BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Selama pelaksanaan program Merdeka Belajar Kampus Merdeka (MBKM) di PT Panca Budi Tbk, peran ganda sebagai Front-End Programmer dan System Analyst diemban. Berdasarkan struktur organisasi perusahaan yang telah dijelaskan pada Gambar 2.2, posisi ini secara struktural berada di bawah supervisi langsung dari IT Project Manager, yang kemudian berkoordinasi dengan Information Technology (IT) Director.

Dalam konteks pembimbingan dan pelaporan, bimbingan langsung diterima dari dua atasan, yaitu Ko Ferry Kurniawan sebagai Pembimbing Lapangan, dan Ko Yos Sugianto sebagai Head Direksi. Ko Ferry Kurniawan berperan sebagai supervisor harian yang memberikan arahan teknis, mengelola tugas-tugas proyek, dan melakukan *code review*. Sementara itu, Ko Yos Sugianto, sebagai Head Direksi, memberikan arahan strategis dan mengawasi keseluruhan progres proyek dari perspektif manajemen tingkat atas, memastikan bahwa pekerjaan yang dilakukan selaras dengan tujuan bisnis perusahaan.

Alur Kerja dan Koordinasi:

Alur kerja dan koordinasi dalam menjalankan peran sebagai Front-End Programmer dan System Analyst di PT Panca Budi Tbk dapat digambarkan sebagai berikut:

- Analisis Kebutuhan Sistem (Peran System Analyst): Keterlibatan dalam tahap awal proyek untuk menganalisis kebutuhan bisnis dan pengguna. Ini

mencakup pengumpulan data, wawancara dengan *user* terkait, serta perumusan spesifikasi fungsional dan non-fungsional sistem. Hasil analisis ini menjadi dasar untuk perancangan antarmuka dan pengembangan *front-end*.

- Perancangan Antarmuka Pengguna (UI/UX) (Peran System Analyst & Front-End Programmer): Berdasarkan hasil analisis, prototipe dan *mockup* antarmuka pengguna dirancang, memastikan desain yang intuitif dan sesuai dengan kebutuhan pengguna. Koordinasi erat dengan *business analyst* atau *product owner* dilakukan untuk validasi konsep desain.
- Pengembangan Front-End (Peran Front-End Programmer): Tanggung jawab dalam menulis kode program untuk membangun antarmuka pengguna aplikasi. Ini meliputi implementasi desain UI/UX menjadi kode fungsional menggunakan teknologi *front-end* yang relevan (misalnya HTML, CSS, JavaScript, *framework*).
- Integrasi dan Koordinasi Back-End: Sebagai Front-End Programmer, koordinasi dengan tim *back-end developer* dilakukan untuk memastikan integrasi yang mulus antara *front-end* dan *back-end* melalui API (*Application Programming Interface*).
- Pengujian (Peran Front-End Programmer & System Analyst):
    - *Unit Testing & Integration Testing*: Pengujian terhadap komponen *front-end* yang dikembangkan dilakukan untuk memastikan fungsionalitas dan kompatibilitasnya.
    - *User Acceptance Testing (UAT)*: Berperan dalam membantu tim *Quality Assurance* atau *Product Owner* dalam UAT, memastikan sistem sesuai dengan kebutuhan dan ekspektasi pengguna yang telah dianalisis sebelumnya.
- Perbaikan *Bug* dan Pemeliharaan: Identifikasi, analisis, dan perbaikan *bug* yang ditemukan pada sisi *front-end* sistem dilakukan, serta pemeliharaan kode untuk optimasi kinerja.

- Dokumentasi: Spesifikasi kebutuhan sistem, alur bisnis, rancangan UI/UX, serta kode program yang telah dikembangkan didokumentasikan untuk referensi di masa mendatang.
- Pelaporan Progres & Konsultasi: Seluruh kegiatan dan progres harian dicatat secara rutin dalam Daily Task MBKM (MBKM 03). Konsultasi dan bimbingan secara berkala dengan Ko Ferry Kurniawan sebagai Pembimbing Lapangan dilakukan untuk mendapatkan arahan teknis, *feedback* terhadap kode, serta solusi atas kendala. Selain itu, pelaporan progres umum dan diskusi strategi juga dilakukan kepada Ko Yos Sugianto sebagai Head Direksi. Koordinasi juga dilakukan dengan pembimbing akademik dari Universitas Multimedia Nusantara untuk memastikan kesesuaian antara pengalaman praktis dan kurikulum MBKM.

3.2 Tugas dan Uraian Kerja Magang

Bagian ini menguraikan secara rinci tujuan dan tugas yang diemban selama periode kerja magang di PT Panca Budi Tbk, dengan fokus pada peran ganda sebagai Front-End Programmer dan System Analyst. Kesempatan didapatkan untuk mengaplikasikan pengetahuan teoritis ke dalam praktik nyata, berkontribusi langsung pada pengembangan dan perbaikan sistem informasi perusahaan.

3.2.1 Tujuan Kerja Magang

Adapun tujuan utama dari pelaksanaan tugas dan uraian kerja magang ini adalah sebagai berikut:

- **Mengimplementasikan Desain Antarmuka Pengguna (UI) yang Responsif:** Menerjemahkan kebutuhan desain menjadi komponen UI yang interaktif dan responsif, memastikan pengalaman pengguna yang optimal di berbagai perangkat. Hal ini krusial untuk memastikan aksesibilitas dan kemudahan penggunaan sistem di berbagai
  platform dan ukuran layar.

- **Menganalisis Kebutuhan Pengguna dan Bisnis:** Mengidentifikasi, mendokumentasikan, dan memvalidasi kebutuhan fungsional dan non-fungsional dari *user* untuk perancangan sistem yang efektif. Proses ini memastikan bahwa sistem yang dikembangkan benar-benar menjawab permasalahan dan memenuhi harapan pemangku kepentingan.
- **Melakukan Pengembangan dan Perbaikan Kode Front-End:** Menulis, menguji, dan memelihara kode *front-end* yang bersih, efisien, dan sesuai standar praktik terbaik. Fokus pada kualitas kode ini bertujuan untuk meningkatkan
  maintainability dan *scalability* sistem.
- **Berpartisipasi dalam Siklus Hidup Pengembangan Sistem (SDLC):** Keterlibatan aktif dalam berbagai fase SDLC, mulai dari analisis, desain, implementasi, hingga pengujian sistem. Ini memberikan pemahaman holistik tentang
  end-to-end process pengembangan perangkat lunak dalam lingkungan profesional.
- **Meningkatkan Kemampuan Pemecahan Masalah Teknis:** Mengidentifikasi akar masalah pada sistem dan merumuskan solusi teknis yang inovatif dan praktis. Latihan ini melatih kemampuan berpikir kritis dan analitis dalam menghadapi kendala.
- **Mengembangkan Keterampilan Kolaborasi Tim:** Berinteraksi dan berkoordinasi secara efektif dengan anggota tim IT lainnya (seperti *back-end developer*, *system analyst*, *project manager*). Keterampilan ini penting untuk mencapai tujuan bersama dalam proyek yang kompleks

3.2.2 Uraian Kerja Magang

Selama periode magang, berbagai tugas yang mencakup aspek *front-end programming* dan *system analysis* dilaksanakan, yang terbagi dalam beberapa proyek utama. Uraian pekerjaan ini mencerminkan kontribusi nyata dalam pengembangan sistem informasi di PT Panca Budi

Tbk, yang dirinci pada Tabel 1.3.3. Berikut adalah rincian tugas yang dikerjakan:

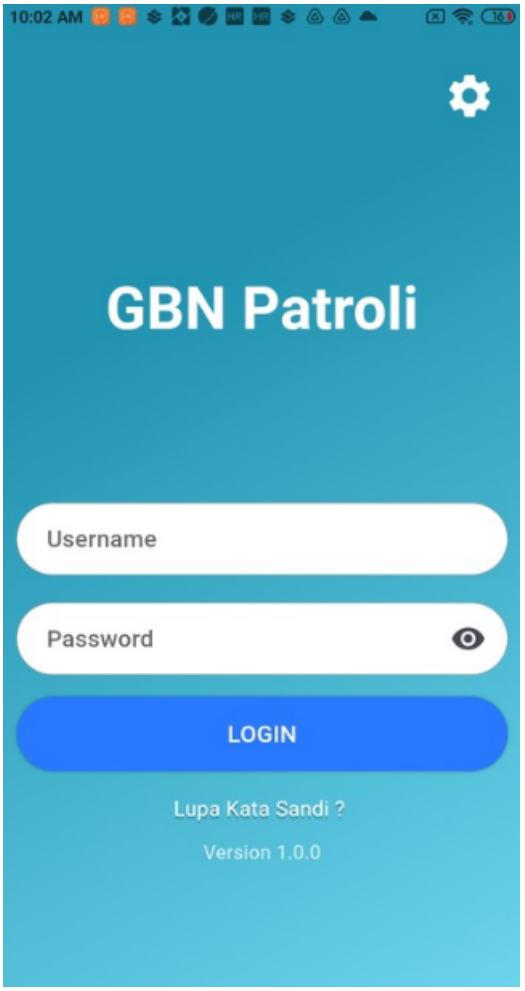
1. **Membuat User Interface *Login* untuk Aplikasi GBN Patroli**

   Kontribusi sebagai Front-End Programmer dimulai dengan fokus pada pengembangan antarmuka pengguna (UI) untuk tampilan *login* pada aplikasi GBN Patroli. Dalam pengerjaan ini, Android Studio dimanfaatkan sebagai *Integrated Development Environment* (IDE) utama. Tahapan Pembuatan UI:

   - Analisis Desain Awal: Proses dimulai dengan mempelajari dan mengadaptasi beberapa *template* UI dasar yang sudah tersedia dari perusahaan. Ini mempercepat proses dengan menyediakan kerangka desain yang konsisten dengan identitas visual perusahaan.
   - Desain dan Penyesuaian dengan Figma: Penyesuaian dan *refinement* desain tampilan *login* dilakukan menggunakan Figma, sebuah *tool* desain kolaboratif. Dalam tahap ini, tata letak, skema warna, tipografi, dan penempatan elemen-elemen UI (seperti *input field* untuk *username* dan *password*, serta tombol *login*) dipastikan sesuai dengan *guideline* perusahaan dan prinsip desain *User Experience* (UX) yang intuitif.
   - Implementasi Kode dengan XML dan Kotlin/Java: Setelah desain final disepakati, rancangan UI diimplementasikan ke dalam kode. Struktur tata letak dan elemen visual dibangun menggunakan XML pada *layout* file di Android Studio. Untuk logika interaksi dasar pada UI, seperti penanganan *input* atau respons tombol, digunakan bahasa pemrograman Kotlin atau Java yang terintegrasi di Android Studio.
   - Pengujian Responsivitas: Pengujian sederhana dilakukan untuk memastikan tampilan *login* responsif dan fungsional di berbagai

ukuran perangkat *mobile* serta kompatibel dengan sistem operasi Android yang dituju.

