PENGEMBANGAN APLIKASI MOBILE *POINT OF SALE* (POS) PADA UMKM PABRIK ABON MENGGUNAKAN PENDEKATAN LEAN STARTUP UNTUK MEWUJUDUKAN KEBERLANJUTAN USAHA

Disusun Oleh:

N a m a : Fareesdzy Akhtarabillah Setiawan

NIM : 21523140

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2026

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PENGEMBANGAN APLIKASI MOBILE *POINT OF SALE* (POS) PADA UMKM PABRIK ABON MENGGUNAKAN PENDEKATAN LEAN STARTUP UNTUK MEWUJUDUKAN KEBERLANJUTAN USAHA

TUGAS AKHIR

Disusun Oleh:

N a m a : Fareesdzy Akhtarabillah Setiawan
NIM : 21523140

Yogyakarta, 22 Desember 2025

Pembimbing,

Kholid Haryono, S.T., M.Kom.

HALAMAN PENGESAHAN DOSEN PENGUJI

PENGEMBANGAN APLIKASI MOBILE *POINT OF SALE* (POS) PADA UMKM PABRIK ABON MENGGUNAKAN PENDEKATAN LEAN STARTUP UNTUK MEWUJUDUKAN KEBERLANJUTAN USAHA

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 13 Januari 2026

Tim Penguji

Kholid Haryono, S.T., M.Kom. _____

Anggota 1

Chandra Kusuma Dewa, S.Kom., M.Cs, Ph.D. _____

Anggota 2

Moh. Idris, S.Kom., M.Kom. _____

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Fareesdzy Akhtarabillah Setiawan

NIM : 21523140

Tugas akhir dengan judul:

PENGEMBANGAN APLIKASI MOBILE *POINT OF SALE* (POS) PADA UMKM PABRIK ABON MENGGUNAKAN PENDEKATAN LEAN STARTUP UNTUK MEWUJUDUKAN KEBERLANJUTAN USAHA

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 22 Desember 2025

Fareesdzy Akhtarabillah Setiawan

HALAMAN PERSEMBAHAN

I dedicate this thesis to everyone who has ever crossed paths with me—
those who stayed, those who left, those who taught me lessons, and those who unknowingly shaped the person I have become.
Each encounter, whether brief or lasting, has played a role in this journey.
This work stands as a quiet tribute to all of you.

HALAMAN MOTO

"If you want to be successful, you need to have total dedication, seek your ultimate limit, and give your best."

-Ayrton Senna-

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Allah SWT atas segala rahmat dan karunia-Nya, sehingga laporan tugas akhir yang berjudul "Pengembangan Aplikasi Mobile Point of Sale (POS) pada UMKM Pabrik Abon Menggunakan Pendekatan Lean Startup untuk Mewujudkan Keberlanjutan Usaha" dapat diselesaikan dengan baik.

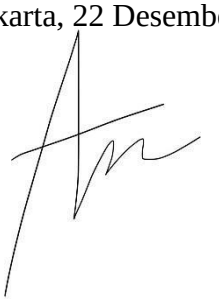
Dalam proses penyusunan tugas akhir ini, penulis banyak menerima bantuan, dukungan, dan bimbingan dari berbagai pihak. Oleh karena itu, dengan segala kerendahan hati dan penuh rasa syukur, penulis menyampaikan terima kasih kepada:

1. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D., selaku Ketua Program Studi Informatika Universitas Islam Indonesia, atas arahan, bimbingan, dan wawasan yang telah memperluas pemahaman penulis dalam bidang informatika.
2. Bapak Kholid Haryono, S.T., M.Kom., selaku dosen pembimbing tugas akhir, atas bimbingan, arahan, serta kesabaran yang telah beliau berikan sepanjang proses penyusunan tugas akhir ini.
3. Ibu Erika Ramadhani, S.T., M.Eng., selaku dosen pembimbing akademik, atas perhatian dan dukungan yang telah diberikan sehingga membantu kelancaran penyelesaian studi penulis.
4. Keluarga Tercinta, Penulis mengucapkan terima kasih yang sebesar-besarnya kepada keluarga tercinta atas doa, dukungan, kesabaran, serta motivasi yang tiada henti selama proses penyelesaian studi dan penyusunan tugas akhir ini.
5. Keluarga Besar Zenith 21, atas kebersamaan, dukungan, dan semangat yang telah mengiringi perjalanan penulis sehingga proses ini terasa lebih bermakna dan tidak dijalani sendiri.
6. Penulis mengucapkan terima kasih kepada seluruh teman yang telah memberikan dukungan selama proses penyelesaian studi, yang tidak dapat disebutkan satu per satu.
7. Penulis menyampaikan terima kasih kepada pemilik pabrik abon Merak Bewangi atas kesediaan, kerja sama, dan keterbukaan yang diberikan kepada penulis selama pelaksanaan penelitian dan pengembangan tugas akhir ini.

Penulis menyadari bahwa laporan tugas akhir ini masih memiliki keterbatasan dan belum sepenuhnya sempurna. Oleh karena itu, kritik dan saran yang bersifat membangun sangat diharapkan sebagai bahan perbaikan dan pengembangan di masa mendatang. Penulis berharap

laporan ini tidak hanya menjadi bagian dari pemenuhan persyaratan akademik, tetapi juga dapat memberikan manfaat bagi pembaca, khususnya dalam pengembangan dan penerapan sistem informasi berbasis mobile pada UMKM. Seluruh proses dan usaha yang telah dilalui penulis persembahkan kepada pihak-pihak yang senantiasa memberikan dukungan, doa, dan semangat, serta kepada Allah SWT yang telah memberikan rahmat dan kekuatan sehingga tugas akhir ini dapat diselesaikan.

Yogyakarta, 22 Desember 2025



Fareesdzy Akhtarabillah Setiawan

SARI

Usaha Mikro, Kecil, dan Menengah (UMKM) memiliki peran penting dalam perekonomian Indonesia, namun masih banyak yang mengalami kendala operasional akibat pencatatan pesanan dan penjualan yang dilakukan secara manual. Kondisi ini berpotensi menimbulkan ketidakteraturan data, kesalahan pencatatan, serta kesulitan dalam memantau perkembangan usaha secara berkelanjutan, sebagaimana yang juga dialami oleh pabrik abon Merak Bewangi.

Penelitian ini bertujuan mengembangkan aplikasi *mobile Point of Sale* (POS) untuk mendukung pencatatan operasional pabrik abon Merak Bewangi secara digital serta menguji kesesuaiannya dalam mendukung keberlanjutan usaha. Pengembangan aplikasi dilakukan menggunakan pendekatan *Lean Startup* melalui dua iterasi, yaitu pengembangan *Minimum Viable Product* (MVP) dan penyempurnaan sistem berdasarkan umpan balik pengguna.

Metode pengumpulan data dilakukan melalui wawancara dan observasi terhadap pemilik dan karyawan pabrik abon Merak Bewangi. Pengujian sistem menggunakan metode *Black Box Testing* untuk memastikan fungsionalitas aplikasi serta *System Usability Scale* (SUS) untuk mengukur kemudahan penggunaan. Hasil pengujian menunjukkan bahwa seluruh fitur berfungsi dengan baik dan aplikasi memiliki tingkat *usability* yang baik serta dapat diterima oleh pengguna.

Temuan penelitian menunjukkan bahwa aplikasi mobile POS yang dikembangkan mampu menggantikan proses pencatatan manual menjadi sistem digital yang lebih terstruktur, efisien, dan mudah digunakan. Kemudahan penggunaan sistem mendukung konsistensi operasional serta mengurangi ketergantungan pada individu tertentu, sehingga informasi usaha dapat terdokumentasi dan dilanjutkan oleh generasi pengelola berikutnya. Dengan demikian, penelitian ini menyimpulkan bahwa penerapan aplikasi mobile POS berbasis pendekatan Lean Startup berpotensi mendukung keberlanjutan usaha UMKM melalui digitalisasi proses pencatatan dan pengelolaan operasional secara berkelanjutan.

Kata kunci: UMKM, Point of Sale, Lean Startup, System Usability Scale, Keberlanjutan Usaha

GLOSARIUM

Activity Diagram	Diagram dalam Unified Modeling Language (UML) yang digunakan untuk menggambarkan alur aktivitas atau proses bisnis dalam suatu sistem secara berurutan.
API (Application Programming Interface)	Antarmuka yang memungkinkan komunikasi dan pertukaran data antara aplikasi frontend dan backend secara terstruktur.
Backend	Bagian dari sistem aplikasi yang menangani logika bisnis, pengolahan data, dan interaksi dengan basis data.
Black Box Testing	Metode pengujian perangkat lunak yang berfokus pada pengujian fungsional sistem tanpa memperhatikan struktur internal atau kode program.
Build–Measure–Learn	Siklus utama dalam metode Lean Startup yang terdiri dari tahap membangun produk, mengukur respons pengguna, dan mengambil pembelajaran untuk perbaikan.
Dashboard	Tampilan antarmuka yang menyajikan ringkasan informasi penting seperti data penjualan dan pesanan secara visual.
Database	Kumpulan data terstruktur yang disimpan secara sistematis dan dapat diakses serta dikelola oleh sistem aplikasi.
Entity Relationship Diagram	Diagram yang menggambarkan hubungan antar entitas dalam basis data beserta atribut dan relasinya.
Flowchart	Diagram alur yang digunakan untuk memvisualisasikan langkah-langkah proses atau logika kerja suatu sistem.
Frontend	Bagian dari aplikasi yang berinteraksi langsung dengan pengguna melalui antarmuka pengguna (user interface).
Lean Startup	Metodologi pengembangan produk yang menekankan validasi berkelanjutan melalui siklus iteratif Build–Measure–Learn untuk meminimalkan risiko kegagalan.
Minimum Viable Product	Versi awal produk yang dikembangkan dengan fitur minimum namun telah mencakup fungsi inti untuk diuji oleh pengguna.
Mobile Application	Aplikasi perangkat lunak yang dirancang untuk dijalankan pada perangkat bergerak seperti smartphone atau tablet.
Point of Sale (POS)	Sistem yang digunakan untuk mencatat dan mengelola transaksi penjualan, data produk, serta laporan penjualan secara terintegrasi.
React Native	Framework open-source yang digunakan untuk membangun aplikasi mobile lintas platform dengan satu basis kode.

Supabase	Platform Backend as a Service (BaaS) yang menyediakan layanan basis data, autentikasi, dan API secara terintegrasi.
System Usability Scale (SUS)	Metode evaluasi usability sistem berbasis kuesioner yang digunakan untuk mengukur tingkat kemudahan penggunaan dan penerimaan pengguna
UMKM (Usaha Mikro, Kecil, dan Menengah)	Unit usaha produktif yang dikelola secara perorangan atau badan usaha dengan skala mikro, kecil, atau menengah sesuai peraturan perundang-undangan.
Usability	Tingkat kemudahan suatu sistem untuk dipahami, dipelajari, dan digunakan oleh pengguna.
Use Case Diagram	Diagram UML yang menggambarkan interaksi antara aktor dan sistem berdasarkan fungsi-fungsi yang disediakan.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Usaha Mikro, Kecil, dan Menengah (UMKM) merupakan tulang punggung perekonomian di Indonesia, memberikan kontribusi signifikan terhadap produk domestik bruto serta penyerapan tenaga kerja. Meskipun perannya krusial, banyak UMKM terutama yang bergerak di sektor industri rumahan seperti pabrik abon masih menghadapi tantangan mendasar dalam hal efisiensi operasional. Salah satu tantangan utama adalah proses manajemen pesanan dan pencatatan penjualan yang masih dilakukan secara manual menggunakan buku tulis (Putra & Hartono, 2024). Praktik ini berpotensi menimbulkan berbagai kendala, seperti ketidaktepatan pencatatan, risiko kehilangan data, waktu proses yang lebih lama, serta sulitnya pemilik usaha memperoleh informasi operasional secara cepat dan akurat.

Pabrik abon Merak Bewangi yang berlokasi di Boyolali merupakan salah satu UMKM yang telah berdiri sejak tahun 1970 dan masih mempertahankan proses operasional secara manual. Pencatatan transaksi, pencatatan data mitra, dan laporan penjualan masih dilakukan menggunakan buku tulis atau catatan terpisah dan mengandalkan ingatan pemiliknya saat ini. Hal ini menimbulkan beberapa kendala, seperti risiko kehilangan data, kesalahan perhitungan, lambatnya proses *input* data, seringnya pemilik usaha lupa dalam mencatat pesanan dan mencatat mitra baru, serta sulitnya pemilik usaha dalam memantau perkembangan usaha secara *real-time*.

Dalam wawancara awal bersama pemilik usaha, Ia menyatakan kondisi usaha pabrik abon ini terancam untuk sulit untuk diturunkan ke generasi berikutnya, sehingga tidak bisa berlanjut jika pengelola saat ini tidak ada, maka perlu solusi agar supaya usaha ini dikembangkan lebih sistematis dan dapat dilanjutkan oleh generasi berikutnya

Untuk menjawab kebutuhan tersebut, dikembangkanlah aplikasi *mobile* POS (*Point of Sales*) yang dirancang khusus untuk mendukung proses transaksi sistematis dan penyajian laporan secara otomatis sehingga pergantian generasi berikutnya meninggalkan sistem yang digunakan untuk melanjutkan usaha. Dalam proses pengembangan aplikasi ini, digunakan pendekatan *Lean Startup*, sebuah metodologi modern yang berfokus pada pengembangan produk digital melalui siklus *Build–Measure–Learn* (Sri Rahayu & Tata Sutabri, 2024). Metode ini cocok dipakai dalam penelitian ini karena pemilik usaha belum mampu mendeskripsikan kebutuhan sistemnya dengan jelas (Ningsih et al., 2023).

Melalui pendekatan *Lean Startup*, pengembangan aplikasi dilakukan secara iteratif dengan melibatkan pemilik dan pegawai sebagai pengguna utama. Pendekatan ini memungkinkan validasi fitur secara langsung, meminimalkan pemborosan, serta memastikan bahwa aplikasi yang dikembangkan benar-benar sesuai dengan kebutuhan operasional pabrik abon Merak Bewangi.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, rumusan masalah dalam penelitian ini adalah:

a. Bagaimana mengembangkan aplikasi mobile POS untuk mewujudkan potensi keberlanjutan usaha?
b. Bagaimana aplikasi ini diuji kesesuainnya untuk mencapai keberlanjutan usaha?

1.3 Batasan Masalah

Agar penelitian tetap terarah dan fokus, batasan masalah yang digunakan adalah sebagai berikut:

a. Proses pengembangan mengikuti pendekatan Lean Startup dengan fokus pada pembuatan MVP (*Minimum Viable Product*).
b. Pengujian dilakukan oleh pihak internal pabrik abon Merak Bewangi (pemilik dan pegawai).

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah mengembangkan aplikasi *mobile* POS yang mampu mendukung proses pencatatan pesanan dan penjualan di pabrik abon Merak Bewangi, sekaligus mengimplementasikan pendekatan *Lean Startup* agar fitur-fitur yang dikembangkan benar-benar sesuai dengan kebutuhan pengguna untuk keberlanjutan usaha. Selain itu, penelitian ini juga bertujuan untuk menguji kesesuaian agar mencapai keberlanjutan usaha.

1.5 Manfaat Penelitian

Penelitian ini diharapkan memberikan manfaat baik secara teoritis maupun praktis. Secara teoritis, penelitian ini dapat menambah referensi penerapan metode *Lean Startup* dalam pengembangan aplikasi *mobile* pada UMKM tradisional, serta menjadi studi pendukung dalam literatur pengembangan sistem POS berbasis *mobile*. Secara praktis, aplikasi yang dikembangkan diharapkan mampu membantu pabrik abon Merak Bewangi dalam melakukan

pencatatan transaksi dan pesanan secara lebih efektif dan efisien, mengurangi risiko kesalahan *input* serta kehilangan data, memudahkan pemilik usaha dalam memantau laporan penjualan, dan mendukung proses digitalisasi UMKM agar lebih adaptif terhadap perkembangan teknologi.

1.6 Metodologi Penelitian

Penelitian ini menggunakan metodologi **Lean Startup**, yang menekankan pada pengembangan produk secara iteratif, efisien, dan berorientasi langsung pada kebutuhan pengguna. Metode ini dipilih karena sesuai dengan tujuan penelitian untuk menghasilkan aplikasi POS yang benar-benar dibutuhkan oleh pengguna pabrik abon Merak Bewangi. Siklus *Build–Measure–Learn* yang dilakukan secara berulang digunakan sebagai dasar dalam setiap tahap pengembangan (Shepherd & Patzelt, 2021), sehingga setiap versi aplikasi yang dihasilkan selalu divalidasi oleh pengguna utama, yaitu pemilik dan pegawai pabrik.

a. *Build*

Tahap ini berfokus pada pembuatan *Minimum Viable Product* (MVP) dari aplikasi POS. Pada setiap siklus *Build–Measure–Learn,* pengguna utama seperti pemilik dan pegawai pabrik Merak Bewangi dilibatkan secara langsung untuk mencoba prototipe awal dan memberikan masukan. Umpan balik dan hasil observasi terhadap penggunaan aplikasi menjadi landasan untuk menganalisis kebutuhan, memvalidasi fitur, serta menentukan perbaikan yang perlu dilakukan pada tahap pengembangan berikutnya. MVP yang dikembangkan terdiri dari fitur inti, yaitu manajemen produk, manajemen mitra, manajemen akun, pencatatan pesanan, dan tampilan dashboard dasar. Versi awal ini dibuat sesederhana mungkin agar dapat segera diuji kepada pengguna dengan investasi pengembangan minimal.

b. *Measure*

Setelah MVP selesai dibangun, tahap Measure dilakukan untuk menilai tingkat penerimaan dan kemudahan penggunaan aplikasi oleh pengguna. Pemilik dan pegawai mencoba langsung fitur inti aplikasi, sementara peneliti mengamati proses penggunaan serta kendala yang ditemui. Masukan terkait alur kerja, tampilan, maupun fungsi aplikasi dicatat secara sistematis sebagai dasar evaluasi.

c. *Learn*

Pada tahap Learn, seluruh data dan masukan yang diperoleh dari tahap Measure dianalisis untuk mendapatkan insight mengenai kebutuhan pengguna, pengalaman penggunaan, serta performa aplikasi. Hasil analisis ini digunakan untuk memvalidasi MVP

dan menentukan perbaikan fitur pada siklus pengembangan berikutnya. Proses ini memastikan bahwa setiap fitur aplikasi yang dikembangkan telah divalidasi secara langsung oleh pengguna sehingga risiko kegagalan dapat diminimalkan.

Proses *Build–Measure–Learn* dilakukan secara berulang, di mana setiap iterasi menghasilkan versi aplikasi yang lebih matang dan sesuai dengan kebutuhan operasional pabrik abon Merak Bewangi. Penelitian ini dibatasi hingga iterasi kedua sesuai ruang lingkup yang telah ditetapkan.

1.7 Sistematika Penelitian

Untuk memberikan gambaran yang jelas mengenai keseluruhan isi laporan skripsi, berikut adalah sistematika penulisan yang digunakan:

a. BAB I PENDAHULUAN

   Bab ini berisi latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, serta sistematika penulisan.

b. BAB II LANDASAN TEORI DAN KAJIAN PUSTAKA

   Bab ini menguraikan teori-teori yang relevan dengan penelitian, seperti UMKM, sistem *Point of Sale* (POS), pengembangan aplikasi *mobile*, serta penelitian terdahuluan yang sejenis.

c. BAB III METODE PENELITIAN

   Bab ini menjelaskan tentang kerangka kerja penelitian, metode pengumpulan data, serta tahapan perancangan dan pengembangan sistem menggunakan metode *Lean Startup*.

d. BAB IV HASIL DAN PEMBAHASAN

   Bab ini menyajikan hasil dari perancangan dan implementasi sistem, pengujian fungsionalitas menggunakan metode *Black Box Testing* dan *System Usability Scale*, serta pembahasan mengenai hasil yang dicapai.

e. BAB V KESIMPULAN DAN SARAN

   Bab ini berisi kesimpulan dari keseluruhan hasil penelitian serta saran untuk pengembangan sistem di masa mendatang.

BAB II
LANDASAN TEORI DAN KAJIAN PUSTAKA

2.1 Landasan Teori

Bab ini membahas landasan teori yang digunakan sebagai dasar konseptual dan teknis dalam penelitian pengembangan aplikasi *Point of Sale* (POS) berbasis Android pada pabrik abon Merak Bewangi. Landasan teori disusun secara sistematis dan terstruktur untuk menjelaskan konsep-konsep utama yang relevan sebagai pijakan ilmiah penelitian, mulai dari karakteristik Usaha Mikro, Kecil, dan Menengah (UMKM) sebagai objek penelitian, konsep dan fungsi sistem *Point of Sale* (POS) sebagai solusi digital yang ditawarkan, metodologi Lean Startup dengan pendekatan *Minimum Viable Product* (MVP) sebagai kerangka pengembangan sistem, hingga teknologi pengembangan perangkat lunak serta metode pengujian dan evaluasi sistem yang digunakan. Penyusunan landasan teori ini bertujuan untuk memberikan kerangka pemahaman yang komprehensif dan terintegrasi sehingga penelitian memiliki dasar ilmiah yang kuat, logis, dan terarah, serta mampu mendukung proses pengembangan sistem secara iteratif yang berfokus pada validasi kebutuhan dan pengalaman pengguna.

2.1.1 Usaha Mikro, Kecil, dan Menengah (UMKM)

Usaha Mikro, Kecil, dan Menengah (UMKM) didefinisikan sebagai unit usaha produktif yang dimiliki oleh entitas perorangan maupun badan usaha yang memenuhi kriteria sebagai usaha mikro sebagaimana diatur dalam perundang-undangan. Menurut Undang-Undang Nomor 20 Tahun 2008 Tentang Usaha Mikro, Kecil, Dan Menengah (UMKM), UMKM memegang peranan strategis dalam struktur perekonomian nasional, utamanya sebagai pilar penyerapan tenaga kerja dan instrumen pemerataan pembangunan ekonomi. Walaupun demikian, eksistensi UMKM kerap dihadapkan pada tantangan operasional yang berpotensi menghambat skalabilitasnya (Mendrofa et al., 2025). Tantangan tersebut mencakup manajemen inventaris yang kurang akurat yang berisiko pada *overstocking* atau *stockout*, proses pencatatan transaksi yang bersifat manual sehingga rentan terhadap kesalahan manusia, serta kesulitan dalam menyusun laporan keuangan yang valid untuk analisis kinerja penjualan dan perencanaan strategis jangka panjang.

2.1.2 Sistem *Point of Sale* (POS)

Point of Sale (POS) merupakan sistem yang digunakan untuk mencatat dan memproses

transaksi penjualan secara terintegrasi dalam satu platform (Ismul Afriza & Kurniawan Pakpahan, 2023). Pada perkembangannya, sistem POS tidak hanya berfungsi sebagai alat pencatat transaksi penjualan, tetapi juga mencakup pengelolaan data produk, mitra atau pelanggan, pengaturan harga, serta penyajian laporan penjualan secara periodik dan *real-time*. Penerapan sistem POS berbasis digital memberikan kemudahan dalam proses pencatatan transaksi, meningkatkan akurasi dan konsistensi data (Andriasari et al., 2024), serta mempercepat akses informasi penjualan yang dibutuhkan oleh pemilik usaha. Selain itu, sistem POS digital mampu mengurangi ketergantungan pada pencatatan manual yang rawan kesalahan dan kehilangan data. Bagi UMKM, sistem POS menjadi solusi strategis untuk meningkatkan efisiensi operasional dan profesionalisme pengelolaan usaha, sekaligus menggantikan proses manual yang kurang efektif dengan sistem terkomputerisasi yang lebih terstruktur, terintegrasi, dan dapat diandalkan sebagai dasar pengambilan keputusan bisnis.

2.1.3 *Lean Startup*

Lean Startup merupakan metodologi pengembangan produk yang diperkenalkan oleh Eric Ries dengan tujuan meminimalkan risiko kegagalan produk melalui proses validasi berkelanjutan terhadap kebutuhan pengguna. Metodologi ini menekankan pengembangan produk secara iteratif melalui siklus *Build–Measure–Learn*, di mana setiap versi produk diuji langsung kepada pengguna untuk memperoleh umpan balik nyata. Pendekatan *Lean Startup* sangat sesuai untuk pengembangan produk digital, khususnya aplikasi, yang kebutuhan dan preferensi penggunanya dapat berkembang seiring waktu.

Dalam konteks penelitian ini, *Lean Startup* dipilih karena aplikasi *Point of Sale* (POS) yang dikembangkan merupakan solusi digital baru yang belum pernah diterapkan sebelumnya pada pabrik abon Merak Bewangi, sehingga tingkat ketidakpastian terhadap kebutuhan dan penerimaan pengguna masih cukup tinggi. Dengan menggunakan pendekatan *Lean Startup*, proses pengembangan aplikasi dapat dilakukan secara bertahap dan adaptif melalui iterasi yang berkelanjutan (Isabel & Pereira, 2022). Pendekatan ini memungkinkan peneliti untuk memperoleh umpan balik langsung dari pengguna sejak tahap awal pengembangan (Amelia & Vidiati, 2025) .Dengan demikian, pengembangan aplikasi tidak hanya berfokus pada penyelesaian aspek teknis semata, tetapi juga menekankan kesesuaian solusi yang dibangun dengan kebutuhan operasional nyata pengguna di lapangan, serta meningkatkan peluang aplikasi untuk diterima dan digunakan secara berkelanjutan.

2.1.4 Siklus *Build-Measure-Learn*

Siklus *Build–Measure–Learn* merupakan inti dari metodologi *Lean Startup* yang menekankan proses pengembangan produk secara iteratif dan berkelanjutan. Tahap Build berfokus pada pembangunan *Minimum Viable Product* (MVP), yaitu versi awal aplikasi POS yang dirancang dengan fitur inti yang paling dibutuhkan oleh pengguna, seperti pencatatan pesanan, pengelolaan produk, dan penyajian *dashboard* penjualan. Pengembangan MVP dilakukan dengan mempertimbangkan kebutuhan utama pengguna dan permasalahan operasional yang dihadapi, tanpa menambahkan fitur yang bersifat kompleks (Cook et al., 2023). Dengan lingkup fitur yang terbatas, MVP dapat dikembangkan dalam waktu yang relatif singkat sehingga memungkinkan sistem untuk segera diuji oleh pengguna sebagai sarana validasi awal terhadap fungsi dan manfaat aplikasi yang dikembangkan.

Tahap *Measure* dilakukan untuk mengukur respons dan pengalaman pengguna terhadap MVP melalui berbagai metode, seperti observasi langsung terhadap penggunaan aplikasi, pengujian fungsional untuk memastikan setiap fitur berjalan sesuai kebutuhan, serta evaluasi usabilitas guna menilai tingkat kemudahan penggunaan sistem. Data yang diperoleh pada tahap ini digunakan untuk menilai efektivitas sistem dalam mendukung proses operasional pengguna, sekaligus mengidentifikasi kendala, kekurangan, maupun potensi perbaikan yang diperlukan (Kaylan et al., 2022).

Tahap *Learn* merupakan proses analisis terhadap seluruh hasil pengukuran yang diperoleh pada tahap sebelumnya untuk mendapatkan pembelajaran yang komprehensif mengenai kelebihan dan kekurangan system (Yoo et al., 2021). Pembelajaran ini tidak hanya berfokus pada aspek teknis, tetapi juga mencakup pengalaman dan persepsi pengguna terhadap sistem. Hasil pembelajaran tersebut selanjutnya dijadikan dasar dalam menentukan perbaikan, penyempurnaan, dan pengembangan fitur pada iterasi berikutnya agar sistem semakin sesuai dengan kebutuhan pengguna.

2.1.5 *Minimum Viable Product* (MVP)

Minimum Viable Product (MVP) adalah versi awal produk yang dikembangkan dengan fitur minimum namun telah mencakup fungsi inti sehingga cukup untuk digunakan dan dievaluasi oleh pengguna. Tujuan utama pengembangan MVP adalah untuk memperoleh umpan balik pengguna secara cepat dan relevan dengan biaya serta waktu pengembangan yang relatif minimal (Lortie et al., 2025). Melalui MVP, pengembang dapat memvalidasi asumsi awal terkait kebutuhan pengguna tanpa harus membangun sistem secara penuh sejak tahap awal. Dalam penelitian ini, MVP aplikasi POS digunakan sebagai sarana validasi awal untuk

memastikan bahwa sistem yang dikembangkan benar-benar mampu menggantikan proses pencatatan manual yang sebelumnya digunakan di pabrik abon Merak Bewangi, sekaligus menjadi dasar dalam menentukan arah pengembangan dan penyempurnaan fitur pada iterasi selanjutnya.

2.1.6 React Native

React Native merupakan sebuah kerangka kerja (*framework*) *open-source* yang dikembangkan oleh Meta Platforms, Inc (Wibowo & Wibawa, 2025). *Framework* ini dirancang untuk membangun aplikasi *mobile* pada platform Android dan iOS dengan menggunakan satu basis kode (*codebase*) tunggal. *React Native* memanfaatkan bahasa pemrograman JavaScript dan pustaka (*library*) *React*, yang memungkinkan para pengembang web untuk mengaplikasikan keahlian mereka dalam ranah pengembangan *mobile* (Yaqin, 2023). Keunggulan utama yang ditawarkan oleh *React Native* adalah efisiensi, di mana pengembang tidak perlu menyusun dan memelihara kode secara terpisah untuk setiap platform. Hal ini secara signifikan dapat mereduksi waktu, biaya, dan alokasi sumber daya pengembangan, serta mempercepat siklus rilis produk ke pasar.

2.1.7 TypeScript

TypeScript adalah sebuah *superset* sintaksis dari JavaScript yang dikembangkan oleh Microsoft. Secara fundamental, *TypeScript* memperkaya JavaScript dengan menambahkan fitur pengetikan statis (*static typing*) opsional (Fathoni et al., 2025). Keberadaan tipe data statis memungkinkan deteksi dini terhadap potensi kesalahan pemrograman, seperti ketidaksesuaian tipe data atau pemanggilan properti yang tidak ada, pada tahap kompilasi, bukan pada saat eksekusi program (*runtime*) (Wijaya, 2025). Implikasinya, kode yang dihasilkan menjadi lebih andal, prediktif, mudah dibaca, serta memiliki tingkat pemeliharaan (*maintainability*) yang lebih tinggi, yang sangat krusial untuk proyek perangkat lunak berskala besar dan kolaboratif.

2.1.8 Supabase

Supabase adalah sebuah platform *Backend as a Service* (BaaS) *open-source* yang diposisikan sebagai alternatif modern dari Firebase (Phan & Yuricha, 2023). Platform ini menyediakan serangkaian layanan *backend* siap pakai, yang mencakup basis data relasional (menggunakan PostgreSQL yang tangguh), sistem otentikasi pengguna yang aman, penyimpanan objek (*storage*) untuk file, serta pembuatan Antarmuka Pemrograman Aplikasi (API) secara otomatis. Pemanfaatan *Supabase* memungkinkan akselerasi proses

pengembangan secara signifikan, karena pengembang tidak perlu membangun, mengkonfigurasi, dan mengelola infrastruktur *backend* dari awal (Valerian Romero et al., 2024). Dalam penelitian ini, Supabase difungsikan sebagai basis data *cloud* untuk penyimpanan data transaksi, produk, dan mitra secara terpusat, aman, dan dapat diakses secara *real-time* oleh aplikasi klien.

2.1.9 Usecase Diagram

Use Case Diagram merupakan salah satu diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan interaksi antara aktor dengan sistem (Khairunnisa & Voutama, 2024). Diagram ini berfokus pada fungsi-fungsi yang disediakan oleh sistem serta hubungan antara aktor dan fungsi tersebut. *Use Case Diagram* membantu dalam mengidentifikasi kebutuhan fungsional sistem dari sudut pandang pengguna, sehingga memudahkan proses analisis dan perancangan sistem (Rospricilia & Ma'ady, 2024).

Dalam penelitian ini, *Use Case Diagram* digunakan untuk memodelkan interaksi pengguna dengan aplikasi *Point of Sale* (POS), seperti proses login, pengelolaan data produk, pencatatan pesanan, pengelolaan pesanan, serta akses terhadap laporan penjualan. Penggunaan *Use Case Diagram* bertujuan untuk memberikan gambaran yang jelas mengenai batasan sistem, peran masing-masing aktor, dan fungsi utama yang dapat dijalankan oleh sistem. Diagram ini juga berfungsi sebagai acuan dalam tahap perancangan dan implementasi sistem agar fitur yang dikembangkan sesuai dengan kebutuhan pengguna yang telah diidentifikasi.

2.1.10 Activity Diagram

Activity Diagram merupakan salah satu diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan alur aktivitas atau aliran kerja (*workflow*) suatu proses dalam sistem secara detail (Tabrani et al., 2021). *Activity Diagram* menekankan urutan aktivitas, kondisi percabangan, serta aliran proses dari awal hingga akhir, sehingga mampu memberikan gambaran yang lebih jelas mengenai bagaimana suatu proses dijalankan di dalam sistem (Narulita et al., 2024).

Dalam penelitian ini, *Activity Diagram* digunakan untuk memodelkan alur aktivitas pengguna dalam aplikasi *Point of Sale* (POS), seperti proses login, pencatatan pesanan, pengelolaan data produk, hingga proses penyelesaian pesanan dan peninjauan laporan penjualan. Penggunaan *Activity Diagram* bertujuan untuk mempermudah pemahaman terhadap alur proses bisnis yang terjadi dalam sistem, serta menjadi acuan dalam perancangan logika sistem dan implementasi fitur agar sesuai dengan kebutuhan operasional pengguna. Simbol

Activity Diagram dapat dilihat pada Gambar 2.1

Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem,aktivitas biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan / Join	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar 2.1 Simbol Activity Diagram

2.2 Metode Pengujian Sistem

2.2.1 Black Box Testing

Black Box Testing adalah metode pengujian perangkat lunak yang berfokus pada pengujian fungsionalitas sistem tanpa memperhatikan struktur internal kode atau logika pemrograman yang digunakan (Maulida et al., 2025). Pengujian dilakukan dengan memberikan berbagai skenario input tertentu dan kemudian mengevaluasi output yang dihasilkan oleh sistem untuk mengetahui apakah hasil yang diperoleh telah sesuai dengan yang diharapkan. Metode ini digunakan dalam penelitian untuk memastikan bahwa setiap fitur aplikasi POS dapat berjalan dengan baik, berfungsi sesuai spesifikasi, serta memenuhi kebutuhan fungsional yang telah ditetapkan berdasarkan hasil analisis kebutuhan pengguna (Putri et al., 2025).

2.2.2 System Usability Scale (SUS)

System Usability Scale (SUS) merupakan metode evaluasi yang digunakan untuk mengukur tingkat kegunaan suatu sistem berdasarkan persepsi pengguna setelah menggunakan sistem tersebut (Maulia et al., 2024). SUS terdiri dari sepuluh pernyataan standar yang disusun

dalam bentuk kuesioner dengan skala penilaian Likert lima tingkat, yang digunakan untuk menghasilkan skor usabilitas secara kuantitatif. Metode ini digunakan karena bersifat sederhana, cepat dalam penerapan, serta mampu memberikan gambaran umum yang jelas mengenai tingkat penerimaan dan kemudahan penggunaan aplikasi dari sudut pandang pengguna.

Dalam instrumen System Usability Scale (SUS), pertanyaan disusun secara bergantian antara pertanyaan positif dan pertanyaan negatif. Pertanyaan dengan nomor ganjil (1, 3, 5, 7, dan 9) merupakan pertanyaan positif yang menggambarkan persepsi kemudahan dan kenyamanan pengguna terhadap sistem, sedangkan pertanyaan dengan nomor genap (2, 4, 6, 8, dan 10) merupakan pertanyaan negatif yang digunakan untuk mengidentifikasi potensi kesulitan, kompleksitas, dan hambatan dalam penggunaan sistem. Penyusunan pertanyaan secara bergantian ini bertujuan untuk menjaga konsistensi jawaban responden serta meningkatkan validitas hasil pengukuran usabilitas sistem.

