terdapat ikon pensil abu-abu untuk mengedit informasi klien. Di bagian kanan bawah layar, terdapat tombol hijau berbentuk lingkaran dengan ikon "+" yang berfungsi untuk menambahkan klien baru. Desain keseluruhan mengedepankan aksesibilitas, kemudahan navigasi, dan tampilan yang rapi, mempermudah pengguna dalam mengelola daftar klien dengan efisien.

4.3.16 Tampilan *Create Client (Create Client Screen)*

Berikut adalah tampilan *create client* dari *Mobile Apps Sales*:

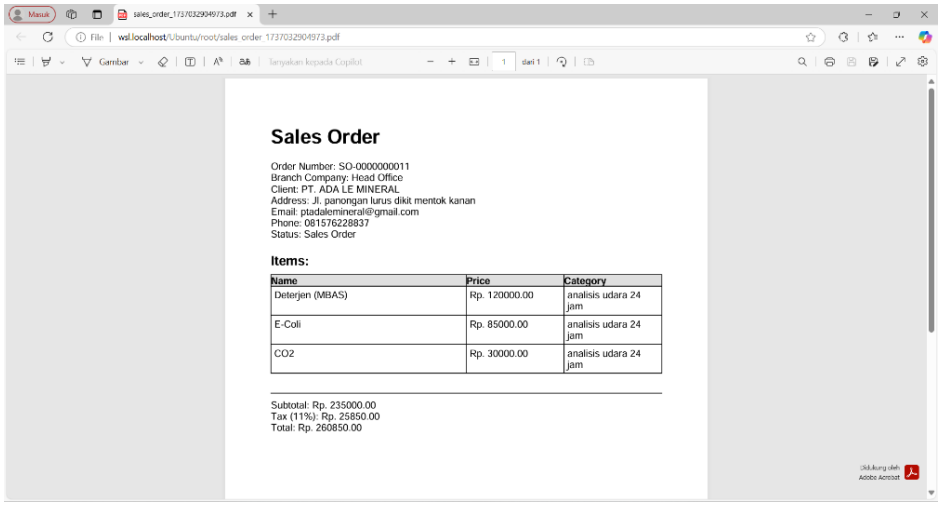


Gambar 4. **30** Tampilan *Create Client Screen*

Tampilan *Create Client* ini dirancang untuk memfasilitasi penambahan data klien baru dengan antarmuka yang sederhana dan intuitif. Bagian atas layar menampilkan *header* hijau dengan judul "*Create Client,"* diikuti oleh ikon panah di sisi kiri untuk kembali ke halaman sebelumnya. Formulir untuk menambahkan klien terdiri dari beberapa *field input,* dimulai dengan *Name,* tempat pengguna memasukkan nama klien (contoh: "PT. Lingkungan"). Di bawahnya terdapat *field* untuk Email (contoh: "ptlingkungan@gmail.com"), *Address* (contoh: "Jl. panongan lurus dikit"), dan *Phone* (contoh: "081254336765"), yang semuanya dirancang dalam format sederhana dengan garis pemisah yang mempermudah pembacaan dan pengisian. Pada bagian bawah, terdapat menu *drop-down* untuk memilih *Branch Company* (contoh: *"Head Office"),* memberikan opsi cabang perusahaan terkait klien. Di bagian paling bawah, terdapat tombol hijau besar bertuliskan *"Create Client"* yang menjadi aksi utama untuk menyimpan data klien baru. Desain ini menggunakan tata letak vertikal yang rapi dengan warna latar yang bersih, memastikan aksesibilitas dan kemudahan penggunaan. Tampilan ini memadukan fungsionalitas dan kesederhanaan untuk mempermudah pengguna dalam menambahkan klien baru dengan cepat dan efisien.