Dengan pendekatan ini, tampilan *login* yang dihasilkan tidak hanya estetis dan sesuai standar perusahaan, tetapi juga fungsional dan siap untuk diintegrasikan dengan logika *back-end*.



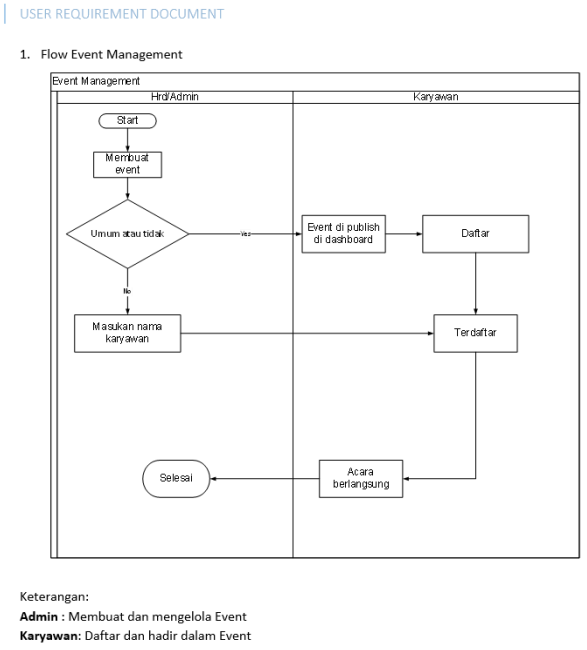
Gambar 3.1 Tampilan User Interface Login Aplikasi GBN Patroli

2. Membuat URD untuk workflow Create Event di sistme HRIS Panca Budi

Dalam peran sebagai System Analyst, tanggung jawab untuk menyusun Dokumen URD (*User Requirement Document*)

yang terperinci untuk *workflow Event Management* pada sistem HRIS (Human Resource Information System) Panca Budi diemban. URD ini berfungsi sebagai *blueprint* yang mendefinisikan secara jelas kebutuhan fungsionalitas dan interaksi pengguna dalam pengelolaan *event*, memastikan keselarasan antara harapan *user* dan kemampuan sistem yang akan dibangun atau dikembangkan. **Komponen Utama URD *Event Management*:**

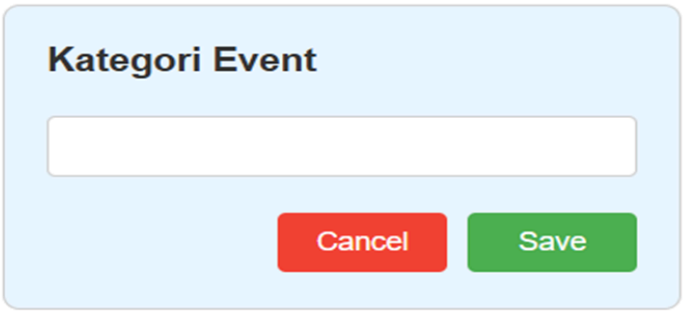
- Alur *Event Management*: Dokumen URD diawali dengan penjelasan alur kerja umum *Event Management*, yang membagi peran utama menjadi Admin dan Karyawan. Admin bertanggung jawab untuk membuat dan mengelola *event*, sementara Karyawan berfungsi untuk mendaftar dan hadir dalam *event* yang tersedia. Visualisasi alur ini disajikan melalui *flow chart* untuk memberikan pemahaman menyeluruh tentang bagaimana setiap proses berinteraksi.



Gambar 3.2 Flow Chart URD Create Event

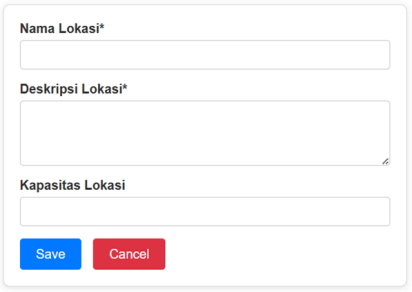
Master Data:

- **Master Kategori**: Bagian ini mendefinisikan kategori *event* (misalnya Seminar, Workshop). Kebutuhan untuk fitur "Tambah Baru" (memasukkan jenis *event* dengan validasi unik) dan "Edit" (mengubah data jenis *event*) dianalisis.



Gambar 3.3 Master Kategori

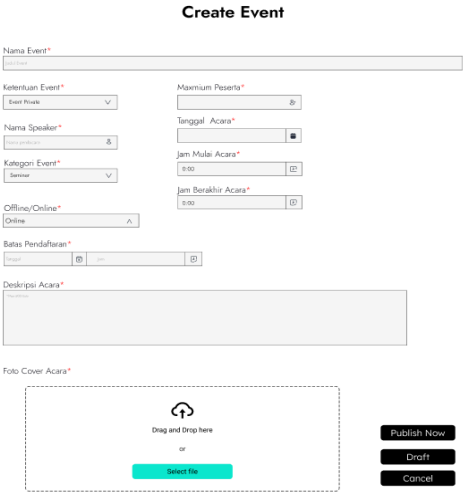
- **Master Lokasi:** Lokasi internal perusahaan yang sering digunakan didefinisikan. Fitur "Tambah Baru" (memasukkan nama lokasi, deskripsi, dan kapasitas, dengan validasi nama lokasi yang unik) dan "Edit" (menghapus atau mengubah data lokasi) diuraikan.



Gambar 3.4 Master Lokasi

Transaksi (*Transaction*)

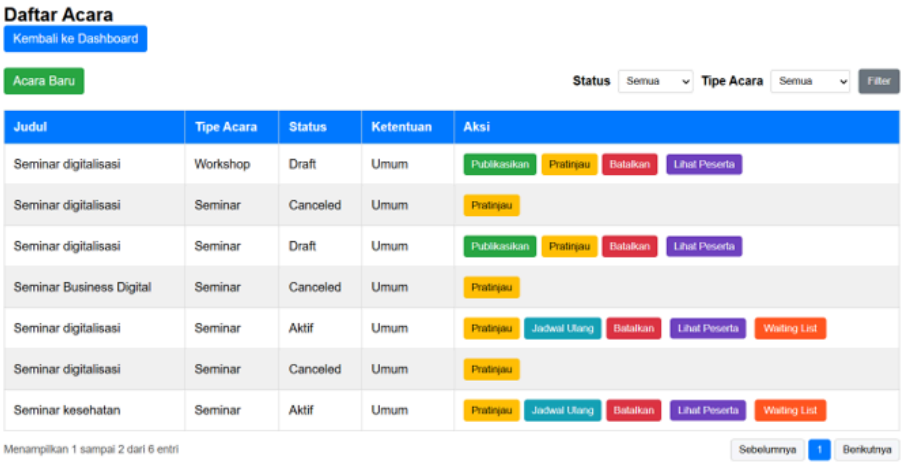
- **Transaksi *Create Event*:** Proses Admin membuat *event* baru didefinisikan. Kebutuhan untuk berbagai *field input* (Nama Acara, Ketentuan Acara, Jumlah Peserta Maksimal, Nama Pembicara, Kategori Acara, Tanggal Acara, Jam Mulai/Berakhir, Batas Pendaftaran, *Online/Offline*, Deskripsi, Foto Cover) dianalisis. Fitur penting seperti "Publikasikan Sekarang" dan "Simpan sebagai Draft" juga didetailkan.



Gambar 3.5 Tampilan User Interface Create Event

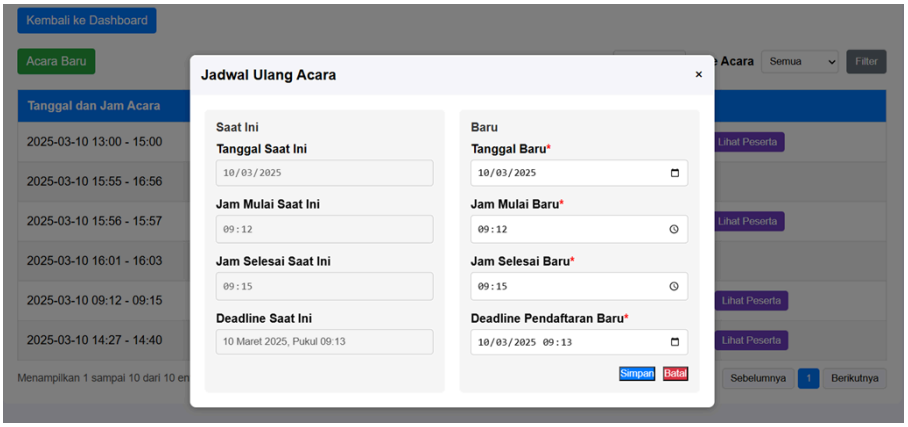
- **Transaksi *Daftar Event*:** Menguraikan fitur yang digunakan Admin untuk mengelola *event* yang telah dibuat. Ini mencakup

fungsi "Publikasikan" (untuk *draft*), "Pratinjau", "Jadwal Ulang" (dengan batasan satu kali), "Batal", "Lihat Peserta", dan "Waiting List".



Gambar 3.6 Tampilan User Interface Daftar Event

- **Transaksi *Reschedule*:** Proses perubahan jadwal *event* didefinisikan. Kebutuhan untuk memasukkan "Tanggal Baru", "Jam Mulai Baru", "Jam Selesai Baru", dan "Deadline Pendaftaran Baru" diidentifikasi.



Gambar 3.7 Tampilan User Interface Reschedule

- ***Transaksi Pendaftaran Karyawan:*** Menguraikan bagaimana Karyawan mendaftar untuk mengikuti event. Fitur-fitur seperti

"More Details" (mengarahkan ke halaman pendaftaran), *"Daftar"* (yang akan disable setelah terdaftar), *"Ajukan Batal"*, dan *"Waiting List"* (jika pendaftaran penuh) didetailkan. Dokumen juga mencatat adanya reminder otomatis H-1 sebelum event.