Tabel 2.1 Daftar Pertanyaan SUS

No	Pertanyaan
1	Saya berpikir bahwa saya akan sering menggunakan sistem ini
2	Saya merasa sistem ini terlalu kompleks
3	Saya merasa sistem ini mudah digunakan
4	Saya membutuhkan bantuan teknis untuk dapat menggunakan sistem ini
5	Saya merasa fitur-fitur dalam sistem ini terintegrasi dengan baik
6	Saya merasa terdapat terlalu banyak inkonsistensi dalam sistem ini
7	Saya merasa kebanyakan orang akan dapat mempelajari sistem ini dengan cepat
8	Saya merasa sistem ini membingungkan untuk digunakan
9	Saya merasa percaya diri dalam menggunakan sistem ini
10	Saya perlu membiasakan diri terlebih dahulu sebelum dapat menggunakan sistem ini

Untuk mengukur tingkat usabilitas aplikasi, responden diminta memberikan penilaian terhadap sepuluh pernyataan SUS dengan pilihan jawaban skala Likert sebagai berikut: sebagaimana ditunjukkan pada Tabel 2.2 berikut.

Tabel 2.2 Pilihan Jawaban SUS

Jawaban	Skor Nilai
Sangat Tidak Setuju	1
Tidak Setuju	2
Netral	3
Setuju	4
Sangat Setuju	5

Untuk memperoleh kesimpulan mengenai kinerja sistem, metode *System Usability Scale* (SUS) menerapkan perhitungan skor berdasarkan rumus yang telah ditentukan. Pada pertanyaan dengan nomor ganjil, perhitungan skor dilakukan dengan mengurangkan nilai jawaban yang diberikan oleh responden (X) dengan angka satu (Prayoga & Kristiana, 2024), sebagaimana dirumuskan pada Persamaan (2.1).

$$\text{Skor Pertanyaan Ganjil} = X - 1 \tag{2.1}$$

Adapun untuk pertanyaan dengan nomor genap, perhitungan skor dilakukan dengan cara mengurangkan nilai maksimum skala, yaitu 5, dengan skor jawaban yang diberikan oleh responden (X) (Rivydho Assidiq et al., 2022), sebagaimana dirumuskan pada Persamaan (2.2).

$$\text{Skor Pertanyaan Genap} = 5 - X \tag{2.2}$$

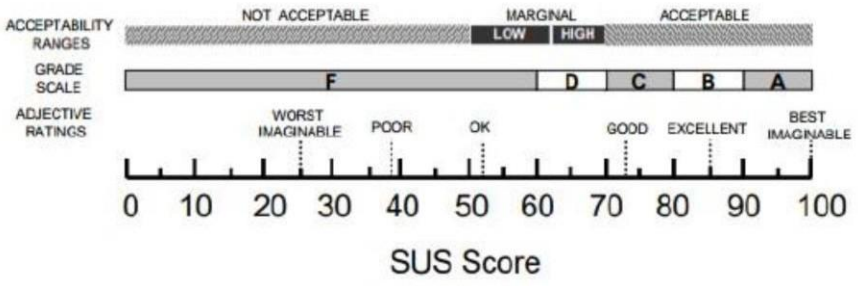
Berdasarkan hasil perhitungan tersebut, selanjutnya dilakukan penghitungan nilai sub total yang diperoleh dari penjumlahan skor seluruh pertanyaan bernomor ganjil dan genap . Proses perhitungan sub total ini ditunjukkan pada Persamaan (2.3)

$$\text{Skor Subtotal} = \text{Total Skor Ganjil} + \text{Total Skor Genap} \tag{2.3}$$

Setelah nilai sub total diperoleh, tahap selanjutnya adalah menghitung skor akhir masing-masing responden dengan cara mengalikan nilai sub total tersebut dengan konstanta sebesar 2,5 (Rivydho Assidiq et al., 2022). Proses perhitungan skor akhir responden ini ditunjukkan pada Persamaan (2.4)

$$\text{Skor Akhir Responden} = \text{Subtotal} \times 2.5 \tag{2.4}$$

Setelah seluruh skor akhir dari masing-masing responden diperoleh, tahap berikutnya adalah menghitung skor akhir *System Usability Scale* (SUS). Perhitungan skor akhir SUS dilakukan dengan cara menjumlahkan seluruh skor akhir responden, kemudian membaginya dengan jumlah responden yang terlibat dalam pengujian. Skor akhir SUS yang dihasilkan selanjutnya digunakan untuk menarik kesimpulan mengenai kinerja sistem dengan menyesuaikan nilai tersebut pada kategori penilaian SUS sebagaimana ditunjukkan pada Gambar 2.2



Gambar 2.2 Skala Penilaian SUS

2.3 Tinjauan Penelitian Terdahulu

Pengembangan aplikasi mobile *Point of Sale* (POS) telah menjadi salah satu fokus utama dalam berbagai penelitian di bidang sistem informasi dan informatika, khususnya dalam konteks digitalisasi Usaha Mikro, Kecil, dan Menengah (UMKM). Seiring dengan meningkatnya kebutuhan akan pengelolaan transaksi yang cepat, akurat, dan terintegrasi, aplikasi POS berbasis mobile dinilai mampu memberikan solusi yang lebih fleksibel dibandingkan sistem konvensional. Berbagai penelitian terdahulu telah mengkaji pengembangan dan penerapan aplikasi POS dengan beragam pendekatan metode, fitur, serta teknik pengujian, guna meningkatkan efisiensi operasional, akurasi pencatatan data, dan kualitas layanan kepada pengguna. Oleh karena itu, tinjauan penelitian terdahulu pada subbab ini bertujuan untuk mengkaji dan menganalisis hasil-hasil penelitian sebelumnya, mengidentifikasi persamaan dan perbedaan dengan penelitian yang dilakukan, serta menemukan celah penelitian (*research gap*) yang menjadi landasan pengembangan penelitian ini. Berikut beberapa studi yang relevan dengan topik ini baik di Indonesia maupun internasional:

a. Suhargo et al., (2025) mengembangkan aplikasi POS berbasis Android untuk UMKM sektor makanan dan minuman, khususnya kafe, dengan menambahkan fitur *Global Positioning System* (GPS) guna membantu pelanggan menemukan lokasi outlet serta mengakses informasi menu. Pengembangan aplikasi pada penelitian tersebut menggunakan metode *Waterfall* yang meliputi tahapan analisis kebutuhan, perancangan, implementasi, dan pengujian. Aplikasi yang dikembangkan menyediakan fitur autentikasi pengguna, manajemen outlet dan kasir, pengelolaan transaksi, histori transaksi, laporan penjualan, manajemen menu dan meja, serta fitur pencarian lokasi outlet berbasis GPS dengan memanfaatkan Mapbox API. Sistem dirancang untuk melayani tiga peran pengguna, yaitu pemilik, kasir, dan pelanggan. Pengujian sistem dilakukan menggunakan metode *Black Box Testing* dengan pendekatan *use case testing* untuk memastikan setiap

fungsi berjalan sesuai dengan skenario yang ditetapkan.

Selain itu, dilakukan *User Acceptance Testing* (UAT) yang melibatkan 50 responden yang terdiri dari pemilik kafe UMKM, kasir, dan pelanggan. Hasil pengujian UAT menunjukkan tingkat penerimaan pengguna yang sangat baik dengan skor sebesar 86,7% pada aspek *functionality*, 86,8% pada aspek *usability*, dan 85,7% pada aspek *efficiency*. Hasil tersebut menunjukkan bahwa aplikasi POS berbasis mobile yang dikembangkan dinilai layak, mudah digunakan, serta efektif dan efisien dalam mendukung aktivitas operasional UMKM kafe. Meskipun penelitian ini berhasil menunjukkan efektivitas aplikasi POS berbasis mobile dengan integrasi GPS, pendekatan pengembangan yang digunakan masih bersifat linear melalui model Waterfall, sehingga proses validasi kebutuhan pengguna dilakukan secara terbatas pada tahap awal dan akhir pengembangan.

b. Sumarto (2023) berfokus pada analisis dan perancangan aplikasi POS berbasis mobile dengan studi kasus pada UMKM Tofan Computer. Permasalahan utama yang diangkat adalah keterbatasan UMKM dalam mengelola transaksi penjualan secara cepat dan akurat, manajemen stok yang belum optimal, serta proses pembuatan laporan transaksi yang masih memerlukan waktu lama. Metode pengembangan yang digunakan dalam penelitian tersebut adalah *Rapid Application Development* (RAD), yang menekankan kecepatan pengembangan sistem melalui pembuatan purwarupa secara iteratif dan keterlibatan pengguna secara langsung. Pada penelitian ini dihasilkan 22 kebutuhan sistem yang terdiri dari kebutuhan fungsional dan nonfungsional, yang kemudian dijadikan dasar dalam perancangan arsitektur sistem, antarmuka pengguna, serta basis data.

   Implementasi purwarupa aplikasi dilakukan menggunakan bahasa pemrograman Dart dengan framework Flutter, sehingga aplikasi dapat berjalan pada berbagai platform mobile dengan satu basis kode. Pengujian aplikasi dilakukan menggunakan teknik *Black Box Testing* untuk memastikan setiap fungsi berjalan sesuai dengan kebutuhan yang telah ditetapkan. Hasil pengujian menunjukkan bahwa purwarupa aplikasi POS yang dikembangkan dapat berjalan dengan baik dan responsif. Penelitian ini menyimpulkan bahwa metode RAD mampu mempercepat proses pengembangan aplikasi POS serta memberikan fleksibilitas yang lebih tinggi dalam mengakomodasi kebutuhan pengguna yang terus berkembang. Meskipun demikian, penelitian ini masih berfokus pada tahap analisis dan perancangan serta pengembangan purwarupa, sehingga evaluasi penggunaan aplikasi secara berkelanjutan pada lingkungan operasional nyata belum menjadi fokus utama penelitian.

c. Wisnu et al., (2022) mengembangkan aplikasi POS berbasis mobile yang dapat membantu

pelaku UMKM atau pemilik warung dalam melakukan pencatatan transaksi penjualan serta menghasilkan laporan penjualan secara otomatis. Metode pengembangan sistem yang digunakan dalam penelitian tersebut adalah Waterfall, karena kebutuhan sistem telah terdefinisi dengan jelas sejak awal dan tidak mengalami perubahan selama proses pengembangan. Aplikasi WarunkQu dikembangkan menggunakan *framework* Flutter dengan bahasa pemrograman Dart, sehingga menghasilkan aplikasi mobile berbasis Android yang dapat digunakan pada berbagai perangkat. Sistem yang dibangun mendukung pengelolaan data produk, pelanggan, transaksi penjualan, serta penyajian laporan penjualan yang terintegrasi.

Pengujian sistem pada penelitian dilakukan melalui pengujian fungsional menggunakan metode *Black Box Testing* dan pengujian *usability* menggunakan metode *System Usability Scale* (SUS). Hasil pengujian fungsional menunjukkan tingkat keberhasilan sebesar 100%, sementara hasil pengujian usability memperoleh skor rata-rata sebesar 91 yang termasuk dalam kategori sangat baik dan dapat diterima oleh pengguna. Hasil tersebut menunjukkan bahwa aplikasi WarunkQu memiliki tingkat fungsionalitas dan kemudahan penggunaan yang tinggi. Meskipun demikian, penelitian ini masih menggunakan pendekatan pengembangan yang bersifat linear dengan model *Waterfall*, sehingga proses evaluasi dan validasi kebutuhan pengguna dilakukan secara terbatas.

d. Nurhidayat et al., (2025) pengembangan aplikasi POS berbasis Android yang terintegrasi dengan website berbasis Laravel serta mendukung metode pembayaran digital menggunakan QRIS. Metode pengembangan sistem yang digunakan adalah *System Development Life Cycle* (SDLC) dengan model *Waterfall*, yang meliputi tahapan analisis kebutuhan, perancangan sistem, implementasi, pengujian, dan pemeliharaan. Sistem yang dikembangkan memiliki fitur pencatatan transaksi otomatis, pengelolaan data produk dan pengguna, penyajian laporan penjualan dalam bentuk digital, serta integrasi pembayaran non-tunai melalui *QRIS Payment*. Selain itu, sistem dirancang dengan dua peran utama, yaitu admin melalui website dan kasir melalui aplikasi mobile, sehingga pengelolaan data dapat dilakukan secara terpusat dan terintegrasi. Pengujian sistem pada penelitian dilakukan menggunakan metode *Black Box Testing* untuk memastikan seluruh fungsi aplikasi berjalan sesuai dengan kebutuhan yang telah ditentukan. Hasil pengujian menunjukkan bahwa seluruh fitur sistem dapat berfungsi dengan baik dan mendukung proses transaksi penjualan secara lebih cepat dan akurat.

Penelitian ini menyimpulkan bahwa penerapan aplikasi POS berbasis Android dengan dukungan QRIS mampu meningkatkan efisiensi operasional, mengurangi

kesalahan pencatatan transaksi, serta membantu UMKM dalam beradaptasi dengan perkembangan teknologi digital. Meskipun demikian, penelitian ini masih menggunakan pendekatan pengembangan *Waterfall* yang bersifat linear, sehingga proses evaluasi dan penyesuaian kebutuhan pengguna dilakukan secara terbatas pada tahapan tertentu. Hal ini menunjukkan adanya peluang untuk mengembangkan sistem POS dengan pendekatan yang lebih iteratif dan berorientasi pada validasi berkelanjutan, seperti metodologi *Lean Startup*, guna menghasilkan solusi yang lebih fleksibel dan sesuai dengan dinamika kebutuhan UMKM.

e. Pratama & Somya (2021) melakukan penelitian yang bertujuan untuk membantu proses transaksi yang sebelumnya masih dilakukan secara konvensional menggunakan pencatatan manual. Penelitian ini mengembangkan aplikasi POS berbasis Android dengan memanfaatkan Firebase Realtime Database sebagai media penyimpanan data penjualan serta SQLite sebagai penyimpanan data sementara pada proses transaksi. Aplikasi yang dikembangkan dilengkapi dengan fitur manajemen data barang, transaksi penjualan, pencarian barang menggunakan barcode dan QR code melalui library ZXing, serta penyajian laporan penjualan secara realtime. Hasil pengujian sistem menggunakan metode Black Box menunjukkan bahwa seluruh fungsi sistem berjalan dengan baik dan sesuai dengan kebutuhan pengguna, sehingga aplikasi POS tersebut dinilai mampu meningkatkan efisiensi dan keakuratan proses transaksi penjualan.

   Penelitian ini memiliki keterkaitan dengan penelitian yang penulis lakukan, khususnya pada penerapan fitur-fitur utama aplikasi POS seperti pengelolaan data produk, pencatatan transaksi penjualan, penggunaan barcode/QR code untuk mempercepat proses input data, serta penyimpanan data secara realtime. Perbedaan penelitian ini terletak pada pendekatan pengembangan yang digunakan, di mana penelitian penulis menerapkan pendekatan Lean Startup untuk memastikan fitur-fitur yang dikembangkan benar-benar sesuai dengan kebutuhan pengguna dan mendukung keberlanjutan usaha, khususnya pada skala usaha kecil dan menengah.

f. Mulyani et al., (2025) melakukan penelitian yang bertujuan untuk mengatasi permasalahan pencatatan transaksi yang masih dilakukan secara manual pada UMKM kuliner. Penelitian ini mengembangkan aplikasi POS berbasis Android dengan metode Agile yang bersifat fleksibel dan adaptif terhadap perubahan kebutuhan pengguna. Aplikasi yang dikembangkan menggunakan bahasa pemrograman Kotlin dan basis data SQLite, sehingga mampu mendukung proses transaksi secara offline tanpa ketergantungan pada server tambahan.

Fitur utama yang diimplementasikan dalam aplikasi POS tersebut meliputi pengelolaan data menu makanan dan minuman, pencatatan transaksi penjualan, manajemen inventaris, serta pembuatan laporan penjualan otomatis dalam format digital (PDF). Hasil pengujian sistem menggunakan metode Black Box dan evaluasi pengguna menunjukkan tingkat penerimaan yang sangat baik, di mana aplikasi dinilai mudah digunakan, mampu meningkatkan efisiensi operasional, serta meminimalkan kesalahan pencatatan transaksi pada UMKM Nyemil Beauty.

Penelitian ini memiliki keterkaitan dengan penelitian yang penulis lakukan, khususnya pada pengembangan aplikasi POS berbasis Android yang berfokus pada digitalisasi pencatatan transaksi dan pengelolaan penjualan. Persamaan terletak pada pemanfaatan fitur inti POS seperti manajemen produk, transaksi penjualan, dan laporan penjualan digital. Adapun perbedaan penelitian ini dengan penelitian penulis terletak pada pendekatan pengembangan dan konteks penerapan sistem, di mana penelitian penulis mengintegrasikan pendekatan Lean Startup untuk memastikan kesesuaian fitur dengan kebutuhan pengguna serta menekankan aspek keberlanjutan usaha sebagai tujuan jangka panjang dari implementasi aplikasi POS.

Berdasarkan tinjauan penelitian terdahulu, dapat disimpulkan bahwa sebagian besar studi telah mengimplementasikan fitur inti POS seperti pengelolaan produk/menu, transaksi penjualan, dan laporan penjualan. Sejumlah penelitian menambahkan fitur spesifik seperti barcode/QR untuk pencarian produk, pembayaran digital QRIS, dukungan *offline* dan laporan PDF, maupun fitur kontekstual seperti manajemen meja/outlet serta GPS. Berbeda dari fokus fitur tambahan tersebut, penelitian ini menitikberatkan pada penguatan alur operasional harian UMKM melalui fitur laporan penjualan *real-time*, pengelolaan pesanan secara *end-to-end* (menambah, memproses, membatalkan), serta pengelolaan entitas bisnis yang mendukung operasional seperti kelola mitra, kelola produk, dan kelola akun/karyawan. Dengan demikian, celah penelitian yang diangkat adalah minimnya kajian yang secara eksplisit mengintegrasikan *lifecycle* pesanan, manajemen mitra, dan manajemen karyawan dalam satu aplikasi POS mobile yang terstruktur, sementara penelitian lain cenderung berfokus pada fitur inti POS atau fitur tambahan tertentu secara terpisah.

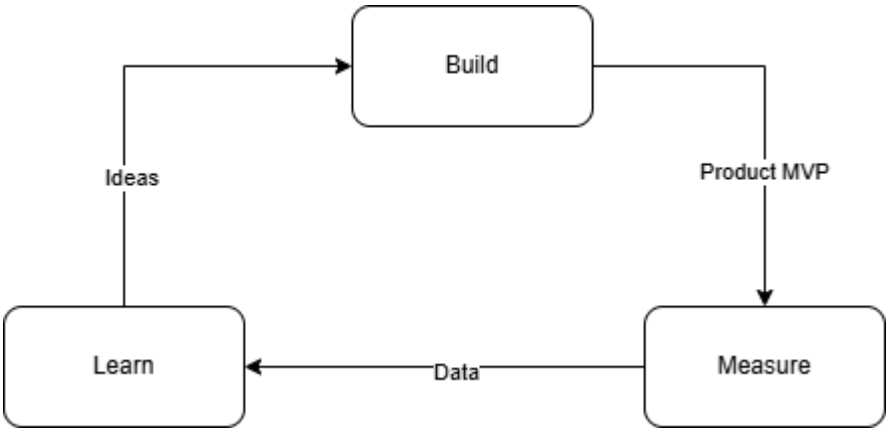
BAB III
METODE PENELITIAN

Berdasarkan kondisi nyata yang terjadi pada pabrik abon Merak Bewangi, diperoleh gambaran bahwa proses bisnis pencatatan pesanan dan penjualan masih dilakukan secara manual. Proses transaksi dimulai dari penerimaan pesanan oleh pemilik melalui *WhatsApp*, kemudian data pesanan dicatat menggunakan buku tulis. Selanjutnya, pemilik akan menghubungi karyawan melalui *WhatsApp* untuk memproses pesanan. Setelah pesanan selesai, karyawan akan menghubungi pemilik untuk memberi tahu apabila pesanan sudah selesai. Kemudian pemilik akan memberi tahu pembeli jika pesanan sudah selesai dan meminta untuk menyelesaikan pembayaran. Setelah pembeli melakukan pembayaran, pesanan akan dikirim ke alamat pembeli dan pemilik juga memberikan resi kepada pembeli. Selanjutnya, pencatatan penjualan dan rekapitulasi transaksi dilakukan secara terpisah berdasarkan ingatan dan catatan manual yang tersedia. Kondisi tersebut menyebabkan data penjualan tidak tersusun secara sistematis, berisiko mengalami kehilangan atau kesalahan pencatatan, serta menyulitkan pemilik usaha dalam melakukan pemantauan penjualan secara cepat dan akurat.

Selain itu, proses rekapitulasi data penjualan membutuhkan waktu yang relatif lama karena harus menghimpun kembali catatan transaksi yang tersebar. Ketiadaan sistem digital juga menyebabkan informasi terkait stok produk dan laporan penjualan tidak dapat diperoleh secara realtime. Berdasarkan kondisi tersebut, diperlukan suatu solusi berbasis sistem informasi yang mampu mendukung proses pencatatan transaksi dan pengelolaan penjualan secara lebih efektif dan efisien.

Oleh karena itu, penelitian ini menggunakan pendekatan Lean Startup yang menekankan pada pengembangan sistem secara bertahap dan iteratif berdasarkan kebutuhan pengguna. Pendekatan ini dipilih agar solusi yang dikembangkan benar-benar sesuai dengan kondisi proses bisnis yang berjalan serta mampu memberikan nilai tambah bagi keberlanjutan usaha.

3.1 Lean Startup



Gambar 3.1 *Lean Startup Methodology*

Penelitian ini mengadopsi metodologi *Lean Startup*, sebuah pendekatan yang berfokus pada pengembangan produk secara cepat dan efisien melalui siklus iteratif *Build–Measure–Learn*. Metodologi ini dipilih untuk meminimalkan risiko pengembangan produk yang tidak sesuai dengan kebutuhan pengguna dengan mengedepankan validasi berkelanjutan melalui pengujian fitur yang dikembangkan (Sri Rahayu & Tata Sutabri, 2024). Pendekatan ini sangat relevan untuk pengembangan aplikasi POS pada pabrik abon Merak Bewangi, karena proses digitalisasi membutuhkan penyesuaian yang adaptif terhadap kebutuhan nyata pengguna di lapangan.

Tahap *Build* merupakan fase awal dalam siklus *Lean Startup* yang berfokus pada proses perancangan dan pembangunan produk awal dalam bentuk *Minimum Viable Product* (MVP). Pada tahap ini, ide dan asumsi awal mengenai solusi permasalahan pengguna diterjemahkan ke dalam fitur-fitur inti yang paling esensial. Pengembangan MVP dilakukan dengan ruang lingkup terbatas untuk meminimalkan biaya dan waktu pengembangan, namun tetap mampu merepresentasikan fungsi utama sistem yang dibutuhkan pengguna (Lortie et al., 2025). Dalam konteks penelitian ini, tahap Build bertujuan untuk menghasilkan aplikasi POS versi awal yang dapat langsung digunakan oleh pemilik dan karyawan pabrik abon sebagai sarana validasi kebutuhan operasional secara nyata, tanpa menambahkan fitur yang bersifat kompleks atau belum tervalidasi.

Tahap *Measure* merupakan tahap pengukuran untuk mengetahui bagaimana respon dan pengalaman pengguna setelah menggunakan aplikasi yang telah dibangun. Pada tahap ini, peneliti mengamati penggunaan aplikasi oleh pemilik dan karyawan, serta mencatat berbagai masukan terkait kemudahan penggunaan, alur kerja, dan fungsi sistem. Selain itu, dilakukan pengujian untuk memastikan bahwa setiap fitur berjalan sesuai dengan kebutuhan pengguna.

Hasil dari tahap *Measure* digunakan untuk mengetahui apakah aplikasi yang dikembangkan sudah membantu proses kerja pengguna atau masih terdapat bagian yang perlu diperbaiki.

Tahap *Learn* merupakan tahap pembelajaran yang dilakukan berdasarkan hasil pengukuran pada tahap sebelumnya. Pada tahap ini, peneliti menganalisis seluruh masukan, temuan, dan hasil pengujian untuk memahami kelebihan dan kekurangan aplikasi. Pembelajaran yang diperoleh digunakan sebagai dasar dalam menentukan perbaikan dan penyempurnaan sistem pada tahap pengembangan berikutnya. Dengan adanya tahap *Learn*, proses pengembangan aplikasi menjadi lebih terarah karena setiap keputusan pengembangan didasarkan pada pengalaman pengguna secara langsung, sehingga aplikasi yang dihasilkan semakin sesuai dengan kebutuhan operasional.

Dalam pendekatan *Lean Startup*, salah satu keputusan penting yang dapat diambil setelah tahap Learn adalah pivot atau persevere. Pivot merupakan perubahan terarah terhadap strategi produk atau fitur berdasarkan hasil evaluasi dan umpan balik pengguna, apabila solusi yang dikembangkan belum sepenuhnya sesuai dengan kebutuhan pengguna. Pivot tidak berarti kegagalan sistem, melainkan penyesuaian arah pengembangan agar produk lebih relevan dan bernilai guna. Dalam konteks penelitian ini, keputusan pivot dilakukan apabila hasil evaluasi pada suatu iterasi menunjukkan bahwa fitur yang dikembangkan belum memberikan manfaat optimal bagi pengguna. Dalam penelitian ini, proses pivot dievaluasi pada setiap akhir iterasi berdasarkan hasil pengujian dan umpan balik pengguna. Namun, perubahan yang dilakukan masih berada dalam ruang lingkup perbaikan fitur dan alur sistem, sehingga penelitian ini tetap dibatasi pada dua iterasi pengembangan.

Apabila diperoleh perbedaan umpan balik dari pengguna, maka keputusan pengembangan diarahkan pada proses pivot dengan menyesuaikan fitur dan alur sistem agar dapat mengakomodasi kebutuhan utama pengguna. Perbedaan umpan balik tidak dipandang sebagai kegagalan sistem, melainkan sebagai dasar pembelajaran untuk menentukan arah pengembangan yang paling relevan. Dalam konteks penelitian ini, pivot dapat berupa penyesuaian fitur laporan, penyederhanaan alur transaksi, maupun penambahan opsi penggunaan sistem yang lebih fleksibel sesuai dengan kebutuhan pengguna.

Secara keseluruhan, strategi pengembangan sistem dalam penelitian ini dilaksanakan dalam dua iterasi utama:

a. Iterasi Pertama yang bertujuan untuk membangun *Minimum Viable Product* (MVP), yaitu versi awal aplikasi dengan fitur inti yang dapat langsung diuji oleh pengguna. MVP ini berfungsi sebagai fondasi untuk memahami kebutuhan dasar dan perilaku pengguna saat berinteraksi dengan sistem.

b. Iterasi Kedua yang berfokus pada penyempurnaan fitur berdasarkan data dan masukan yang diperoleh dari tahap Learn pada iterasi pertama. Perbaikan dilakukan dengan tetap mengacu pada kebutuhan pengguna dan hasil pengamatan penggunaan MVP sebelumnya.

Pada penelitian ini, penerapan metode Lean Startup dibatasi hingga dua iterasi pengembangan. Pembatasan jumlah iterasi ini bukan disebabkan oleh keterbatasan konsep Lean Startup, melainkan merupakan keputusan metodologis yang disesuaikan dengan ruang lingkup penelitian dan keterbatasan waktu pelaksanaan. Secara konseptual, Lean Startup merupakan pendekatan yang bersifat iteratif dan berkelanjutan melalui siklus Build–Measure–Learn, di mana jumlah iterasi tidak ditentukan secara baku dan dapat dilakukan berulang kali hingga tercapai kesesuaian antara produk dan kebutuhan pengguna. Oleh karena itu, dua iterasi dalam penelitian ini dipandang telah cukup untuk merepresentasikan proses validasi kebutuhan pengguna, pengujian fitur utama aplikasi POS, serta evaluasi perbaikan sistem berdasarkan umpan balik pengguna. (Golovin, 2025)

Pendekatan *Lean Startup* dinilai tepat karena memberikan fleksibilitas dalam pengambilan keputusan berbasis data, memungkinkan pengembangan sistem yang adaptif terhadap perubahan kebutuhan, serta mendorong penghematan sumber daya dalam proses rekayasa teknologi. Dengan iterasi yang berulang dan berbasis validasi langsung dari pengguna, kualitas dan relevansi fitur aplikasi dapat ditingkatkan secara signifikan.

3.2 Metode Pengumpulan Data

Metode pengumpulan data pada penelitian ini dilakukan untuk memperoleh pemahaman yang akurat mengenai kebutuhan, permasalahan, serta alur kerja operasional pabrik abon Merak Bewangi. Proses ini bertujuan untuk memastikan bahwa fitur yang dikembangkan dalam aplikasi POS benar-benar mencerminkan kondisi dan kebutuhan pengguna di lapangan. Data dikumpulkan melalui dua teknik utama, yaitu wawancara mendalam dan observasi langsung, sehingga hasil analisis kebutuhan dapat divalidasi dari perspektif pengguna nyata sebagai dasar penyusunan MVP dan pelaksanaan iterasi *Lean Startup*.

3.2.1 Wawancara

Wawancara dilakukan secara mendalam dengan pihak-pihak yang terlibat langsung dalam operasional harian, yaitu satu orang pemilik usaha (*owner*) dan dua orang karyawan (*employee*). Tujuan utama dari wawancara ini adalah untuk menggali informasi kualitatif terkait alur kerja, tantangan, dan ekspektasi terhadap sistem. Beberapa poin penting yang berhasil diidentifikasi melalui wawancara adalah:

a. Permasalahan dalam pencatatan data masih manual menggunakan buku tulis menyebabkan berbagai kendala, seperti ketidakteraturan data, risiko tinggi kehilangan atau kerusakan catatan, dan kesulitan dalam melakukan rekapitulasi penjualan secara cepat dan akurat.
b. Kebutuhan akan sistem digital yang diungkapkan pemilik mencakup beberapa kemampuan utama, seperti pencatatan pesanan secara otomatis dan terstruktur, penyajian laporan penjualan secara *real-time*, serta ketersediaan antarmuka yang intuitif dan mudah dioperasikan.
c. Harapan terhadap aplikasi yang dikembangkan memiliki dua tujuan utama, yaitu peningkatan efisiensi pencatatan serta fungsinya sebagai alat bantu pengambilan keputusan melalui fitur visualisasi data penjualan.

3.2.2 Observasi

Observasi langsung dilaksanakan dengan mengunjungi lokasi pabrik abon untuk mengamati secara langsung aktivitas operasional, khususnya yang berkaitan dengan proses manajemen pesanan. Melalui pengamatan ini, peneliti memvalidasi temuan dari wawancara dan memperoleh pemahaman kontekstual mengenai alur kerja yang ada. Hasil observasi mengonfirmasi bahwa proses pencatatan manual memakan waktu yang signifikan dan rentan terhadap kesalahan input data (*human error*) (Putra & Hartono, 2024). Selain itu, observasi juga mengidentifikasi bahwa seluruh subjek penelitian telah akrab dengan penggunaan perangkat *mobile* berbasis Android, yang mengindikasikan bahwa implementasi aplikasi *mobile* merupakan solusi yang sangat memungkinkan dan dapat diadopsi dengan baik.

3.3 Iterasi 1

Iterasi pertama dalam penelitian ini merupakan tahap awal penerapan siklus *Build–Measure–Learn* yang bertujuan untuk membangun dan menguji MVP (*Minimum Viable Product*) dari aplikasi POS pabrik abon Merak Bewangi. Iterasi ini berfokus pada pengembangan versi awal aplikasi yang memuat fitur inti yang paling dibutuhkan oleh pengguna. Melalui iterasi pertama ini, peneliti berupaya memahami bagaimana pemilik dan pegawai pabrik berinteraksi dengan sistem, hambatan yang mereka hadapi, serta efektivitas solusi yang diberikan terhadap permasalahan pencatatan manual yang sebelumnya dilakukan secara konvensional.

3.3.1 Build

a. Analisis Kebutuhan Sistem

Analisis kebutuhan sistem pada iterasi pertama merupakan tahap dasar yang bertujuan untuk mengidentifikasi fitur inti yang benar-benar diperlukan oleh pengguna dalam mendukung proses operasional pabrik abon Merak Bewangi. Tahap ini dilakukan untuk memastikan bahwa MVP yang dikembangkan mampu menjawab permasalahan utama yang ditemukan pada proses pencatatan manual, sekaligus menyediakan alur kerja yang lebih efisien dan mudah dipahami oleh pemilik serta pegawai pabrik. Analisis kebutuhan dilakukan berdasarkan informasi dari wawancara dan observasi sehingga rancangan MVP dapat disusun secara tepat sasaran dan mencerminkan kebutuhan nyata di lapangan.

1. Pengumpulan Data Primer

Data primer diperoleh melalui wawancara langsung dengan pemilik dan 2 karyawan pabrik abon. Tujuannya untuk menggali proses bisnis, kendala operasional, serta ekspektasi terhadap aplikasi POS yang akan dikembangkan. Pertanyaan berfokus pada alur pencatatan pesanan dan kebutuhan pelaporan. Hasil dari wawancara ini menjadi fondasi dalam menentukan fitur utama dan desain sistem yang relevan dengan kebutuhan pengguna di lapangan. Daftar pertanyaan dapat dilihat pada Tabel 3.1.

Tabel 3.1 Daftar Pertanyaan Wawancara

No	Pertanyaan Wawancara	Tujuan Pertanyaan
1	Bagaimana proses pencatatan pesanan dilakukan saat ini?	Mengetahui metode manual yang digunakan dan potensi kesalahannya.
2	Apakah proses pencatatan pesanan memakan banyak waktu?	Mengidentifikasi kendala waktu dalam pengelolaan pesanan.
3	Seberapa sering terjadi kesalahan dalam pencatatan pesanan?	Mengetahui tingkat kesalahan (human error) yang sering terjadi.
4	Fitur apa saja yang diharapkan ada dalam aplikasi pencatatan pesanan?	Menentukan kebutuhan fungsional utama dari sisi pengguna.
5	Apakah Anda dan karyawan sudah terbiasa menggunakan smartphone dalam pekerjaan?	Menilai kesiapan pengguna terhadap penerapan aplikasi mobile.
6	Bagaimana Anda saat ini membuat laporan penjualan bulanan atau harian?	Mengidentifikasi kebutuhan digitalisasi laporan penjualan.
7	Apa kendala terbesar yang Anda hadapi dalam pencatatan dan pelaporan pesanan?	Menentukan akar masalah yang perlu diselesaikan aplikasi.
8	Apakah data stok produk dicatat secara manual atau terintegrasi dengan penjualan?	Mengetahui potensi pengembangan fitur integrasi stok.
9	Apakah sistem manual yang digunakan saat ini cukup membantu	Mengukur tingkat kepuasan terhadap sistem lama.

No	Pertanyaan Wawancara	Tujuan Pertanyaan
	dalam operasional?	
10	Apa harapan Anda terhadap aplikasi yang akan dikembangkan ini?	Mengetahui ekspektasi pengguna terhadap manfaat aplikasi.