4.3.17 Tampilan PDF *(SO, PO, CO)*

Berikut adalah tampilan PDF dari Mobile Apps Sales:



Gambar 4. 31 Tampilan PDF Pesanan Penjualan

Tampilan PDF Sales Order berikut dirancang untuk memberikan informasi detail terkait pesanan penjualan. Bagian atas dokumen mencantumkan judul "Sales Order" diikuti dengan informasi penting, seperti Order Number, Branch Company, Client (misalnya, "PT. ADA LE MINERAL"), Address, Email, Phone, dan Status yang menunjukkan bahwa ini adalah dokumen Sales Order. Bagian Items menyajikan daftar produk yang dipesan dalam tabel yang terstruktur dengan kolom Name, Price, dan Category. Contohnya, produk seperti "Deterjen (MBAS)" memiliki harga Rp. 120,000.00 dengan kategori "analisis udara 24 jam." Tabel ini memudahkan pembaca untuk memahami daftar barang beserta detailnya secara cepat. Pada bagian bawah, terdapat rincian keuangan, termasuk Subtotal (Rp. 235,000.00), Tax (11%) (Rp. 25,850.00), dan Total yang dihitung secara otomatis (Rp. 260,850.00). Rincian ini memastikan transparansi dan akurasi dalam total pesanan. Desain dokumen ini sederhana dan profesional, memudahkan pengguna untuk memeriksa detail pesanan dengan cepat serta memberikan gambaran lengkap tentang transaksi dalam format yang jelas dan rapi. Begitupun pada tampilan untuk Purchase Order dan Cancel Order.

BAB 5
KESIMPULAN

Aplikasi Mobile Sales yang dirancang untuk PT Chemviro Buana Indonesia bertujuan meningkatkan efisiensi operasional, khususnya dalam manajemen aktivitas *Sales Order*. Aplikasi ini mengatasi masalah pencatatan data secara manual yang rentan kesalahan dan memakan waktu, serta keterbatasan data *real-time*. Dalam perancangan aplikasi, metode *Agile* diterapkan untuk memastikan bahwa setiap fitur yang dirancang memenuhi kebutuhan operasional dengan koordinasi yang erat antara tim pengembang dan pemangku kepentingan. Masalah utama yang diidentifikasi meliputi kurangnya pelatihan teknik bagi tim sales dan sistem manual yang terpisah, yang dapat mengganggu kinerja operasional. Untuk menganalisis akar masalah ini, metode *Fishbone* digunakan, dengan fokus pada kurangnya pelatihan teknis.

Aplikasi ini menawarkan berbagai fitur unggulan, seperti autentikasi pengguna untuk keamanan data dan pengelolaan pesanan yang mencakup *Sales Order*, *Purchase Order*, dan *Cancel Order*. Selain itu, aplikasi ini menyediakan pelaporan pendapatan dan kinerja secara real-time serta integrasi dengan sistem CRM perusahaan untuk meningkatkan transparansi antar-departemen. Hal ini mempermudah pemantauan pesanan dan pengambilan keputusan strategis dengan lebih akurat. Implementasi aplikasi terbukti meningkatkan produktivitas tim sales, mengurangi proses manual, mempercepat alur kerja, dan meminimalkan kesalahan operasional. Dengan arsitektur *Layered* dan penggunaan *Flutter*, aplikasi ini mudah untuk dikembangkan lebih lanjut untuk memenuhi kebutuhan perusahaan di masa depan. Pada dasarnya, perancangan aplikasi ini tidak hanya mendukung efisiensi operasional, tetapi juga menjadi fondasi untuk pertumbuhan bisnis yang lebih kompetitif dan berkelanjutan di era digital.

DAFTAR REFERENSI

[1] S. Mulyani, *Metode Analisis dan Perancangan Sistem*, Edisi Kedua. bandung: Abdi Sistematika, 2016.

[2] E. Sutanto, *Pemrograman Android Dengan Menggunakan Eclipse & StarUML*. Airlangga University Press, 2020.

[3] R. D. Prehanto and D. kadek I. Nuryana, *BUKU AJAR KONSEP SISTEM INFORMASI*. Scopindo Media Pustaka, 2020.

[4] F. R. Sari and A. Utami, *REKAYASA PERANGKAT LUNAK BERORIENTASI OBJEK MENGGUNAKAN PHP*, 1st, Cetakan 1 ed. Yogyakarta: Andi, 2021.