Gambar 3.8 Tampilan User Interface Pendaftaran Karyawan

3. **Membuat User Interface yang baru untuk "Create Event" pada sistem HRIS Panca Budi**

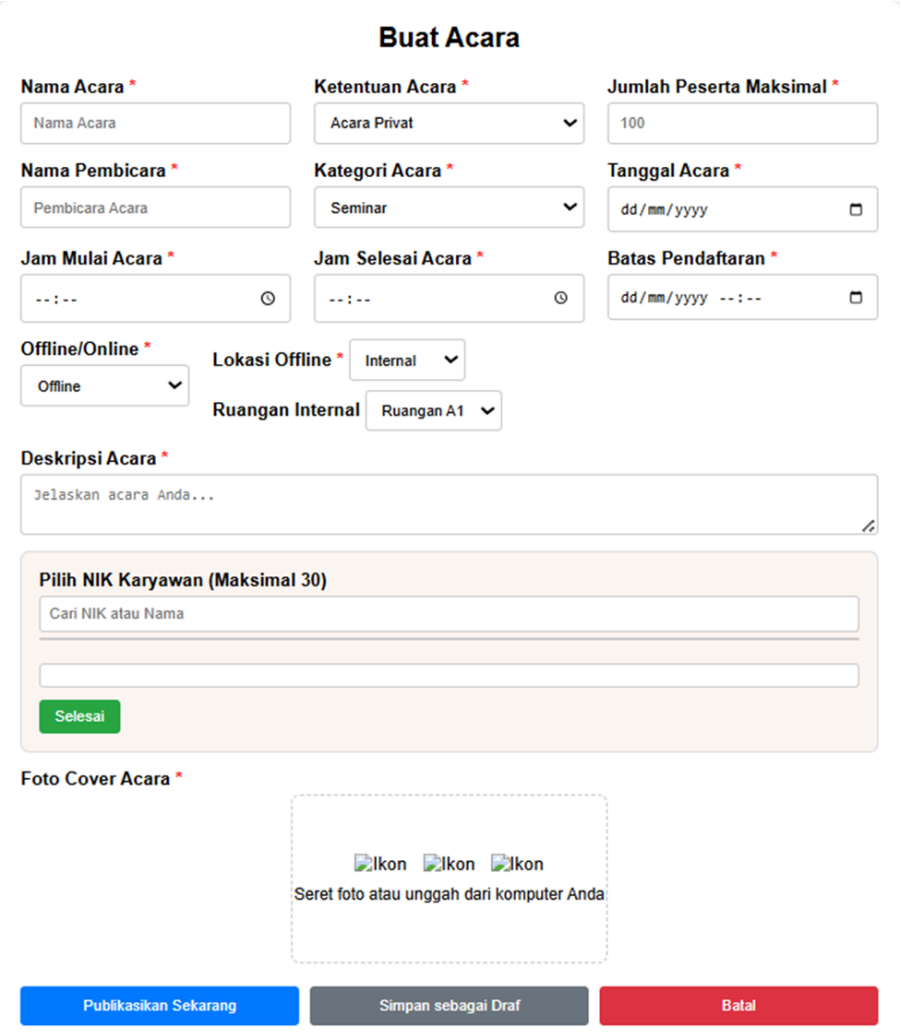
Sebagai Front-End Programmer, kontribusi dilanjutkan dengan merancang dan mengimplementasikan antarmuka pengguna (UI) baru untuk fitur "Create Event" di sistem HRIS Panca Budi. Pengerjaan ini dilakukan setelah URD terkait fitur ini selesai dan disetujui, sehingga desain UI dapat secara akurat merepresentasikan kebutuhan fungsional yang telah didefinisikan..

Tahapan Pembuatan UI:

- Interpretasi URD dan Desain Awal: Interpretasi spesifikasi dari Dokumen URD, khususnya bagian Transaksi "Create Event", dilakukan. Berdasarkan kebutuhan tersebut, desain awal UI dibuat dengan konsep yang masih mentah.

- Prototyping dan Kolaborasi dengan Figma: **Figma** digunakan untuk membuat prototipe interaktif dari tampilan "Create Event". Dalam fase ini, kolaborasi dengan System Analyst (peran yang juga diemban) atau *UI/UX designer* lain, serta *back-end programmer* terkait, dilakukan untuk mengumpulkan *feedback* dan melakukan iterasi desain. Hal ini memastikan UI tidak hanya estetis tetapi juga fungsional dan mudah digunakan.
- Implementasi Kode dengan HTML, CSS, JavaScript (jQuery) di VSCode: Setelah prototipe desain final disetujui, implementasi kode dilakukan menggunakan **VSCode** sebagai *Integrated Development Environment* (IDE) utama.
    - Struktur halaman dan elemen-elemen *form* (seperti *input field* untuk nama acara, pembicara, tanggal, waktu, kapasitas, serta pilihan *online/offline*) dibangun menggunakan **HTML**.
    - Aspek visual dan tata letak, termasuk *styling* tombol, penataan *form*, dan responsivitas, diatur menggunakan **CSS**. Fokus diberikan pada menciptakan tampilan yang bersih, modern, dan konsisten dengan *design system* perusahaan.
    - Fungsionalitas interaktif, seperti validasi *input*, efek dinamis saat memilih opsi *online/offline* (misalnya menampilkan *field* untuk lokasi atau *link*), serta interaksi dengan elemen *form* lainnya, diimplementasikan menggunakan **JavaScript**. Dalam hal ini, **jQuery** dimanfaatkan untuk mempermudah manipulasi DOM dan penanganan *event*, mempercepat proses pengembangan interaktivitas UI.
- Integrasi Awal dan Pengujian Fungsionalitas UI: Meskipun integrasi penuh dengan *back-end* dilakukan oleh tim *developer* yang lebih luas, pengujian fungsionalitas sisi *front-end* secara

mandiri dilakukan. Ini termasuk memastikan semua *field input* berfungsi dengan baik, validasi *client-side* berjalan sesuai harapan, dan elemen interaktif merespons dengan benar.



Gambar 3.12 Tampilan Antarmuka Buat Acara (Create Event)

Berikut adalah cuplikan kode yang menunjukkan implementasi *front-end* dari tampilan "Create Event":

createeevent.html > ...

```html
<!DOCTYPE html>
<html lang="id">
<head>
  <meta charset="UTF-8">
  <title>Buat Acara</title>
  <!-- Hubungkan ke file CSS eksternal -->
  <link rel="stylesheet" href="styles/styles.css">

  <script>
    /* ===== Variabel Global ===== */
    let selectedEmployees = [];
    let eventImageBase64 = "";
    // Array multi-date: setiap item adalah { date, startTime, endTime }
    let selectedDates = [];

    /* ===== Fungsi DRAG & DROP Foto Cover ===== */
    function handleDragOver(e) {
      e.preventDefault();
      document.getElementById("drop-area").classList.add("drag-over");
    }
    function handleDragLeave(e) {
      e.preventDefault();
      document.getElementById("drop-area").classList.remove("drag-over");
    }
    function handleDrop(e) {
      e.preventDefault();
      document.getElementById("drop-area").classList.remove("drag-over");
      const files = e.dataTransfer.files;
      if (files && files.length > 0) {
        processFile(files[0]);
      }
    }
    function triggerBrowse() {
      document.getElementById("event-image").click();
    }
    function handleFileSelect(e) {
      const file = e.target.files[0];
      if (file) {
        processFile(file);
      }
    }
    function processFile(file) {
      const reader = new FileReader();
      reader.onload = function(e) {
        eventImageBase64 = e.target.result;
      };
      reader.readAsDataURL(file);
    }
```

Gambar 3.13 Cuplikan Kode HTML, CSS, dan JavaScript untuk UI "Create Event"

4. **Membuat tampilan untuk membuat promo dalam Odoo**

Pada proyek ini, peran sebagai Front-End Programmer diemban untuk mengembangkan antarmuka pengguna (UI) bagi fitur "Promotion Program" di dalam sistem ERP Odoo yang digunakan oleh PT Panca Budi Tbk. Proyek ini bertujuan untuk memungkinkan pengguna internal, seperti tim pemasaran atau penjualan, untuk mendefinisikan dan mengelola berbagai jenis program promosi secara mandiri melalui UI yang intuitif.

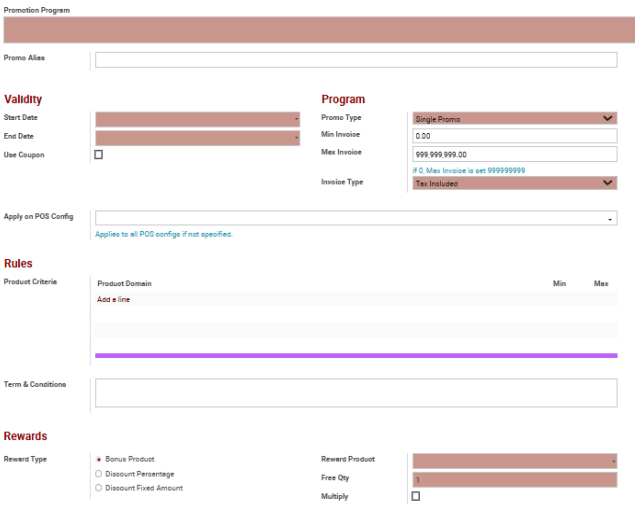
Tahapan Pembuatan UI:

- Pemahaman Modul Odoo dan Kebutuhan Bisnis: Pemahaman struktur dan fungsionalitas modul "Promosi" yang ada di Odoo dilakukan, serta koordinasi dengan tim *marketing* untuk mengidentifikasi kebutuhan spesifik dalam pembuatan program promosi. Ini mencakup parameter seperti jenis promosi (diskon, bonus produk), periode validitas, kriteria produk, dan ketentuan.
- Desain UI/UX dengan Figma: Berbekal pemahaman kebutuhan dan struktur Odoo, **Figma** digunakan untuk merancang *mockup* dan prototipe tampilan "Promotion Program". Desain ini mencakup elemen-elemen *form* untuk *input* data promosi, seperti "Promo Alias", "Validity" (tanggal mulai dan berakhir), "Program" (tipe promosi, minimal/maksimal invoice), "Rules" (kriteria produk), dan "Rewards" (jenis *reward*). Penekanan diberikan pada tata letak yang bersih dan navigasi yang logis agar pengguna dapat dengan mudah mengkonfigurasi promosi kompleks.
- Implementasi Kode pada Platform Odoo: Implementasi UI dilakukan dengan menyesuaikan *template* dan struktur yang sudah ada di Odoo. Meskipun Odoo memiliki kerangka kerja sendiri (misalnya QWeb untuk *templating* dan JavaScript yang

terintegrasi), kemampuan **HTML** untuk struktur dasar elemen, **CSS** untuk *styling* agar tampilan selaras dengan *look and feel* Odoo, serta **JavaScript (jQuery)** untuk menambahkan interaktivitas dinamis pada *form* dimanfaatkan. Contoh interaktivitas meliputi perubahan *field* berdasarkan pilihan "Promo Type" atau "Reward Type".