2. Pengumpulan Data Sekunder

Data sekunder diperoleh dari referensi penelitian terdahulu, jurnal ilmiah, serta dokumentasi operasional pabrik. Informasi ini digunakan untuk memperkuat pemahaman teoritis dan memastikan kesesuaian rancangan aplikasi dengan kebutuhan nyata di lapangan. Data sekunder juga memberikan dasar untuk membandingkan hasil penelitian ini dengan penelitian serupa sebelumnya.

b. Design

Tahap desain pada iterasi pertama merupakan proses perumusan struktur awal aplikasi berdasarkan kebutuhan pengguna yang telah dianalisis sebelumnya. Pada tahap ini, peneliti memvisualisasikan alur kerja sistem, interaksi antarpengguna, struktur basis data, serta rancangan antarmuka sebagai dasar pengembangan *Minimum Viable Product* (MVP). Tahap desain pada iterasi 1 mencakup desain proses bisnis, desain *database,* desain antarmuka yang secara keseluruhan memberikan gambaran menyeluruh mengenai cara kerja sistem sebelum memasuki tahap implementasi. Perancangan ini tidak hanya bertujuan menciptakan representasi visual sistem, tetapi juga memastikan kesesuaian fitur dengan kebutuhan aktual pemilik dan pegawai pabrik abon Merak Bewangi.

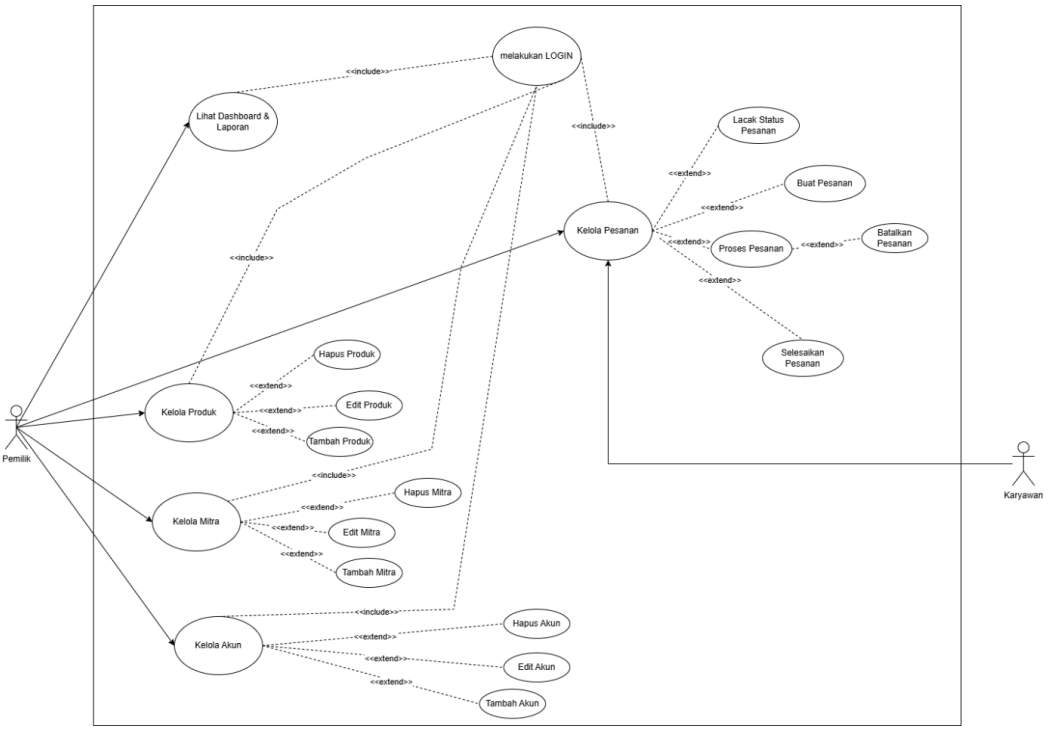
1. Desain Proses Bisnis

Desain proses bisnis bertujuan untuk menggambarkan alur dan mekanisme kerja sistem aplikasi *Point of Sale* (POS) yang dikembangkan secara terstruktur dan mudah dipahami. Desain proses bisnis disusun untuk merepresentasikan bagaimana sistem mendukung aktivitas operasional pengguna, mulai dari proses awal hingga menghasilkan keluaran berupa informasi dan laporan yang dibutuhkan. Pada tahap ini, pemodelan proses bisnis divisualisasikan menggunakan *activity diagram,* dan *use case diagram. Activity diagram* digunakan untuk memodelkan aktivitas serta aliran kerja pengguna secara detail, sedangkan *use case diagram* digunakan untuk mengidentifikasi aktor dan interaksi mereka dengan sistem. Ketiga diagram tersebut saling melengkapi dalam memberikan gambaran menyeluruh mengenai proses bisnis aplikasi POS yang dikembangkan.

a) Use Case Diagram

Diagram ini menunjukkan interaksi antara aktor utama, yaitu Pemilik dan Karyawan,

dengan sistem. Pemilik memiliki hak akses penuh, mencakup pengelolaan data karyawan, produk, dan laporan penjualan, sedangkan Karyawan berfokus pada pengelolaan pesanan dan pembaruan status transaksi. Diagram ini berfungsi sebagai panduan penting dalam mendefinisikan batasan tanggung jawab masing-masing pengguna serta membantu pengembang memahami kebutuhan autentikasi, otorisasi, dan hak akses pada setiap modul sistem. Dengan demikian, representasi ini tidak hanya berperan sebagai alat desain visual, tetapi juga sebagai dokumentasi konseptual yang menghubungkan hasil analisis kebutuhan dengan perancangan arsitektur aplikasi. Hasil rancangan *usecase* dapat dilihat pada Gambar 3.2 *Use Case* Diagram



Gambar 3.2 *Use Case* Diagram

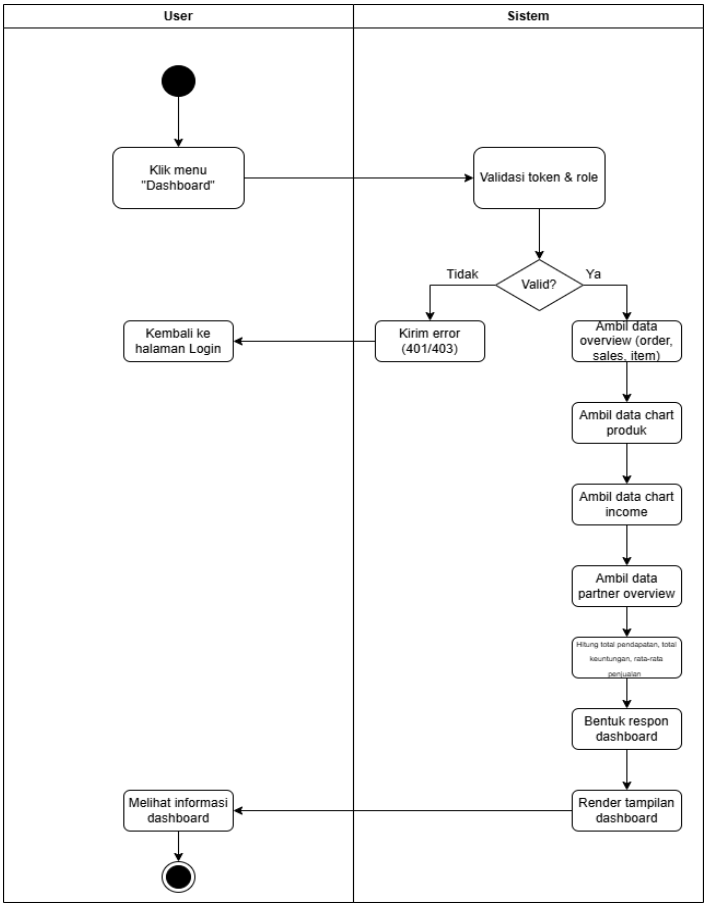
Use case diagram pada Gambar 3.2 menggambarkan interaksi antara dua aktor utama dalam sistem, yaitu Pemilik dan Karyawan, dengan berbagai fungsi yang tersedia pada aplikasi POS pabrik abon Merak Bewangi. Pemilik memiliki hak akses penuh terhadap seluruh fitur sistem, termasuk melihat *dashboard*, membuat pesanan, memproses, menyelesaikan pesanan, dan membatalkan pesanan, serta melakukan pengelolaan data seperti menambah dan menghapus produk, mitra, dan karyawan. Sementara itu, Karyawan memiliki akses terbatas pada fungsi operasional yang berkaitan dengan pemrosesan pesanan, yaitu menambah pesanan, memproses pesanan, dan menyelesaikan pesanan, dan membatalkan

pesanan. Diagram ini menunjukkan struktur kewenangan yang jelas antara kedua aktor, di mana Pemilik bertindak sebagai pengelola penuh sistem dan Karyawan berfokus pada tugas eksekusi di lapangan. Pembatasan akses fitur bagi Karyawan diterapkan untuk menjaga keamanan data, konsistensi operasional, serta memastikan bahwa proses bisnis tetap berjalan sesuai struktur organisasi pabrik abon Merak Bewangi.

b) Activity Diagram

Activity diagram digunakan untuk memvisualisasikan langkah-langkah operasional dalam setiap fitur, seperti proses pembuatan pesanan, pembaruan status pengiriman, dan pelaporan transaksi. Diagram ini memberikan gambaran alur aktivitas yang terjadi dalam sistem, memperlihatkan urutan aksi dan keputusan yang dilakukan oleh pengguna serta sistem secara simultan. Selain menunjukkan urutan logis aktivitas, diagram ini juga menggambarkan percabangan keputusan yang mungkin muncul berdasarkan input pengguna atau kondisi tertentu. Dengan demikian, pengembang dapat menganalisis efisiensi alur kerja, mengidentifikasi titik potensi keterlambatan, dan memahami interaksi antar proses dengan lebih mendalam. Selain itu, activity diagram membantu dalam merancang algoritma dan struktur kode program yang sesuai dengan alur aktivitas sebenarnya di lapangan, memastikan bahwa setiap skenario penggunaan dapat dijalankan secara konsisten dan bebas konflik.

1) *Activity Diagram* Lihat Dashboard & Laporan



Gambar 3.3 *Activity Diagram* Lihat Dashboard & Laporan

Activity diagram pada Gambar 3.3 menggambarkan alur aktivitas pengguna dan sistem dalam proses menampilkan *dashboard* dan laporan pada aplikasi *mobile Point of Sale* (POS). Proses dimulai ketika pengguna memilih menu *Dashboard* pada aplikasi. Setelah aksi tersebut dilakukan, sistem terlebih dahulu melakukan proses validasi token dan peran pengguna (*role*) untuk memastikan bahwa pengguna memiliki hak akses yang sesuai.

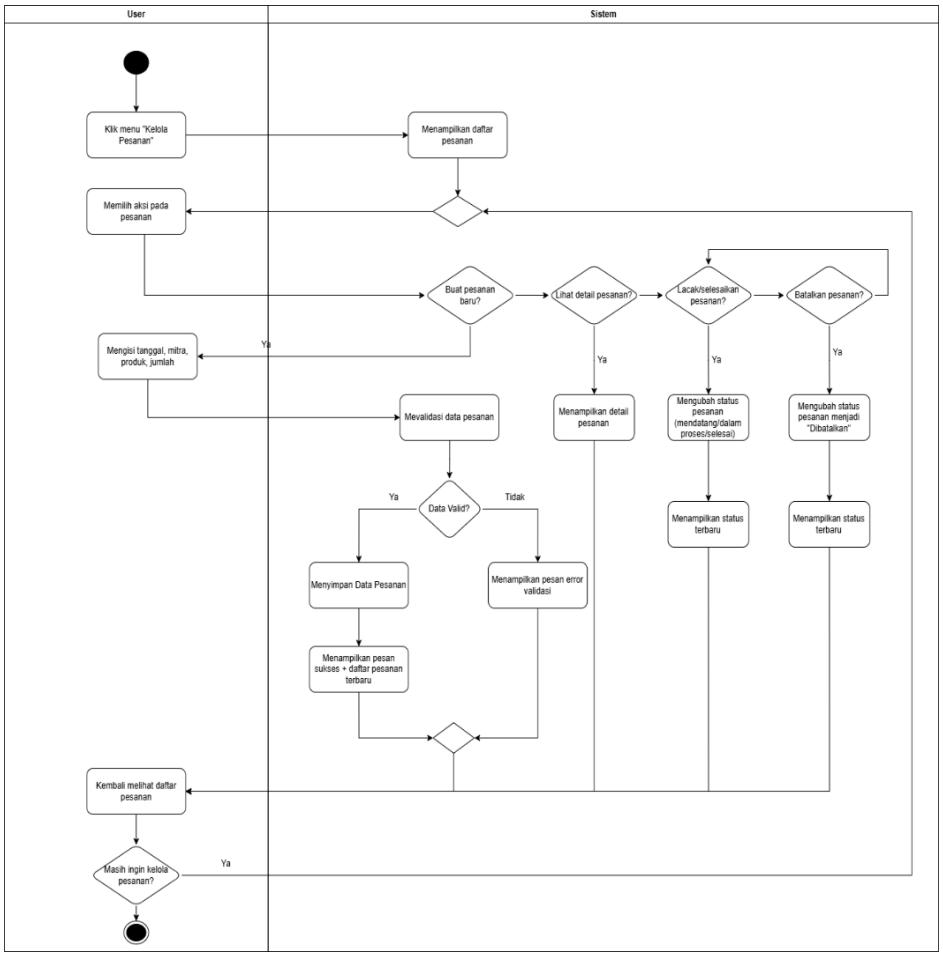
Apabila hasil validasi menunjukkan bahwa token atau peran pengguna tidak valid, sistem akan mengirimkan pesan kesalahan dengan kode 401 atau 403, kemudian pengguna diarahkan kembali ke halaman login. Mekanisme ini diterapkan untuk menjaga keamanan sistem serta mencegah akses yang tidak berwenang terhadap data *dashboard* dan laporan. Jika proses validasi berhasil, sistem akan melanjutkan dengan mengambil data ringkasan yang diperlukan untuk ditampilkan pada dashboard. Data tersebut meliputi informasi overview transaksi, data penjualan, serta jumlah item yang

tercatat dalam sistem. Selanjutnya, sistem mengambil data pendukung berupa data grafik produk, grafik pendapatan, dan ringkasan mitra untuk menyajikan informasi secara visual dan mudah dipahami.

Setelah seluruh data berhasil diperoleh, sistem melakukan proses perhitungan, seperti total pendapatan, total keuntungan, dan rata-rata penjualan. Hasil pengolahan data tersebut kemudian disusun menjadi satu kesatuan respon *dashboard*. Tahap akhir pada sisi sistem adalah melakukan proses render tampilan dashboard berdasarkan data yang telah diolah.

Pada sisi pengguna, hasil dari proses tersebut ditampilkan dalam bentuk informasi dashboard yang dapat digunakan sebagai bahan pemantauan kondisi usaha dan pengambilan keputusan. Dengan demikian, activity diagram ini menunjukkan bahwa sistem dashboard dirancang untuk menyajikan informasi secara aman, terstruktur, dan informatif guna mendukung kebutuhan pelaporan dan evaluasi kinerja usaha.

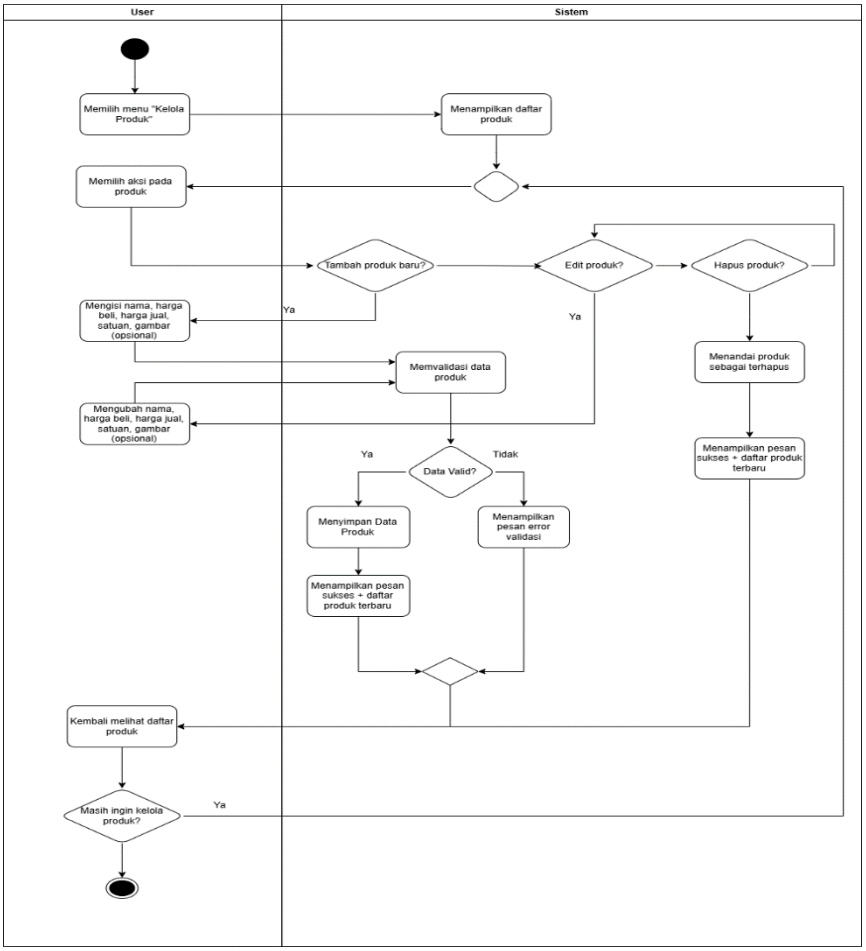
2) *Activity Diagram* Kelola Pesanan



Gambar 3.4 *Activity Diagram* Kelola Pesanan

Activity diagram pada Gambar 3.4 *Activity Diagram* Kelola Pesanan menggambarkan alur aktivitas pengguna dalam mengelola data pesanan pada aplikasi *Point of Sale* (POS). Proses dimulai ketika pengguna memilih menu Kelola Pesanan, kemudian sistem menampilkan daftar pesanan yang tersedia. Pengguna dapat melakukan beberapa aksi, yaitu membuat pesanan baru, melihat detail pesanan, memperbarui status pesanan, atau membatalkan pesanan. Pada proses pembuatan pesanan, pengguna mengisi data yang diperlukan dan sistem melakukan validasi sebelum menyimpan data ke dalam basis data. Apabila data valid, pesanan berhasil disimpan dan ditampilkan pada daftar pesanan, sedangkan jika tidak valid sistem akan menampilkan pesan kesalahan. Selain itu, pengguna dapat melihat detail pesanan serta memperbarui status pesanan sesuai proses pengerjaan, seperti mendatang, dalam proses, atau selesai. Sistem juga menyediakan fitur pembatalan pesanan yang akan mengubah status pesanan menjadi dibatalkan dan menampilkan status terbaru. Setelah seluruh proses pengelolaan pesanan selesai, pengguna akan kembali ke halaman daftar pesanan dan dapat melanjutkan pengelolaan atau mengakhiri proses. Activity diagram ini menunjukkan bahwa sistem dirancang untuk mendukung pengelolaan pesanan secara terstruktur, efisien, dan mudah digunakan.

3) *Activity Diagram* Kelola Produk



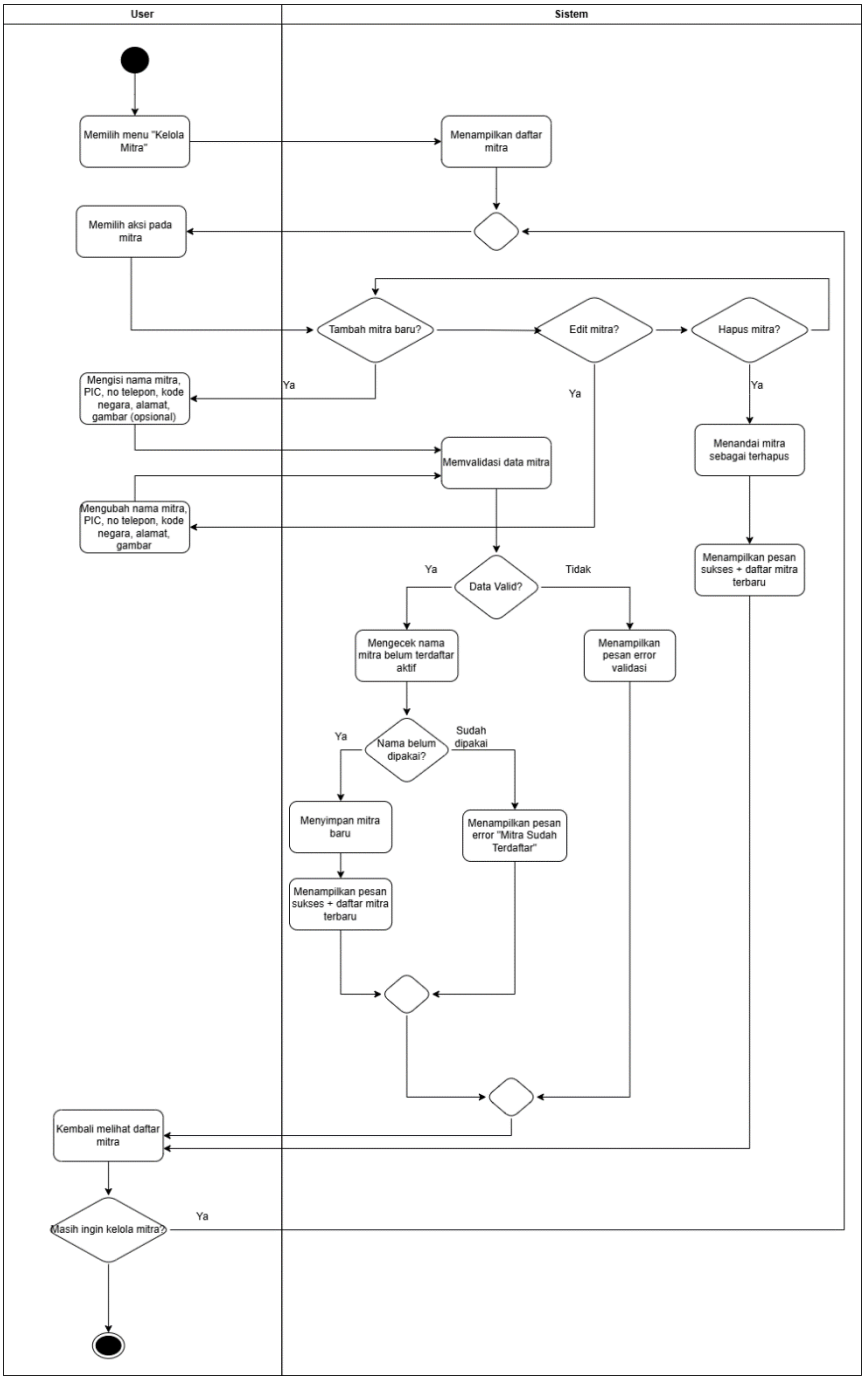
Gambar 3.5 *Activity Diagram* Kelola Produk

Activity diagram pada Gambar 3.5 menggambarkan alur aktivitas pengguna dalam mengelola data produk pada aplikasi *Point of Sale* (POS). Proses dimulai ketika pengguna memilih menu Kelola Produk, kemudian sistem menampilkan daftar produk yang tersedia. Pengguna dapat memilih beberapa aksi, yaitu menambahkan produk baru, mengubah data produk, atau menghapus produk. Pada proses penambahan maupun pengeditan produk, pengguna mengisi atau memperbarui data produk seperti nama, harga beli, harga jual, satuan, serta gambar produk (opsional). Selanjutnya, sistem melakukan validasi data sebelum menyimpan perubahan.

Apabila data produk dinyatakan valid, sistem akan menyimpan data dan menampilkan pesan keberhasilan beserta daftar produk yang telah diperbarui. Namun, jika data tidak valid, sistem akan menampilkan pesan kesalahan validasi kepada pengguna. Untuk proses penghapusan, sistem menandai produk sebagai terhapus dan menampilkan daftar produk terbaru. Setelah proses pengelolaan selesai, pengguna akan

kembali ke halaman daftar produk dan dapat memilih untuk melanjutkan pengelolaan atau mengakhiri proses. Activity diagram ini menunjukkan bahwa sistem mendukung pengelolaan data produk secara terstruktur dan efisien.

4) *Activity Diagram* Kelola Mitra



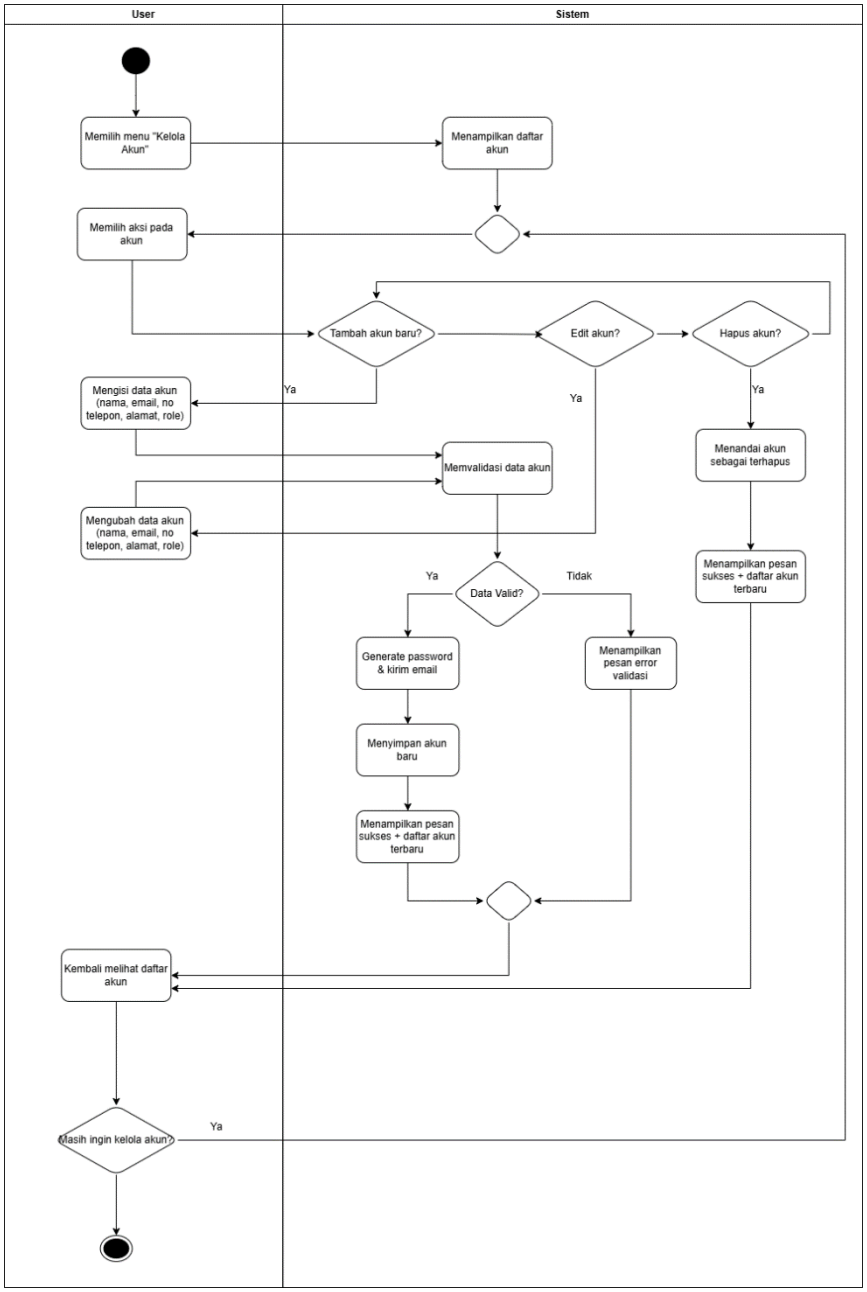
Gambar 3.6 *Activity Diagram* Kelola Mitra

Activity diagram pada Gambar 3.6 menggambarkan alur aktivitas pengguna dalam mengelola data mitra pada aplikasi *Point of Sale* (POS). Proses dimulai ketika

pengguna memilih menu Kelola Mitra, kemudian sistem menampilkan daftar mitra yang tersedia. Pengguna dapat melakukan beberapa aksi, yaitu menambahkan mitra baru, mengubah data mitra, atau menghapus mitra. Pada proses penambahan dan pengeditan mitra, pengguna mengisi atau memperbarui data mitra seperti nama mitra, PIC, nomor telepon, kode negara, alamat, serta gambar (opsional). Selanjutnya, sistem melakukan validasi data mitra.

Apabila data dinyatakan valid, sistem akan melakukan pengecekan untuk memastikan bahwa nama mitra belum terdaftar sebelumnya. Jika nama mitra belum digunakan, sistem menyimpan data mitra dan menampilkan pesan keberhasilan beserta daftar mitra terbaru. Namun, jika nama mitra sudah terdaftar, sistem akan menampilkan pesan kesalahan. Apabila data tidak valid, sistem menampilkan pesan error validasi. Pada proses penghapusan, sistem menandai data mitra sebagai terhapus dan menampilkan daftar mitra yang telah diperbarui. Setelah proses pengelolaan selesai, pengguna kembali ke halaman daftar mitra dan dapat memilih untuk melanjutkan pengelolaan atau mengakhiri proses. Activity diagram ini menunjukkan bahwa sistem mendukung pengelolaan data mitra secara terstruktur dan terkontrol.

5) *Activity Diagram* Kelola Akun



Gambar 3.7 *Activity Diagram* Kelola Akun

Activity diagram pada Gambar 3.7 menggambarkan alur aktivitas pengguna dalam mengelola data akun pada aplikasi *Point of Sale* (POS). Proses dimulai ketika pengguna memilih menu Kelola Akun, kemudian sistem menampilkan daftar akun yang tersedia. Pengguna dapat melakukan beberapa aksi, yaitu menambahkan akun baru, mengubah data akun, atau menghapus akun. Pada proses penambahan dan pengeditan akun, pengguna mengisi atau memperbarui data akun yang meliputi nama, email, nomor

telepon, alamat, serta peran (*role*). Selanjutnya, sistem melakukan validasi terhadap data akun yang dimasukkan.

Apabila data akun dinyatakan valid, sistem akan menghasilkan kata sandi secara otomatis dan mengirimkan informasi akun melalui email, kemudian menyimpan data akun ke dalam sistem serta menampilkan pesan keberhasilan beserta daftar akun terbaru. Namun, jika data tidak valid, sistem akan menampilkan pesan kesalahan validasi. Pada proses penghapusan, sistem menandai akun sebagai terhapus dan menampilkan daftar akun yang telah diperbarui. Setelah proses pengelolaan akun selesai, pengguna akan kembali ke halaman daftar akun dan dapat memilih untuk melanjutkan pengelolaan atau mengakhiri proses. *Activity diagram* ini menunjukkan bahwa sistem mendukung pengelolaan akun secara terstruktur dan aman.

2. Desain Database

Desain database merupakan tahapan penting dalam pengembangan aplikasi *Point of Sale* (POS) karena berfungsi sebagai fondasi utama dalam pengelolaan dan penyimpanan data sistem. Pada penelitian ini, layanan Supabase digunakan sebagai *Backend as a Service (BaaS)* yang menyediakan sistem manajemen basis data berbasis PostgreSQL, autentikasi pengguna, serta layanan *Application Programming Interface* (API) secara terintegrasi. Perancangan *database* dilakukan untuk memastikan bahwa data yang berkaitan dengan pengguna, produk, transaksi penjualan, serta laporan dapat disimpan secara terstruktur, konsisten, dan aman. Pemanfaatan Supabase memungkinkan pengelolaan data dilakukan secara *real-time*, mendukung skalabilitas sistem, serta mempermudah integrasi antara aplikasi mobile dan *backend*. Desain *database* pada penelitian ini disusun berdasarkan kebutuhan fungsional yang telah diidentifikasi, sehingga mampu mendukung proses bisnis aplikasi POS secara optimal, meminimalkan redundansi data, serta menjaga integritas data sesuai dengan kebutuhan operasional UMKM.

a) Relasi Tabel

Perancangan basis data pada sistem *Point of Sale* (POS) ini disusun untuk mendukung proses pengelolaan data pengguna, mitra, produk, serta transaksi pemesanan secara terstruktur dan terintegrasi. Basis data terdiri dari lima tabel utama, yaitu *User, Partner, Product, Order*, dan *OrderItem*, yang saling berelasi untuk membentuk alur data transaksi yang utuh.

Tabel user berfungsi menyimpan informasi akun pengguna aplikasi, meliputi identitas, kredensial, serta peran (*role*) pengguna. Pada rancangan ini, tabel user tidak memiliki relasi langsung dengan tabel transaksi karena difokuskan pada kebutuhan autentikasi dan otorisasi akses sistem.

Tabel *partner* digunakan untuk menyimpan data mitra yang melakukan pemesanan.

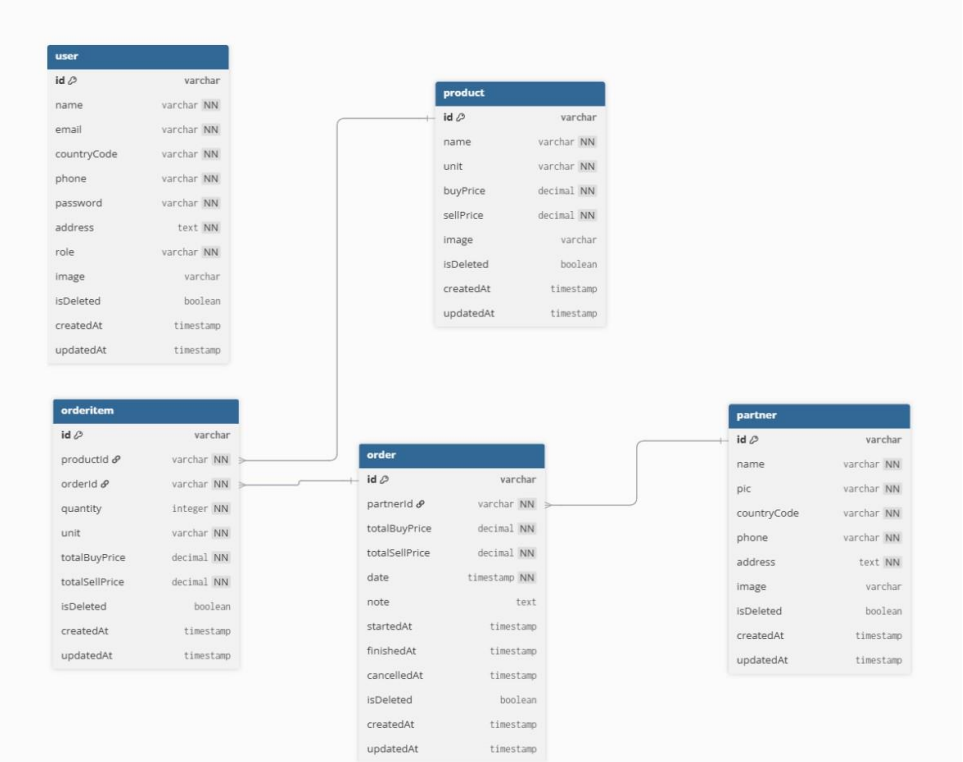
Tabel ini memiliki relasi *one-to-many* dengan tabel order, yang berarti satu mitra dapat memiliki banyak pesanan, sedangkan satu pesanan hanya terkait dengan satu mitra. Relasi ini direpresentasikan melalui atribut partnerId pada tabel *order* sebagai *foreign key* yang mengacu pada *primary key tabel partner*.

Tabel order merupakan entitas transaksi utama yang menyimpan ringkasan pemesanan, seperti total harga beli, total harga jual, tanggal pemesanan, catatan tambahan, serta penanda status proses pesanan (startedAt, finishedAt, cancelledAt). Selanjutnya, tabel order memiliki relasi *one-to-many* dengan tabel orderitem, karena satu pesanan dapat terdiri dari beberapa item produk.

Tabel orderitem menyimpan rincian produk pada setiap pesanan, seperti produk yang dipilih, jumlah, satuan, serta nilai total harga beli dan harga jual per item. Tabel orderitem menghubungkan tabel order dan product melalui dua atribut *foreign key*, yaitu orderId (mengacu pada tabel order) dan productId (mengacu pada tabel product). Dengan demikian, setiap orderitem hanya terkait pada satu pesanan dan satu produk, namun satu pesanan dapat memiliki banyak orderitem, dan satu produk dapat muncul di banyak orderitem.