[5] Y. A. Rozzi, J. Fredricka, and E. P. Arimi, *Sistem Monitoring Kualitas Udara dengan Aplikasi Thinger.io*. NEM, 2023.

[6] H. I. Yunarto, *Business Concepts Implementation Series IN SALES AND DISTRIBUTION MANAGEMENT*. Jakarta: PT Elex Media Komputindo, 2006.

[7] S. I. Kalb, *Zero-Budget Marketing*, 2nd Edition revised. United States: K&A Press, 1996.

[8] United States. Department of the Army, *Army Handbook for Purchasing and Contracting Officers*. Washington: U.S. Government Printing Office, 1954.

[9] C. Garner, *Trading Commodities, Commodity Options and Currencies (Collection)*. FT Press, 2012.

[10] SURYANI, *CUSTOMER RELATIONSHIP MANAGEMENT (CRM) dalam RISET PEMASARAN*. Pascal Books, 2022.

[11] I. R. Akbar, *MANAJEMEN HUBUNGAN PELANGGAN (Customer Relationship Management)*, 01 ed. Yayasan Sahabat Alam Rafflesia, 2021.

[12] F. J. Hair, R. Anderson, R. Mehta, and B. Babin, *Sales Force Management*. Wiley, 2020.

[13] R. Manglik, *Mobile Operating Systems*. EduGorilla Publication, 2024.

[14] L. Collins and R. S. Ellis, Eds., *Mobile Devices:Tools and Technologies*, 1st Edition. New York: Chapman and Hall/CRC, 2015. doi: 10.1201/b18165.

[15] T. Bailey and A. Biessek, *Flutter for Beginners:Cross-platform Mobile Development from Hello, World! to App Release with Flutter 3.10+ and Dart 3.x*. Packt Publishing, 2023.

[16] D. Chopra and R. Khurana, *Flutter and Dart: Up and Running: Build Native Apps for Both IOS and Android Using a Single Codebase (English Edition)*. Walter de Gruyter GmbH, 2023.

[17] A. Vemula, *Flutter for Mobile App Development: From Idea to App Store*. Anand Vemula, 2023.

[18] K. Singh, *Laravel for Beginners*. Karamvir Singh, 2021.

[19] M. Stuffer, *Laravel: Up & Running*. O'Reilly Media, 2023.

[20] M. Masse, *REST API Design Rulebook*. O'Reilly Media, 2011.

[21] S. Patni, *Pro RESTful APIs: Design, Build and Integrate with REST, JSON, XML and JAX-RS*. Apress, 2017.

[22] bin S. Uzayr, *Learning WordPress REST API*. Packt Publishing.

[23] J. Turnbull, *The Docker Book: Containerization Is the New Virtualization*. James Turnbull, 2014.

[24] *How Docker Enables DevOps, Continuous Integration and Continuous Delivery Docker From Code to Container EBOOK*.

[25] C. Ürtekin, *Advanced Flutter: Databases and Layered Architecture*. Cihan Ürtekin, 2023.

[26] Harahap. Febriani Elfira, S. Adisuwiryo, and R. Fitriana, *Analisis dan Perancangan Sistem Informasi*, Pertama. Banyumas: Wawasan Ilmu, 2022.

[27] D. Effendi *et al.*, *Panduan dalam Pengembangan Perangkat Lunak*, Cetakan Pertama. Bandung: Kaizen Media Publishing, 2024.

[28] B. Sucipto and H. D. Kushendar, *PENGAMBILAN KEPUTUSAN dan KEPEMIMPINAN (Panduan Teori dan Konsep Bagi Mahasiswa Program Sarjana dan Magister)*, Cetakan Pertama. Indramayu: Adab, 2023.

[29] Y. B. Setyawan, *ALUR PIKIR AKTUALISASI*, Cetakan Pertama. Pusat Pengembangan Pendidikan dan Penelitian Indonesia, 2022.

[30] N. Hidayat *et al.*, *TOTAL QUALITY CONTROL*. Yayasan Cendikia Mulia Mandiri, 2024.