- Pengujian Fungsionalitas UI: Pengujian terhadap UI yang dikembangkan dilakukan untuk memastikan semua *field input* berfungsi dengan benar, validasi *client-side* berjalan, dan elemen interaktif merespons seperti yang diharapkan. Hal ini memastikan pengalaman pengguna yang lancar sebelum diintegrasikan lebih lanjut dengan logika *back-end* Odoo.

Melalui proyek ini, pemahaman lebih dalam tentang pengembangan *front-end* dalam ekosistem ERP yang terintegrasi seperti Odoo diperoleh, serta kemampuan untuk menerjemahkan kebutuhan bisnis menjadi solusi UI yang praktis.



Gambar 3.14 User Interface Create Promotion in Odoo

5. **Membuat Modul - Modul dalam aplikasi Pricing Management System (PMS)**

Pada proyek ini, kontribusi diberikan dalam pengembangan berbagai modul inti pada aplikasi Pricing Management System (PMS). Sistem ini dirancang untuk mengelola dan mengotomatisasi proses penetapan harga layanan logistik dan ekspedisi. Keterlibatan mencakup peran ganda sebagai *System Analyst* untuk memahami dan mendefinisikan kebutuhan fungsional setiap modul, serta sebagai *Front-End Programmer* dalam merancang dan mengimplementasikan antarmuka pengguna. Lingkungan pengembangan utama yang digunakan adalah **VSCode** dengan kombinasi **HTML** untuk struktur dasar, **CSS** untuk pengaturan tata letak dan responsivitas, serta **JavaScript (jQuery)** untuk implementasi fungsionalitas interaktif.

Modul-Modul yang Dikerjakan:

- **Master Moda:** Modul ini berfungsi untuk mendefinisikan berbagai moda transportasi (darat, laut, udara) yang digunakan dalam perhitungan biaya distribusi. Sebagai *System Analyst*, analisis kebutuhan *user* untuk mencatat informasi detail seperti jenis moda, golongan kendaraan, periode bahan bakar, serta rasio efisiensi konsumsi bahan bakar dilakukan. Dari sisi *Front-End*, desain UI yang *simple* namun *powerful* dirancang agar Admin dapat dengan mudah menambah, mengedit, atau menghapus data moda. Antarmuka ini mencakup *input field* untuk 'Moda', *dropdown* untuk 'Golongan' dan 'Periode BBM', serta *input* numerik untuk 'Ratio (KM/Ltr)'. Fitur *auto-numbering* pada 'Kode Moda' juga diimplementasikan untuk memfasilitasi pengelolaan data. Tujuan modul ini adalah menyediakan data moda transportasi yang akurat dan terstandardisasi sebagai dasar perhitungan harga.

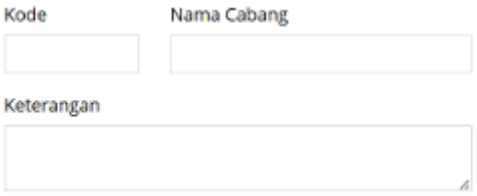
Gambar 3.15 User Interface Master Moda

- **Master Gerbang Tol:** Modul ini dikembangkan untuk mendefinisikan tarif tol berdasarkan asal perjalanan, gerbang masuk, dan golongan kendaraan. Sebagai System Analyst, data dikumpulkan dan kebutuhan diidentifikasi untuk mencatat informasi seperti nama tol masuk ('Tol'), nama tol keluar ('Gerbang Tol'), pilihan golongan kendaraan, tanggal mulai berlaku tarif, dan nilai tarif tol. Dari sisi *Front-End*, UI dirancang agar *user* dapat dengan mudah menginput data-data tersebut. Interaksi diimplementasikan menggunakan JavaScript/jQuery untuk memastikan *field input* sesuai dengan format yang dibutuhkan dan memberikan *feedback* visual yang jelas kepada *user*. Modul ini esensial untuk memastikan perhitungan biaya distribusi yang akurat dan transparan.

Gambar 3.16 User Interface Master Gerbang Tol

- **Master Cabang:** Modul Master Cabang ini digunakan untuk mencatat seluruh data cabang yang dimiliki oleh PT Panca Budi Logistindo. Kontribusi diberikan dalam pengembangan *front-end*

modul ini. Antarmuka penggunanya mencakup *field input* untuk 'Kode' cabang, 'Nama Cabang', serta 'Keterangan'. Pentingnya modul ini juga terletak pada fungsinya sebagai referensi utama dalam pengelolaan hak akses pengguna *login* berdasarkan area kerja masing-masing cabang, sehingga setiap *user* hanya memiliki akses data sesuai dengan cabang tempatnya terdaftar. Desain UI dioptimalkan untuk kesederhanaan dan fungsionalitas, mempermudah Admin dalam mengelola data cabang.

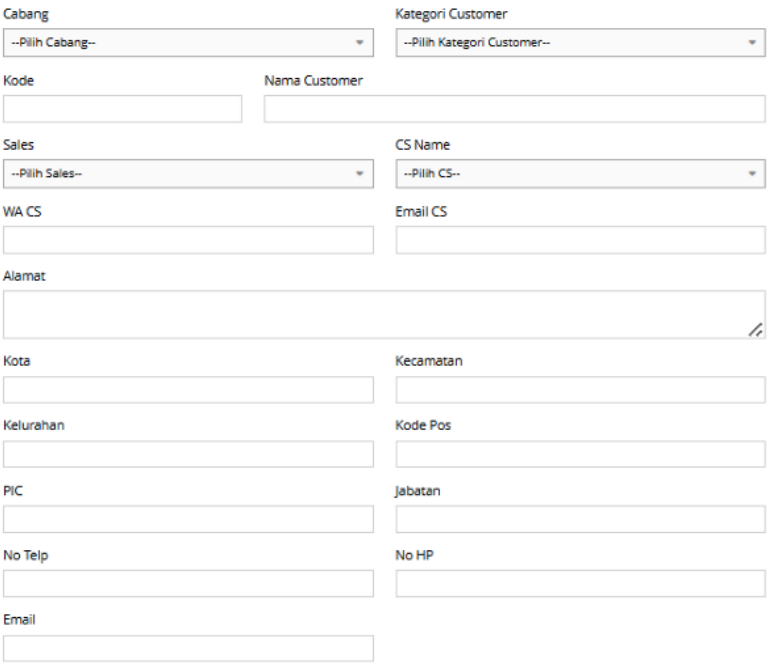


Gambar 3.17 User Interface Master Cabang

- **Master Sales:** Modul Master Sales ini digunakan untuk mencatat dan mengelola data karyawan *sales* di masing-masing cabang. Bantuan diberikan dalam perancangan UI yang memungkinkan pengguna untuk menambah data *sales* baru, mengedit, atau menghapusnya. Antarmuka ini terdapat *field input* untuk 'Kode' *sales*, 'Nama Sales', serta *dropdown* untuk memilih 'Cabang' (data diambil dari Master Cabang), dan 'Keterangan'. UI dirancang untuk mempercepat proses pengisian data oleh pengguna, dan integrasi dengan Master Cabang menggunakan JavaScript/jQuery memastikan validitas data yang dipilih.

Gambar 3.18 User Interface Master Sales

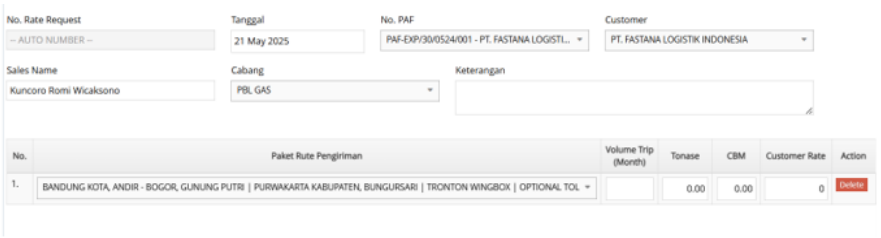
- **Master Customer:** Modul Master Customer ini berfungsi untuk mencatat informasi detail pelanggan yang bekerja sama dengan perusahaan. Sebagai *System Analyst*, analisis dan definisi berbagai *field* yang diperlukan dilakukan, seperti informasi cabang terkait, penanggung jawab *sales* dan *customer service* (CS), serta identitas lengkap pelanggan termasuk alamat, kontak, dan kategori bisnis. Dari perspektif *Front-End*, UI yang komprehensif namun tetap intuitif dibangun, dengan berbagai tipe *input field* dan *dropdown* yang terhubung ke master data lain. Tujuan modul ini adalah menyediakan basis data pelanggan yang lengkap dan terorganisir untuk mendukung proses penawaran harga dan layanan.



Gambar 3.19 User Interface Master Customer

- **Rate Request (Permintaan Harga):** Modul ini merupakan fitur transaksi krusial dalam Pricing Management System (PMS), berfungsi untuk memfasilitasi proses permintaan harga layanan

logistik dari *user*, baik itu internal tim maupun dari pelanggan. Sebagai *System Analyst*, analisis alur kerja permintaan harga dari inisiasi hingga tahap peninjauan dan respons dilakukan. Kebutuhan yang diidentifikasi mencakup: detail pengirim/penerima, jenis barang, dimensi/berat/volume, rute (asal-tujuan), moda transportasi yang diinginkan, dan *service level*. Dari sisi *Front-End*, UI dirancang dan diimplementasikan yang memungkinkan *user* untuk mengisi formulir permintaan harga dengan mudah dan jelas. Antarmuka ini mencakup *field input* untuk 'No. Rate Request' (otomatis), 'Tanggal', 'No. PAF', 'Customer', 'Sales Name', 'Cabang', dan 'Keterangan'. Fitur utama adalah kemampuan untuk menambahkan 'Paket Rute Pengiriman' secara dinamis, di mana *user* dapat memasukkan detail rute, Volume Trip, Tonase, CBM, dan Customer Rate.