Tabel *product* merupakan tabel yang menyimpan data produk, seperti nama, satuan, harga beli, harga jual, dan atribut pendukung lainnya. Relasi antara *product* dan orderitem bersifat *one-to-many*, di mana satu produk dapat tercatat pada banyak item pesanan, sehingga transaksi dapat dilakukan tanpa duplikasi data produk pada tabel *order*.

Secara keseluruhan, relasi antar tabel pada rancangan ini bertujuan untuk menjaga integritas data transaksi, mengurangi redundansi penyimpanan, serta memudahkan proses pencatatan pesanan dan pelaporan penjualan secara lebih konsisten dan terstruktur. Hasil perancangan struktur database dapat dilihat pada Gambar 3.8



Gambar 3.8 Relasi Tabel

b) Tabel User

Tabel 3.2 menampilkan struktur tabel User yang digunakan sebagai salah satu tabel utama dalam sistem. Tabel ini dirancang untuk menyimpan seluruh data pengguna yang berinteraksi dengan sistem, baik pemilik maupun karyawan, sehingga sistem dapat mengenali identitas setiap pengguna secara tepat. Informasi yang disimpan dalam tabel User mencakup data identitas dasar, informasi kontak, serta data pendukung lainnya yang diperlukan dalam proses penggunaan sistem.

Selain berfungsi sebagai penyimpanan data identitas, tabel User juga berperan penting dalam mendukung proses autentikasi dan pengelolaan hak akses pengguna. Melalui pengaturan kolom peran pengguna, sistem dapat membedakan hak akses antar pengguna sesuai dengan tanggung jawab dan kewenangan masing-masing. Oleh karena itu, setiap kolom pada tabel User dirancang dengan tipe data dan constraint tertentu untuk menjaga konsistensi data, mencegah terjadinya duplikasi, serta memastikan keamanan data yang tersimpan dalam basis data. Perancangan struktur tabel ini diharapkan dapat mendukung operasional sistem secara optimal dan berkelanjutan.

Tabel 3.2 Tabel *User*

Nama Kolom	Tipe Data	Data Length	Constraint
Id	UUID	-	PK
name	Varchar	255	NOT NULL
email	Varchar	255	UNIQUE, NOT NULL
countryCode	Varchar	10	NOT NULL
phone	Varchar	20	NOT NULL
password	Varchar	255	NOT NULL
address	Text	-	NOT NULL
role	ENUM (’Admin’, ’Employee’)	-	-
image	Varchar	500	NULL
isDeleted	Boolean	-	DEFAULT: false
createdAt	Timestamp	-	DEFAULT: now()
updatedAt	Timestamp	-	AUTO UPDATE

Pada Tabel User, kolom isDeleted (Boolean, DEFAULT: false) digunakan sebagai penanda status penghapusan logis (soft delete) untuk data akun. Secara default, nilai false menunjukkan akun masih aktif dan dapat ditampilkan pada daftar akun. Ketika pengguna melakukan aksi hapus akun pada fitur Kelola Akun, sistem tidak menghapus data akun secara permanen, melainkan menandai akun sebagai terhapus dan menampilkan daftar akun yang telah diperbarui. Mekanisme ini membuat akun yang dihapus “hilang dari daftar” tanpa benar-benar menghilangkan record dari basis data.

Pendekatan penandaan seperti ini selaras dengan pola pada backend yang menampilkan data “aktif” dengan memfilter isDeleted: false, sehingga data yang sudah dihapus secara logis tidak ikut ditampilkan, namun tetap tersimpan dan tidak dihapus permanen.

c) Tabel OrderItem

Tabel 3.3 menyajikan struktur tabel OrderItem yang digunakan dalam sistem sebagai media penyimpanan data rincian item pada setiap pesanan yang dilakukan oleh pengguna. Tabel ini dirancang untuk mencatat secara detail informasi produk yang terlibat dalam suatu transaksi, mulai dari nama produk, jumlah pembelian, satuan produk, hingga nilai harga yang dihasilkan dari proses pemesanan. Dengan adanya tabel OrderItem, sistem mampu memisahkan data pesanan utama dengan data rincian item, sehingga pengelolaan informasi transaksi dapat dilakukan secara lebih terstruktur dan sistematis.

Selain berfungsi sebagai penyimpanan detail transaksi, tabel OrderItem juga berperan dalam mendukung proses perhitungan total harga serta pencatatan riwayat penjualan. Informasi

harga yang disimpan pada tabel ini mencerminkan nilai transaksi berdasarkan jumlah item yang dipesan, sehingga dapat membantu sistem dalam menyajikan informasi keuangan yang lebih akurat. Hubungan antara tabel OrderItem dengan tabel pesanan dan tabel produk memungkinkan sistem untuk menjaga keterkaitan data antar entitas, serta memudahkan proses penelusuran data transaksi apabila diperlukan.

Perancangan struktur tabel OrderItem dilengkapi dengan penerapan tipe data dan constraint yang disesuaikan dengan fungsi masing-masing kolom. Hal ini bertujuan untuk menjaga konsistensi dan keutuhan data, serta mencegah terjadinya kesalahan pencatatan selama proses transaksi berlangsung. Dengan perancangan yang terstruktur, tabel OrderItem diharapkan dapat mendukung operasional sistem secara optimal, khususnya dalam proses pengelolaan pesanan dan pencatatan data penjualan secara berkelanjutan.

Tabel 3.3 Tabel OrderItem

Nama Kolom	**Tipe Data**	**Data Length**	**Constraint**
id	UUID	-	PK
quantity	Integer	-	NOT NULL
unit	Varchar	50	NOT NULL
totalBuyPrice	Float	-	-
totalSellPrice	Float	-	-
productId	UUID	-	FK ke tabel Product
orderId	UUID	-	FK ke tabel Order
isDeleted	Boolean	-	DEFAULT: false
createdAt	Timestamp	-	DEFAULT: now()
updatedAt	Timestamp	-	AUTO UPDATE

Pada Tabel OrderItem, kolom isDeleted (Boolean, DEFAULT: false) juga dipakai sebagai penanda soft delete untuk rincian item dalam pesanan. Hal ini relevan karena tabel OrderItem menyimpan detail transaksi (kuantitas, total harga, serta relasi ke Order dan Product) dan berperan mendukung perhitungan total harga serta pencatatan riwayat penjualan. Karena data OrderItem merupakan bagian penting dari histori transaksi, penyediaan isDeleted memungkinkan suatu item dianggap "tidak aktif/terhapus" secara logis tanpa menghapus catatan detail transaksi secara permanen.

Konsep ini konsisten dengan penjelasan implementasi soft delete di sistem (data aktif

ditampilkan melalui filter isDeleted: false) untuk menjaga integritas data tanpa penghapusan permanen dari basis data.

Kolom totalBuyPrice dan totalSellPrice pada tabel OrderItem digunakan untuk menyimpan nilai total harga beli dan harga jual pada setiap item produk dalam suatu pesanan. Nilai tersebut merepresentasikan hasil perhitungan harga berdasarkan kuantitas yang dipesan sehingga mendukung proses perhitungan total harga serta pencatatan riwayat penjualan dan penyajian informasi keuangan yang lebih akurat.

Kolom quantity pada tabel OrderItem digunakan untuk mencatat jumlah pembelian pada suatu item pesanan, sedangkan kolom unit digunakan untuk mencatat satuan produk yang menjelaskan bentuk kuantitas tersebut (misalnya kg/pcs/pack). Kombinasi keduanya diperlukan karena OrderItem dirancang untuk menyimpan rincian transaksi secara lengkap, termasuk informasi "jumlah" dan "satuan" pada setiap produk yang dipesan.

d) Tabel Product

Tabel 3.4 menampilkan struktur tabel Product yang digunakan dalam sistem sebagai media penyimpanan data produk yang dikelola. Tabel ini dirancang untuk menyimpan seluruh informasi yang berkaitan dengan produk, seperti nama produk, satuan, harga beli, dan harga jual, yang menjadi dasar dalam proses pencatatan transaksi penjualan. Keberadaan tabel Product memungkinkan sistem untuk mengelola data produk secara terpusat sehingga informasi yang digunakan pada setiap proses transaksi tetap konsisten dan terintegrasi.

Selain berfungsi sebagai penyimpanan data produk, tabel Product juga berperan dalam mendukung proses pengelolaan harga dan pencatatan penjualan. Informasi harga yang tersimpan pada tabel ini digunakan oleh sistem sebagai acuan dalam perhitungan transaksi, sehingga dapat membantu pengguna dalam melakukan pencatatan penjualan secara lebih terstruktur. Dengan adanya tabel Product, sistem dapat memastikan bahwa setiap transaksi merujuk pada data produk yang sama, sehingga meminimalkan terjadinya kesalahan pencatatan.

Perancangan struktur tabel Product dilengkapi dengan tipe data dan constraint yang disesuaikan dengan kebutuhan sistem. Hal ini bertujuan untuk menjaga konsistensi dan keutuhan data produk yang tersimpan dalam basis data, serta mendukung keberlangsungan penggunaan sistem dalam jangka panjang. Dengan perancangan yang terstruktur, tabel Product diharapkan dapat menunjang operasional sistem secara optimal, khususnya dalam proses pengelolaan data produk dan transaksi penjualan.

Tabel 3.4 Tabel Product

Nama Kolom	Tipe Data	Data Length	Constraint
id	UUID	-	PK
name	Varchar	255	NOT NULL
unit	Varchar	50	NOT NULL
buyPrice	Float	-	-
sellPrice	Float	-	-
image	Varchar	500	NULL
isDeleted	Boolean	-	DEFAULT: false
createdAt	Timestamp	-	DEFAULT: now()
updatedAt	Timestamp	-	AUTO UPDATE

Kolom pada tabel Product digunakan sebagai penanda penghapusan logis (*soft delete*). Setiap data produk yang tersimpan secara default memiliki nilai false, yang menunjukkan bahwa data masih aktif dan dapat digunakan dalam proses operasional. Ketika pengguna melakukan penghapusan produk, sistem tidak menghapus data tersebut secara permanen dari basis data, melainkan mengubah status **isDeleted** untuk menandai bahwa produk telah terhapus. Dengan mekanisme ini, sistem dapat menampilkan daftar produk yang diperbarui dengan hanya mengambil data yang memiliki status aktif (isDeleted = false), sekaligus menjaga integritas data tanpa menghilangkan informasi produk secara permanen.

e) Tabel Order

Tabel 3.5 menyajikan struktur tabel Order yang digunakan dalam sistem untuk menyimpan data pesanan yang dilakukan oleh pengguna. Tabel ini berfungsi sebagai penyimpanan utama informasi transaksi, mulai dari nilai total harga, waktu pemesanan, hingga status proses pesanan. Keberadaan tabel Order memungkinkan sistem untuk mencatat setiap transaksi secara terstruktur sehingga pengelolaan data pesanan dapat dilakukan secara sistematis dan terintegrasi dengan tabel lainnya.

Selain itu, tabel Order juga berperan dalam mendukung pemantauan alur proses pesanan, baik pesanan yang sedang diproses, telah selesai, maupun dibatalkan. Informasi waktu pada setiap tahapan pesanan dicatat untuk membantu sistem dalam menelusuri riwayat transaksi dan status pesanan. Perancangan tabel ini dilengkapi dengan tipe data dan constraint yang disesuaikan dengan kebutuhan sistem guna menjaga konsistensi dan keutuhan data transaksi.

Tabel 3.5 Tabel Order

Nama Kolom	Tipe Data	Data Length	Constraint
Id	UUID	-	PK
totalBuyPrice	Float	-	-
totalSellPrice	Float	-	-
date	Timestamp	-	NOT NULL
note	Text	-	NULL
startedAt	Timestamp	-	NULL
finishedAt	Timestamp	-	NULL
cancelledAt	Timestamp	-	NULL
partnerId	UUID	-	FK ke tabel Partner
isDeleted	Boolean	-	DEFAULT: false
createdAt	Timestamp	-	DEFAULT: now()
updatedAt	Timestamp	-	AUTO UPDATE

Pada Tabel Order, kolom isDeleted (Boolean, DEFAULT: false) berfungsi sebagai flag penghapusan logis untuk data pesanan/transaksi. Tabel Order sendiri memuat data inti pesanan seperti total harga, tanggal pesanan, status proses (startedAt, finishedAt, cancelledAt), serta relasi ke mitra (partnerId).

Penggunaan isDeleted tampak pada implementasi backend dashboard: pada fungsi overview, query menggunakan isDeleted: false sehingga pesanan yang sudah ditandai terhapus tidak ikut dihitung/ditampilkan dalam ringkasan dashboard (disebut sebagai "tidak termasuk data yang dihapus").

Selain itu, pada penjelasan fungsi trackOrder, disebutkan bahwa jika pesanan tidak ditemukan atau telah dihapus, sistem mengembalikan respons 404, yang menunjukkan bahwa status "terhapus" diperlakukan sebagai kondisi yang membuat pesanan tidak dapat diproses/diakses seperti biasa.

Kolom totalBuyPrice dan totalSellPrice pada tabel Order digunakan untuk menyimpan ringkasan total harga beli dan total harga jual dari seluruh item yang terdapat pada satu transaksi/pesanan. Penyimpanan total ini memudahkan sistem dalam menampilkan ringkasan pesanan serta mendukung proses pelaporan, termasuk perhitungan pendapatan dan keuntungan pada dashboard, tanpa perlu melakukan agregasi ulang dari seluruh data item setiap kali informasi ringkasan dibutuhkan.

f) Tabel Partner

Tabel 3.6 menyajikan struktur tabel Partner yang digunakan dalam sistem untuk menyimpan data mitra atau pihak yang bekerja sama dalam proses operasional. Tabel ini

berfungsi untuk mencatat informasi identitas mitra, seperti nama mitra, penanggung jawab, serta informasi kontak dan alamat. Keberadaan tabel Partner memungkinkan sistem untuk mengelola data mitra secara terpusat sehingga informasi yang berkaitan dengan pihak terkait dapat diakses dan digunakan secara konsisten dalam proses transaksi.

Selain sebagai penyimpanan data identitas mitra, tabel Partner juga berperan dalam mendukung proses pencatatan transaksi yang melibatkan pihak mitra. Informasi mitra yang tersimpan pada tabel ini digunakan sebagai acuan dalam setiap pesanan yang terkait, sehingga sistem dapat mengaitkan data transaksi dengan mitra yang bersangkutan. Dengan adanya tabel Partner, sistem mampu menjaga keterkaitan data antar tabel serta memudahkan penelusuran riwayat transaksi yang melibatkan mitra tertentu.

Perancangan struktur tabel Partner dilengkapi dengan tipe data dan constraint yang disesuaikan dengan kebutuhan sistem. Hal ini bertujuan untuk menjaga konsistensi dan keutuhan data mitra, serta mendukung keberlangsungan pengelolaan data dalam jangka panjang. Dengan struktur tabel yang terorganisir dengan baik, tabel Partner diharapkan dapat menunjang operasional sistem secara optimal, khususnya dalam proses pengelolaan data mitra dan transaksi yang berkaitan.

Tabel 3.6 Tabel Partner

Nama Kolom	Tipe Data	Data Length	Constraint
Id	UUID	-	PK
name	Varchar	255	NOT NULL
pic	Varchar	255	NOT NULL
countryCode	Varchar	10	NOT NULL
phone	Varchar	20	NOT NULL
address	Text	-	NOT NULL
image	Varchar	500	NULL
isDeleted	Boolean	-	DEFAULT: false
createdAt	Timestamp	-	DEFAULT: now()
updatedAt	Timestamp	-	AUTO UPDATE

Pada Tabel Partner, kolom isDeleted (Boolean, DEFAULT: false) digunakan sebagai penanda soft delete pada data mitra. Tabel Partner sendiri menyimpan informasi mitra yang terhubung dengan pesanan/transaksi, sehingga data mitra bersifat penting untuk keterkaitan dengan data pesanan.

Di bagian perancangan proses (activity diagram), ketika pengguna menghapus mitra, sistem melakukan penghapusan dengan cara menandai data mitra sebagai terhapus dan

menampilkan daftar mitra terbaru.

Secara implementasi, isDeleted juga digunakan untuk menjaga validitas data mitra yang dipakai sistem:

- Saat tambah mitra, sistem mengecek duplikasi nama; jika mitra ditemukan dan masih aktif (isDeleted = false), sistem menolak agar tidak terjadi duplikasi data aktif.
- Saat buat pesanan, sistem memvalidasi mitra yang dipilih; jika mitra tidak ditemukan atau statusnya terhapus (checkPartner.isDeleted), sistem mengembalikan pesan "Mitra tidak ditemukan". Ini memastikan pesanan tidak dibuat menggunakan mitra yang sudah tidak aktif.

Kolom image pada tabel **Partner** digunakan untuk menyimpan logo atau gambar mitra sebagai bagian dari identitas visual mitra di dalam sistem. Keberadaan atribut ini mendukung fitur pada aplikasi yang menyediakan unggah logo mitra saat proses penambahan data mitra, sehingga setiap mitra dapat dikenali lebih mudah secara visual. Identitas visual tersebut membantu meningkatkan kejelasan tampilan, terutama ketika data mitra ditampilkan kembali pada halaman daftar mitra maupun pada konteks pesanan yang terkait dengan mitra tertentu. Kolom image bersifat opsional (NULL) karena tidak semua mitra wajib memiliki logo/gambar, sehingga sistem tetap dapat menyimpan data mitra walaupun pengguna tidak mengunggah gambar.

3. Desain Antarmuka

Desain antarmuka merupakan tahapan penting dalam pengembangan aplikasi *Point of Sale* (POS) karena berperan langsung dalam menentukan kemudahan penggunaan dan pengalaman pengguna dalam berinteraksi dengan sistem. Pada penelitian ini, desain antarmuka disusun dengan memperhatikan prinsip *Prototyping* agar aplikasi yang dikembangkan tidak hanya fungsional, tetapi juga mudah dipahami dan nyaman digunakan oleh pengguna. Proses desain antarmuka diawali dengan pembuatan *wireframe* sebagai gambaran awal tata letak dan alur navigasi sistem, yang berfungsi untuk memvisualisasikan struktur halaman secara sederhana sebelum dilakukan perancangan visual secara detail. Selanjutnya, desain *Prototype* dikembangkan untuk menentukan tampilan visual, konsistensi elemen antarmuka, serta alur interaksi pengguna, sehingga aplikasi POS yang dihasilkan mampu mendukung aktivitas operasional pengguna secara efektif dan efisien.

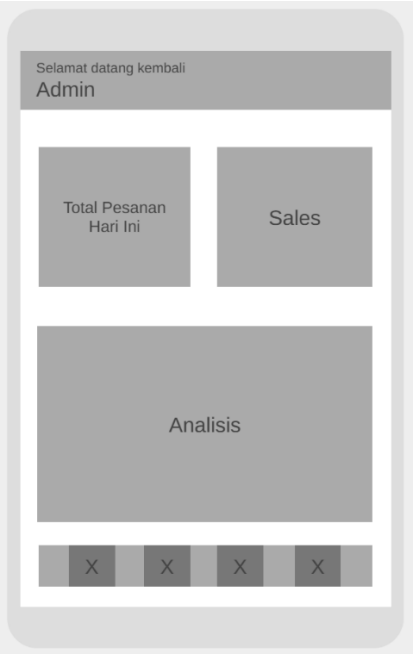
a) Wireframe

Wireframe digunakan untuk memvisualisasikan rancangan awal antarmuka aplikasi sebelum memasuki tahap pengembangan *prototype* secara detail. Pada tahap ini, wireframe berperan penting dalam menggambarkan struktur halaman, penempatan elemen, hierarki

informasi, serta alur navigasi dasar yang akan digunakan oleh pengguna. Selain sebagai representasi awal, wireframe juga membantu mengidentifikasi kebutuhan fungsional, memvalidasi alur kerja, serta memastikan bahwa setiap elemen antarmuka telah ditempatkan sesuai prioritas dan konteks pengguna. Dengan demikian, wireframe berfungsi sebagai blueprint awal yang memastikan setiap fitur dan alur kerja dalam aplikasi POS disusun secara sistematis, intuitif, dan selaras dengan kebutuhan operasional pabrik abon Merak Bewangi.

Untuk menghasilkan wireframe yang informatif, efisien, dan representatif, penelitian ini menggunakan pendekatan *low-fidelity wireframing* yang dibuat dengan bantuan perangkat lunak desain seperti Figma. Pendekatan ini dipilih karena memungkinkan proses iterasi yang cepat, fleksibel, dan mudah direvisi berdasarkan masukan pengguna pada tahap awal. Selain itu, *low-fidelity wireframe* menjadi sarana eksplorasi ide tanpa perlu terikat pada detail visual, sehingga fokus tetap berada pada alur interaksi dan struktur informasi. Wireframe yang dihasilkan memberikan gambaran menyeluruh mengenai alur interaksi pengguna, hubungan antarhalaman, serta mekanisme penggunaan fitur inti sebelum pengembangan desain visual akhir dilakukan secara lebih mendalam.

1) Dashboard

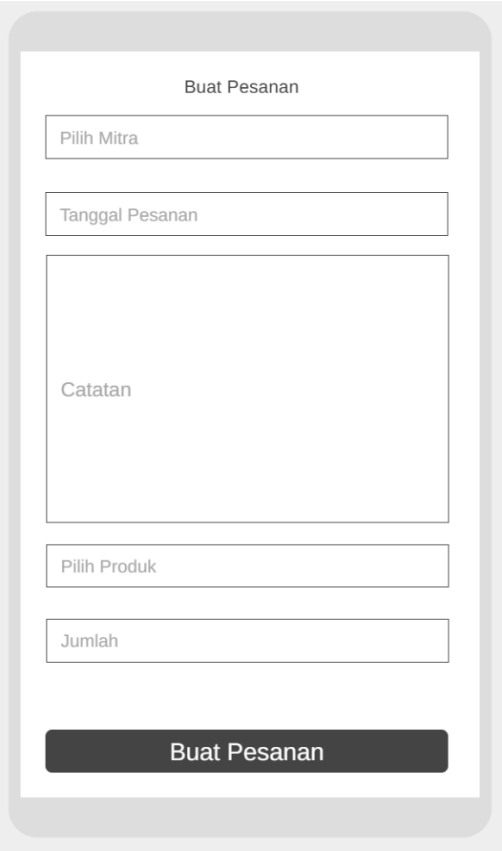


Gambar 3.9 *Wireframe Dashboard*

Pada Gambar 3.9 menampilkan rancangan tampilan awal yang berfungsi sebagai pusat informasi bagi pengguna. Pada bagian atas terdapat dua kartu utama yang menampilkan total pesanan hari ini dan informasi penjualan. Di bagian bawahnya, terdapat komponen analisis yang menampilkan ringkasan performa penjualan dalam

bentuk visual sederhana. Navigasi utama ditempatkan pada bagian bawah layar untuk memudahkan pengguna berpindah antarhalaman. Desain ini disusun untuk menyajikan informasi penting secara ringkas, mudah dibaca, dan cepat diakses sehingga mendukung kebutuhan monitoring harian oleh pemilik usaha.

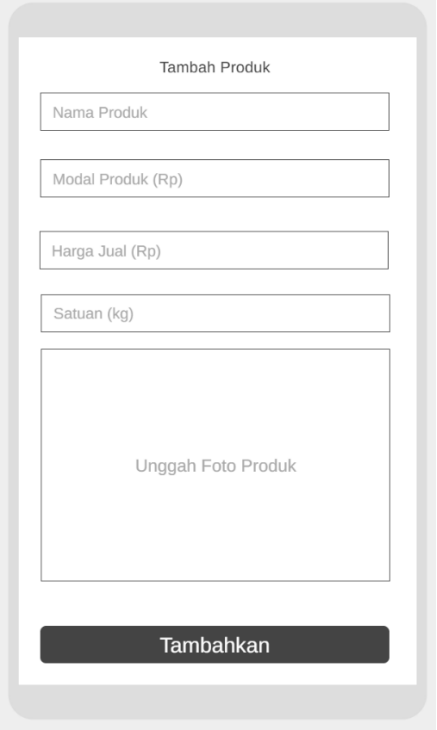
2) Buat Pesanan



Gambar 3.10 *Wireframe* Buat Pesanan

Pada Gambar 3.10 menggambarkan rancangan antarmuka untuk proses pembuatan pesanan baru. Halaman ini terdiri dari beberapa elemen input utama, yaitu pemilihan mitra, tanggal pesanan, kolom catatan, pemilihan produk, serta jumlah pesanan. Struktur ini dirancang agar pengguna dapat memasukkan data pesanan secara cepat dan terstruktur. Tombol "Buat Pesanan" ditempatkan pada bagian bawah untuk memudahkan eksekusi setelah seluruh informasi terisi. Desain ini mendukung proses operasional pabrik abon dengan memastikan setiap pesanan dicatat secara lengkap, konsisten, dan mudah dipahami pada tahap pemrosesan selanjutnya.

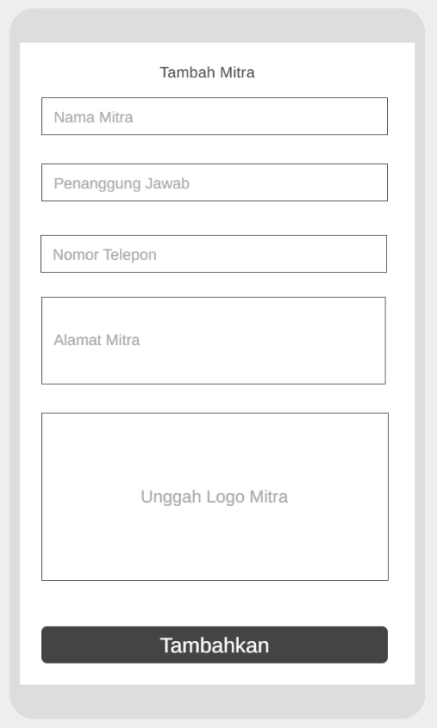
3) Tambah Produk



Gambar 3.11 *Wireframe* Tambah Produk

Pada Gambar 3.11 menggambarkan rancangan antarmuka untuk proses penambahan produk baru dalam aplikasi. Halaman ini menyediakan beberapa elemen input penting, meliputi nama produk, modal produk, harga jual, satuan, serta fitur unggah foto produk. Susunan elemen formulir dibuat sederhana dan terstruktur agar pengguna dapat memasukkan data produk dengan cepat dan minim kesalahan. Tombol "Tambahkan" ditempatkan pada bagian bawah sebagai aksi utama untuk menyimpan produk baru. Desain *wireframe* ini dibuat untuk memastikan bahwa proses penambahan produk dapat dilakukan secara efisien oleh pemilik usaha, serta mendukung integritas data dalam pengelolaan inventaris pada aplikasi POS.

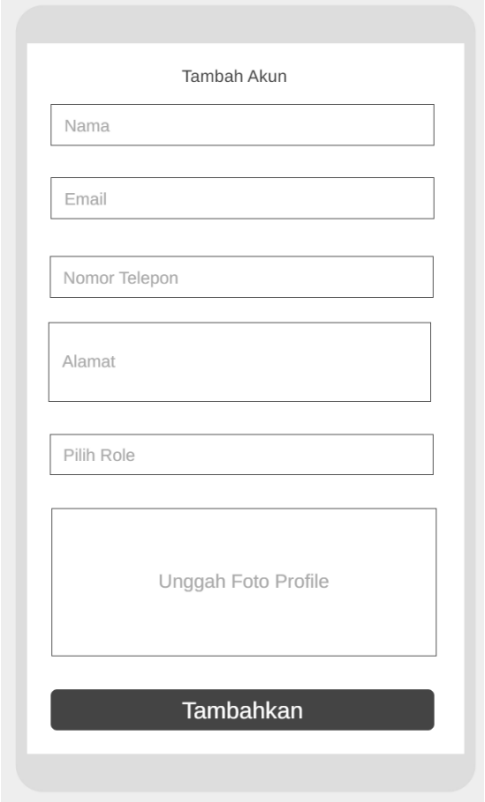
4) Tambah Mitra



Gambar 3.12 *Wireframe* Tambah Mitra

Pada Gambar 3.12 menampilkan rancangan antarmuka untuk proses penambahan mitra baru dalam aplikasi. Halaman ini menyediakan beberapa elemen input utama, seperti nama mitra, penanggung jawab, nomor telepon, dan alamat mitra. Selain itu, terdapat fitur unggah logo mitra untuk memudahkan identifikasi visual pada sistem. Setiap elemen formulir disusun secara terstruktur untuk memastikan proses input data dilakukan dengan cepat dan akurat. Tombol "Tambahkan" ditempatkan di bagian bawah sebagai aksi utama untuk menyimpan data mitra baru. Desain ini dibuat untuk mendukung proses administrasi pabrik abon Merak Bewangi, sehingga penambahan mitra dapat dilakukan secara efisien dan konsisten.

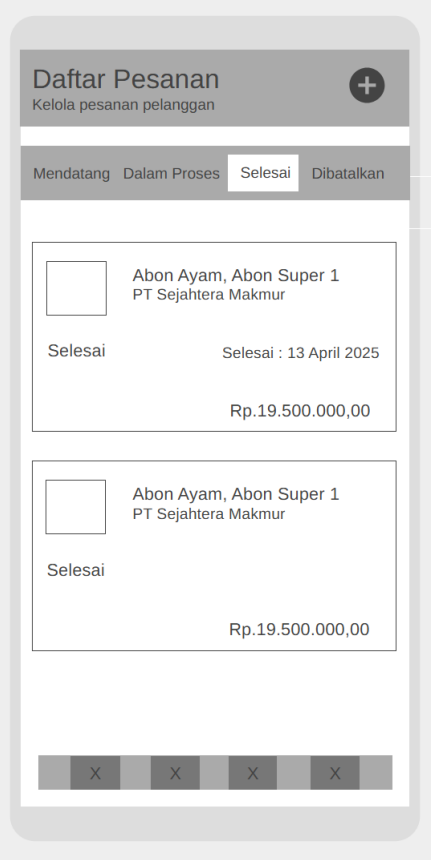
5) Tambah Akun



Gambar 3.13 *Wireframe* Tambah Akun

Pada Gambar 3.13 menggambarkan tampilan antarmuka untuk proses penambahan akun pengguna baru dalam aplikasi. Halaman ini memuat beberapa elemen input penting, seperti nama, email, nomor telepon, alamat, serta pemilihan role pengguna (*Owner* atau *Employee*). Selain itu, terdapat fitur unggah foto profil untuk melengkapi identitas visual akun. Seluruh elemen formulir disusun secara vertikal dan terstruktur untuk memastikan proses input data berlangsung dengan mudah, cepat, dan minim kesalahan. Tombol "Tambahkan" ditempatkan pada bagian bawah sebagai aksi utama untuk menyimpan akun baru. Desain ini mendukung kebutuhan administrasi pabrik abon Merak Bewangi dalam mengelola pengguna aplikasi secara lebih efektif dan terorganisir.

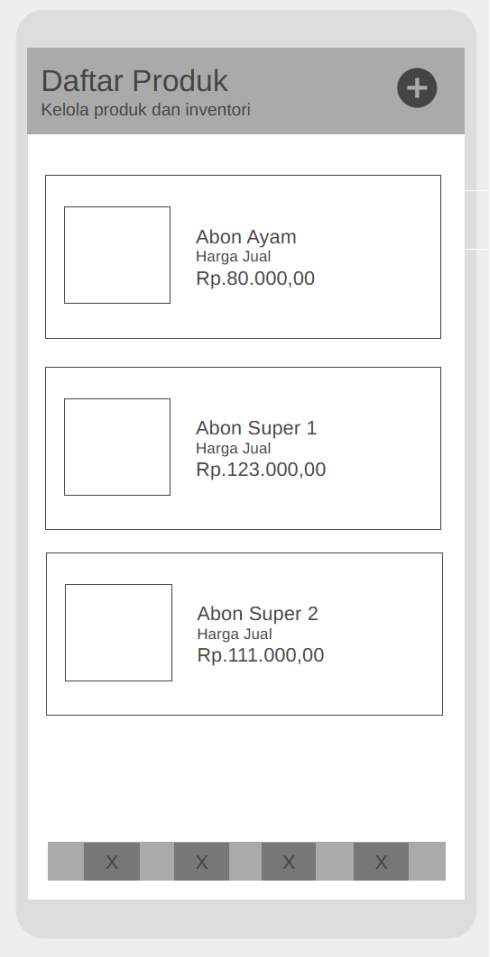
6) Daftar Pesanan



Gambar 3.14 *Wireframe* Daftar Pesanan

Pada Gambar 3.14 menggambarkan tampilan halaman Daftar Pesanan yang digunakan untuk mengelola seluruh pesanan pelanggan. Pada bagian atas terdapat judul halaman dan ikon tambah yang berfungsi untuk mengarahkan pengguna ke proses pembuatan pesanan baru. Tepat di bawahnya disediakan tab status pesanan, yaitu *Mendatang*, *Dalam Proses*, *Selesai*, dan *Dibatalkan*, sehingga pengguna dapat memfilter pesanan berdasarkan tahap pengerjaan. Setiap pesanan ditampilkan dalam bentuk kartu yang memuat informasi ringkas, seperti nama produk, nama mitra, status pesanan, tanggal penyelesaian, serta total nilai transaksi. Di bagian bawah halaman terdapat bilah navigasi utama yang memudahkan pengguna berpindah ke menu lain dalam aplikasi. Rancangan ini mendukung kebutuhan *monitoring* pesanan secara cepat dan terstruktur bagi pemilik maupun karyawan.

7) Daftar Produk



Gambar 3.15 *Wireframe* Daftar Produk

Wireframe pada Gambar 3.1 menampilkan rancangan antarmuka untuk halaman Daftar Produk, yang berfungsi sebagai pusat informasi seluruh produk yang tersedia dalam aplikasi POS. Pada bagian atas terdapat judul halaman dan ikon tambah yang memudahkan pengguna untuk menambahkan produk baru. Setiap produk ditampilkan dalam bentuk kartu yang memuat foto produk, nama produk, serta harga jual sehingga mempermudah proses identifikasi. Penyajian produk dalam format daftar vertikal ini dirancang agar pemilik maupun karyawan dapat menelusuri data produk dengan cepat dan efisien. Navigasi utama di bagian bawah halaman memastikan perpindahan antarfitur dapat dilakukan dengan mudah. Desain *wireframe* ini mendukung kebutuhan operasional pabrik abon Merak Bewangi dalam mengelola inventaris produk secara praktis, terstruktur, dan mudah diakses.