Gambar 3.20 User Interface Transaction Rate Request

6. Merancang Modul Pendaftaran Aset dalam Odoo

Dalam peran ganda sebagai System Analyst dan Front-End Programmer, kontribusi diberikan dalam perancangan dan tahap awal implementasi modul pendaftaran aset di dalam sistem ERP Odoo yang digunakan oleh PT Panca Budi Tbk. Modul ini dirancang untuk mendigitalisasi dan menyederhanakan proses pencatatan serta pengelolaan aset perusahaan secara terpusat, mulai dari fase akuisisi hingga perhitungan depresiasi dan penarikan aset.

- **Sebagai System Analyst:**
    - **Analisis Kebutuhan Bisnis Mendalam:** Melakukan serangkaian wawancara dan diskusi terstruktur dengan pemangku kepentingan utama dari departemen yang relevan, seperti departemen keuangan (akuntansi aset), logistik (manajemen inventaris aset fisik), dan departemen operasional (pengguna aset). Tujuan dari analisis ini adalah untuk secara komprehensif memahami alur kerja pendaftaran aset manual yang saat ini berlaku, mengidentifikasi *pain points* (titik-titik masalah) yang sering muncul, serta mengumpulkan daftar lengkap kebutuhan fungsional dan non-fungsional dari pengguna. Informasi kunci yang perlu dicatat untuk setiap aset diidentifikasi secara spesifik, termasuk, namun tidak terbatas pada, jenis aset (misalnya, kendaraan, mesin, peralatan kantor), tanggal akuisisi, nilai perolehan awal, lokasi penempatan aset yang tepat, nama penanggung jawab aset, estimasi masa manfaat, metode depresiasi yang digunakan (garis lurus, saldo menurun, dll.), dan nomor seri unik atau identifikasi aset lainnya.
    - **Pembuatan Spesifikasi Fungsional Rinci:** Menyusun dokumen spesifikasi fungsional yang terperinci (*Functional Specification Document*) yang secara jelas menjabarkan setiap fitur yang dibutuhkan oleh modul aset ini. Ini mencakup rincian struktur formulir input data aset yang akan dikembangkan, fungsionalitas pencarian dan filter aset yang fleksibel (berdasarkan lokasi, jenis, status, dll.), tampilan daftar aset yang mudah dipahami, serta mempertimbangkan potensi dan mekanisme integrasi dengan modul akuntansi yang sudah ada di Odoo untuk

otomatisasi pencatatan jurnal depresiasi atau pelepasan aset.

- **Perancangan Alur Proses (Workflow Design):** Membuat diagram alur proses (*process flowcharts*) yang visual dan jelas untuk menggambarkan seluruh *workflow* pendaftaran aset. Alur ini mencakup langkah-langkah mulai dari pengajuan pembelian aset, verifikasi, persetujuan, hingga pencatatan aset sebagai aset tetap dalam sistem. Selain itu, alur untuk pembaruan data aset (misalnya perubahan lokasi atau penanggung jawab) dan proses penghapusan aset (misalnya penjualan atau *disposal*) juga dirancang untuk memastikan kelengkapan dan konsistensi data.

- **Sebagai Front-End Programmer:**
  - **Desain UI/UX Kontekstual:**
    - Merancang *wireframe* awal dan *mockup* UI untuk tampilan utama modul aset, seperti halaman *"Create Asset"* (untuk memasukkan data aset baru) dan *"Asset List"* (untuk melihat daftar aset yang ada), menggunakan *tool* desain seperti Figma. Perancangan ini mempertimbangkan aspek konsistensi dengan *design system* Odoo yang sudah ada, memastikan antarmuka terasa *native* dan intuitif bagi pengguna Odoo. Fokus diberikan pada kemudahan navigasi, minimasi *error* saat *input* data, dan efisiensi alur kerja pengguna.
  - **Implementasi Komponen UI Dasar:**
    - Mengimplementasikan struktur dasar *form* pendaftaran aset menggunakan HTML dan CSS yang sesuai dengan kerangka kerja Odoo (misalnya, QWeb untuk *templating*). Tugas ini mencakup pembuatan elemen-elemen seperti *dropdown* untuk

pemilihan jenis aset, komponen *date picker* untuk tanggal akuisisi, *input field* numerik dengan validasi untuk nilai perolehan dan masa manfaat, serta *text area* untuk deskripsi aset yang lebih panjang.

- **Eksplorasi Integrasi dengan Modul Odoo:**
  - Melakukan eksplorasi awal bagaimana komponen UI yang dibangun dapat diintegrasikan secara efektif ke dalam arsitektur modul Odoo yang lebih luas. Ini meliputi pemahaman tentang model data yang digunakan Odoo untuk aset, serta bagaimana *controller* Odoo memproses data dari *front-end* ke *back-end*. Langkah ini penting untuk memastikan kompatibilitas dan fungsionalitas yang mulus dalam ekosistem Odoo.

7. **Melakukan Analisis Gap pada Aplikasi Akuntansi PT Panca Budi**

Dalam peran sebagai System Analyst, tugas ini berfokus pada analisis mendalam dan sistematis terhadap aplikasi akuntansi yang sedang berjalan di PT Panca Budi Tbk. Tujuan utama dari analisis ini adalah untuk mengidentifikasi ***gap*** atau kesenjangan yang signifikan antara fungsionalitas dan kinerja sistem akuntansi yang saat ini digunakan dengan kebutuhan bisnis yang terus berkembang, serta praktik akuntansi terbaik dan standar kepatuhan terbaru yang berlaku di industri.

- **Fase Pengumpulan Data Komprehensif:**
  - **Wawancara Terstruktur dengan Pemangku Kepentingan:**
    - Melakukan serangkaian wawancara terstruktur dan tidak terstruktur dengan pengguna kunci dari departemen akuntansi dan keuangan, termasuk

manajer keuangan, akuntan, dan staf terkait lainnya. Wawancara ini bertujuan untuk memperoleh pemahaman mendalam tentang alur kerja harian mereka, mengidentifikasi tantangan operasional yang sering mereka hadapi dengan sistem saat ini (misalnya, proses manual yang memakan waktu, kesulitan dalam pelaporan, *error* data), serta mengumpulkan daftar lengkap fitur-fitur baru atau peningkatan yang diinginkan untuk mendukung operasional yang lebih efisien.

- **Tinjauan Dokumentasi Sistem Eksisting:**
  - Mempelajari secara cermat seluruh dokumentasi teknis dan operasional yang ada untuk aplikasi akuntansi, termasuk panduan pengguna, spesifikasi fungsional, diagram arsitektur sistem, dan diagram basis data (jika tersedia). Tinjauan ini membantu memahami batasan dan kapabilitas sistem saat ini.
- **Observasi Langsung Proses Bisnis:**
  - Melakukan observasi langsung terhadap bagaimana transaksi akuntansi diproses sehari-hari menggunakan sistem yang ada. Ini mencakup pengamatan dari tahap *input* data (misalnya, faktur pembelian, penjualan), proses posting jurnal, pengelolaan buku besar, rekonsiliasi akun, hingga proses pembuatan laporan keuangan (laporan laba rugi, neraca, arus kas). Observasi ini memberikan wawasan praktis tentang *workflow* dan interaksi pengguna dengan sistem.

- **Fase Analisis Sistem Saat Ini (*As-Is Analysis*):**
    - **Pemetaan Proses Bisnis Akuntansi:**
        - Membuat diagram alur proses (*process flowcharts*) yang detail dan jelas untuk seluruh operasional akuntansi yang didukung oleh aplikasi saat ini. Pemetaan ini mencakup setiap langkah dalam pencatatan jurnal, pengelolaan buku besar, proses piutang dan hutang, manajemen aset tetap, serta siklus pelaporan keuangan. Tujuannya adalah untuk memvisualisasikan bagaimana proses-proses ini secara aktual berjalan di sistem.
    - **Identifikasi Fungsionalitas Aplikasi:**
        - Mendokumentasikan secara rinci fungsionalitas yang tersedia pada aplikasi akuntansi yang sedang berjalan. Ini mencakup daftar modul-modul yang ada (misalnya, General Ledger, Accounts Payable, Accounts Receivable, Fixed Assets), fitur-fitur utama yang disediakan oleh setiap modul, serta jenis-jenis laporan keuangan dan operasional yang dapat dihasilkan oleh sistem.
- **Fase Definisi Kebutuhan Ideal (*To-Be Requirements*):**
    - ***Benchmarking* dengan Praktik Terbaik Industri:**
        - Melakukan riset dan *benchmarking* terhadap praktik akuntansi terbaik dalam industri manufaktur dan distribusi, serta menganalisis fitur-fitur standar yang ada pada sistem ERP akuntansi modern yang terkemuka. Ini membantu menetapkan standar fungsionalitas yang diinginkan.
    - **Penyusunan Kebutuhan Fungsional dan Non-Fungsional Baru:**

        - Berdasarkan analisis *As-Is* dan *benchmarking*, mengidentifikasi dan menyusun daftar kebutuhan fungsional dan non-fungsional yang belum terpenuhi oleh sistem akuntansi saat ini. Contoh kebutuhan fungsional meliputi integrasi yang lebih baik dengan modul lain (misalnya manajemen persediaan, penjualan, atau penggajian), otomatisasi proses rekonsiliasi bank, kemampuan *drill-down* laporan keuangan, atau fitur untuk mendukung pelaporan analitis yang lebih canggih. Kebutuhan non-fungsional dapat mencakup peningkatan kinerja sistem, keamanan data yang lebih kuat, atau skalabilitas untuk mendukung pertumbuhan perusahaan.

- **Fase Analisis Kesenjangan (*Gap Analysis*) Mendalam:**
    - **Perbandingan Sistematis *As-Is* dan *To-Be*:**
        - Melakukan perbandingan secara sistematis antara fungsionalitas dan proses yang ada pada sistem akuntansi saat ini (*As-Is*) dengan daftar kebutuhan ideal (*To-Be Requirements*) yang telah disusun.
    - **Identifikasi *Gap* Spesifik:**
        - Mengidentifikasi secara spesifik dan terperinci area-area di mana sistem saat ini tidak memenuhi kebutuhan atau menghambat efisiensi operasional. *Gap* ini dapat berupa kurangnya fitur krusial (misalnya, tidak adanya modul *fixed asset* yang memadai), proses yang masih bersifat manual dan rawan kesalahan, masalah kinerja sistem yang menyebabkan keterlambatan, atau ketidaksesuaian dengan standar akuntansi terkini (misalnya, IFRS atau PSAK baru).

- **Analisis Dampak *Gap*:**
  - Menganalisis potensi dampak dari setiap *gap* yang teridentifikasi terhadap operasional bisnis PT Panca Budi Tbk. Dampak ini dapat berupa peningkatan biaya operasional, risiko kesalahan finansial yang lebih tinggi, keterlambatan dalam penyusunan laporan keuangan, atau hambatan dalam pengambilan keputusan strategis yang berbasis data.

- **Fase Rekomendasi Awal yang Terukur:**
  - Berdasarkan hasil analisis *gap* yang komprehensif, merumuskan rekomendasi awal yang konkret dan terukur untuk perbaikan atau pengembangan sistem akuntansi. Rekomendasi ini dapat mencakup peningkatan fitur yang ada, modifikasi proses bisnis untuk mengoptimalkan penggunaan sistem, atau bahkan pertimbangan untuk evaluasi dan implementasi sistem baru (misalnya upgrade ERP atau implementasi modul baru di Odoo) jika *gap* yang ditemukan bersifat signifikan dan strategis. Rekomendasi akan disertai dengan justifikasi bisnis dan perkiraan manfaat yang dapat diperoleh

8. **Merancang URD untuk Pembuatan Peminjaman Truk di PT Panca Budi Logistindo**

   Dalam peran sebagai System Analyst, tugas ini melibatkan perancangan dokumen *User Requirement Document* (URD) yang komprehensif untuk sistem peminjaman truk di PT Panca Budi Logistindo (PBL). Sistem ini bertujuan untuk mendigitalisasi dan mengoptimalkan proses manajemen peminjaman truk internal, dari pengajuan hingga pengembalian.

- **Fase Pengumpulan Kebutuhan:**
  - **Wawancara dengan Departemen Terkait:**

        - Melakukan wawancara terstruktur dengan berbagai departemen yang terlibat dalam proses peminjaman truk, seperti departemen logistik (yang mengelola armada truk), departemen operasional (pengguna truk), dan departemen keuangan (untuk pencatatan biaya dan aset). Fokus pada pemahaman alur kerja peminjaman truk saat ini, termasuk bagaimana permintaan diajukan, bagaimana persetujuan diberikan, bagaimana penjadwalan dilakukan, dan bagaimana penggunaan serta pengembalian truk dicatat.
    - **Tinjauan Proses Manual/Eksisting:**
        - Menganalisis dokumen, formulir, atau catatan manual yang digunakan dalam proses peminjaman truk saat ini. Mengidentifikasi informasi penting yang dicatat, *bottleneck* dalam proses manual, dan potensi risiko yang ada.
    - **Identifikasi Kebutuhan Fungsional:**
        - Mengumpulkan dan mendokumentasikan kebutuhan fungsional yang spesifik untuk sistem peminjaman truk, seperti:
        - Kemampuan untuk mengajukan permohonan peminjaman truk (tanggal, waktu, tujuan, jenis barang, supir, asisten supir).
        - Proses persetujuan atau penolakan permohonan oleh manajer yang berwenang.
        - Fungsionalitas penjadwalan truk yang tersedia.
        - Pencatatan detail penggunaan truk (jarak tempuh, konsumsi bahan bakar, biaya operasional).
        - Pencatatan kondisi truk sebelum dan sesudah peminjaman.

        - Manajemen *master data* truk (jenis, kapasitas, nomor polisi, status ketersediaan).
        - Fitur pelaporan riwayat peminjaman truk.
    - **Identifikasi Kebutuhan Non-Fungsional:**
        - Menentukan kebutuhan non-fungsional, seperti kinerja sistem (kecepatan respon), keamanan data (hak akses pengguna), ketersediaan, dan *usability* (kemudahan penggunaan antarmuka).
- **Fase Perancangan Dokumen URD:**
    - **Definisi Tujuan dan Ruang Lingkup Sistem:** Menjelaskan secara jelas mengapa sistem peminjaman truk ini dibutuhkan dan batasan fungsionalitasnya.
    - **Definisi Aktor Sistem:** Mengidentifikasi semua entitas yang akan berinteraksi dengan sistem (misalnya, Pengaju Peminjaman, Supervisor Logistik, Admin Truk, Kepala Departemen Keuangan).
    - **Pembuatan Diagram Alur Proses (*Flowchart*)/*Use Case Diagram*:** Memvisualisasikan *workflow* peminjaman truk dalam bentuk diagram, menunjukkan langkah-langkah, keputusan, dan interaksi antar aktor. *Use Case Diagram* dapat digunakan untuk menggambarkan fungsionalitas utama dari perspektif pengguna.
    - **Deskripsi Fungsionalitas Detail:** Menjabarkan setiap fungsi secara rinci, termasuk *input* yang dibutuhkan, proses yang terjadi, dan *output* yang dihasilkan. Misalnya, untuk fungsi "Pengajuan Peminjaman Truk", akan dijelaskan *field* yang harus diisi, validasi data, dan notifikasi yang terkirim.
    - **Perancangan *Wireframe* / *Mockup* Awal:** Membuat sketsa awal antarmuka pengguna untuk modul-modul kunci (misalnya, formulir pengajuan, daftar truk tersedia, riwayat

peminjaman) untuk memberikan gambaran visual kepada *stakeholder* dan mempermudah validasi.

- **Manajemen Data:** Mendefinisikan entitas data yang diperlukan (misalnya, data truk, data pengemudi, data peminjaman) dan atribut-atributnya.
- **Kebutuhan Laporan:** Menentukan jenis laporan yang dibutuhkan dari sistem, seperti laporan penggunaan truk bulanan, laporan ketersediaan truk, atau laporan biaya operasional per truk.

- **Fase Validasi URD:**
  - Mempresentasikan dokumen URD kepada pemangku kepentingan untuk mendapatkan *feedback* dan persetujuan. Melakukan revisi berdasarkan masukan untuk memastikan dokumen URD mencerminkan kebutuhan bisnis secara akurat dan lengkap sebelum berlanjut ke tahap desain teknis.

9. Membuat Report Sales dari Odoo

Dalam peran sebagai Front-End Programmer dan System Analyst, kontribusi diberikan dalam pengembangan dan penyesuaian laporan penjualan (*Sales Report*) di dalam sistem ERP Odoo, khususnya untuk menampilkan data target dan pencapaian penjualan per tim dan *salesman* seperti yang terlihat pada "Sales Target Report". Proyek ini bertujuan untuk menyediakan visualisasi data penjualan yang akurat dan mudah diinterpretasi bagi manajemen untuk pengambilan keputusan strategis.

- **Sebagai System Analyst:**
  - **Analisis Kebutuhan Laporan:** Melakukan diskusi dengan tim *sales* dan manajemen untuk memahami kebutuhan spesifik laporan penjualan. Ini mencakup identifikasi metrik kunci yang harus ditampilkan (misalnya, *Target*,

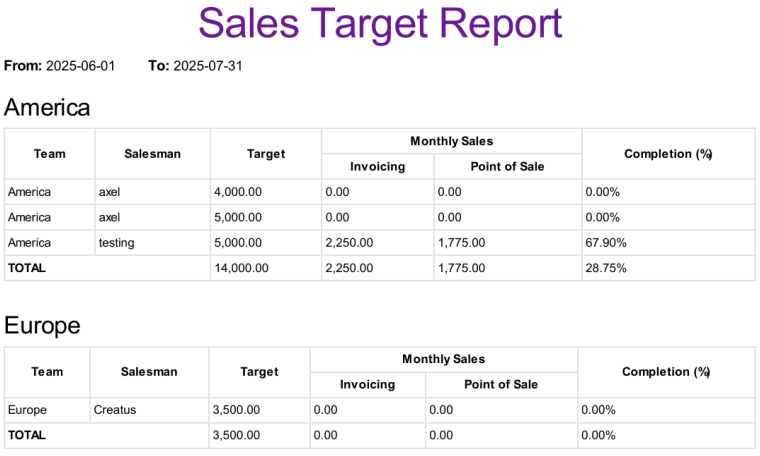
Monthly Sales yang terdiri dari *Invoicing* dan *Point of Sale*, dan *Completion Percentage*), serta bagaimana data harus dikelompokkan (per *Team* dan *Salesman*) dan difilter (berdasarkan periode waktu seperti *From: 2025-06-01 To: 2025-07-31*). Memahami sumber data di Odoo (modul penjualan, akuntansi, atau data *custom*) yang relevan untuk setiap metrik.

- **Perancangan Logika Laporan:** Mendefinisikan logika perhitungan untuk setiap metrik, terutama untuk *Completion (%)* (misalnya, `(Invoicing + Point of Sale) / Target * 100%`). Memastikan bagaimana data dari berbagai sumber di Odoo akan diakumulasikan dan disajikan.
- **Pembuatan Spesifikasi Laporan:** Menyusun spesifikasi laporan yang merinci struktur data, format tampilan, kriteria *filtering*, dan logika perhitungan.

- **Sebagai Front-End Programmer:**
  - **Identifikasi dan Penyesuaian *View* Odoo:** Mengidentifikasi *view* atau *template* laporan yang paling sesuai di Odoo yang dapat disesuaikan. Jika tidak ada yang cocok, merancang *view* baru.
  - **Implementasi Struktur Laporan (QWeb/XML/HTML):** Menggunakan *template engine* Odoo (QWeb) atau kombinasi XML/HTML untuk membangun struktur tabel laporan sesuai dengan kebutuhan. Memastikan baris dan kolom laporan terdefinisi dengan baik untuk menampilkan *Team*, *Salesman*, *Target*, *Invoicing*, *Point of Sale*, dan *Completion (%)*.
  - **Pemformatan Data dan *Styling* (CSS):** Menerapkan CSS untuk memformat tampilan laporan agar mudah dibaca, termasuk *styling* untuk *header* tabel, baris data, dan total.

Memastikan data numerik diformat dengan benar (misalnya, dua angka di belakang koma untuk persentase).