8) Daftar Akun

Gambar 3.16 *Wireframe* Daftar Akun

Wireframe pada Gambar 3.16 menampilkan rancangan antarmuka untuk halaman Daftar Akun, yang berfungsi sebagai pusat pengelolaan data mitra dan pegawai dalam aplikasi. Pada bagian atas halaman terdapat judul dan ikon tambah yang memberikan akses cepat untuk menambahkan akun baru. Setiap akun ditampilkan dalam format kartu yang memuat foto profil, nama, informasi perusahaan atau instansi terkait, serta jenis peran pengguna. Penyajian data dalam bentuk daftar vertikal memudahkan pengguna untuk menelusuri, mengenali, dan mengelola akun secara efisien. Navigasi utama di bagian bawah halaman memastikan akses yang mudah ke fitur lain dalam aplikasi. Desain ini dibuat untuk mendukung proses administrasi pengguna secara terstruktur, jelas, dan mudah dioperasikan.

b) Desain Prototype

Bagian *Prototype* membahas rancangan antarmuka yang dikembangkan berdasarkan

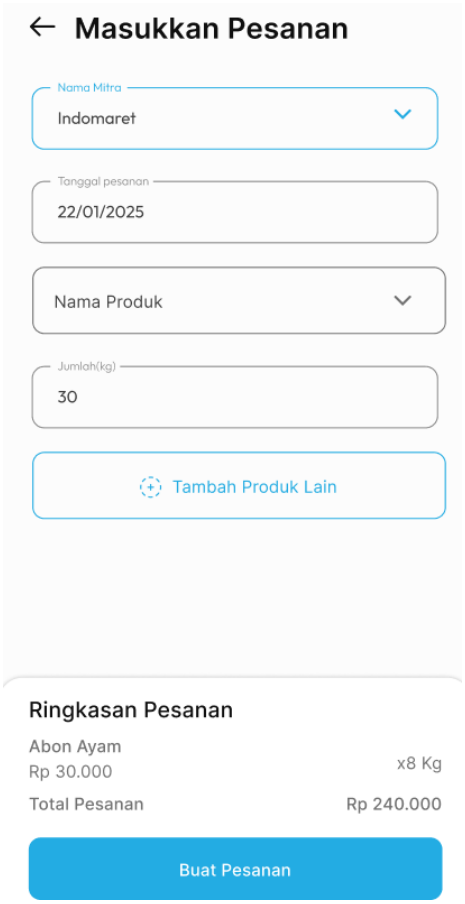
wireframe awal. Tahap ini berfokus pada penyempurnaan visual, pemilihan warna, tipografi, ikon, serta pengaturan tata letak agar aplikasi tidak hanya fungsional tetapi juga nyaman dan mudah digunakan. Pendekatan desain prototype dalam penelitian ini menekankan kesederhanaan, keterbacaan, serta konsistensi antarhalaman. Desain yang dihasilkan diharapkan mampu meningkatkan efisiensi operasional sekaligus memberikan pengalaman penggunaan yang optimal. Pendekatan desain *prototype* dalam penelitian ini mengutamakan prinsip kesederhanaan (*simplicity*), konsistensi komponen antarmuka, dan kejelasan informasi. Selain itu, proses perancangan mengikuti metode *design iteration* melalui pembuatan *high-fidelity mockup* menggunakan perangkat lunak Figma. Dengan metode ini, desain dapat diuji dan direvisi secara cepat berdasarkan umpan balik yang diterima pada tahap awal. Pembuatan protoype juga mengacu pada *design system* sederhana untuk memastikan keseragaman gaya visual, seperti penggunaan warna utama, gaya tombol, ukuran teks, serta ikonografi. Hasil akhir dari proses *prototype* diharapkan mampu meningkatkan efisiensi operasional, mengurangi beban kognitif pengguna, serta memberikan pengalaman penggunaan aplikasi yang lebih unggul, responsif, dan nyaman dalam konteks operasional pabrik abon.

1) Dashboard

Gambar 3.17 Desain Dashboard

Desain *dashboard* pada Gambar 3.17 menampilkan tampilan antarmuka yang berfungsi sebagai pusat informasi utama bagi pengguna. Halaman ini menyajikan ringkasan pesanan dan penjualan dalam bentuk kartu informasi yang mudah dibaca, seperti jumlah pesanan yang harus diproses hari ini serta total penjualan. Selain itu, terdapat visualisasi data penjualan dalam bentuk grafik batang yang menampilkan tren penjualan bulanan, sehingga memudahkan pemilik usaha dalam memantau performa bisnis secara cepat dan akurat. Navigasi utama ditempatkan di bagian bawah layar agar pengguna dapat berpindah antarmuka dengan mudah. Desain ini disusun dengan mempertimbangkan kenyamanan visual, keterbacaan, serta penyajian informasi yang ringkas namun tetap informatif.

2) Buat Pesanan



Gambar 3.18 Desain Buat Pesanan

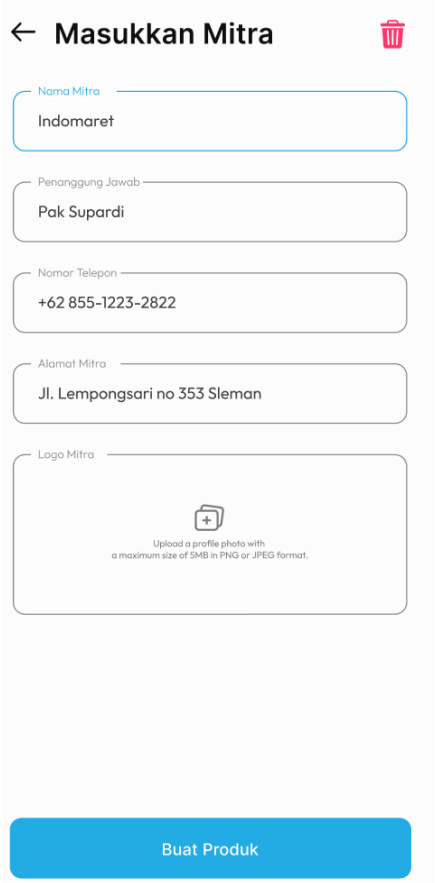
Desain pada Gambar 3.18 menampilkan antarmuka untuk proses pembuatan pesanan baru. Halaman ini menyediakan beberapa elemen input utama seperti pemilihan mitra, tanggal pesanan, pemilihan produk, serta jumlah produk yang dipesan. Terdapat pula tombol untuk menambah produk lain apabila pesanan terdiri dari lebih dari satu item. Pada bagian bawah halaman, ditampilkan ringkasan pesanan yang berfungsi untuk memberikan informasi awal mengenai rincian pesanan sebelum dikonfirmasi. Tombol "Buat Pesanan" ditempatkan pada posisi yang mudah dijangkau untuk memudahkan pengguna menyelesaikan proses input. Desain ini dirancang agar proses pencatatan pesanan dapat dilakukan dengan cepat, akurat, dan sesuai kebutuhan operasional pabrik abon Merak Bewangi.

3) Tambah Produk

Gambar 3.19 Desain Tambah Produk

Desain pada Gambar 3.19 menampilkan antarmuka untuk proses penambahan produk baru dalam aplikasi. Halaman ini menyediakan beberapa elemen input seperti nama produk, modal produk, harga jual, dan unggahan foto produk. Tata letak dirancang sederhana dan terstruktur agar pengguna dapat mengisi data produk secara cepat dan akurat. Tombol "Buat Produk" ditempatkan pada bagian bawah sebagai aksi utama untuk menyimpan produk baru. Desain ini memastikan bahwa proses penambahan produk berjalan efisien dan mendukung pengelolaan inventaris secara optimal dalam pabrik abon Merak Bewangi.

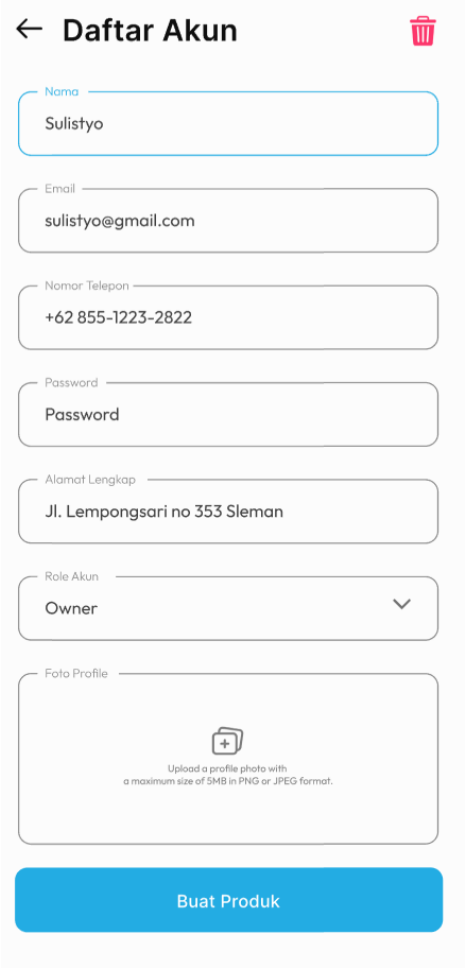
4) Tambah Mitra



Gambar 3.20 Desain Tambah Mitra

Desain pada Gambar 3.20 menampilkan antarmuka untuk proses penambahan mitra baru dalam aplikasi. Halaman ini menyediakan beberapa elemen input utama, yaitu nama mitra, nama penanggung jawab, nomor telepon, alamat mitra, serta unggahan logo mitra. Setiap komponen formulir disusun secara vertikal dengan jarak yang cukup agar mudah dibaca dan diisi oleh pengguna. Di bagian bawah terdapat tombol aksi untuk menyimpan data mitra yang telah dimasukkan. Rancangan ini bertujuan memudahkan pemilik usaha dalam menambahkan mitra secara cepat dan terstruktur, sehingga proses administrasi relasi bisnis dapat dikelola dengan lebih rapi dan konsisten melalui aplikasi.

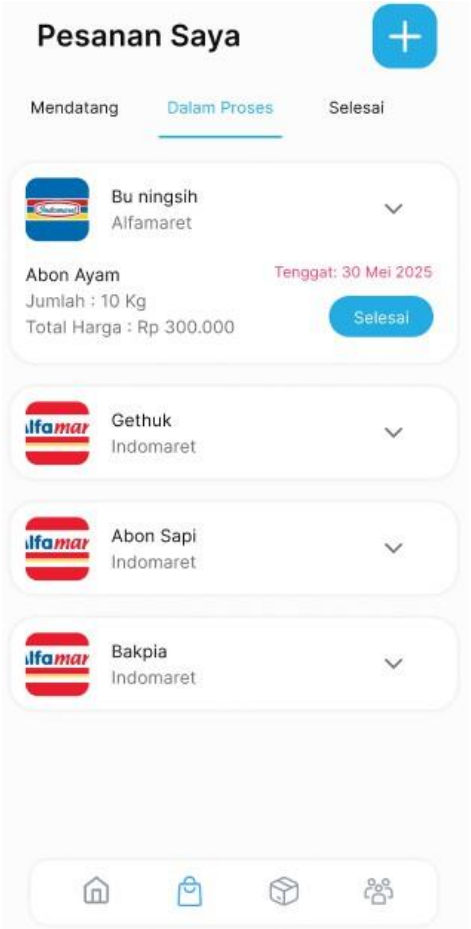
5) Tambah Akun



Gambar 3.21 Desain Tambah Akun

Desain pada Gambar 3.21 menampilkan antarmuka untuk proses penambahan akun pengguna baru. Halaman ini terdiri dari beberapa elemen input seperti nama lengkap, email, nomor telepon, password, alamat, serta pemilihan role pengguna (*Owner* atau *Employee*). Selain itu, terdapat fitur unggah foto profil untuk melengkapi identitas visual akun. Tata letak formulir dirancang sederhana dan mudah dipahami agar proses pembuatan akun dapat dilakukan dengan cepat dan minim kesalahan. Tombol aksi ditempatkan di bagian bawah halaman untuk menyimpan akun baru.

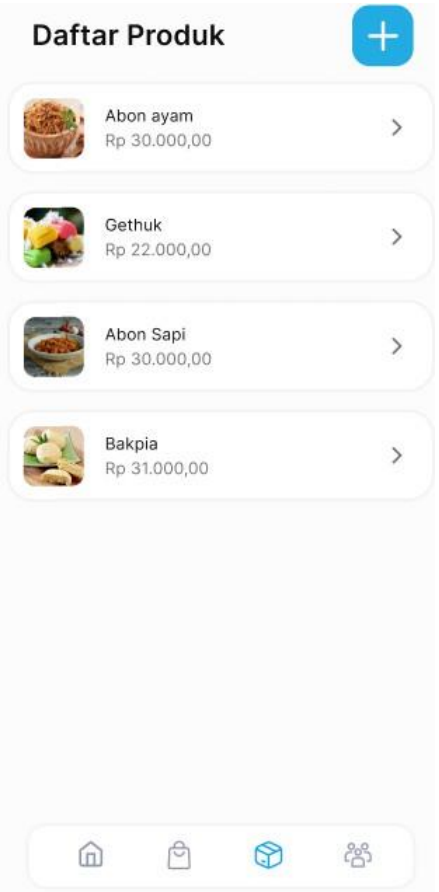
6) Daftar Pesanan



Gambar 3.22 Desain Daftar Pesanan

Desain pada Gambar 3.22 menampilkan tampilan antarmuka untuk halaman Daftar Pesanan yang digunakan untuk memantau seluruh pesanan berdasarkan statusnya. Halaman ini menyediakan beberapa tab kategori, seperti *Mendatang*, *Dalam Proses*, dan *Selesai*, sehingga pengguna dapat melihat pesanan sesuai tahap pengerjaan. Setiap pesanan ditampilkan dalam bentuk kartu yang memuat informasi penting seperti nama mitra, jenis produk, jumlah pesanan, total harga, serta tenggat pemesanan. Terdapat pula tombol aksi untuk memperbarui status pesanan dengan cepat. Ikon tambah pada bagian atas memungkinkan pengguna membuat pesanan baru secara langsung. Desain ini disusun untuk memudahkan monitoring dan pengelolaan pesanan secara efisien, khususnya bagi pemilik dan karyawan yang menangani proses operasional harian.

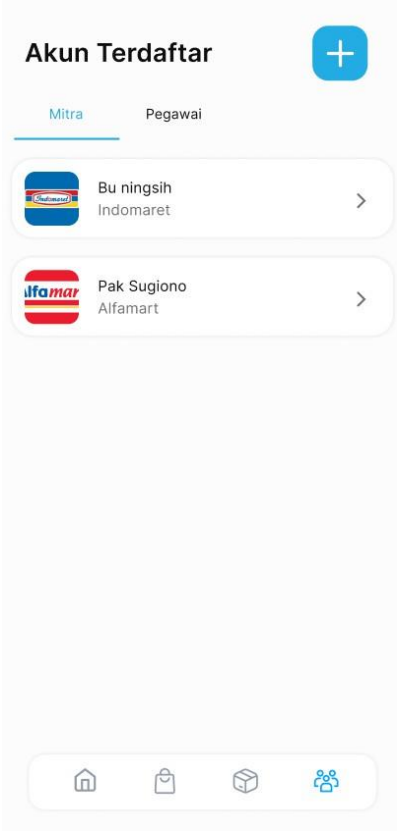
7) Daftar Produk



Gambar 3.23 Desain Daftar Produk

Desain pada Gambar 3.23 menampilkan antarmuka untuk halaman Daftar Produk, yang berfungsi sebagai pusat pengelolaan seluruh produk yang tersedia dalam aplikasi. Setiap produk ditampilkan dalam bentuk kartu yang memuat foto produk, nama produk, dan harga jual sehingga memudahkan pengguna dalam mengenali dan menelusuri daftar produk. Ikon tambah pada bagian atas halaman memudahkan pengguna untuk menambahkan produk baru secara cepat. Tampilan yang bersih dan terstruktur ini dirancang untuk mendukung proses administrasi produk secara efisien, memastikan bahwa pemilik atau karyawan dapat mengelola inventaris dengan mudah dan akurat.

8) Daftar Akun



Gambar 3.24 Desain Daftar Akun

Desain pada Gambar 3.24 menampilkan antarmuka untuk halaman Daftar Akun yang digunakan untuk mengelola data mitra dan pegawai. Halaman ini menyediakan dua tab utama, yaitu *Mitra* dan *Pegawai*, yang memungkinkan pengguna menelusuri akun berdasarkan kategorinya. Setiap akun ditampilkan dalam format kartu yang memuat foto profil, nama pengguna, serta informasi perusahaan atau instansi terkait. Ikon tambah pada bagian atas memberikan akses cepat untuk menambahkan akun baru. Navigasi utama pada bagian bawah halaman memastikan pengguna dapat berpindah antarfitur dengan mudah. Desain ini disusun untuk mendukung proses pengelolaan akun secara terstruktur, rapi, dan efisien dalam aplikasi.

c. Pengembangan

Tahap pengembangan merupakan proses realisasi dari seluruh perancangan desain yang telah disusun. Pada tahap ini, rancangan sistem yang sebelumnya divisualisasikan melalui *use case diagram*, *activity diagram*, desain *database*, serta *wireframe* mulai diterjemahkan menjadi kode program yang berfungsi secara nyata. Pada tahap ini, seluruh fitur inti mulai diimplementasikan agar rancangan sistem dapat berfungsi secara nyata

sebagai bagian dari *Minimum Viable Product* (MVP). Fitur yang dikembangkan pada iterasi pertama mencakup proses autentikasi pengguna, pengelolaan data produk, akun, mitra, pencatatan pesanan yang terintegrasi dengan alur operasional pabrik, serta penyajian tampilan *dashboard* penjualan untuk mendukung monitoring secara *real time*. Seluruh proses pengembangan dilakukan dengan pendekatan iteratif agar hasil implementasi sesuai dengan kebutuhan pengguna dan dapat divalidasi pada tahap berikutnya.

Metode pembuatan pada tahap pengembangan menerapkan pendekatan *component-based development* dengan struktur modul yang terorganisasi. Setiap fitur dipecah menjadi komponen fungsional yang bersifat *reusable*, sehingga memudahkan proses *refactoring*, unit testing, dan pengembangan berkelanjutan. Implementasi *frontend* dilakukan menggunakan *React Native* dengan arsitektur berbasis *state management* yang konsisten melalui Context API, sehingga alur data antar-komponen dapat terjaga stabil. TypeScript digunakan untuk menyediakan *static type checking* sehingga error dapat terdeteksi sejak tahap kompilasi, memperkuat reliabilitas aplikasi. Pada sisi *backend*, Supabase dimanfaatkan sebagai *Backend-as-a-Service (BaaS)* yang menyediakan PostgreSQL, REST API otomatis, dan autentikasi terintegrasi. Integrasi antara frontend dan backend dilakukan melalui query terstruktur menggunakan *Supabase Client*, yang mendukung operasi *CRUD* dengan latensi rendah. Seluruh proses pengembangan dicatat menggunakan *version control* GitHub, termasuk *branching workflow* untuk memisahkan fitur dalam *feature branch*, *pull request review*, dan dokumentasi commit yang sistematis. Selain itu, proses *hot reload* dan *live debugging* pada *React Native* digunakan untuk mempercepat iterasi antarmuka selama tahap pengembangan. Dengan kombinasi metode ini, proses development menjadi lebih skalabel, modular, dan selaras dengan prinsip *Lean Startup* yang menekankan eksperimen cepat, validasi berkelanjutan, serta kemampuan adaptasi terhadap masukan pengguna secara langsung.

3.3.2 Measure

Tahap *Measure* pada iterasi pertama dilakukan untuk mengukur sejauh mana MVP yang telah dikembangkan mampu menjawab kebutuhan pengguna dan menyelesaikan permasalahan utama pada proses operasional pabrik abon Merak Bewangi. Pengukuran dilakukan dengan cara mengamati langsung penggunaan aplikasi oleh pemilik dan karyawan, kemudian mencatat respons pengguna terhadap alur kerja, kecepatan pencatatan, serta kemudahan navigasi pada fitur-fitur inti. Selain itu, peneliti juga mengumpulkan umpan balik melalui diskusi terarah untuk mengetahui bagian mana yang dianggap membantu, kurang jelas, atau memerlukan

perbaikan. Data yang diperoleh pada tahap ini mencakup waktu yang dibutuhkan untuk mencatat sebuah pesanan, tingkat kesalahan input, serta kenyamanan pengguna ketika mengoperasikan aplikasi. Hasil dari proses pengukuran ini menjadi dasar penting dalam menentukan prioritas perbaikan pada tahap *Learn*, sehingga iterasi berikutnya dapat menghasilkan peningkatan fitur yang lebih relevan dan tepat sasaran.

a. Black Box Testing

Black Box Testing dilakukan sebagai metode pengujian fungsional untuk memastikan setiap fitur pada MVP berjalan sesuai dengan kebutuhan pengguna tanpa memeriksa struktur internal kode. Pengujian ini berfokus pada input yang diberikan pengguna dan output yang dihasilkan sistem. Setiap fungsi utama seperti *login*, pembuatan pesanan, pengelolaan produk, pengelolaan mitra, dan perubahan status pesanan diuji dengan berbagai skenario untuk mengevaluasi keakuratan proses, ketahanan terhadap kesalahan input, serta konsistensi hasil. Metode ini dipilih karena sesuai dengan konteks penelitian berbasis *Lean Startup*, di mana fokus utama adalah memastikan fungsi inti bekerja dengan benar pada tahap awal sebelum dilakukan penyempurnaan lebih lanjut pada iterasi berikutnya.

b. Wawancara

Wawancara dilakukan pada tahap *Measure* iterasi pertama sebagai metode pengumpulan data kualitatif untuk memperoleh umpan balik langsung dari pengguna setelah melakukan demo penggunaan aplikasi MVP. Wawancara ini bertujuan untuk menggali persepsi pengguna terkait kesesuaian fitur, kemudahan penggunaan, serta alur kerja aplikasi dalam mendukung aktivitas operasional pabrik abon Merak Bewangi. Proses wawancara dilakukan secara langsung kepada pemilik dan karyawan setelah mereka mencoba menggunakan aplikasi, dengan pendekatan diskusi terarah agar pengguna dapat menyampaikan pendapat, pengalaman, serta kendala yang dirasakan selama penggunaan sistem.

Pertanyaan wawancara difokuskan pada aspek fungsionalitas fitur inti, kejelasan alur proses pencatatan pesanan, kemudahan navigasi antarmuka, serta kecepatan aplikasi dalam mendukung pekerjaan pengguna. Selain itu, wawancara juga digunakan untuk mengidentifikasi bagian sistem yang dianggap paling membantu, fitur yang masih membingungkan, serta saran perbaikan yang diharapkan pengguna. Hasil wawancara ini digunakan sebagai dasar evaluasi pada tahap *Measure* dan menjadi masukan utama pada tahap *Learn* untuk menentukan prioritas perbaikan dan pengembangan fitur pada iterasi selanjutnya.

3.3.3 Learn

Tahap Learn merupakan proses analisis terhadap seluruh temuan yang diperoleh pada tahap Measure, dengan tujuan untuk memahami pola penggunaan, kendala yang dialami pengguna, serta efektivitas fitur-fitur yang telah diimplementasikan pada MVP. Pada tahap ini, peneliti meninjau kembali data hasil observasi, hasil *Black Box Testing*, serta hasil dari wawancara yang berupa *feedback* langsung dari pengguna. Temuan tersebut kemudian digunakan sebagai landasan dalam menentukan langkah perbaikan dan penyempurnaan fitur pada iterasi berikutnya.

Seluruh temuan tersebut dianalisis untuk mengidentifikasi bagian sistem yang telah berjalan dengan baik, fitur yang masih kurang optimal, serta aspek-aspek yang memerlukan penyempurnaan. Hasil analisis pada tahap *Learn* kemudian digunakan sebagai dasar dalam menentukan prioritas perbaikan dan arah pengembangan pada iterasi berikutnya, sehingga proses pengembangan aplikasi dapat dilakukan secara lebih terarah, adaptif, dan sesuai dengan kebutuhan nyata pengguna di lingkungan operasional pabrik abon Merak Bewangi.

3.4 Iterasi 2

Iterasi kedua dilakukan sebagai tindak lanjut dari temuan pada tahap Learn iterasi pertama. Fokus utama pada iterasi ini adalah melakukan penyempurnaan fitur, peningkatan pengalaman pengguna, serta optimalisasi performa aplikasi agar lebih responsif dan sesuai dengan kebutuhan operasional pabrik abon Merak Bewangi. Iterasi ini tetap mengikuti alur *Build– Measure–Learn*, namun dengan ruang lingkup pengembangan yang lebih terarah berdasarkan data validasi pengguna dari MVP sebelumnya.

3.4.1 Build

Tahap *Build* pada iterasi kedua merupakan tahap pengembangan lanjutan yang dilakukan setelah peneliti memperoleh pembelajaran (*Learn*) dari hasil pengukuran dan evaluasi pada iterasi pertama. Pada tahap ini, pengembangan aplikasi difokuskan pada penerapan hasil analisis terhadap temuan sebelumnya. Perbaikan dan penyempurnaan yang dilakukan pada iterasi kedua tidak diarahkan pada penambahan fitur yang bersifat kompleks, melainkan pada optimalisasi fitur-fitur inti agar sistem dapat berjalan lebih stabil, responsif, dan efisien dalam mendukung proses operasional. Melalui tahap *Build* ini, MVP yang telah dikembangkan sebelumnya disempurnakan berdasarkan kebutuhan nyata pengguna, sehingga aplikasi diharapkan mampu memberikan peningkatan kualitas penggunaan serta menjadi solusi yang lebih tepat guna bagi pabrik abon Merak Bewangi.

3.4.2 Measure

Tahap *Measure* pada iterasi kedua dilakukan untuk mengevaluasi hasil penyempurnaan sistem yang telah diimplementasikan pada tahap *Build* iterasi kedua. Pada tahap ini, pengukuran difokuskan tidak hanya pada kinerja teknis aplikasi, tetapi juga pada tingkat penerimaan dan kenyamanan pengguna dalam menggunakan sistem yang telah dioptimalkan. Proses pengukuran dilakukan menggunakan tiga metode utama, yaitu *Black Box Testing*, wawancara, dan *System Usability Scale* (SUS).

Black Box Testing digunakan untuk memastikan bahwa seluruh fungsi inti aplikasi berjalan sesuai dengan kebutuhan dan skenario penggunaan setelah dilakukan perbaikan pada iterasi kedua. Pengujian ini difokuskan pada validasi input dan output sistem tanpa melihat struktur internal kode, sehingga dapat memastikan stabilitas dan konsistensi fungsi aplikasi dalam mendukung proses operasional.

Selain itu, wawancara dilakukan untuk memperoleh umpan balik langsung dari

pengguna setelah menggunakan aplikasi dalam aktivitas operasional sehari-hari. Wawancara ini bertujuan untuk menggali persepsi pengguna terkait kemudahan penggunaan, kejelasan alur kerja, serta dampak perbaikan sistem terhadap efisiensi kerja. Pendekatan ini memungkinkan peneliti memperoleh pemahaman kualitatif yang lebih mendalam mengenai pengalaman pengguna.ung proses operasional.

Pengukuran juga dilengkapi dengan metode *System Usability Scale* (SUS) untuk menilai tingkat kegunaan aplikasi secara kuantitatif. Kuesioner SUS diberikan kepada pengguna setelah mereka menggunakan aplikasi pada iterasi kedua, guna memperoleh skor *usability* yang merepresentasikan tingkat kemudahan, kenyamanan, dan penerimaan sistem. Hasil pengukuran dari ketiga metode tersebut digunakan secara komprehensif untuk menilai keberhasilan iterasi kedua serta menjadi dasar dalam penarikan kesimpulan penelitian.

3.4.3 Learn

Tahap *Learn* pada iterasi kedua merupakan tahapan pembelajaran akhir yang berfokus pada analisis mendalam terhadap seluruh hasil pengukuran yang telah diperoleh pada tahap *Measure* iterasi kedua. Pada tahap ini, peneliti melakukan evaluasi secara komprehensif dengan mengintegrasikan temuan dari *Black Box Testing*, hasil wawancara pengguna, serta nilai *System Usability Scale* (SUS) untuk memperoleh gambaran yang menyeluruh mengenai kinerja dan kualitas aplikasi yang telah disempurnakan. Proses pembelajaran ini bertujuan untuk memahami sejauh mana perbaikan dan optimalisasi yang dilakukan pada iterasi kedua mampu meningkatkan stabilitas sistem, memperbaiki alur kerja, serta meningkatkan kenyamanan dan penerimaan pengguna dalam penggunaan aplikasi *Point of Sale* (POS).

Analisis pada tahap *Learn* tidak hanya berfokus pada keberhasilan fungsi sistem secara teknis, tetapi juga pada pengalaman pengguna dalam konteks operasional nyata. Umpan balik kualitatif dari wawancara digunakan untuk mengidentifikasi persepsi pengguna terhadap kemudahan penggunaan, kejelasan navigasi, serta efektivitas aplikasi dalam mendukung aktivitas kerja sehari-hari. Sementara itu, hasil *Black Box Testing* memastikan bahwa seluruh fungsi inti sistem berjalan secara konsisten dan bebas dari kesalahan yang dapat mengganggu proses operasional. Nilai SUS yang diperoleh digunakan sebagai indikator kuantitatif untuk menilai tingkat kegunaan aplikasi secara objektif.

Hasil pembelajaran pada tahap *Learn* iterasi kedua kemudian dijadikan dasar untuk menilai keberhasilan keseluruhan proses pengembangan aplikasi berdasarkan metodologi *Lean Startup*. Tahap ini berperan penting dalam menentukan apakah sistem yang dikembangkan telah memenuhi kebutuhan pengguna, mencapai tujuan penelitian, serta layak untuk digunakan

sebagai solusi operasional. Dengan demikian, tahap *Learn* pada iterasi kedua menjadi penutup dari siklus *Build–Measure–Learn* sekaligus menjadi landasan dalam penarikan kesimpulan dan perumusan saran pada tahap akhir penelitian.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil

Bagian hasil menyajikan temuan penelitian yang diperoleh dari proses implementasi dan pengujian sistem yang telah dikembangkan. Hasil penelitian disajikan secara sistematis berdasarkan data yang dikumpulkan selama pengujian, sehingga memberikan gambaran mengenai kinerja sistem serta tingkat kesesuaian sistem terhadap kebutuhan pengguna.

4.1.1 Iterasi 1

Iterasi pertama merupakan implementasi awal dari siklus Build–Measure–Learn untuk menghasilkan *Minimum Viable Product* (MVP). Pada iterasi ini, pengembangan sistem difokuskan pada penyusunan fitur inti, validasi kebutuhan pengguna, serta pengujian awal fungsi-fungsi dasar aplikasi. Hasil dari iterasi ini digunakan sebagai pijakan untuk melakukan perbaikan pada iterasi berikutnya.

a. Build

1. Hasil Analisis Kebutuhan Sistem

Hasil analisis kebutuhan pada iterasi pertama diperoleh dari wawancara mendalam dan observasi langsung terhadap pemilik serta dua karyawan pabrik abon Merak Bewangi. Berdasarkan proses tersebut, ditemukan beberapa kebutuhan inti yang harus dipenuhi oleh aplikasi POS agar mampu menggantikan proses pencatatan manual yang selama ini digunakan.

Dari sisi operasional, sistem yang dikembangkan harus mampu menyediakan mekanisme pencatatan pesanan yang lebih cepat, terstruktur, dan minim kesalahan. Pengguna membutuhkan fitur untuk mencatat pesanan beserta detail mitra, tanggal pesanan, jumlah produk, dan catatan tambahan. Selain itu, pemilik usaha membutuhkan informasi penjualan yang dapat ditampilkan secara ringkas dan mudah dipahami, sehingga sistem perlu menyediakan dashboard dengan ringkasan pesanan dan visualisasi penjualan.

Dari sisi manajemen data, pengguna menginginkan fitur pengelolaan produk, mitra, dan akun agar seluruh informasi usaha tidak lagi tersebar pada buku catatan terpisah. Setiap entitas harus dapat ditambahkan, diperbarui, dan ditampilkan kembali dengan mudah melalui aplikasi. Analisis kebutuhan juga menunjukkan pentingnya autentikasi pengguna karena terdapat perbedaan akses antara pemilik dan karyawan, sehingga sistem wajib mendukung *role-based*

access.

Secara keseluruhan, hasil analisis kebutuhan sistem menunjukkan bahwa aplikasi harus mampu:

a) Mendukung pencatatan pesanan secara cepat dan terstruktur.
b) Menyediakan dashboard penjualan sebagai dasar pengambilan keputusan.
c) Mengelola data produk, mitra, dan akun dengan rapi.
d) Memfasilitasi akses berbeda antara *owner* dan *employee*.
e) Memberikan pengalaman penggunaan yang sederhana dan mudah dipahami.

Temuan kebutuhan ini menjadi dasar dalam menentukan fitur inti yang akan diwujudkan pada tahap penge mbangan MVP dalam iterasi pertama.

2. Hasil Implementasi MVP

Tahap *Build* pada iterasi pertama menghasilkan sebuah MVP aplikasi POS yang telah memuat fitur inti sesuai kebutuhan pengguna pabrik abon Merak Bewangi. Implementasi dilakukan berdasarkan rancangan pada tahap sebelumnya yang mencakup diagram alur, ERD, *wireframe*, dan UI/UX.

a) Dashboard

Gambar 4.1 Dashboard

Gambar 4.1 menampilkan hasil implementasi halaman Dashboard pada MVP aplikasi POS pabrik abon Merak Bewangi. Dashboard berfungsi sebagai pusat informasi utama yang

memberikan ringkasan kondisi operasional secara real-time. Pada bagian atas, pengguna disambut dengan informasi peran (Owner) dan status harian, termasuk jumlah pesanan yang harus diproses pada hari tersebut. Selain itu, terdapat kartu informasi *Sales* yang menampilkan total pendapatan secara ringkas sehingga pemilik usaha dapat memantau performa bisnis tanpa harus membuka halaman lain.

Bagian berikutnya menampilkan grafik analisis penjualan dalam bentuk diagram batang tahunan yang memperlihatkan total pesanan per bulan. Visualisasi ini membantu pengguna memahami tren permintaan produk dari waktu ke waktu dan mendukung pengambilan keputusan strategis terkait produksi dan distribusi. Menu navigasi bawah disusun secara intuitif agar pengguna dapat berpindah ke fitur lain seperti pesanan, produk, dan akun dengan cepat. Secara keseluruhan, desain dashboard ini dirancang untuk menyajikan informasi penting secara ringkas, mudah dipahami, dan mendukung efisiensi operasional pabrik abon Merak Bewangi.

Import

```
import express from 'express'
import prisma from '../db/prisma.js'
const router = express.Router()
import    {    chartProductAnnual,    chartProductMonth,
chartProductWeek } from './chart-product.js'
import    {    chartIncomeAnnual,    chartIncomeMonth,
chartIncomeWeek } from './chart-income.js'
```

Gambar 4.2

Kode Implementasi

```
const overview = async (req, res) => {
    try {
        const now = new Date()
        now.setHours(0, 0, 0, 0)
        const endDay = new Date(now.getTime() + 86400000)
        const [countOrder, orders, orderItems] = await
Promise.all([
            prisma.order.count({
                where: {
                    AND: [
                        {
                            isDeleted: false
                        },
                        {
                            date: {
                                gte: now,
                                lte: endDay,
```

```
                            }
                        },
                        {
                            startedAt: null
                        },
                        {
                            cancelledAt: null
                        }
                    ]
                }
            }),
            prisma.order.findMany({
                where: {
                    AND: [
                        {
                            isDeleted: false
                        },
                        {
                              finishedAt: {
                                  not: null
                            }
                        }
                    ]
                },
                select: {
                    totalSellPrice: true,
                    totalBuyPrice: true,
                    orderItems: {
                        select: {
                            quantity: true
                        }
                    }
                }
            }),
            prisma.orderItem.findMany({
                where: {
                    order: {
                        AND: [
                            {
                                isDeleted: false

                            },
                            {
                                  finishedAt: {
                                      not: null
                                }
                            }
                        ]
                    }
                },
                select: {
                    quantity: true,
                }
            })
        ])
```

Gambar 4.3

Penjelasan : Fungsi overview berperan sebagai *controller* untuk menyajikan data ringkasan (*dashboard*) pada sistem. Pada fungsi ini, sistem terlebih dahulu menentukan rentang waktu hari berjalan dengan mengatur waktu ke awal hari dan menambahkan 24 jam sebagai batas akhir. Selanjutnya, dilakukan tiga buah query secara paralel menggunakan *Promise.all* agar proses pengambilan data lebih efisien. Query pertama menggunakan *prisma.order.count()* untuk menghitung jumlah pesanan pada hari tersebut yang belum diproses, tidak dibatalkan, serta tidak termasuk data yang dihapus. Query kedua menggunakan *prisma.order.findMany()* untuk mengambil daftar pesanan yang telah selesai diproses beserta informasi total harga jual, total harga beli, dan jumlah item yang dijual. Sementara itu, query ketiga menggunakan *prisma.orderItem.findMany()* untuk memperoleh data seluruh item pesanan yang berasal dari order yang telah selesai, khususnya pada atribut jumlah (*quantity*). Data hasil ketiga query ini selanjutnya dimanfaatkan sebagai dasar perhitungan statistik dan visualisasi pada halaman *dashboard* sistem.

```
let totalIncome = 0
let totalProfit = 0
let averageTransactionValue = 0
let totalItem = 0

    for (const order of orders) {
                totalIncome += order.totalSellPrice
                   const profit = order.totalSellPrice -
order.totalBuyPrice
                totalProfit += profit
                                          totalItem     +=
order.orderItems.reduce((total,    item)    =>    total    +
item.quantity, 0)
            }

            averageTransactionValue = Number(totalProfit)
/ Number(totalItem) || 0

            let totalOrderItem = 0

            for (const orderItem of orderItems) {
                totalOrderItem += orderItem.quantity
            }
```

Gambar 4.4

Penjelasan : Pada tahap ini dilakukan proses perhitungan statistik penjualan yang diperlukan untuk ditampilkan pada halaman *dashboard*. Sistem menghitung total pendapatan (*totalIncome*), total keuntungan (*totalProfit*), total jumlah barang yang terjual (*totalItem*), serta nilai rata-rata keuntungan per item (*averageTransactionValue*). Pendapatan dihitung

dari penjumlahan total harga jual pada setiap transaksi, sedangkan keuntungan diperoleh dari selisih antara total harga jual dan total harga beli pada setiap transaksi. Jumlah item terjual dihitung menggunakan fungsi *reduce()* untuk menjumlahkan kuantitas setiap barang yang termasuk dalam transaksi tersebut. Selanjutnya, sistem menghitung nilai rata-rata keuntungan per item dengan membagi total keuntungan dengan total item yang terjual, serta melakukan pengecekan nilai nol untuk menghindari kesalahan pembagian. Selain itu, sistem juga menghitung total keseluruhan item terjual melalui iterasi pada objek *orderItems*, yang digunakan sebagai dasar penyajian data pada grafik penjualan. Sistem dapat menampilkan laporan laba bulanan dengan memanfaatkan data transaksi yang tersimpan pada tabel Order, khususnya atribut totalSellPrice dan totalBuyPrice. Laba dihitung sebagai selisih antara total harga jual dan total harga beli pada setiap transaksi yang telah selesai diproses, kemudian diakumulasi dalam periode satu bulan untuk memperoleh total laba bulanan. Mekanisme ini konsisten dengan proses perhitungan keuntungan pada dashboard, di mana keuntungan diperoleh dari totalSellPrice- totalBuyPrice. Untuk kebutuhan laporan bulanan, proses perhitungan dilakukan dengan membatasi data transaksi pada rentang tanggal bulan yang dipilih dan hanya mengambil pesanan yang valid (tidak terhapus dan telah selesai), sehingga hasil laporan merepresentasikan keuntungan bulanan secara akurat.

```
const chartProduct = async (req, res) => {
    const { type } = req.query

    if (type === 'weekly') {
        return await chartProductWeek(req, res)
    } else if (type === 'monthly') {
        return await chartProductMonth(req, res)
    } else {
        return await chartProductAnnual(req, res)
    }
}
```

Gambar 4.5

Penjelasan : Fungsi *chartProduct* merupakan bagian dari *controller dashboard* yang bertugas menentukan jenis grafik penjualan produk yang akan disajikan kepada pengguna. Parameter *type* diambil dari query string untuk mengidentifikasi apakah pengguna meminta grafik mingguan, bulanan, atau tahunan. Berdasarkan nilai parameter tersebut, fungsi akan mendelegasikan proses pengolahan data kepada fungsi *chartProductWeek*, c*hartProductMonth*, atau *chartProductAnnual*. Pendekatan ini membuat sistem menjadi modular, karena fungsi *chartProduct* hanya berperan sebagai pengarah logika (*dispatcher*), sementara proses perhitungan statistik dilakukan secara terpisah pada fungsi khusus masing-

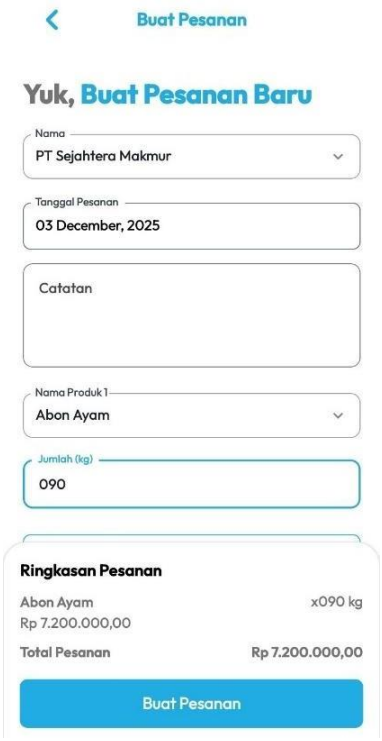
masing. Dengan demikian, struktur kode menjadi lebih terorganisir dan memudahkan pengembangan maupun pemeliharaan di masa mendatang. Produk paling laris pada sistem ditentukan berdasarkan akumulasi jumlah item terjual (quantity) pada tabel OrderItem. Sistem mengambil data OrderItem yang berasal dari pesanan (Order) dengan status selesai (finishedAt bernilai tidak null) dan tidak berada pada kondisi terhapus (isDeleted = false). Selanjutnya, data tersebut dikelompokkan berdasarkan productId dan dilakukan penjumlahan total kuantitas untuk setiap produk. Produk dengan nilai total kuantitas terbesar kemudian ditetapkan sebagai produk paling laris. Karena relasi Product dan OrderItem bersifat one-to-many, proses agregasi ini dapat menggambarkan tingkat permintaan produk secara akurat berdasarkan riwayat transaksi yang terjadi.

```
const chartIncome = async (req, res) => {
    const { type } = req.query

    if (type === 'weekly') {
        return await chartIncomeWeek(req, res)
    } else if (type === 'monthly') {
        return await chartIncomeMonth(req, res)
    } else {
        return await chartIncomeAnnual(req, res)
    }
}
```

Gambar 4.6

Penjelasan : Fungsi *chartIncome* berfungsi untuk menentukan jenis grafik pendapatan yang akan ditampilkan pada *dashboard*. Sistem membaca parameter *type* dari query string untuk mengidentifikasi apakah pengguna meminta data pendapatan mingguan, bulanan, atau tahunan. Berdasarkan nilai tersebut, *controller* akan mendelegasikan proses perhitungan kepada fungsi *chartIncomeWeek*, *chartIncomeMonth*, atau *chartIncomeAnnual*. Dengan pendekatan ini, fungsi *chartIncome* tidak melakukan pengolahan data secara langsung, tetapi berperan sebagai pengatur logika (*dispatcher*) sehingga struktur kode menjadi lebih modular, mudah dirawat, dan memenuhi prinsip *Single Responsibility* dalam pengembangan perangkat lunak.

b) Buat Pesanan



Gambar 4.7 Buat Pesanan

Gambar 4.7 menampilkan hasil implementasi halaman *Buat Pesanan* pada MVP aplikasi POS pabrik abon Merak Bewangi. Halaman ini berfungsi sebagai komponen utama dalam proses pencatatan pesanan yang sebelumnya dilakukan secara manual. Pada fitur Buat Pesanan, sistem mewajibkan pengguna memilih mitra terlebih dahulu melalui elemen *dropdown* sebelum pesanan dapat dibuat. Hal ini menunjukkan bahwa pesanan dicatat hanya atas nama mitra yang sudah tersedia/terdaftar di sistem, karena daftar pada dropdown berasal dari data mitra yang telah tersimpan. Dari sisi backend, aturan ini diperkuat melalui validasi pada fungsi createOrder. Sistem mewajibkan partnerId terisi sebagai syarat pembuatan pesanan, sehingga pesanan tidak dapat disimpan jika identitas mitra tidak dipilih. Validasi selanjutnya memastikan bahwa mitra tersebut benar-benar ada dan tidak dalam status terhapus (isDeleted). Apabila mitra tidak ditemukan atau sudah tidak aktif, sistem mengembalikan respons gagal dengan pesan "Mitra tidak ditemukan", sehingga order hanya bisa dibuat untuk mitra yang valid/terdaftar dan masih aktif. Kemudian menentukan tanggal pesanan serta menuliskan catatan tambahan apabila diperlukan.

Selanjutnya, pengguna dapat memilih produk yang dipesan beserta jumlah satuan yang diinginkan. tambah produk lain apabila pesanan lebih dari satu produk. Setiap perubahan pada jumlah atau produk akan otomatis diperbarui pada bagian *Ringkasan Pesanan* di

bawahnya. Ringkasan ini menampilkan detail pesanan, termasuk nama produk, kuantitas, harga per unit, dan total biaya yang harus dibayar.

Tombol “Buat Pesanan” ditempatkan di bagian bawah sebagai langkah final untuk menyimpan pesanan ke dalam database. Desain halaman ini menekankan kejelasan alur kerja, kemudahan input, serta minimnya risiko kesalahan, sehingga proses pencatatan pesanan menjadi lebih efisien dibandingkan metode manual.

Import

```
import express from 'express'
import prisma from '../db/prisma.js' const
router = express.Router()
import verification from '../middleware/verification.js'
```

Gambar 4.8

Kode Implementasi

```
     const createOrder = async (req, res) => {
        const { date, partnerId, note, orderItems } =
req.body
        if (!date || !partnerId || !orderItems) {
             return res.status(400).json({ status: 400,
message: 'Harap isi semua field' })
        }
        const parsedDate = Date.parse(date);
        if (isNaN(parsedDate)) {
             return res.status(400).json({ status: 400,
message: 'Tanggal tidak valid' });
        }

        if (typeof orderItems !== "object") {
             return res.status(400).json({ status: 400,
message: 'Format pesanan tidak valid' })
        }
        if (!Array.isArray(orderItems)) {
             return res.status(400).json({ status: 400,
message: 'Format pesanan tidak valid' })
        }
        if (orderItems.length === 0) {
             return res.status(400).json({ status: 400,
message: 'Pesanan tidak boleh kosong' })
        }
        const allowedKeys = ["productId", "quantity"];
        const productIds = []
        orderItems.forEach(item => {
             const keys = Object.keys(item);
                  const invalidKeys = keys.filter(key =>
!allowedKeys.includes(key));
             if (invalidKeys.length > 0) {
                  invalidKeys.forEach(key => delete
item[key]);
           }
              if (!item.productId || !item.quantity ||
isNaN(Number(item.quantity))) {
              return res.status(400).json({ status: 400,
message: 'Format pesanan tidak valid' })
           }
           productIds.push(item.productId)
      })
```

Gambar 4.9

Penjelasan: Bagian awal fungsi *createOrder* digunakan untuk melakukan validasi terhadap data pesanan yang masuk sebelum proses penyimpanan dilakukan. Validasi dimulai dengan memastikan bahwa parameter utama seperti tanggal, identitas mitra (*partnerId*), dan daftar item pesanan (*orderItems*) telah terisi. Sistem kemudian melakukan pemeriksaan format tanggal untuk memastikan bahwa nilai yang dikirimkan dapat diproses secara benar. Selanjutnya, struktur *orderItems* diverifikasi agar benar-benar berupa array dan tidak dalam keadaan kosong. Setiap elemen dalam *orderItems* kemudian diperiksa kesesuaiannya dengan struktur data yang diperbolehkan, yaitu hanya mengandung *productId* dan quantity. *Key* yang tidak relevan akan dihapus secara otomatis untuk mencegah manipulasi data. Setelah itu, setiap item diperiksa apakah memiliki nilai *productId* dan *quantity* yang valid, serta memastikan bahwa jumlah (*quantity*) merupakan angka. Validasi berlapis ini bertujuan untuk menjaga integritas data sehingga hanya data yang valid dan aman yang dapat diproses oleh sistem.

```
const [checkProducts, checkPartner] = await Promise.all([
                prisma.product.findMany({
                    where: {
                        AND: [
                            {
                                id: {
                                    in: productIds
                                }
                            },
                            {
                                isDeleted: false
                            }
                        ]
                    },
                }),
                prisma.partner.findUnique({
                    where: {
                        id: partnerId
                    }
                })
            ])
            if (!checkPartner || checkPartner.isDeleted) {
                return res.status(400).json({ status: 400,
message: 'Mitra tidak ditemukan' })
            }
           if (checkProducts.length !== productIds.length)
{
                return res.status(400).json({ status: 400,
message: 'Produk tidak ditemukan' })
            }
```

Gambar 4.10

Penjelasan : Pada tahap selanjutnya, sistem melakukan validasi terhadap data produk dan mitra sebelum pesanan disimpan. Proses ini dilakukan menggunakan *Promise.all* agar dua query database dapat dijalankan secara paralel sehingga mempercepat waktu respon. Query pertama mengambil daftar produk berdasarkan *productIds* yang dikirim dari klien, dengan ketentuan bahwa produk tersebut harus memiliki ID yang sesuai dan tidak berada dalam status terhapus (*isDeleted = false*). Query kedua memeriksa keberadaan mitra yang dipilih dengan mencocokkan nilai *partnerId*. Setelah mendapatkan hasil query, sistem melakukan pemeriksaan logis dengan memastikan bahwa mitra ditemukan dan masih aktif. Selanjutnya, sistem memvalidasi bahwa jumlah produk yang ditemukan dalam database harus sama dengan jumlah ID produk yang dikirim oleh pengguna. Jika terjadi ketidaksesuaian, sistem mengembalikan respons gagal dengan pesan "Produk tidak ditemukan". Validasi ini penting dilakukan untuk memastikan integritas data dan mencegah proses pembuatan pesanan menggunakan data yang tidak valid.

```
let totalBuyPriceOrder = 0 let
totalSellPriceOrder = 0

const dataOrderItems = orderItems.map(item => {
const buyPrice = checkProducts.find(product => product.id
=== item.productId).buyPrice || 0
const sellPrice = checkProducts.find(product => product.id
=== item.productId).sellPrice || 0
const totalBuyPrice = buyPrice * Number(item.quantity)
const totalSellPrice = sellPrice * Number(item.quantity)
const image = checkProducts.find(product => product.id ===
item.productId).image || null
const unit = checkProducts.find(product => product.id ===
item.productId).unit || "pcs"
const name = checkProducts.find(product => product.id ===
item.productId).name || "Produk Tidak Ditemukan"
                totalBuyPriceOrder += totalBuyPrice
                totalSellPriceOrder += totalSellPrice return
                {
                    productId: item.productId, quantity:
                    Number(item.quantity),
                    totalBuyPrice,
                    totalSellPrice, image,
                    unit,
                    name
                }
        })
```

Gambar 4.11

Penjelasan : Setelah seluruh produk dan mitra tervalidasi, sistem melakukan perhitungan harga pesanan berdasarkan setiap item yang dikirimkan. Pada tahap ini, fungsi *map()* digunakan untuk mengekstraksi informasi harga beli, harga jual, dan detail produk lainnya dari data yang telah diverifikasi. Sistem menghitung total harga beli (*totalBuyPrice*) dan total harga jual (*totalSellPrice*) dengan mengalikan harga satuan dengan jumlah pesanan. Nilai total tersebut kemudian dijumlahkan ke dalam variabel akumulasi *totalBuyPriceOrder* dan *totalSellPriceOrder*. Selain itu, informasi tambahan seperti nama produk, satuan, dan gambar juga diambil untuk melengkapi struktur data pesanan sebelum disimpan. Hasil akhir proses ini adalah suatu array *dataOrderItems* yang berisi objek-objek produk dengan informasi lengkap dan akurat, yang nantinya digunakan pada proses penyimpanan transaksi ke *database*.

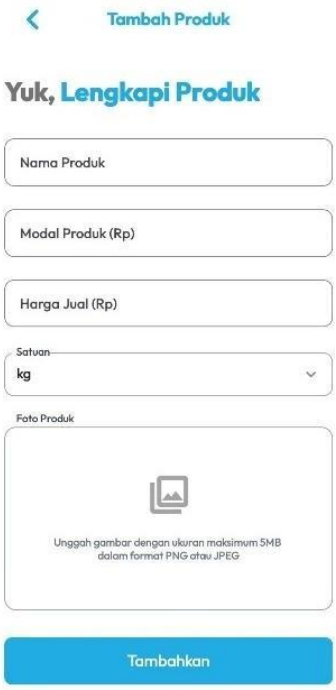
```
    const order = await prisma.order.create({
                data: {
                    date: new Date(date),
                    partnerId,
                    totalBuyPrice: totalBuyPriceOrder,
                    totalSellPrice: totalSellPriceOrder,
                    note,
                    orderItems: {
                        createMany: {
                            data: dataOrderItems
                        }
                    }
                },
                include: {
                    orderItems: {
                        include: {
                            product: true
                        }
                    }
                }
            })

            order.status = "Mendatang"

              return res.status(200).json({ status: 200,
message: 'Berhasil membuat pesanan!', data: order })
```

Gambar 4.12

Penjelasan : Setelah seluruh proses validasi dan perhitungan selesai, sistem menyimpan data pesanan ke dalam basis data menggunakan fungsi *prisma.order.create()*. Pada tahap ini, sistem mencatat informasi tanggal pesanan, identitas mitra, total harga beli, total harga jual, serta catatan tambahan. Selain itu, sistem juga melakukan penyimpanan massal terhadap

rincian item pesanan melalui fitur *createMany* pada relasi *orderItems*, di mana setiap item telah diperkaya dengan informasi harga total, kuantitas, dan atribut produk lainnya. Untuk memudahkan proses penampilan data di sisi klien, sistem turut menyertakan relasi *orderItems* beserta detail produk yang terkait menggunakan parameter *include*. Setelah pesanan berhasil dibuat, sistem menambahkan atribut status dengan nilai "Mendatang" pada objek pesanan sebagai penanda bahwa pesanan tersebut masih berada pada tahap awal. Terakhir, sistem mengirimkan respon HTTP dengan kode status 200 yang berisi pesan keberhasilan dan data pesanan yang baru dibuat, sehingga frontend dapat menampilkan informasi pesanan secara lengkap kepada pengguna.

c) Tambah Produk



Gambar 4.13 Tambah Produk

Gambar 4.13 menampilkan hasil implementasi halaman *Tambah Produk* pada aplikasi POS pabrik abon Merak Bewangi. Halaman ini berfungsi sebagai antarmuka bagi pemilik usaha untuk menambahkan data produk baru secara cepat, terstruktur, dan konsisten. Pada bagian atas tampilan, pengguna diberikan judul halaman *Tambah Produk* serta ajakan "Lengkapi Produk" untuk memperjelas tujuan langkah yang sedang dilakukan.

Form input terdiri dari beberapa komponen penting, yaitu **Nama Produk**, **Modal Produk**, **Harga Jual**, serta **Satuan** yang dapat dipilih melalui *dropdown* (misalnya kg, pcs, atau pack). Keempat elemen ini merupakan informasi dasar yang dibutuhkan untuk

menghitung profit dan menampilkan data produk dalam daftar inventori.

Di bawahnya, terdapat komponen unggah gambar produk yang memungkinkan pengguna menambahkan foto sebagai identitas visual produk. Sistem memberikan batasan ukuran maksimal 5MB dan format PNG/JPEG untuk memastikan kualitas gambar optimal dan kompatibel dengan sistem penyimpanan.

Pada bagian paling bawah, terdapat tombol **Tambahkan** yang digunakan untuk menyimpan data produk baru ke *database*. Desain halaman ini dibuat sederhana, bersih, dan mudah dipahami sehingga pemilik atau karyawan dapat menambahkan produk tanpa mengalami kebingungan. Implementasi halaman ini mendukung proses digitalisasi inventori yang sebelumnya dilakukan secara manual.

Import

```
    import express from 'express'
    import prisma from '../db/prisma.js'
    const router = express.Router()
    import                verification                from
'../middleware/verification.js'
```

Gambar 4.14

Kode Implementasi

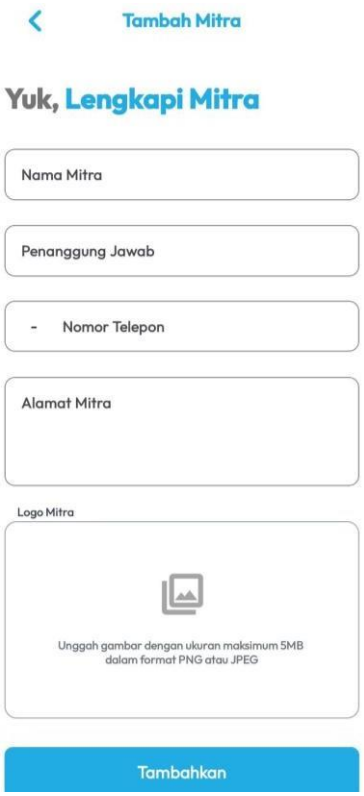
```
    const createProduct = async (req, res) => {
        const { name, buyPrice, sellPrice, unit, image } =
req.body
        if (!name || !buyPrice || !sellPrice || !unit) {
              return res.status(400).json({ status: 400,
message: 'Harap isi semua field' })
        }
                  if      (isNaN(Number(buyPrice))       ||
isNaN(Number(sellPrice))) {
              return res.status(400).json({ status: 400,
message: 'Harga harus berupa angka' })
        }
        try {
            const product = await prisma.product.create({
                data: {
                    name,
                    buyPrice: Number(buyPrice),
                    sellPrice: Number(sellPrice),
                    unit,
                    image: image || null,
                }
            })
              return res.status(200).json({ status: 200,
message: 'Berhasil menambahkan produk!', data: product })
```

```
        } catch (error) { console.log(error)
            return res.status(500).json({ status: 500,
message: 'Terjadi Kesalahan Sistem!' })
        }
    }
```

Gambar 4.15

Penjelasan : Fungsi *createProduct* merupakan *endpoint backend* yang digunakan untuk menambahkan produk baru ke dalam sistem. Pada tahap awal, sistem mengekstraksi data produk yang dikirimkan melalui *body request* dan melakukan validasi untuk memastikan bahwa seluruh *field* yang dibutuhkan, seperti nama produk, harga beli, harga jual, dan satuan produk, telah terisi. Selanjutnya, sistem melakukan validasi format harga untuk memastikan bahwa kedua nilai tersebut merupakan angka yang dapat diproses. Jika seluruh validasi terpenuhi, sistem menggunakan Prisma ORM untuk menyimpan data produk ke dalam basis data melalui perintah *prisma.product.create()*. Proses penyimpanan dilakukan di dalam blok *try* untuk mengantisipasi potensi kesalahan, dan apabila berhasil, sistem akan mengembalikan status 200 beserta data produk yang baru ditambahkan. Sementara itu, jika terjadi kesalahan pada saat penyimpanan, blok *catch* akan menangani *error* tersebut dan mengembalikan respons status 500 sebagai indikator bahwa terjadi kesalahan sistem. Dengan demikian, fungsi ini memastikan bahwa data produk yang masuk selalu valid, terstruktur, dan aman untuk disimpan.

d) Tambah Mitra



Gambar 4.16 Tambah Mitra

Gambar 4.16 menampilkan hasil implementasi halaman *Tambah Mitra* pada aplikasi POS pabrik abon Merak Bewangi. Halaman ini berfungsi untuk memfasilitasi proses pendaftaran mitra baru secara digital, menggantikan proses pencatatan manual yang sebelumnya dilakukan melalui buku tulis. Tampilan antarmuka dirancang sederhana dan mudah dipahami agar pengguna dapat mengisi data mitra tanpa mengalami kebingungan.

Formulir yang tersedia terdiri dari beberapa komponen utama, yaitu **Nama Mitra**, **Penanggung Jawab**, **Nomor Telepon**, dan **Alamat Mitra**. Informasi ini merupakan data dasar yang penting untuk mengelola relasi bisnis serta memastikan pencatatan pesanan dapat ditautkan dengan mitra yang tepat. Penempatan elemen input dibuat secara terurut dan konsisten untuk meningkatkan keterbacaan dan mengurangi potensi kesalahan pengguna saat mengisi data.

Bagian berikutnya menyediakan fitur unggah logo mitra, yang memungkinkan pengguna menambahkan identitas visual setiap mitra. Sistem memberikan batasan berupa ukuran maksimal file 5 MB serta format PNG atau JPEG untuk menjaga kualitas gambar sekaligus memastikan kompatibilitas dengan penyimpanan backend. Kehadiran logo mitra juga membantu meningkatkan kejelasan tampilan pada halaman daftar mitra maupun

halaman pesanan.

Pada bagian bawah, terdapat tombol **Tambahkan** yang berfungsi untuk menyimpan data mitra baru ke dalam sistem. Desain halaman ini mengedepankan kesederhanaan, kejelasan, dan kelancaran alur kerja sehingga proses penambahan mitra menjadi lebih efisien dan bebas dari pencatatan manual.

Import

```
    import express from 'express'
    import prisma from '../db/prisma.js' const
    router = express.Router()
import verification from '../middleware/verification.js'
```

Gambar 4.17

Kode Implementasi

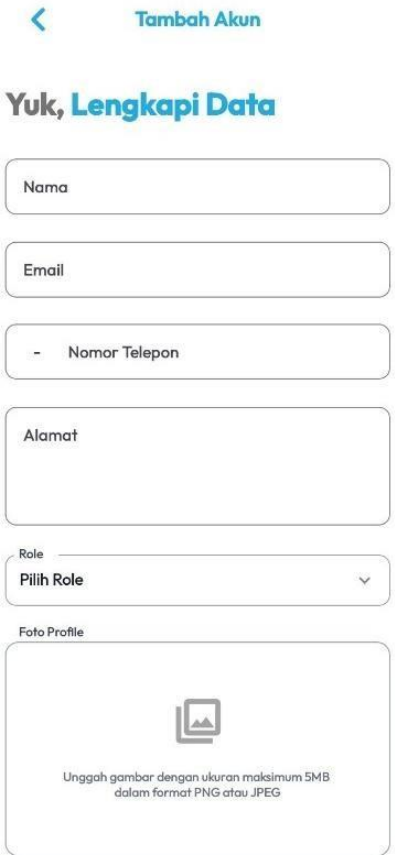
```
      const createPartner = async (req, res) => {
            const { name, pic, countryCode, phone, address,
  image } = req.body
            if (!name || !pic || !countryCode || !phone ||
  !address) {
                return res.status(400).json({ status: 400,
  message: 'Harap isi semua field' })
            }
            try {
                              const      checkName     = await
  prisma.partner.findFirst({
                 where: {
                    name: {
                          equals: name,
                          mode: "insensitive"
                    }
                 }
             })
            if (checkName && checkName.isDeleted === false)
  {
                return res.status(400).json({ status: 400,
  message: "Mitra sudah terdaftar" })
            }
            const partner = await prisma.partner.create({
                data: {
                    name,
                    pic,
                    countryCode,
                    phone,
                    address,
                    image
                }
            })
                return res.status(200).json({ status: 200,
  message: 'Berhasil menambahkan mitra!', data: partner })
        } catch (error) {
             console.log(error)
                 return res.status(500).json({ status: 500,
  message: 'Terjadi Kesalahan Sistem!' })
```

```
        }
    }
```

Gambar 4.18

Penjelasan : Fungsi *createPartner* digunakan untuk menambahkan data mitra ke dalam sistem melalui proses validasi dan penyimpanan yang terstruktur. Fungsi ini memulai proses dengan mengekstraksi informasi mitra dari *body request* dan memastikan bahwa seluruh *field* yang dibutuhkan, seperti nama mitra, penanggung jawab, kode negara, nomor telepon, dan alamat, telah terisi. Selanjutnya, sistem memeriksa kemungkinan duplikasi dengan mencari data mitra lain berdasarkan nama menggunakan metode pencocokan *case- insensitive*. Jika mitra dengan nama yang sama ditemukan dan masih aktif, proses dihentikan untuk mencegah duplikasi data. Apabila seluruh validasi terpenuhi, sistem menyimpan data mitra ke dalam basis data menggunakan Prisma ORM. Setelah penyimpanan berhasil, sistem mengembalikan respons dengan status 200 dan data mitra yang baru dibuat. Apabila terjadi kesalahan selama proses, blok *catch* menangani *error* tersebut dan mengembalikan respons 500 sebagai indikasi kesalahan sistem. Dengan demikian, fungsi *createPartner* memastikan bahwa data mitra yang masuk selalu konsisten, valid, dan bebas dari duplikasi.

e) Tambah Akun



Gambar 4.19 Tambah Akun

Gambar 4.19 menampilkan hasil implementasi halaman *Tambah Akun* pada aplikasi POS pabrik abon Merak Bewangi. Halaman ini digunakan oleh pemilik usaha untuk membuat akun baru bagi pegawai atau menambahkan akun owner tambahan sesuai kebutuhan operasional. Fitur ini merupakan bagian penting dari pengelolaan akses sistem karena setiap akun memiliki peran (*role*) yang menentukan batasan fitur yang dapat digunakan.

Formulir yang disediakan terdiri dari beberapa elemen utama, yaitu **Nama**, **Email**, **Nomor Telepon**, dan **Alamat**, yang berfungsi sebagai data identitas dasar pengguna. Di bawahnya terdapat kolom **Role**, yang ditampilkan dalam bentuk *dropdown* dan memungkinkan pemilik memilih peran pengguna, misalnya sebagai *Owner* atau *Employee*. Pengaturan peran ini diterapkan untuk menjaga kontrol akses dan memastikan bahwa hanya pengguna tertentu yang dapat mengakses fitur-fitur sensitif seperti manajemen produk dan akun.

Selain itu, halaman ini juga menyediakan komponen unggah foto profil dengan batas ukuran maksimum 5MB dalam format PNG atau JPEG. Penambahan foto profil berfungsi

untuk memberikan identitas visual pada setiap pengguna, sehingga mempermudah proses verifikasi dan interaksi di dalam aplikasi.

Tombol **Tambahkan** pada bagian bawah berfungsi untuk menyimpan seluruh data yang telah diinput ke dalam database. Struktur halaman ini dirancang dengan pendekatan sederhana dan intuitif, sehingga proses pembuatan akun dapat dilakukan dengan cepat, konsisten, dan bebas dari kesalahan input.

Import

```
    import express from 'express'
    import prisma from '../db/prisma.js'
    import { sendEmail } from '../utils/node-mailer/send-
email.js';
    import                verification                from
'../middleware/verification.js';
    import              randomCharacter               from
'../utils/randomCharacter.js';
    import bcrypt from 'bcryptjs'
    import validateEmail from '../utils/validateEmail.js';
const router = express.Router()
```

Gambar 4.20

Kode Implementasi

```
    const createUser = async (req, res) => {
        const { name, email, countryCode, phone, address,
role, image } = req.body
        if (!name || !email || !countryCode || !phone ||
!address || !role) {
            return res.status(400).json({ status: 400,
message: 'Harap isi semua field' })
        }

        if (role !== "Admin" && role !== "Employee") {
            return res.status(400).json({ status: 400,
message: 'Role tidak valid' })
        }

        if (!validateEmail(email)) {
            return res.status(400).json({ status: 400,
message: 'Email tidak valid' })
        }
        try {
                        const    checkEmail    =    await
prisma.user.findUnique({
                where: {
                    email
                }
            })
            if (checkEmail) {
                return res.status(400).json({ status: 400,
message: "Email sudah terdaftar" })
            }
            const password = randomCharacter(8)
```

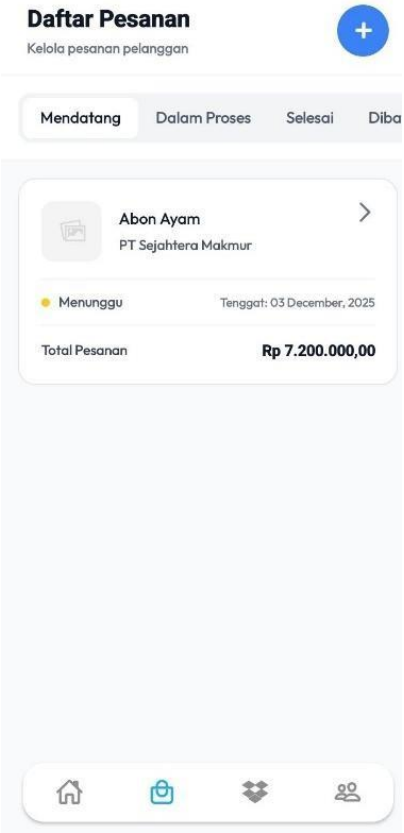
```
                    const hashedPassword = await
bcrypt.hash(password, 10)
                const send = await sendEmail(email,
"CREATE_ACCOUNT", name, password)
            if (send instanceof Error) {
                return res.status(500).json({ status: 500,
message: 'Gagal mengirim email' })
            }
            const user = await prisma.user.create({ data:
                {
                    address,
                    countryCode, email,
                    name,
                    password: hashedPassword, phone,
                    role,
                    image: image || null
                }
            })

                return res.status(200).json({ message:
"Berhasil membuat akun, harap cek email!", data: user })
        } catch (error) { console.log(error)
            return res.status(500).json({ status: 500,
message: 'Terjadi Kesalahan Sistem!' })
        }
}
```

Gambar 4.21

Penjelasan : Fungsi *createUser* digunakan untuk melakukan proses pembuatan akun pengguna baru pada sistem. Fungsi ini diawali dengan melakukan validasi terhadap kelengkapan data yang dikirim melalui *body request*, meliputi nama, email, kode negara, nomor telepon, alamat, serta peran (*role*) pengguna. Selain itu, sistem juga membatasi nilai role hanya pada dua jenis, yaitu "*Admin*" dan "*Employee*", guna menjaga konsistensi hak akses. Validasi format email dilakukan menggunakan fungsi *validateEmail*, kemudian sistem memeriksa apakah email tersebut sudah terdaftar di basis data dengan memanfaatkan Prisma ORM. Apabila email belum digunakan, sistem akan menghasilkan kata sandi secara acak melalui fungsi *randomCharacter*, kemudian mengenkripsinya menggunakan algoritma *bcrypt* sebelum disimpan. Selanjutnya, sistem mengirimkan informasi akun dan kata sandi tersebut ke email pengguna melalui fungsi *sendEmail*. Apabila pengiriman email berhasil, data pengguna disimpan ke dalam basis data menggunakan *prisma.user.create()*. Fungsi ini kemudian mengembalikan respons dengan status 200 dan pesan "Berhasil membuat akun,

harap cek email!". Di sisi lain, seluruh proses dibungkus dalam blok *try–catch* untuk menangani kemungkinan kesalahan sistem, di mana jika terjadi *error*, sistem akan mengembalikan status 500 sebagai indikasi kegagalan *internal*.

f) Daftar Pesanan



Gambar 4.22 Daftar Pesanan

Gambar 4.22 menampilkan hasil implementasi halaman *Daftar Pesanan* pada aplikasi POS pabrik abon Merak Bewangi. Halaman ini berfungsi sebagai pusat pengelolaan seluruh pesanan dari berbagai mitra, yang sebelumnya dicatat secara manual dalam buku tulis. Tampilan antarmuka dirancang agar pengguna dapat memantau status pesanan secara cepat, akurat, dan terstruktur.

Pada bagian atas halaman, pengguna disajikan judul *Daftar Pesanan* beserta tombol tambah ("+") yang memungkinkan pembuatan pesanan baru. Di bawahnya terdapat empat kategori status, yaitu **Mendatang**, **Dalam Proses**, **Selesai**, dan **Dibatalkan**. Pengelompokan status ini mempermudah pengguna dalam melacak progres pesanan berdasarkan tahapan operasional.

Tampilan pada gambar memperlihatkan tab **Mendatang**, yang berisi daftar pesanan yang telah dibuat tetapi belum diproses. Setiap item pesanan menampilkan informasi lengkap seperti nama produk, nama mitra, status pesanan (misalnya *Menunggu*), tenggat

waktu penyelesaian, serta total nilai pesanan. Informasi ini disajikan secara ringkas agar pemilik atau karyawan dapat mengambil tindakan dengan cepat, seperti memulai proses produksi atau melakukan *follow-up* kepada mitra.