- **Implementasi Logika Tampilan (JavaScript/jQuery):** Menggunakan JavaScript atau jQuery yang terintegrasi di Odoo untuk menambahkan fungsionalitas dinamis jika diperlukan, seperti *filtering* berdasarkan tanggal atau *refresh* data. Jika perhitungan *Completion (%)* tidak dilakukan di *back-end*, logika ini dapat diimplementasikan di *front-end*.
- **Pengujian Tampilan Laporan:** Melakukan pengujian untuk memastikan bahwa laporan menampilkan data dengan benar, perhitungan akurat, dan *filtering* berfungsi sebagaimana mestinya. Membandingkan output laporan dengan data sumber untuk validasi. Kontribusi dalam proyek ini meningkatkan pemahaman tentang bagaimana data bisnis dari Odoo dapat diekstraksi dan disajikan dalam format laporan yang informatif untuk mendukung analisis kinerja penjualan.



Sales Target Report

From: 2025-06-01 **To:** 2025-07-31

America

Team	Salesman	Target	Monthly Sales		Completion (%)
			Invoicing	Point of Sale	
America	axel	4,000.00	0.00	0.00	0.00%
America	axel	5,000.00	0.00	0.00	0.00%
America	testing	5,000.00	2,250.00	1,775.00	67.90%
TOTAL		14,000.00	2,250.00	1,775.00	28.75%

Europe

Team	Salesman	Target	Monthly Sales		Completion (%)
			Invoicing	Point of Sale	
Europe	Creatus	3,500.00	0.00	0.00	0.00%
TOTAL		3,500.00	0.00	0.00	0.00%

Gambar 3.21 User Interface Sales Target Report (Odoo)

3.3 Kendala yang Ditemukan

Selama menjalani program magang di PT Panca Budi Tbk, beberapa tantangan signifikan dan kompleks ditemui, terutama dalam hal adaptasi dengan keragaman aplikasi dan sistem informasi yang harus dikerjakan dalam lingkungan industri. Karena masih tergolong baru di dunia industri dan menghadapi teknologi yang spesifik, proses adaptasi ini menjadi kendala utama yang menuntut pembelajaran dan penyesuaian yang cepat serta mendalam. Kendala-kendala ini terasa di setiap proyek yang dikerjakan, dan dapat dijelaskan lebih lanjut sebagai berikut:

- **Kendala Adaptasi Pengembangan Aplikasi *Mobile* (GBN Patroli):** Pada awal magang, saat pengerjaan tampilan *login* untuk aplikasi GBN Patroli, tantangan adaptasi dengan paradigma pengembangan aplikasi *mobile* mulai teridentifikasi. Penggunaan Android Studio sebagai alat utama untuk *coding*, termasuk struktur proyek Android dan cara kerja *layout* berbasis XML, masih memerlukan familiarisasi lebih lanjut. Meskipun *template* dari perusahaan cukup membantu, proses modifikasi dan penyesuaian desain menggunakan Figma agar tampilan konsisten dan responsif di berbagai ukuran perangkat *mobile* membutuhkan pemahaman mendalam tentang desain UI/UX *mobile* yang belum sepenuhnya dikuasai. Selain itu, pengembangan logika interaksi dasar menggunakan bahasa Kotlin atau Java juga menjadi tantangan, mengingat perbedaan signifikan dengan bahasa pemrograman yang umum diajarkan di lingkungan akademik. Kesulitan juga muncul dalam *debugging* aplikasi *mobile* pada perangkat fisik atau *emulator* yang berbeda, serta pemahaman siklus hidup aktivitas Android.
- **Kendala Pembuatan URD (User Requirement Document) untuk *Workflow Event Management* HRIS Panca Budi:** Waktu beralih ke peran System Analyst, kendala utamanya adalah

menyesuaikan diri dengan cara analisis dan pendokumentasian kebutuhan yang formal dan sangat lengkap sesuai standar perusahaan. Proses ini menuntut kemampuan untuk menerjemahkan kebutuhan bisnis yang kadang masih belum jelas dan seringkali bervariasi dari berbagai *user* (misalnya dari tim HRD atau manajemen) menjadi spesifikasi fungsional dan non-fungsional yang rinci, terstruktur, tidak ambigu, dan dapat diimplementasikan. Tantangan juga muncul saat mencoba memodelkan alur proses yang rumit, yang melibatkan berbagai *stakeholder* dan cabang keputusan, ke dalam *flow chart* atau diagram alur yang mudah dimengerti dan akurat. Memastikan bahwa semua detail kebutuhan, termasuk *edge cases* dan skenario pengecualian, tidak ada yang terlewat adalah aspek krusial. Proses validasi dan mendapatkan persetujuan dari atasan atau *stakeholder* juga menuntut kemampuan komunikasi, negosiasi, dan *conflict resolution* yang perlu dilatih secara intensif.

- **Kendala Implementasi User Interface "*Create Event*" Sistem HRIS Panca Budi:** Setelah URD selesai dan disetujui, kendala berikutnya muncul di tahap implementasi tampilan "*Create Event*" yang formulirnya cukup kompleks dan memiliki banyak *field* serta validasi. Diperlukan adaptasi dengan standar *coding* HTML, CSS, dan JavaScript (jQuery) yang spesifik di lingkungan perusahaan, yang mungkin memiliki konvensi penulisan kode, penamaan variabel, dan struktur direktori yang berbeda dari praktik umum. Membiasakan diri dengan sistem desain perusahaan dan cara-cara terbaik (*best practices*) dalam menulis kode *front-end* yang rapi, modular, dan efektif di VSCode menjadi esensial. Selain itu, membuat fitur interaktif yang dinamis, seperti validasi *input real-time*, efek visual saat memilih opsi *online/offline* (misalnya menampilkan *field* untuk lokasi atau *link* berdasarkan pilihan), serta interaksi kompleks lainnya dengan elemen *form*,

membutuhkan pemahaman yang kuat tentang cara mengatur dan memanipulasi elemen di halaman *web* (manipulasi DOM) dan bagaimana cara kerja *event handling* menggunakan jQuery, yang saat itu masih dalam proses pembelajaran dan eksperimen.

- **Kendala Pemahaman Aplikasi Odoo Erp :** Kendala utama yang dihadapi selama masa magang adalah kesulitan dalam beradaptasi dengan sistem ERP Odoo. Meskipun antarmuka pengguna Odoo cukup intuitif, proses pengembangan di dalamnya memiliki struktur yang kompleks, terutama terkait relasi antarmodel data dan alur bisnis internal. Sistem Odoo belum secara menyeluruh dikenalkan dalam perkuliahan reguler, sehingga proses pemahaman harus dilakukan dari awal. Tantangan tersebut semakin besar karena modul yang dikembangkan memiliki banyak dependensi antar entitas dan memerlukan pemahaman konsep QWeb, struktur modul XML, dan pendekatan Object-Relational Mapping (ORM). Adaptasi terhadap hal ini membutuhkan waktu serta ketekunan untuk memahami alur sistem secara menyeluruh.
- **Kendala Pengembangan Tampilan "*Create Promo*" dalam Odoo:** Proyek Odoo ini memberikan kendala adaptasi yang berbeda dan cukup signifikan karena Odoo merupakan sistem ERP yang sudah matang dan memiliki arsitektur serta konvensi pengembangannya sendiri yang unik. Kurang familiar dalam navigasi, pemahaman struktur modul internal Odoo, serta bagaimana cara membuat tampilan baru bisa terintegrasi dengan baik ke kerangka kerja Odoo (seperti QWeb untuk *templating* atau JavaScript yang sudah terintegrasi di dalamnya dengan API Odoo) merupakan tantangan utama. Pembelajaran alur pengembangan di *platform* Odoo, yang sangat berbeda dengan pengembangan *website standalone* atau *mobile native*, membutuhkan kurva pembelajaran yang cepat dan kemauan untuk memahami filosofi di balik arsitektur Odoo. Tantangan lain juga ada di penyesuaian

desain UI yang sudah dibuat di Figma agar kompatibel dengan komponen UI yang telah ada di Odoo, dan memastikan semua fiturnya berjalan sesuai dengan alur bisnis promosi yang kompleks dan spesifik. Pengujian fungsionalitas dalam lingkungan Odoo yang terintegrasi penuh juga memerlukan pemahaman tentang dampak perubahan UI terhadap *back-end* sistem.

- **Kendala Pembuatan Modul-Modul dalam Aplikasi Pricing Management System (PMS):** Proyek PMS ini, yang merupakan sistem besar dengan banyak modul *master* dan transaksi yang saling terhubung, menjadi kendala adaptasi yang paling menantang. Tingkat kompleksitasnya sangat tinggi karena melibatkan data dari berbagai sumber dan interkoneksi antar modul. Kurang terbiasa mengatur kompleksitas hubungan data antar modul (misalnya data Master Moda harus terhubung ke Rate Request, atau Master Customer ke Project Assessment) dan memastikan integritas data di seluruh sistem merupakan hambatan besar. Tantangan muncul saat harus memastikan tampilan UI/UX di semua modul itu konsisten dan *user-friendly*, serta bagaimana cara membuat fitur *auto-generate* data dari satu *field* ke *field* lain di modul yang berbeda itu bisa akurat, dan bagaimana cara menangani validasi data yang ketat di setiap formulir *input* yang macam-macam (misalnya validasi numerik, format tanggal, atau validasi *dropdown* yang bergantung pada pilihan lain). Pembelajaran mendalam tentang alur bisnis logistik dan ekspedisi, termasuk terminologi spesifik dan proses operasional, juga menjadi bagian dari proses adaptasi agar dapat membuat fitur sistem yang sesuai dengan kebutuhan operasional yang sebenarnya.
- **Kendala Perancangan Modul Pendaftaran Aset dalam Odoo:** Saat merancang modul pendaftaran aset di Odoo, kendala utama adalah minimnya pengalaman sebelumnya dalam mengelola aset secara sistematis dari perspektif akuntansi dan operasional.

Diperlukan pemahaman mendalam tentang siklus hidup aset, mulai dari akuisisi, depresiasi, hingga disposisi, dan bagaimana setiap tahap ini direfleksikan dalam sistem. Tantangan muncul dalam menerjemahkan kebutuhan lintas departemen (keuangan, logistik, operasional) menjadi spesifikasi fungsional yang koheren, terutama dalam menentukan *field* data esensial yang harus dicatat dan bagaimana mereka saling berelasi. Penyesuaian dengan konvensi *customization* Odoo yang spesifik, terutama dalam mendefinisikan model data baru dan membuat tampilan (*views*) yang sesuai dengan arsitektur Odoo, juga menjadi kendala. Memastikan bahwa desain UI tidak hanya intuitif tetapi juga dapat dengan mudah diimplementasikan dalam kerangka Odoo membutuhkan eksplorasi teknis yang signifikan.

- **Kendala Analisis *Gap* pada Aplikasi Akuntansi PT Panca Budi:** Kendala terbesar dalam melakukan analisis *gap* pada aplikasi akuntansi adalah kompleksitas sistem akuntansi itu sendiri, yang seringkali melibatkan banyak modul yang saling terhubung dan logika bisnis yang rumit, ditambah lagi dengan kemungkinan adanya *legacy code* atau dokumentasi yang tidak lengkap. Kesulitan muncul dalam menggali informasi yang detail dan akurat dari *stakeholder* yang mungkin memiliki perspektif berbeda atau tidak terbiasa menjelaskan proses mereka secara teknis. Proses membandingkan sistem *As-Is* dengan kebutuhan *To-Be* memerlukan pemahaman mendalam tentang standar akuntansi dan kebutuhan bisnis PT Panca Budi Tbk secara menyeluruh, bukan hanya dari sudut pandang teknis. Mengidentifikasi *gap* yang bukan hanya fitur yang hilang tetapi juga masalah efisiensi proses atau kepatuhan regulasi menjadi tantangan. Selain itu, menyusun rekomendasi yang realistis, terukur, dan dapat diterima oleh manajemen, dengan mempertimbangkan kendala sumber daya dan anggaran, juga memerlukan kemampuan analisis bisnis yang kuat.

- **Kendala Perancangan URD untuk Peminjaman Truk di PT Panca Budi Logistindo:** Kendala utama dalam merancang URD untuk sistem peminjaman truk adalah kompleksitas operasional logistik yang melibatkan banyak variabel dan skenario. Diperlukan pemahaman detail tentang alur peminjaman truk yang ada, termasuk pengecekan ketersediaan, persetujuan multi-level, penjadwalan, dan pelaporan penggunaan. Menggali kebutuhan yang jelas dan konsisten dari berbagai departemen (misalnya logistik, operasional, dan pengemudi) yang memiliki sudut pandang berbeda merupakan tantangan. Proses memetakan *workflow* peminjaman truk ke dalam *flowchart* atau *use case diagram* yang komprehensif, mencakup semua skenario *happy path* dan *exception path*, memerlukan ketelitian tinggi. Kesulitan juga muncul dalam mendefinisikan secara tepat *master data* yang dibutuhkan (misalnya data truk, supir, rute) dan bagaimana data tersebut akan diintegrasikan serta divalidasi dalam sistem untuk memastikan akurasi dan integritas. Perancangan UI/UX awal yang intuitif namun dapat mengakomodasi semua kebutuhan fungsional juga menjadi bagian dari tantangan ini.

3.4 Solusi atas Kendala yang Ditemukan

Menghadapi berbagai kendala adaptasi yang kompleks selama pelaksanaan magang, beberapa solusi diterapkan secara proaktif dan strategis untuk memastikan keberlangsungan dan kualitas pekerjaan yang optimal. Pendekatan ini didasari oleh semangat belajar, keinginan untuk berkontribusi maksimal, serta dukungan dari lingkungan kerja. Solusi-solusi tersebut diuraikan sebagai berikut:

- **Inisiatif Belajar Mandiri melalui Sumber Daring (Online Resources) yang Terstruktur:** Ketika dihadapkan pada *tools* atau teknologi baru yang belum sepenuhnya familiar, seperti Android

Studio, Figma, VSCode, serta bahasa pemrograman dan *framework* seperti XML, Kotlin/Java, HTML, CSS, JavaScript, dan jQuery, pencarian referensi tambahan secara mandiri menjadi prioritas utama. Salah satu solusi paling efektif adalah dengan memanfaatkan *platform* berbagi video seperti YouTube, Udemy, atau Coursera, yang menyediakan berbagai tutorial, *webinar*, atau *walkthrough* yang relevan dengan topik yang sedang dikerjakan. Selain itu, aktif membaca dokumentasi resmi (*official documentation*) dari setiap teknologi, berpartisipasi dalam forum diskusi pengembang (seperti Stack Overflow), dan membaca artikel teknis di blog-blog profesional sangat membantu dalam memahami konsep dasar, alur kerja, serta *best practices* dalam pengembangan *front-end* dan analisis sistem. Sebagai contoh, untuk memahami struktur *layout* di Android Studio atau manipulasi DOM dengan jQuery, tutorial visual dan contoh kode seringkali lebih mudah dicerna dan diimplementasikan.

- **Pembelajaran Menggunakan AI :** Untuk mengatasi kesulitan dalam memahami struktur sistem Odoo ERP, pendekatan belajar mandiri dilakukan secara bertahap. Proses pembelajaran dilakukan dengan mengeksplorasi dokumentasi resmi Odoo, membaca referensi daring, dan memanfaatkan teknologi kecerdasan buatan (AI) seperti ChatGPT. Permasalahan teknis dikaji melalui proses *prompting*, yaitu dengan memberikan pertanyaan-pertanyaan spesifik kepada AI, menelaah jawaban yang diberikan, dan kemudian diimplementasikan langsung pada sistem yang sedang dikerjakan. Setiap kesalahan yang ditemukan dijadikan bahan evaluasi untuk memahami logika Odoo secara lebih mendalam. Pendekatan ini terbukti efektif dalam mempercepat proses pemahaman dan penerapan konsep teknis secara praktis.
- **Memperhatikan dan Mengikuti Arahan Mentor Secara Aktif dan Proaktif:** Peran kedua atasan langsung, yaitu Ko Ferry

Kurniawan sebagai Pembimbing Lapangan dan Ko Yos Sugianto sebagai Head Direksi, sangat krusial dalam proses pembelajaran. Setiap arahan dan instruksi yang diberikan, baik itu terkait teknis implementasi kode, alur analisis kebutuhan sistem, maupun strategi proyek secara keseluruhan, berusaha untuk selalu diperhatikan dan diterapkan secara aktif. Saran dan *feedback* yang diberikan oleh mentor, yang memiliki pengalaman luas dan mendalam di industri, menjadi panduan yang sangat berharga dalam mengatasi kompleksitas pekerjaan, memahami konteks bisnis yang lebih luas, dan memastikan solusi yang diterapkan sesuai dengan standar kualitas dan kebutuhan perusahaan. Sikap proaktif dalam meminta *feedback* setelah menyelesaikan tugas juga membantu mempercepat proses belajar.

- **Budaya Tidak Malu Bertanya dan Berdiskusi Aktif:** Salah satu solusi paling fundamental adalah dengan membiasakan diri untuk tidak ragu atau malu bertanya ketika menemui kesulitan atau kebingungan yang tidak dapat diselesaikan melalui inisiatif belajar mandiri. Diskusi aktif dengan Ko Ferry Kurniawan dilakukan untuk masalah-masalah teknis yang spesifik, seperti optimasi kode, teknik *debugging* yang efisien, atau cara mengintegrasikan UI dengan *back-end* di lingkungan Odoo yang kompleks. Selain itu, inisiatif untuk berdiskusi dengan sesama rekan magang atau *developer* lain di tim juga dilakukan untuk berbagi pengetahuan, mencari perspektif berbeda terhadap suatu masalah, atau bahkan melakukan *pair programming* singkat. Pendekatan proaktif ini sangat membantu dalam mempercepat pemahaman, menghindari kesalahan yang berlarut-larut, dan menemukan solusi yang efisien dalam waktu yang lebih singkat, sekaligus membangun jaringan profesional.
- **Praktik Langsung dan Eksperimen Berkelanjutan (*Learning by Doing*):** Selain belajar teori dan aktif bertanya, pendekatan

learning by doing secara konsisten diterapkan. Setiap konsep atau *tool* baru yang dipelajari, atau setiap masalah teknis yang dihadapi, langsung dipraktikkan melalui eksperimen pada proyek-proyek kecil atau bagian-bagian terpisah dari modul yang sedang dikembangkan. Misalnya, untuk memahami cara kerja *layout* di Android, percobaan dengan berbagai kombinasi elemen UI dilakukan. Untuk memahami struktur Odoo, mencoba membuat modul sederhana atau memodifikasi *view* yang ada. Pendekatan ini membantu menginternalisasi pengetahuan dan membangun intuisi teknis yang kuat, sehingga adaptasi terhadap aplikasi dan sistem yang berbeda dapat berjalan lebih lancar seiring waktu dan pengalaman yang terakumulasi. Pendekatan ini juga membantu mengidentifikasi *error* di awal dan mempercepat proses *troubleshooting*.

- **Membuat *Checklist* dan *To-Do List* untuk Proyek Kompleks:** Untuk mengatasi kompleksitas proyek-proyek besar seperti PMS dan analisis akuntansi, pembuatan *checklist* dan *to-do list* yang detail menjadi solusi efektif. Setiap tugas besar dipecah menjadi sub-tugas yang lebih kecil dan terkelola. Ini membantu memvisualisasikan progres, memastikan tidak ada langkah yang terlewat, dan mempertahankan fokus pada detail-detail penting. Untuk analisis *gap*, *checklist* item-item yang harus diverifikasi dalam sistem dan pertanyaan-pertanyaan kunci untuk *stakeholder* disusun.
- **Membangun Jaringan Internal dan Memanfaatkan Pengetahuan Kolektif:** Secara aktif membangun jaringan komunikasi dengan *developer* atau *system analyst* lain di perusahaan, baik yang senior maupun rekan sejawat. Ini memungkinkan untuk memanfaatkan pengetahuan kolektif tim, belajar dari pengalaman mereka, dan mendapatkan *insight* yang tidak selalu tersedia dalam dokumentasi formal. Diskusi informal

mengenai tantangan proyek atau *best practices* seringkali memberikan solusi yang tidak terduga. Khususnya dalam memahami sistem *legacy* atau alur bisnis yang sudah lama berjalan, berbicara dengan *user* berpengalaman atau *developer* lama sangat membantu.

- **Fokus pada Pemahaman Konsep Inti daripada Hanya Sintaks:** Ketika dihadapkan pada banyak bahasa dan *framework* baru, solusi yang diterapkan adalah dengan lebih dulu fokus pada pemahaman konsep inti di balik teknologi tersebut (misalnya, konsep MVC, manipulasi DOM, *event-driven programming*, atau prinsip basis data) daripada hanya menghafal sintaks. Pemahaman konsep yang kuat memungkinkan transisi yang lebih cepat antara teknologi yang berbeda dan mempermudah pembelajaran *framework* baru. Ini sangat relevan dalam lingkungan kerja yang dinamis di mana teknologi dapat terus berubah