Navigasi di bagian bawah halaman menyediakan akses cepat menuju halaman lain seperti dashboard, pembuatan pesanan, daftar produk, dan daftar akun. Dengan desain yang sederhana dan intuitif, halaman ini membantu meningkatkan efisiensi pemantauan pesanan serta meminimalkan risiko kesalahan dalam operasional sehari-hari.

Import

```
    import express from 'express'
    import prisma from '../db/prisma.js'
    const router = express.Router()
import verification from '../middleware/verification.js'
```

Gambar 4.23

Kode Implementasi

```
    const trackOrder = async (req, res) => {
        const { id } = req.params
        try {
            const check = await prisma.order.findUnique({
                where: {
                    id
                }
            })
            if (!check || check.isDeleted) {
                return res.status(404).json({ status: 404,
message: 'Pesanan tidak ditemukan' })
            }

            if (check.finishedAt) {
                return res.status(400).json({ status: 400,
message: 'Pesanan sudah selesai' })
            }

            if (!check.startedAt) {
                const order = await prisma.order.update({
                    where: {
                        id
                    },
                    data: {
                        startedAt: new Date()
                    }
                })
                return res.status(200).json({ status: 200,
message: 'Berhasil memulai pesanan!', data: order })
            } else {
                const order = await prisma.order.update({
                    where: {
                        id
```

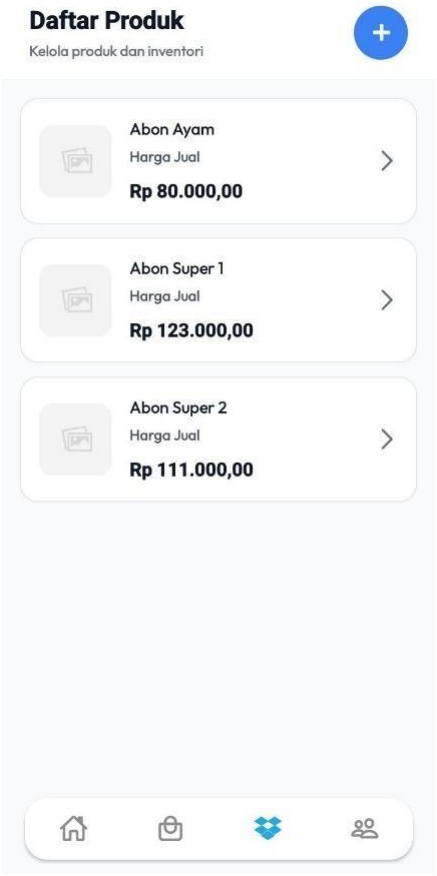
```
                    },
                    data: {
                        finishedAt: new Date()
                    }
                })
                return res.status(200).json({ status: 200,
message: 'Berhasil menyelesaikan pesanan!', data: order
})
            }
        } catch (error) {
            console.log(error)
              return res.status(500).json({ status: 500,
message: 'Terjadi Kesalahan Sistem!' })
        }
    }
```

Gambar 4.24

Penjelasan : Fungsi *trackOrder* berfungsi untuk mengatur status pengerjaan pesanan, yaitu memulai dan menyelesaikan suatu pesanan. Proses ini dimulai dengan mengambil ID pesanan dari parameter URL, kemudian sistem melakukan pencarian data pesanan menggunakan Prisma ORM. Jika pesanan tidak ditemukan atau telah dihapus, fungsi akan mengembalikan respons 404. Apabila pesanan sudah memiliki nilai *finishedAt*, maka sistem menolak permintaan karena pesanan tersebut telah selesai sebelumnya.

Untuk pesanan yang belum dimulai, sistem akan mengisi atribut *startedAt* dengan waktu saat ini sebagai penanda bahwa proses pengerjaan telah dimulai. Sebaliknya, apabila pesanan sudah dimulai tetapi belum selesai, sistem akan mengisi atribut *finishedAt* sebagai penanda bahwa pesanan telah diselesaikan. Setiap perubahan status disimpan kembali ke basis data dan dikembalikan ke klien bersama pesan keberhasilan. Seluruh proses juga dilindungi dengan blok *try–catch* untuk menangani kemungkinan kesalahan sistem.

g) Daftar Produk



Gambar 4.25 Daftar Produk

Gambar 4.25 menampilkan hasil implementasi halaman *Daftar Produk* pada aplikasi POS pabrik abon Merak Bewangi. Halaman ini berfungsi sebagai pusat pengelolaan seluruh produk yang tersedia dalam operasional pabrik, menggantikan pencatatan manual yang sebelumnya dilakukan pada buku atau catatan terpisah. Tampilan antarmuka dirancang sederhana, bersih, dan terorganisasi untuk memudahkan pengguna dalam meninjau serta memperbarui informasi produk.

Pada bagian atas halaman, terdapat judul *Daftar Produk* beserta tombol tambah ("+") yang digunakan untuk menambahkan produk baru ke dalam inventori. Pengguna dapat mengakses tombol ini ketika ingin memasukkan produk baru dengan informasi seperti harga modal, harga jual, satuan, dan foto produk.

Setiap kartu produk dalam daftar menampilkan informasi inti yaitu **nama produk**, kategori **harga jual**, serta ikon navigasi yang mengarahkan pengguna ke halaman detail atau pengeditan produk. Format penyajian daftar dibuat ringkas agar pemilik atau karyawan dapat melihat berbagai produk dengan cepat tanpa perlu membuka menu tambahan.

Di bagian bawah halaman terdapat bilah navigasi utama yang memudahkan perpindahan ke halaman lain seperti dashboard, pesanan, dan akun. Dengan implementasi halaman ini, proses pengelolaan produk menjadi jauh lebih efisien, rapi, dan mudah diakses, mendukung tujuan digitalisasi inventori di pabrik abon Merak Bewangi.

Import

```
import express from 'express'
import prisma from '../db/prisma.js'
const router = express.Router()
import verification from '../middleware/verification.js'
```

Gambar 4.26

Kode Implementasi

```
const getAllProducts = async (req, res) => {
    try {
        const products = await prisma.product.findMany({
            where: {
                isDeleted: false
            },
            orderBy: {
                name: "asc"
            }
        })
            return res.status(200).json({ status: 200,
message: "Success", data: products })

    } catch (error) {
        console.log(error)
            return res.status(500).json({ status: 500,
message: 'Terjadi Kesalahan Sistem!' })
    }
}
```

Gambar 4.27

Penjelasan : Fungsi *getAllProducts* merupakan bagian dari lapisan *backend* yang bertanggung jawab untuk menangani permintaan (*request*) pengambilan seluruh data produk dari basis data. Fungsi ini diimplementasikan menggunakan konsep *asynchronous programming* dengan tujuan memastikan proses pengambilan data berjalan secara efisien tanpa menghambat eksekusi sistem lainnya.

Pada awal fungsi, proses pengambilan data dilakukan menggunakan metode *findMany* dari Prisma ORM yang berfungsi untuk mengambil lebih dari satu data produk sekaligus. Pengambilan data difilter menggunakan kondisi *isDeleted: false*, yang bertujuan untuk memastikan bahwa hanya data produk yang masih aktif dan tidak terhapus secara logis (*soft delete*) yang ditampilkan kepada pengguna. Pendekatan ini digunakan untuk menjaga integritas data tanpa harus menghapus data secara permanen dari basis data.

Selanjutnya, data produk yang diperoleh diurutkan berdasarkan atribut name secara menaik (*ascending*). Proses pengurutan ini bertujuan untuk meningkatkan keterbacaan data dan memudahkan pengguna dalam menemukan produk pada antarmuka aplikasi, khususnya pada sistem *Point of Sales* (POS).

Apabila proses pengambilan data berhasil, sistem akan mengembalikan respons dengan kode status HTTP 200 (*OK*) disertai dengan pesan keberhasilan serta data produk dalam format JSON. Format respons ini digunakan agar mudah diproses oleh *frontend* maupun aplikasi klien lainnya.

Sebaliknya, apabila terjadi kesalahan selama proses pengambilan data, sistem akan menangkap kesalahan tersebut melalui blok *catch*. Kesalahan yang terjadi akan dicatat ke dalam *console* untuk keperluan *debugging*, kemudian sistem akan mengembalikan respons dengan kode status HTTP 500 (*Internal Server Error*) beserta pesan kesalahan umum. Mekanisme ini bertujuan untuk meningkatkan keandalan sistem serta memberikan penanganan kesalahan yang terstruktur.

Secara keseluruhan, fungsi *getAllProducts* dirancang untuk memastikan proses pengambilan data produk berjalan secara aman, terstruktur, dan efisien, serta mendukung kebutuhan fungsional sistem POS dalam menampilkan data produk yang valid dan terurut dengan baik.

b. Measure

Tahap *Measure* pada iterasi pertama merupakan fase evaluasi yang bertujuan untuk menilai sejauh mana MVP (*Minimum Viable Product*) yang telah dikembangkan

mampu memenuhi kebutuhan dan ekspektasi pengguna di lingkungan pabrik abon Merak Bewangi. Pada tahap ini, fokus utama adalah mengamati performa aplikasi ketika digunakan dalam kondisi operasional nyata serta mengukur tingkat keterpakaian, keandalan, dan kemudahan penggunaan setiap fitur inti. Melalui proses evaluasi ini, peneliti tidak hanya memverifikasi apakah fungsi dasar aplikasi berjalan sesuai rancangan, tetapi juga mengidentifikasi kendala, kekurangan, dan potensi peningkatan yang diperlukan sebelum memasuki iterasi selanjutnya.

1. Black Box Testing

Pengujian *Black Box* dilakukan untuk memastikan seluruh fitur inti pada aplikasi berfungsi sesuai dengan kebutuhan fungsional yang telah ditetapkan. Pengujian ini dilakukan oleh pemilik usaha dan dua karyawan sebagai pengguna langsung, sehingga proses evaluasi benar-benar mencerminkan kondisi operasional nyata di lapangan. Setiap skenario pengujian dirancang untuk mensimulasikan tindakan-tindakan umum yang dilakukan pengguna, mulai dari proses autentikasi, penambahan data baru, pengelolaan pesanan, hingga navigasi antarmuka aplikasi. Pada setiap skenario, pengguna memberikan input tertentu kemudian memverifikasi apakah output yang dihasilkan sistem telah sesuai dengan perilaku yang diharapkan. Pendekatan ini memungkinkan peneliti untuk menilai reliabilitas, konsistensi, serta ketahanan sistem dalam memproses berbagai jenis input, sekaligus memastikan bahwa setiap fitur berjalan stabil tanpa menimbulkan *error* selama proses pengujian dilakukan. Hasil pengujian *Black Box* dapat dilihat pada Tabel 4.1

Tabel 4.1 Hasil Black Box Testing Iterasi 1

No	Fitur yang Diuji	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian
1	Login Pengguna	Memasukkan email dan password benar	Sistem berhasil memverifikasi dan masuk ke *dashboard*	Berhasil
2	Login Pengguna	Memasukkan email dan password salah	Sistem menampilkan pesan *error* "Password salah"	Berhasil
3	Login Pengguna	Mengosongkan salah satu field	Sistem menampilkan pesan *error* "Harap isi semua field"	Berhasil
4	Tampilkan Dashboard	Sistem memuat data penjualan dan grafik penjualan	Sistem menampilkan informasi sesuai *database*	Berhasil
5	Tambah Produk	Mengisi semua form dan tekan tombol Tambahkan	Data produk tersimpan dan sistem menampilkan di daftar produk	Berhasil

No	Fitur yang Diuji	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian
6	Tambah Produk	Mengosongkan salah satu field	Sistem menampilkan pesan *error* "Harap isi semua data"	Berhasil
7	Edit Produk	Memperbarui data produk dan tekan tombol Perbarui	Data produk baru tersimpan dan sistem menampilkan data produk terbaru	Berhasil
8	Tambah Mitra	Mengisi semua form dan tekan tombol Tambahkan	Data mitra tersimpan dan sistem menampilkan di daftar Mitra	Berhasil
9	Tambah Mitra	Mengosongkan salah satu field	Sistem menampilkan pesan *error* "Harap isi semua data"	Berhasil
10	Edit Mitra	Memperbarui data mitra dan tekan tombol Perbarui	Data mitra baru tersimpan dan sistem menampilkan data mitra terbaru	Berhasil
11	Tambah Akun	Mengisi semua data dan memilih role	Akun baru tersimpan dan dapat login	Berhasil
12	Tambah Akun	Mengosongkan salah satu field	Sistem menampilkan pesan *error* "Harap isi semua data"	Berhasil
13	Edit Akun	Memperbarui data akun dan tekan tombol Perbarui	Data akun baru tersimpan dan sistem menampilkan data mitra terbaru	Berhasil
14	Buat Pesanan	input mitra, tanggal pesanan, produk, kuantitas dan menekan tombol Buat Pesanan	Pesanan tersimpan dalam status "Mendatang"	Berhasil
15	Buat Pesanan	Mengosongkan salah satu field input	Sistem menampilkan pesan *error* "Harap isi semua data"	Berhasil
16	Ubah Status Pesanan	Menekan tombol "Mulai"	Status berubah sesuai aksi dan sistem menampilkan pesanan di daftar "Dalam Proses"	Berhasil
17	Ubah Status Pesanan	Menekan tombol "Selesai"	Status berubah sesuai aksi dan sistem menampilkan pesanan di daftar "Selesai"	Berhasil
18	Batalkan Pesanan	Menekan tombol "Batalkan"	Status berubah sesuai aksi dan sistem menampilkan pesanan di daftar "Dibatalkan"	Berhasil
19	Hapus Produk	Pengguna menekan tombol Hapus	Produk terhapus dari daftar produk	Berhasil
20	Hapus Mitra	Pengguna menekan tombol	Mitra terhapus dari daftar	Berhasil

No	Fitur yang Diuji	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian
		Hapus	mitra	
21	Hapus akun	Pengguna menekan tombol Hapus	Akun terhapus dari daftar akun	Berhasil

Hasil pengujian menunjukkan bahwa seluruh fitur inti berjalan sesuai kebutuhan fungsional, menandakan bahwa sistem mampu menjalankan seluruh proses operasional yang dirancang tanpa hambatan berarti. Semua skenario uji yang melibatkan aktivitas penting seperti autentikasi pengguna, manajemen data produk dan mitra, proses pembuatan dan pembaruan pesanan, hingga penyajian informasi pada dashboard berhasil dieksekusi dengan baik. Konsistensi performa sistem ini memperlihatkan bahwa aplikasi telah memiliki fondasi teknis yang stabil dan dapat diandalkan untuk mendukung kegiatan bisnis sehari-hari.

Selain memastikan bahwa fungsi inti bekerja sebagaimana mestinya, pengujian juga memberikan gambaran mengenai tingkat responsivitas aplikasi serta kestabilan alur kerja dalam berbagai kondisi input. Seluruh fitur menunjukkan perilaku yang sesuai ekspektasi, tanpa terjadi kendala besar maupun kesalahan sistem yang dapat mengganggu proses operasional. Hasil ini memperkuat bahwa MVP telah memenuhi standar fungsional minimum sebagaimana yang ditentukan pada tahap analisis kebutuhan.

2. Wawancara

Tahap wawancara pada iterasi pertama dilakukan setelah pengguna mengikuti demo aplikasi yang telah dikembangkan pada tahap Build iterasi pertama. Wawancara ini bertujuan untuk memperoleh umpan balik awal dari pengguna terkait pengalaman penggunaan aplikasi dalam mendukung aktivitas operasional. Proses wawancara dilakukan secara langsung dengan pemilik sebagai pengguna utama sistem setelah mencoba seluruh fitur yang tersedia pada aplikasi.

Berdasarkan hasil wawancara yang dilakukan, pengguna menyampaikan bahwa secara umum aplikasi telah berfungsi dengan baik dan mampu membantu proses pencatatan serta pengelolaan data. Namun demikian, pengguna juga memberikan masukan terkait kecepatan perpindahan antar halaman aplikasi yang masih dirasakan kurang optimal. Kondisi ini menyebabkan alur penggunaan aplikasi belum sepenuhnya terasa lancar, terutama ketika pengguna berpindah dari satu halaman ke halaman lainnya dalam proses operasional.

Masukan yang diperoleh dari tahap wawancara pada iterasi pertama ini menjadi temuan penting dalam proses pengembangan sistem. Umpan balik tersebut menunjukkan bahwa meskipun fungsionalitas aplikasi telah berjalan dengan baik, masih diperlukan perbaikan pada aspek performa untuk meningkatkan kenyamanan penggunaan. Temuan ini selanjutnya dijadikan

sebagai dasar pembelajaran pada tahap Learn iterasi pertama dan menjadi acuan dalam perencanaan perbaikan pada iterasi berikutnya.

c. Learn

Tahap *Learn* pada iterasi pertama merupakan proses refleksi yang bertujuan mengubah seluruh temuan pada fase *Measure* menjadi pengetahuan yang dapat ditindaklanjuti dalam pengembangan berikutnya. Pada fase ini, peneliti menganalisis umpan balik pengguna, hasil pengujian *Black Box* dan *feedback* langsung dari penggunauntuk menarik kesimpulan mengenai efektivitas aplikasi dan area yang masih memerlukan peningkatan. Pembelajaran yang diperoleh menjadi landasan utama dalam merumuskan prioritas fitur dan perbaikan yang akan dilakukan pada iterasi kedua.

Secara umum, pembelajaran pada iterasi pertama menunjukkan bahwa aplikasi telah mampu memenuhi kebutuhan dasar operasional pabrik abon Merak Bewangi, terutama dalam hal pencatatan pesanan, pengelolaan data produk dan mitra, serta penyediaan ringkasan penjualan melalui *dashboard*. Hal ini dibuktikan dengan hasil pengujian *Black Box* yang menyatakan bahwa seluruh fitur inti berjalan sesuai kebutuhan fungsional tanpa menimbulkan kendala teknis yang signifikan. Dengan demikian, dari perspektif fungsionalitas, MVP sudah berada pada jalur yang tepat.

Namun demikian, dari hasil diskusi dan pengamatan penggunaan aplikasi, ditemukan beberapa masukan yang berkaitan dengan kenyamanan pengguna saat menunggu data ditampilkan di layar. Pengguna menyampaikan bahwa pada beberapa kondisi, aplikasi terasa sedikit lambat ketika berpindah halaman atau saat membuka daftar data, seperti daftar pesanan atau data produk. Masukan ini disampaikan dengan ungkapan sederhana, seperti *"kadang harus menunggu sebentar sebelum datanya muncul"* atau *"kalau membuka daftar pesanan, rasanya agak lama dibanding halaman lain"*. Pengguna tidak mengaitkan kondisi tersebut dengan aspek teknis tertentu, melainkan lebih pada pengalaman menunggu yang dirasakan saat menggunakan aplikasi.

Selain itu, pengguna juga menyampaikan harapan agar tampilan data dapat muncul lebih cepat sehingga alur kerja menjadi lebih lancar, terutama ketika aktivitas sedang padat dan pencatatan pesanan dilakukan secara berulang. Menurut pengguna, apabila data dapat ditampilkan dengan lebih responsif, maka proses kerja akan terasa lebih ringan dan tidak terputus oleh waktu tunggu, meskipun hanya dalam hitungan beberapa detik.

Masukan tersebut menunjukkan bahwa meskipun MVP telah memenuhi kebutuhan

fungsional utama, aspek kenyamanan penggunaan masih dapat ditingkatkan, khususnya dalam hal kecepatan aplikasi saat menampilkan data. Temuan ini menjadi pembelajaran penting pada iterasi pertama, karena menegaskan bahwa persepsi pengguna terhadap kualitas aplikasi tidak hanya ditentukan oleh kelengkapan fitur, tetapi juga oleh kelancaran dan kecepatan respons aplikasi dalam penggunaan sehari-hari.

Berdasarkan hasil pembelajaran ini, peneliti menyimpulkan bahwa perbaikan pada iterasi berikutnya perlu difokuskan pada upaya meningkatkan kenyamanan pengguna dengan meminimalkan waktu tunggu saat aplikasi menampilkan data. Dengan demikian, pengembangan pada iterasi kedua diarahkan tidak pada penambahan fitur yang kompleks, melainkan pada penyempurnaan performa aplikasi agar pengalaman penggunaan menjadi lebih cepat, lebih lancar, dan sesuai dengan ekspektasi pengguna di lapangan.

4.1.2 Iterasi 2

Iterasi kedua dilakukan sebagai tindak lanjut dari hasil pembelajaran pada iterasi pertama yang telah mengungkap beberapa aspek penggunaan aplikasi yang masih dapat ditingkatkan. Fokus utama pada iterasi ini adalah melakukan penyempurnaan terhadap pengalaman penggunaan aplikasi berdasarkan masukan langsung dari pengguna, khususnya yang berkaitan dengan kenyamanan, kelancaran alur kerja, serta respons aplikasi saat menampilkan data pada berbagai halaman. Masukan tersebut menjadi dasar penting dalam menentukan arah pengembangan agar aplikasi dapat digunakan secara lebih optimal dalam aktivitas operasional sehari-hari.

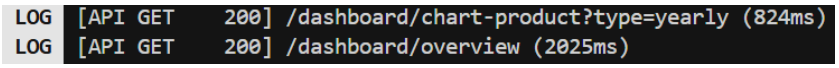
Pengembangan pada iterasi kedua tetap mengikuti siklus Build–Measure–Learn sebagaimana pada iterasi sebelumnya, namun dengan ruang lingkup yang lebih terarah dan spesifik. Peneliti tidak lagi berfokus pada penambahan fitur baru, melainkan pada peningkatan kualitas penggunaan aplikasi yang telah ada, sehingga setiap perbaikan yang dilakukan benar-benar ditujukan untuk menjawab kebutuhan dan harapan pengguna berdasarkan pengalaman nyata mereka dalam menggunakan MVP.

a Build

Pada tahap *Build* iterasi kedua, pengembangan aplikasi difokuskan pada upaya meningkatkan performa dan kenyamanan penggunaan tanpa menambahkan fitur baru yang kompleks. Fokus ini dipilih agar sistem yang telah ada dapat dioptimalkan secara maksimal sebelum dilakukan pengembangan lanjutan. Perbaikan dilakukan berdasarkan keluhan pengguna pada iterasi pertama yang merasa perlu menunggu saat membuka halaman tertentu atau melihat daftar data, terutama ketika aplikasi digunakan dalam kondisi

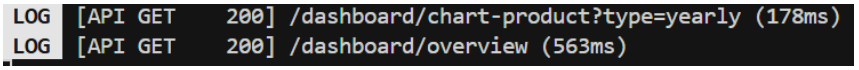
operasional yang padat. Kondisi tersebut menjadi perhatian utama karena waktu tunggu, meskipun singkat, dapat mengganggu kelancaran alur kerja dan menurunkan kenyamanan pengguna dalam penggunaan aplikasi sehari-hari. Berikut adalah perbandingan kecepatan dalam pengambilan dan pengiriman data pada setiap fitur:

1. Dashboard



Gambar 4.28 *Fetch* Data *dashboard* Iterasi 1

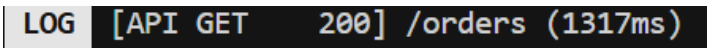
Pada Gambar 4.28 menunjukkan hasil pengujian pemanggilan data pada iterasi pertama yang ditampilkan melalui log sistem. Berdasarkan tangkapan layar tersebut, terlihat bahwa waktu yang dibutuhkan aplikasi untuk menampilkan data *dashboard*, baik grafik produk tahunan maupun ringkasan *overview*, masih relatif lama. Kondisi ini menyebabkan pengguna harus menunggu sebelum seluruh informasi tampil secara lengkap di layar. Temuan ini memperkuat masukan pengguna pada iterasi pertama yang menyatakan bahwa aplikasi terasa kurang responsif saat membuka halaman tertentu, sehingga menjadi salah satu dasar dilakukan penyempurnaan pada iterasi berikutnya.



Gambar 4.29 *Fetch* Data *dashboard* Iterasi 2

Pada Gambar 4.29 menunjukkan hasil pengujian kecepatan penampilan data setelah dilakukan penyempurnaan pada iterasi kedua. Berdasarkan log pemanggilan API, terlihat bahwa waktu yang dibutuhkan aplikasi untuk menampilkan grafik produk tahunan dan data ringkasan dashboard menjadi jauh lebih singkat dibandingkan dengan iterasi pertama. Perubahan ini berdampak pada pengalaman penggunaan yang lebih lancar, di mana pengguna tidak lagi merasakan waktu tunggu yang mengganggu saat membuka halaman dashboard. Hasil ini menunjukkan bahwa perbaikan yang dilakukan pada iterasi kedua berhasil meningkatkan respons aplikasi dalam menampilkan data.

2. Daftar Pesanan



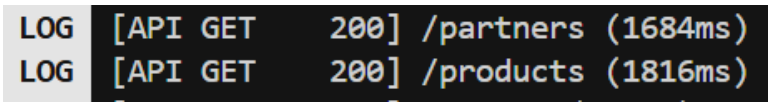
Gambar 4.30 *Fetch Data* Daftar Pesanan Iterasi 1

Pada Gambar 4.30 menampilkan hasil pengujian pemanggilan data pada halaman daftar pesanan pada iterasi pertama. Berdasarkan tangkapan layar log sistem, terlihat bahwa waktu yang dibutuhkan aplikasi untuk menampilkan data pesanan masih relatif lama. Kondisi ini menyebabkan pengguna harus menunggu sebelum seluruh daftar pesanan dapat ditampilkan secara lengkap. Temuan ini sejalan dengan masukan pengguna yang menyatakan bahwa pada iterasi pertama, proses membuka daftar pesanan terasa kurang lancar dan membutuhkan waktu tunggu, sehingga menjadi salah satu aspek yang perlu diperhatikan dalam pengembangan pada iterasi berikutnya.

Gambar 4.31 *Fetch Data* Daftar Pesanan Iterasi 2

Pada Gambar 4.31 menunjukkan hasil pengujian pemanggilan data pada halaman daftar pesanan setelah dilakukan penyempurnaan pada iterasi kedua. Berdasarkan log sistem yang ditampilkan, waktu yang dibutuhkan aplikasi untuk menampilkan data pesanan menjadi jauh lebih singkat dibandingkan dengan iterasi pertama. Perubahan ini membuat daftar pesanan dapat muncul dengan lebih cepat dan responsif, sehingga pengguna tidak lagi merasakan waktu tunggu yang mengganggu saat membuka halaman tersebut. Hasil perbandingan ini menunjukkan bahwa perbaikan yang dilakukan pada iterasi kedua berhasil meningkatkan kelancaran aplikasi dalam menampilkan data daftar pesanan dan mendukung aktivitas operasional pengguna secara lebih efektif.

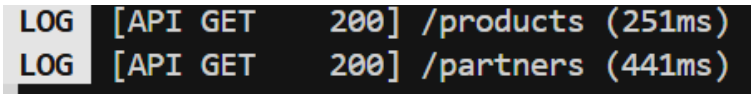
3. Buat Pesanan



Gambar 4.32 *Fetch Data* Buat Pesanan Iterasi 1

Pada Gambar 4.32 menampilkan waktu respons sistem ketika pengguna membuka halaman buat pesanan pada iterasi pertama. Berdasarkan tangkapan layar log pemanggilan API, terlihat bahwa aplikasi membutuhkan waktu yang relatif lama untuk menampilkan data pendukung, seperti daftar mitra dan daftar produk. Kondisi ini menyebabkan proses pembuatan pesanan belum berjalan secara optimal karena pengguna harus menunggu hingga seluruh data tersedia di layar. Temuan ini menunjukkan bahwa pada iterasi pertama, halaman buat pesanan masih memerlukan penyempurnaan agar alur kerja pengguna dapat berjalan lebih lancar dan

efisien.



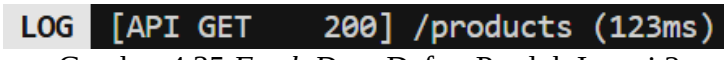
Gambar 4.33 *Fetch Data* Buat Pesanan Iterasi 2

Pada Gambar 4.33 menunjukkan hasil pengujian waktu respons sistem pada halaman buat pesanan setelah dilakukan penyempurnaan pada iterasi kedua. Berdasarkan log pemanggilan API yang ditampilkan, waktu yang dibutuhkan aplikasi untuk menampilkan data mitra dan produk menjadi jauh lebih singkat dibandingkan dengan iterasi pertama. Perbaikan ini membuat halaman buat pesanan dapat diakses dengan lebih cepat dan responsif, sehingga pengguna dapat langsung melanjutkan proses pembuatan pesanan tanpa harus menunggu lama. Hasil perbandingan ini menunjukkan bahwa penyempurnaan pada iterasi kedua berhasil meningkatkan kelancaran alur kerja pengguna pada fitur buat pesanan.

4. Daftar Produk

Gambar 4.34 *Fetch Data* Daftar Produk Iterasi 1

Pada Gambar 4.34 menampilkan hasil pengujian waktu respons sistem ketika pengguna membuka halaman daftar produk pada iterasi pertama. Berdasarkan tangkapan layar log pemanggilan API, terlihat bahwa aplikasi membutuhkan waktu tertentu untuk menampilkan data produk secara keseluruhan. Meskipun data dapat ditampilkan dengan baik, pengguna masih harus menunggu beberapa saat sebelum seluruh informasi produk muncul di layar. Kondisi ini menunjukkan bahwa pada iterasi pertama, halaman daftar produk masih memiliki potensi untuk ditingkatkan agar proses penampilan data dapat berlangsung lebih cepat dan mendukung kenyamanan pengguna dalam mengakses informasi produk.

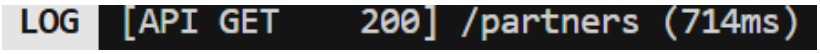


Gambar 4.35 *Fetch Data* Daftar Produk Iterasi 2

Pada Gambar 4.35 menunjukkan hasil pengujian waktu respons sistem pada halaman daftar produk setelah dilakukan penyempurnaan pada iterasi kedua. Berdasarkan log pemanggilan API yang ditampilkan, waktu yang dibutuhkan aplikasi untuk menampilkan data produk menjadi

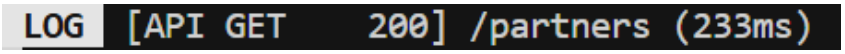
lebih singkat dibandingkan dengan iterasi pertama. Perbaikan ini membuat daftar produk dapat ditampilkan dengan lebih cepat dan responsif, sehingga pengguna tidak lagi merasakan waktu tunggu yang mengganggu saat mengakses informasi produk. Hasil perbandingan ini menunjukkan bahwa penyempurnaan pada iterasi kedua berhasil meningkatkan kenyamanan dan efisiensi penggunaan aplikasi pada fitur daftar produk.

5. Daftar Mitra



Gambar 4.36 *Fetch Data* Daftar Mitra Iterasi 1

Pada Gambar 4.36 menampilkan hasil pengujian waktu respons sistem ketika pengguna membuka halaman daftar mitra pada iterasi pertama. Berdasarkan tangkapan layar log pemanggilan API, terlihat bahwa aplikasi membutuhkan waktu tertentu untuk menampilkan data mitra secara keseluruhan. Kondisi ini menyebabkan pengguna harus menunggu sebelum seluruh informasi mitra dapat ditampilkan dengan lengkap. Temuan ini menunjukkan bahwa pada iterasi pertama, halaman daftar mitra masih memiliki potensi untuk ditingkatkan agar proses penampilan data dapat berlangsung lebih cepat dan mendukung kelancaran aktivitas pengguna dalam mengelola data mitra.



Gambar 4.37 *Fetch Data* Daftar Mitra Iterasi 2

Pada Gambar 4.37 menunjukkan hasil pengujian waktu respons sistem pada halaman daftar mitra setelah dilakukan penyempurnaan pada iterasi kedua. Berdasarkan log pemanggilan API yang ditampilkan, waktu yang dibutuhkan aplikasi untuk menampilkan data mitra menjadi lebih singkat dibandingkan dengan iterasi pertama. Perbaikan ini membuat data mitra dapat ditampilkan dengan lebih cepat dan responsif, sehingga pengguna tidak lagi merasakan waktu tunggu yang mengganggu saat mengakses halaman tersebut. Hasil perbandingan ini menunjukkan bahwa penyempurnaan pada iterasi kedua berhasil meningkatkan efisiensi dan kenyamanan pengguna dalam mengelola data mitra.

6. Daftar Akun

Gambar 4.38 *Fetch Data* Daftar Akun Iterasi 1

Pada Gambar 4.38 menampilkan hasil pengujian waktu respons sistem ketika pengguna membuka halaman daftar akun pada iterasi pertama. Berdasarkan tangkapan layar log pemanggilan API, terlihat bahwa aplikasi membutuhkan waktu tertentu untuk menampilkan data akun secara keseluruhan. Kondisi ini menyebabkan pengguna harus menunggu sebelum seluruh informasi akun dapat ditampilkan dengan lengkap. Temuan ini menunjukkan bahwa pada iterasi pertama, halaman daftar akun masih memiliki potensi untuk ditingkatkan agar proses penampilan data dapat berlangsung lebih cepat dan mendukung kenyamanan pengguna dalam mengelola data akun.

Gambar 4.39 Fetch Data Daftar Akun Iterasi 2

Pada Gambar 4.39 menunjukkan hasil pengujian waktu respons sistem pada halaman daftar akun setelah dilakukan penyempurnaan pada iterasi kedua. Berdasarkan log pemanggilan API yang ditampilkan, waktu yang dibutuhkan aplikasi untuk menampilkan data akun menjadi jauh lebih singkat dibandingkan dengan iterasi pertama. Perbaikan ini membuat daftar akun dapat ditampilkan dengan lebih cepat dan responsif, sehingga pengguna tidak lagi merasakan waktu tunggu yang mengganggu saat mengakses halaman tersebut. Hasil perbandingan ini menunjukkan bahwa penyempurnaan pada iterasi kedua berhasil meningkatkan efisiensi dan kenyamanan pengguna dalam mengelola data akun.

b Measure

Tahap *Measure* pada iterasi kedua dilakukan untuk mengukur efektivitas perbaikan sistem yang telah diterapkan berdasarkan hasil pembelajaran pada iterasi sebelumnya. Pada iterasi pertama, meskipun hasil pengujian fungsional menunjukkan bahwa aplikasi telah berjalan dengan baik, pengguna memberikan masukan terkait kecepatan perpindahan halaman yang masih dirasakan kurang optimal. Oleh karena itu, pada iterasi kedua fokus pengukuran diarahkan untuk memastikan bahwa perbaikan pada aspek performa, khususnya kecepatan perpindahan halaman, telah memberikan dampak positif terhadap pengalaman pengguna.

Pengukuran pada tahap ini dilakukan melalui beberapa metode, yaitu *Black Box Testing*, demo aplikasi, dan wawancara langsung dengan pengguna. *Black Box Testing* digunakan untuk memastikan bahwa seluruh fungsi aplikasi tetap berjalan dengan baik setelah dilakukan perbaikan. Selanjutnya, aplikasi didemonstrasikan kembali kepada

pemilik untuk memperlihatkan perubahan yang telah diterapkan, sekaligus dilakukan wawancara guna memperoleh umpan balik terkait kenyamanan dan kecepatan penggunaan aplikasi. Setelah pengguna menyatakan kepuasan terhadap hasil perbaikan yang dilakukan, tahap *Measure* pada iterasi kedua dilengkapi dengan memberikan kuesioner *System Usability Scale* (SUS) untuk memperoleh penilaian tingkat *usability* aplikasi secara lebih terukur.

1. Black Box Testing

Black Box Testing pada iterasi kedua dilakukan untuk memastikan bahwa seluruh fungsionalitas aplikasi tetap berjalan dengan baik setelah dilakukan perbaikan pada tahap *Build* iterasi kedua. Perbaikan yang dilakukan berfokus pada peningkatan kecepatan perpindahan halaman sebagai respons terhadap umpan balik pengguna pada iterasi pertama. Oleh karena itu, pengujian pada iterasi kedua ini bertujuan untuk memverifikasi bahwa perubahan yang diterapkan tidak menimbulkan kesalahan baru pada fungsi sistem yang telah berjalan sebelumnya.

Black Box Testing dilakukan dengan cara menguji setiap fitur aplikasi berdasarkan masukan dan keluaran yang dihasilkan, tanpa memperhatikan struktur internal sistem. Pengujian ini mencakup fungsi-fungsi utama yang digunakan oleh pengguna dalam proses operasional, seperti pengelolaan data, proses transaksi, serta navigasi antar halaman. Hasil dari pengujian ini digunakan sebagai dasar untuk menilai kesiapan aplikasi sebelum dilakukan pengujian lebih lanjut melalui demo dan wawancara dengan pengguna.

Melalui pelaksanaan *Black Box Testing* pada iterasi kedua, diharapkan aplikasi tidak hanya mengalami peningkatan dari sisi performa, tetapi juga tetap mempertahankan kestabilan dan keandalan fungsi sistem. Dengan demikian, hasil pengujian ini menjadi bagian penting dalam tahap Measure iterasi kedua sebelum dilakukan penilaian usability menggunakan kuesioner *System Usability Scale* (SUS). Hasil Pengujian *Black Box Testing* dapat dilihat pada Tabel 4.2

Tabel 4.2 Hasil Black Box Testing Iterasi 2

No	Fitur yang Diuji	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian
1	Login Pengguna	Memasukkan email dan password benar	Sistem berhasil memverifikasi dan masuk ke *dashboard*	Berhasil
2	Login Pengguna	Memasukkan email dan password salah	Sistem menampilkan pesan *error* “Password salah”	Berhasil
3	Login Pengguna	Mengosongkan salah satu field	Sistem menampilkan pesan *error* “Harap isi semua field”	Berhasil

No	Fitur yang Diuji	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian
4	Tampilkan Dashboard	Sistem memuat data penjualan dan grafik penjualan	Sistem menampilkan informasi sesuai *database*	Berhasil
5	Tambah Produk	Mengisi semua form dan tekan tombol Tambahkan	Data produk tersimpan dan sistem menampilkan di daftar produk	Berhasil
6	Tambah Produk	Mengosongkan salah satu field	Sistem menampilkan pesan *error* "Harap isi semua data"	Berhasil
7	Edit Produk	Memperbarui data produk dan tekan tombol Perbarui	Data produk baru tersimpan dan sistem menampilkan data produk terbaru	Berhasil
8	Tambah Mitra	Mengisi semua form dan tekan tombol Tambahkan	Data mitra tersimpan dan sistem menampilkan di daftar Mitra	Berhasil
9	Tambah Mitra	Mengosongkan salah satu field	Sistem menampilkan pesan *error* "Harap isi semua data"	Berhasil
10	Edit Mitra	Memperbarui data mitra dan tekan tombol Perbarui	Data mitra baru tersimpan dan sistem menampilkan data mitra terbaru	Berhasil
11	Tambah Akun	Mengisi semua data dan memilih role	Akun baru tersimpan dan dapat login	Berhasil
12	Tambah Akun	Mengosongkan salah satu field	Sistem menampilkan pesan *error* "Harap isi semua data"	Berhasil
13	Edit Akun	Memperbarui data akun dan tekan tombol Perbarui	Data akun baru tersimpan dan sistem menampilkan data mitra terbaru	Berhasil
14	Buat Pesanan	input mitra, tanggal pesanan, produk, kuantitas dan menekan tombol Buat Pesanan	Pesanan tersimpan dalam status "Mendatang"	Berhasil
15	Buat Pesanan	Mengosongkan salah satu field input	Sistem menampilkan pesan *error* "Harap isi semua data"	Berhasil
16	Ubah Status Pesanan	Menekan tombol "Mulai"	Status berubah sesuai aksi dan sistem menampilkan pesanan di daftar "Dalam Proses"	Berhasil
17	Ubah Status Pesanan	Menekan tombol "Selesai"	Status berubah sesuai aksi dan sistem menampilkan pesanan di daftar "Selesai"	Berhasil
18	Batalkan	Menekan tombol	Status berubah sesuai aksi	Berhasil

No	Fitur yang Diuji	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian
	Pesanan	"Batalkan"	dan sistem menampilkan pesanan di daftar "Dibatalkan"	
19	Hapus Produk	Pengguna menekan tombol Hapus	Produk terhapus dari daftar produk	Berhasil
20	Hapus Mitra	Pengguna menekan tombol Hapus	Mitra terhapus dari daftar mitra	Berhasil
21	Hapus akun	Pengguna menekan tombol Hapus	Akun terhapus dari daftar akun	Berhasil

Berdasarkan hasil pelaksanaan Black Box Testing di iterasi kedua pada Tabel 4.2, seluruh fungsionalitas aplikasi dapat berjalan dengan baik sesuai dengan kebutuhan pengguna. Setiap fitur yang diuji menunjukkan hasil yang sesuai dengan keluaran yang diharapkan, baik pada proses pengelolaan data, transaksi, maupun navigasi antar halaman. Pengujian ini juga menunjukkan bahwa perbaikan yang dilakukan pada tahap Build iterasi kedua tidak menimbulkan gangguan atau kesalahan baru pada fungsi sistem yang telah berjalan sebelumnya.

Selain itu, pengujian fungsional pada iterasi kedua memastikan bahwa peningkatan kecepatan perpindahan halaman tidak memengaruhi stabilitas aplikasi secara keseluruhan. Seluruh skenario pengujian dapat dieksekusi dengan baik tanpa ditemukan kendala atau kegagalan fungsi. Dengan demikian, dapat disimpulkan bahwa aplikasi berada dalam kondisi stabil dan siap untuk digunakan oleh pengguna pada tahap pengujian selanjutnya.

2. Wawancara

Tahap wawancara pada iterasi kedua dilakukan setelah pengguna mengikuti demo aplikasi yang telah mengalami perbaikan pada aspek performa, khususnya kecepatan perpindahan halaman. Wawancara ini bertujuan untuk memperoleh umpan balik langsung dari pengguna mengenai pengalaman penggunaan aplikasi setelah dilakukan penyempurnaan pada tahap Build iterasi kedua. Proses wawancara dilakukan secara langsung dengan pemilik sebagai pengguna utama sistem setelah mencoba kembali seluruh fungsi aplikasi.

Berdasarkan hasil wawancara yang dilakukan, pengguna menyampaikan bahwa aplikasi telah mengalami peningkatan performa dibandingkan dengan iterasi sebelumnya. Perpindahan antar halaman dirasakan lebih cepat dan responsif, sehingga proses penggunaan aplikasi menjadi lebih nyaman. Selain itu, pengguna menyatakan bahwa alur penggunaan aplikasi secara keseluruhan sudah berjalan dengan baik dan tidak lagi menimbulkan kendala seperti yang dirasakan pada iterasi pertama.

Hasil wawancara ini menunjukkan bahwa perbaikan yang dilakukan pada iterasi kedua telah memberikan dampak positif terhadap pengalaman pengguna. Dengan meningkatnya performa aplikasi dan berkurangnya hambatan dalam penggunaan, pengguna menyatakan kepuasan terhadap kondisi aplikasi saat ini. Temuan dari tahap wawancara ini selanjutnya menjadi dasar untuk melanjutkan proses pengukuran usability menggunakan kuesioner *System Usability Scale* (SUS).

3. System Usability Scale

Pengujian *System Usability Scale* (SUS) dilakukan untuk menilai tingkat kemudahan penggunaan aplikasi berdasarkan persepsi subjektif pengguna. Kuesioner SUS diberikan kepada pemilik dan dua karyawan setelah mereka mencoba seluruh fitur inti aplikasi. Setiap responden diminta memberikan penilaian terhadap sepuluh pernyataan yang mencakup aspek kemudahan penggunaan, konsistensi antarmuka, kompleksitas sistem, serta keyakinan pengguna dalam mengoperasikan aplikasi. Hasil penilaian dari pemilik dan dua karyawan dapat dilihat pada Tabel 4.3.

Tabel 4.3 Rekap Jawaban SUS

No	**Nama**	**Jabatan**	**Pertanyaan**										**Subtotal**
			1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
1	Tuti	Pemilik	4	3	3	3	3	3	3	3	3	2	30
2	Yanti	Karyawan	4	3	3	3	4	3	3	3	3	2	31
3	Hendi	Karyawan	4	3	3	3	4	3	3	3	3	3	32

Berdasarkan data pada Tabel 4.2, dilakukan perhitungan skor SUS menggunakan persamaan (2.1), (2.2), (2.3), dan (2.4) seperti yang sudah dijelaskan pada BAB II. Hasil dari perhitungan metode SUS dapat dilihat pada Tabel 4.4.

Tabel 4.4 Hasil Perhitungan Skor SUS

No	**Nama**	**Subtotal**	**Skor SUS (Subtotal x 2.5)**	**Rata-Rata**
1	Tuti	30	75.0	77.5
2	Yanti	31	77.5	
3	Hendi	32	80.0	

Berdasarkan Tabel 4.4 Hasil dari perhitungan SUS menunjukan Skor 77.5 menempatkan aplikasi pada kategori **Acceptable**, yang tidak hanya menunjukkan bahwa aplikasi sudah mudah digunakan dan diterima oleh pengguna, tetapi juga mencerminkan bahwa mayoritas responden merasa proses navigasi, interaksi antarmuka, dan akses terhadap fitur inti berjalan lancar tanpa menimbulkan kebingungan berarti. Skor ini mengindikasikan bahwa pengalaman pengguna berada pada level yang baik untuk sebuah MVP, di mana sistem telah memenuhi ekspektasi dasar dalam hal kemudahan penggunaan, kejelasan fungsi, serta konsistensi tampilan. Hasil pengujian *usability* yang diperoleh dari pengujian *System Usability Scale* (SUS) juga menunjukkan bahwa pendekatan *Lean Startup* yang diterapkan dalam pengembangan aplikasi telah berhasil menghasilkan sistem yang sesuai dengan kebutuhan pengguna. Aplikasi yang mudah digunakan dan diterima dengan baik oleh pengguna berpotensi untuk terus dimanfaatkan dalam kegiatan operasional pabrik abon Merak Bewangi. Dengan demikian, hasil pengujian SUS ini tidak hanya mencerminkan kualitas antarmuka dan kemudahan penggunaan sistem, tetapi juga menjadi indikator bahwa aplikasi POS yang dikembangkan mampu mendukung keberlanjutan usaha melalui digitalisasi proses pencatatan yang lebih terstruktur, konsisten, dan mudah diwariskan kepada generasi pengelola berikutnya.

c Learn

Tahap *Learn* pada iterasi kedua merupakan proses evaluasi akhir terhadap keseluruhan perbaikan yang telah dilakukan selama siklus pengembangan iterasi kedua. Pada tahap ini, peneliti meninjau kembali seluruh hasil pengukuran dan pengamatan untuk memastikan bahwa perubahan yang diterapkan benar-benar selaras dengan kebutuhan dan harapan pengguna. Berdasarkan hasil pengukuran yang telah dilakukan pada tahap *Measure*, dapat disimpulkan bahwa fokus pengembangan pada penyempurnaan performa aplikasi telah memberikan dampak positif terhadap pengalaman penggunaan aplikasi secara menyeluruh.

Pengguna menyatakan bahwa aplikasi kini terasa lebih ringan dan nyaman digunakan dibandingkan dengan iterasi sebelumnya. Waktu tunggu yang sebelumnya dirasakan saat membuka halaman atau menampilkan data sudah tidak lagi mengganggu alur kerja pengguna. Kondisi ini membuat aktivitas operasional, seperti pencatatan pesanan dan pengelolaan data, dapat dilakukan dengan lebih lancar dan berkesinambungan. Selain itu, pengguna juga menyampaikan bahwa aplikasi lebih responsif ketika digunakan secara

berulang dalam waktu yang cukup lama, sehingga mendukung efisiensi kerja sehari-hari.

Lebih lanjut, pengguna menilai bahwa fitur-fitur yang tersedia pada aplikasi sudah mampu memenuhi kebutuhan operasional pabrik abon Merak Bewangi secara memadai. Tidak terdapat permintaan penambahan fitur baru dalam waktu dekat, karena pengguna merasa bahwa fungsi utama aplikasi telah berjalan sesuai dengan yang diharapkan. Hal ini menunjukkan bahwa penyempurnaan yang dilakukan pada iterasi kedua telah tepat sasaran dan memberikan nilai tambah yang nyata bagi pengguna.

Dengan demikian, hasil pembelajaran pada iterasi kedua menunjukkan bahwa tujuan pengembangan telah tercapai. Aplikasi POS yang dikembangkan tidak hanya memenuhi kebutuhan fungsional pengguna, tetapi juga memberikan kenyamanan dan kelancaran dalam penggunaan sehari-hari. Hasil ini menegaskan bahwa pendekatan pengembangan berbasis iterasi yang berfokus pada masukan pengguna efektif dalam menghasilkan sistem yang siap digunakan dan mendukung kegiatan operasional pabrik abon Merak Bewangi secara optimal.

4.2 Pembahasan

Berdasarkan hasil pengujian dan evaluasi yang telah dilakukan, aplikasi mobile *Point of Sale* (POS) yang dikembangkan menunjukkan kesesuaian yang baik dalam mendukung potensi keberlanjutan usaha. Keberlanjutan usaha pada penelitian ini ditekankan karena permasalahan awal yang terjadi pada pabrik abon Merak Bewangi berkaitan dengan pencatatan pesanan dan penjualan yang masih dilakukan secara manual, sehingga berpotensi menimbulkan ketidakteraturan data, kesalahan pencatatan, serta kesulitan dalam memantau perkembangan usaha secara berkelanjutan. Kondisi tersebut apabila dibiarkan dapat menghambat konsistensi operasional dan menyulitkan pengambilan keputusan, sehingga dibutuhkan sistem yang mampu mendukung pencatatan yang lebih terstruktur dan terdokumentasi.

Pengembangan aplikasi difokuskan pada kebutuhan operasional pengguna, seperti pencatatan pesanan, pengelolaan produk, pengelolaan mitra, serta pemantauan transaksi secara terintegrasi. Pendekatan ini menghasilkan sistem yang mampu menggantikan proses manual menjadi digital, sehingga aktivitas operasional menjadi lebih tertata, efisien, dan terdokumentasi dengan baik. Dengan adanya digitalisasi proses pencatatan, informasi usaha tidak lagi bergantung pada pencatatan terpisah yang rentan hilang atau sulit ditelusuri, melainkan tersimpan dalam sistem dan dapat diakses kembali ketika dibutuhkan.

Selain aspek implementasi fitur, kelayakan sistem dalam mendukung operasional juga diperkuat melalui pengujian fungsional. Pengujian *Black Box Testing* dilakukan untuk

memastikan seluruh fungsionalitas aplikasi tetap berjalan sesuai kebutuhan fungsional serta tidak menimbulkan kesalahan baru setelah dilakukan perbaikan pada iterasi pengembangan. Pengujian ini mencakup fungsi-fungsi utama yang digunakan pengguna dalam kegiatan operasional, seperti pengelolaan data, proses transaksi, serta navigasi antar halaman. Dengan hasil pengujian fungsional yang baik, aplikasi dinilai memiliki reliabilitas yang memadai untuk digunakan dalam aktivitas kerja sehari-hari tanpa mengganggu alur operasional.

Hasil pengujian *System Usability Scale* (SUS) menunjukkan bahwa aplikasi memiliki tingkat kegunaan yang dapat diterima oleh pengguna. Skor SUS yang diperoleh mengindikasikan bahwa aplikasi mudah dipelajari, alur penggunaan jelas, serta fitur-fitur yang disediakan relevan dengan kebutuhan pengguna. Tingkat kegunaan yang baik ini menjadi faktor penting dalam memastikan aplikasi dapat digunakan secara berkelanjutan tanpa menimbulkan kesulitan atau penolakan dari pengguna. Selain itu, hasil pengujian usability juga menunjukkan bahwa aplikasi yang mudah digunakan dan diterima dengan baik berpotensi terus dimanfaatkan dalam kegiatan operasional, sehingga mendukung keberlanjutan usaha melalui digitalisasi pencatatan yang lebih terstruktur, konsisten, dan mudah diwariskan kepada pengelola berikutnya.

Hasil wawancara memperkuat temuan dari pengujian SUS. Pengguna menyampaikan bahwa aplikasi membantu mempercepat proses kerja, mempermudah pencatatan dan pemantauan pesanan, serta meningkatkan keteraturan data operasional. Pengguna juga menilai bahwa aplikasi sesuai dengan alur kerja usaha yang dijalankan dan layak digunakan dalam jangka panjang. Dengan demikian, aplikasi mobile POS yang dikembangkan tidak hanya berhasil diimplementasikan secara teknis, tetapi juga telah diuji kesesuaiannya melalui pengukuran kegunaan dan umpan balik langsung dari pengguna.

Setelah aplikasi POS dikembangkan dan diimplementasikan, penelitian ini tidak secara langsung mengklaim bahwa keberlanjutan usaha telah tercapai sepenuhnya, karena keberlanjutan usaha merupakan kondisi yang bersifat jangka panjang dan membutuhkan pembuktian melalui penggunaan sistem secara kontinu dalam periode waktu yang lebih panjang. Namun demikian, hasil penelitian menunjukkan bahwa aplikasi yang dibangun **berpotensi mendukung keberlanjutan usaha**, karena beberapa indikator penting telah terpenuhi. Pertama, aplikasi mampu menggantikan proses pencatatan manual menjadi sistem digital yang lebih terstruktur sehingga data penjualan dan pesanan dapat tersimpan dan dikelola dengan lebih baik. Kedua, berdasarkan pengujian Black Box, fungsi-fungsi inti sistem berjalan sesuai kebutuhan fungsional, yang mengindikasikan bahwa aplikasi layak digunakan dalam operasional. Ketiga, hasil uji usability menggunakan metode SUS memperoleh nilai rata-rata

77,5 dengan kategori *Acceptable*, yang menunjukkan aplikasi dapat diterima dan mudah digunakan oleh pengguna. Keempat, hasil wawancara pengguna menunjukkan bahwa aplikasi membantu mempercepat proses kerja, mempermudah pengelolaan pesanan dan pencatatan, serta dinilai layak digunakan dalam jangka panjang. Dengan terpenuhinya aspek fungsional, kemudahan penggunaan, dan penerimaan pengguna tersebut, aplikasi yang dikembangkan dapat diposisikan sebagai solusi yang mendukung kelangsungan operasional dan efisiensi kerja, yang merupakan bagian penting dalam mewujudkan keberlanjutan usaha.

Secara keseluruhan, integrasi hasil pengujian fungsional, pengukuran usability, dan wawancara pengguna menunjukkan bahwa aplikasi mobile POS mampu memenuhi kebutuhan operasional pabrik abon Merak Bewangi dan berpotensi mendukung keberlanjutan usaha melalui peningkatan efisiensi operasional, kemudahan penggunaan, penerimaan sistem oleh pengguna, serta pencatatan data yang lebih terstruktur dan terdokumentasi.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil perancangan, pengembangan, serta evaluasi aplikasi mobile *Point of Sale* (POS) pada pabrik abon Merak Bewangi yang telah dilakukan melalui pendekatan *Lean Startup*, dapat ditarik beberapa kesimpulan sebagai berikut.

Pertama, penelitian ini berhasil merancang dan mengimplementasikan aplikasi mobile POS berbasis Android yang mampu menggantikan proses pencatatan manual menjadi sistem digital yang lebih terstruktur. Aplikasi yang dikembangkan telah menyediakan fitur utama berupa manajemen produk, manajemen mitra, manajemen akun, pencatatan dan pemrosesan pesanan, serta *dashboard* ringkasan penjualan. Keberadaan fitur-fitur tersebut mampu menjawab permasalahan utama yang sebelumnya dihadapi oleh pabrik abon Merak Bewangi, khususnya terkait keterlambatan pencatatan, risiko kesalahan input, serta kesulitan dalam memantau data penjualan.

Kedua, penerapan pendekatan *Lean Startup* melalui siklus *Build–Measure–Learn* terbukti efektif dalam menghasilkan sistem yang sesuai dengan kebutuhan nyata pengguna. Pengembangan aplikasi dilakukan secara iteratif dengan melibatkan pemilik dan karyawan sebagai pengguna utama, sehingga setiap perbaikan dan penyempurnaan fitur didasarkan pada umpan balik langsung dari pengguna. Pendekatan ini memungkinkan sistem berkembang secara adaptif dan relevan terhadap kondisi operasional UMKM, serta meminimalkan risiko pengembangan fitur yang tidak dibutuhkan. Dengan demikian, aplikasi yang dihasilkan memiliki potensi untuk terus digunakan dan dikembangkan seiring dengan pertumbuhan usaha, yang merupakan salah satu aspek penting dalam keberlanjutan usaha.

Ketiga, hasil pengujian sistem menggunakan metode Black Box Testing menunjukkan bahwa seluruh fitur aplikasi dapat berfungsi sesuai dengan kebutuhan fungsional yang telah ditetapkan. Selain itu, hasil pengujian *System Usability Scale* (SUS) menunjukkan bahwa aplikasi memiliki tingkat usability yang baik dan dapat diterima oleh pengguna. Tingkat kemudahan penggunaan ini menjadi faktor pendukung keberlanjutan usaha, karena sistem yang mudah dipahami dan dioperasikan memungkinkan proses pencatatan dan pengelolaan usaha dapat dijalankan secara konsisten tanpa memerlukan pembelajaran yang kompleks. Hal ini juga mempermudah adaptasi pengguna baru apabila terjadi pergantian pengelola atau generasi penerus dalam UMKM pabrik abon Merak Bewangi.

Keempat, pada iterasi kedua evaluasi sistem difokuskan pada pengujian langsung oleh pengguna. Evaluasi pada iterasi ini dilakukan dengan cara langsung menyerahkan aplikasi kepada pengguna, yaitu satu orang *owner* dan dua orang *employee*, untuk menguji peningkatan performa sistem, khususnya pada aspek kecepatan *fetch data*. Hasil pengujian menunjukkan bahwa optimasi yang dilakukan berhasil meningkatkan kecepatan pengambilan data secara signifikan, sehingga aplikasi dapat digunakan dengan lebih responsif dan mendukung proses operasional pengguna secara lebih efisien.

Namun demikian, penelitian ini memiliki keterbatasan pada jumlah responden yang terlibat dalam proses evaluasi sistem, yaitu sebanyak tiga orang pengguna internal UMKM pabrik abon Merak Bewangi. Oleh karena itu, hasil dan kesimpulan penelitian ini bersifat kontekstual serta hanya merepresentasikan kondisi dan kebutuhan pengguna pada UMKM tersebut. Temuan penelitian ini tidak dimaksudkan untuk digeneralisasi pada UMKM secara umum, melainkan sebagai studi kasus penerapan aplikasi POS dan pendekatan Lean Startup pada konteks internal objek penelitian. Meskipun demikian, hasil penelitian ini tetap memiliki nilai praktis sebagai referensi awal bagi pengembangan sistem serupa pada UMKM dengan karakteristik yang sejenis.

Secara keseluruhan, aplikasi mobile POS yang dikembangkan dalam penelitian ini tidak hanya berfungsi sebagai alat bantu pencatatan transaksi, tetapi juga berperan sebagai sistem pendukung keberlanjutan usaha. Melalui digitalisasi proses operasional, penyimpanan data yang terstruktur, serta kemudahan penggunaan sistem, aplikasi ini mampu membantu UMKM pabrik abon Merak Bewangi dalam menjaga kontinuitas operasional, mendukung proses alih pengetahuan antar generasi, dan meningkatkan kesiapan usaha untuk berkembang secara berkelanjutan di masa mendatang.

5.2 Saran

Berdasarkan hasil penelitian dan kesimpulan yang telah diperoleh, terdapat beberapa saran yang dapat dipertimbangkan sebagai bahan evaluasi dan acuan dalam pengembangan sistem lebih lanjut, baik dari sisi teknis maupun penerapan di lapangan, serta sebagai referensi bagi penelitian selanjutnya yang memiliki konteks dan karakteristik serupa.

Pertama, aplikasi POS yang dikembangkan dapat ditingkatkan dengan menambahkan fitur lanjutan, seperti manajemen stok bahan baku secara otomatis, laporan keuangan yang lebih komprehensif (laba rugi dan arus kas), serta ekspor data laporan ke dalam format dokumen atau *spreadsheet* untuk kebutuhan administrasi.

Kedua, optimalisasi performa sistem masih dapat dikembangkan lebih lanjut, khususnya

pada proses pemuatan data dalam jumlah besar dan visualisasi grafik penjualan. Hal ini penting untuk memastikan aplikasi tetap responsif ketika jumlah data transaksi semakin meningkat seiring pertumbuhan usaha.

Ketiga, pengembangan sistem selanjutnya dapat mempertimbangkan penambahan *role* pengguna baru, seperti *customer*, yang memungkinkan pelanggan melakukan proses pembelian secara langsung melalui aplikasi. Selain itu, fitur pendukung seperti sistem pembayaran digital juga dapat ditambahkan untuk memfasilitasi transaksi secara *end-to-end*, sehingga aplikasi tidak hanya berfungsi sebagai alat pencatatan internal, tetapi juga sebagai media transaksi yang terintegrasi dan lebih mendukung proses bisnis secara menyeluruh.

Keempat, dari sisi keamanan, pengembangan lanjutan dapat mempertimbangkan penerapan mekanisme keamanan tambahan, seperti audit log aktivitas pengguna dan pengaturan hak akses yang lebih rinci, untuk meningkatkan keandalan sistem dalam jangka panjang.

Dengan adanya saran-saran tersebut, diharapkan aplikasi mobile POS yang dikembangkan dapat terus disempurnakan dan dikembangkan secara berkelanjutan sebagai solusi digital yang relevan bagi UMKM. Pengembangan lanjutan ini diharapkan mampu memperluas fungsi aplikasi tidak hanya sebagai alat bantu operasional internal, tetapi juga sebagai sistem pendukung proses bisnis yang lebih terintegrasi. Selain itu, penerapan solusi digital ini diharapkan dapat mempercepat proses digitalisasi UMKM, meningkatkan efisiensi dan efektivitas operasional usaha, serta membantu pelaku UMKM dalam mengambil keputusan bisnis secara lebih tepat berdasarkan data yang akurat dan terstruktur.

DAFTAR PUSTAKA

Amelia, M., & Vidiati, C. (2025). Strategi Penerapan Lean Startup untuk Meningkatkan Efisiensi dan Inovasi UMKM Berbasis Syariah di Era Digital (Studi di Desa Pesantren Kota Cirebon). *RIGGS: Journal of Artificial Intelligence and Digital Business*, *4*(4), 611–620. https://doi.org/10.31004/riggs.v4i4.3450

Andriasari, S., Nizam, N., Nurhasanah, I. A., & Wulandari, K. S. (2024). Aplikasi Point of Sale untuk Meningkatkan Profitabilitas dan Digitalisasi UMKM. *EXPERT: Jurnal Manajemen Sistem Informasi Dan Teknologi*, *14*(2), 89. https://doi.org/10.36448/expert.v14i2.3974

Anggi Aditiya Ningsih, Rizky Suhanry Rambe, Yuli Novia Munthe, & Purnama Ramadani Sialalahi. (2023). Pendekatan Lean Startup Pada Desain Produk Dan Teknik Minimum Viable Product Dalam Menyikapi Skeptisisme Pada Iklim Bisnis. *Jurnal Publikasi Sistem Informasi Dan Manajemen Bisnis*, *2*(1), 183–198. https://doi.org/10.55606/jupsim.v2i1.867

Fathoni, A., Afirianto, T., & Akbar, M. A. (2025). Analisis Perbandingan Performa Antara Prisma dan KnexJS pada Aplikasi API Berbasis Typescript. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *9*(7), 1–10.

Isabel, A., & Pereira, S. (2022). *Lean Startup: Improving an MVP with service design tools*. Ismul Afriza, F., & Kurniawan Pakpahan, R. (2023). Pengembangan Aplikasi Point of Sales Berbasis Mobile dan Web pada Browenz Coffee. *Julyxxxx*, *24*(2), 28–38. https://doi.org/https://doi.org/10.55601/jsm.24i2.pg

Kaylan, K. B., Russel, S. M., Justice, C. N., Sheena, M. K., Hirshfield, L. E., Heiman, H. L., & Curry, R. H. (2022). Applying the Lean Startup Method to Structure Project- Based, Student-Driven Curricular Enhancements. *Teaching and Learning in Medicine*, *34*(4), 434–443. https://doi.org/10.1080/10401334.2021.1928501

Khairunnisa, G., & Voutama, A. (2024). Peminjaman Inventaris Berbasis Web di BEM Fasilkom Unsika. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *8*(2), 48–55.

Lortie, J., Cox, K., DeRosset, S., Thompson, R., & Kelly, S. (2025). Unpacking the minimum viable product (MVP): a framework for use, goals and essential elements. *Journal of Small Business and Enterprise Development*, *32*(1), 212–235. https://doi.org/10.1108/JSBED-02-2024-0075

Mendrofa, L., Zendrato, B., & Zai, I. (2025). *Pengaruh Digitalisasi Pada Peningkatan*

Efisiensi Operasional Usaha Mikro, Kecil, dan Menengah (UMKM) di Indonesia Tahun 2023.

Mulyani, L. N., Iqbal, M., Islamaya, A., Rati, & Rachmadianti, F. (2025). Pengembangan Aplikasi Point of Sale Berbasis Android pada UMKM Nyemil Beauty Menggunakan Metode Agile. *Jurnal Sains Informatika Terapan*.

Narulita, S., Nugroho, A., & Abdillah, M. Z. (2024). Diagram Unified Modelling Language (UML) untuk Perancangan Sistem Informasi Manajemen Penelitian dan Pengabdian Masyarakat (SIMLITABMAS) Universitas Nasional Karangturi Semarang , Indonesia (deskripsi) dan perancangan sistem , khususnya pada pemrogr. *BRIDGE : Jurnal Publikasi Sistem Informasi Dan Telekomunikasi*, *2*(3), 244–256.

Nurhidayat, A. R., Nugraha, B., & Hendriadi, A. A. (2025). Perancangan Aplikasi Point of Sales (Pos) Berbasis Android Dengan Qris Payment (Studi Kasus: Warung Seblak Tonjong). *Jurnal Informatika Dan Teknik Elektro Terapan*, *13*(2). https://doi.org/10.23960/jitet.v13i2.6323

Phan, I. K., & Yuricha, Y. (2023). Implementasi Pendakatan Backendless Dalam Rapid Prototyping Aplikasi Manajem Penugasan Karyawan. *Jurnal Cahaya Mandalika*, *4*(1), 111–118. https://ojs.cahayamandalika.com/index.php/JCM/article/view/1304

Pratama, R. Y., & Somya, R. (2021). Perancangan Aplikasi Point Of Sales (POS) Berbasis Android (Studi Kasus: Warkop Vape Salatiga). *JATISI (Jurnal Teknik Informatika Dan Sistem Informasi)*, *8*(4), 1923–1938. https://doi.org/10.35957/jatisi.v8i4.1218

Putra, R. R. S., & Hartono, B. (2024). Sistem Informasi Penjualan Berbasis Web Pada Toko Buku Putra Ilmu Semarang. *Jurnal Ilmiah Sistem Informasi*, *3*(3), 11–18.

Rivydho Assidiq, M., Arianti, & Bahri, S. (2022). Analisis Usability Fitur Rating Pada Aplikasi Ladder Menggunakan Metode System Usability Scale. *Jtriste*, *9*(2), 12–21. https://doi.org/10.55645/jtriste.v9i2.374

Rospricilia, T. A., & Ma'ady, M. N. P. (2024). Pemodelan Integration Use Case (IUC): Perancangan Use Case Diagram (UML) untuk Sistem-sistem yang Terintegrasi. *INTEGER: Journal of Information Technology*, *9*(2). https://doi.org/10.31284/j.integer.2024.v9i2.6345

Sri Rahayu, & Tata Sutabri. (2024). Pendekatan Metode Lean Startup dalam Menganalisa Kepuasan Pelanggan pada Menu Masakan "Rajanya Seafood." *Uranus : Jurnal Ilmiah Teknik Elektro, Sains Dan Informatika*, *2*(4), 150–164. https://doi.org/10.61132/uranus.v2i4.489

Suhargo, A., Vardi, P., & Megawan, S. (2025). Pengembangan Aplikasi Point of Sale (POS)

berbasis Mobile untuk Membantu UMKM Development of a Mobile-based Point of Sale (POS) Application to Help MSMEs. *Jurnal Janitra Informatika Dan Sistem Informasi*, *5*(2), 199–214.

Sumarto, M. A. (2023). Analisis dan Perancangan Aplikasi Point of Sale (POS) untuk Usaha Mikro, Kecil, dan Menengah (UMKM) dengan Metode Rapid Application Development (RAD). *Jurnal Studi Komunikasi Dan Media*, *27*(1), 17–34. https://doi.org/10.17933/jskm.2023.5115

Tabrani, M., Suhardi, S., & Priyandaru, H. (2021). Sistem Informasi Manajemen Berbasis Website Pada Unl Studio Dengan Menggunakan Framework Codeigniter. *Jurnal Ilmiah M-Progress*, *11*(1). https://doi.org/10.35968/m-pu.v11i1.598

Undang-Undang Nomor 20 Tahun 2008 Tentang Usaha Mikro, Kecil, Dan Menengah (UMKM).

Valerian Romero, A., Kusnadi, K., & Fahrudin, R. (2024). Membangun Marketplace Untuk Penjualan Produk Kreatif Mahasiswa Berbasis Mobile Menggunakan Metode Fdd. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *7*(6), 3400–3405. https://doi.org/10.36040/jati.v7i6.7278

Wijaya, T. (2025). Teknik Coding Hybrid Antara Typescript Dan Javascript Dalam Framework Angular. *Information System Journal*, *8*(01), 21–28. https://doi.org/10.24076/infosjournal.2025v8i01.2048

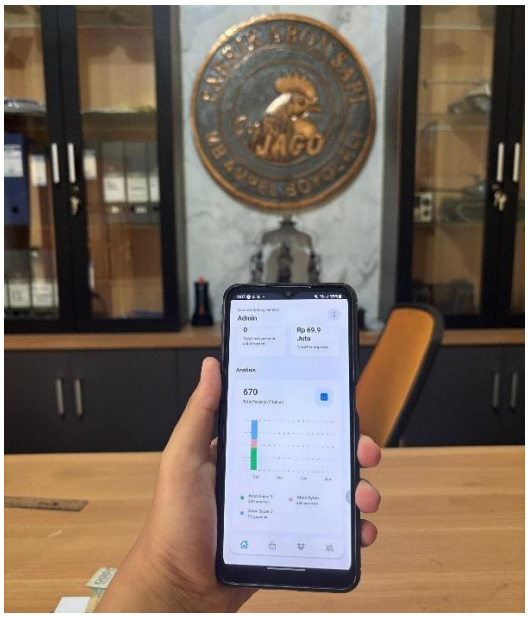
Wisnu, G., Prasetyo, T., Pradana, F., & Prakoso, B. S. (2022). Pengembangan Aplikasi Point of Sales Warung dan UMKM "WarunkQu" menggunakan Framework Flutter. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *6*(10), 2548–2964. http://j-ptiik.ub.ac.id

Yaqin, M. A. (2023). Pengembangan Aplikasi Marketplace Ikan Di Kabupaten Probolinggo Berbasis Frontend Backend Menggunakan React Js. *NJCA (Nusantara Journal of Computers and Its Applications)*, *8*(2), 63. https://doi.org/10.36564/njca.v8i2.342

Yoo, O. S., Huang, T., & Arifo, K. (2021). *A Theoretical Analysis of the Lean Startup Method*. https://ssrn.com/abstract=3070613

LAMPIRAN

Lampiran A: Implementasi Aplikasi pada Pabrik Abon



Lampiran B: Demo Aplikasi Pemilik

Lampiran C: Demo Aplikasi Karyawan

Lampiran D: Demo Aplikasi Karyawan