BAB III
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Selama 4 bulan dan 2 minggu masa pelaksanaan kerja magang berlangsung di *Back*slah Creative Nusantara, saya di berikan tanggung jawab untuk sebagai *front-end developer* untuk membuat tampilan *website asset management* dan *Expense tracker* yang menarik untuk keperluan dari dokumentasi manajemen aset dan finansial pada PT. *Back*slash Creative Nusantara. Selain itu saya juga diberikan tanggung jawab sebagai IT Associate yang bertugas untuk membantu PT. Paul Buana Indonesia yang bertugas untuk membantu *project daily task automation*. Dalam masa pelaksanaan kerja magang ini dibimbing oleh bapak Robert Theo sebagai Senior IT dari pihak *Back*slash dan bapak Damar Sinatrio sebagai pihak dari *Back*slash yang mengawasi dan melakukan laporan untuk melakukan *update* pekerjaan yang sudah dilakukan setiap harinya. Dengan bantuan dari Bapak Robert Theo dan Bapak Damar Sinatrio membuat pelaksanaan magang ini berkesan dengan mendapatkan pelajaran baru mulai dari pendalaman bahasa pemograman, manajemen tugas dan juga masukan serta saran yang berguna dalam memperbaiki kesalahan kesalahan yang terjadi selama menjalani masa kerja magang ini.

3.2 Tugas dan Uraian Kerja Magang

Dalam proses pengembangan website Asset Management dan *Expense* Tracker, dibutuhkan kolaborasi antara berbagai peran, termasuk pengembang *back*-end dan front-end. Oleh karena itu, penulis bekerja sama dengan rekan magang lainnya dalam membangun sistem ini secara terpadu. Selama masa pelaksanaan magang di PT. *Back*slash Creative Nusantara sebagai IT Associate, penulis bertanggung jawab dalam pengembangan sistem berbasis web, khususnya pada bagian front-end, serta terlibat dalam koordinasi teknis bersama tim *back*-end. Website ini dikembangkan menggunakan bahasa

pemrograman PHP dengan framework Laravel untuk sisi *back*-end serta HTML, CSS, dan JavaScript pada sisi front-end. Dalam membangun antarmuka pengguna yang responsif dan modern, digunakan framework antarmuka Bootstrap serta CoreUI. Untuk pengelolaan data, sistem ini menggunakan MySQL sebagai basis data utama. Tabel 3.1 berikut adalah rangkaian kegiatan yang dilakukan selama masa magang dalam proses pengembangan website ini.

Tabel 3.1 Rincian Kerja Magang

No.	Aktivitas Kerja	Rentang Tanggal
1	Mengerjakan fitur login, termasuk autentikasi, otorisasi, dan pengujian awal.	21 Apr - 25 Apr 2025
2	Menyelesaikan login, menambahkan fitur logout, redirect, dan penggantian password.	28 Apr - 02 Mei 2025
3	Membuat dan menguji fitur manajemen user (CRUD).	05 Mei - 13 Mei 2025
4	Mengimplementasikan pencatatan inventaris dan pembelian barang.	13 Mei - 19 Mei 2025
5	Menyempurnakan serta menguji fitur pembelian dan permintaan barang.	19 Mei - 26 Mei 2025
6	Menambahkan fitur persetujuan pembelian dan mulai fitur reimbursement.	26 Mei - 02 Jun 2025
7	Melakukan revisi pada fitur persetujuan dan pengajuan reimbursement.	02 Jun - 09 Jun 2025
8	Penyelesaian dan testing fitur reimbursement. Pengujian menyeluruh untuk fitur reimbursement.	09 Jun - 20 Jun 2025
9	Revisi dan penyempurnaan fitur asset dan fitur reimbursement. Testing akhir pada keduanya.	23 Jun - 27 Jun 2025

3.2.1 Mengerjakan Fitur Login

Pada minggu pertama, fokus utama pengerjaan adalah membangun fondasi sistem berupa fitur login. Fitur ini menjadi elemen penting karena berfungsi sebagai pintu masuk ke dalam sistem dan menjadi basis untuk keamanan data serta pembatasan akses pengguna. Pengerjaan mencakup proses autentikasi untuk memverifikasi identitas *user* berdasarkan *username* dan *password* yang valid, serta otorisasi yang berfungsi untuk memberikan hak akses tertentu sesuai peran, seperti admin, staf, atau pengguna biasa. Penerapan autentikasi dan otorisasi dilakukan dengan memanfaatkan standar keamanan

sistem informasi, termasuk hashing *password* dan penggunaan session atau token untuk menjaga keamanan selama sesi pengguna berlangsung. Pengujian awal juga dilakukan untuk memastikan bahwa sistem login dapat menangani berbagai kemungkinan, seperti input yang tidak valid, akun tidak ditemukan, atau *password* yang salah. Dengan diselesaikannya bagian ini, sistem telah memiliki dasar keamanan yang cukup untuk dilanjutkan ke tahap fitur berikutnya.

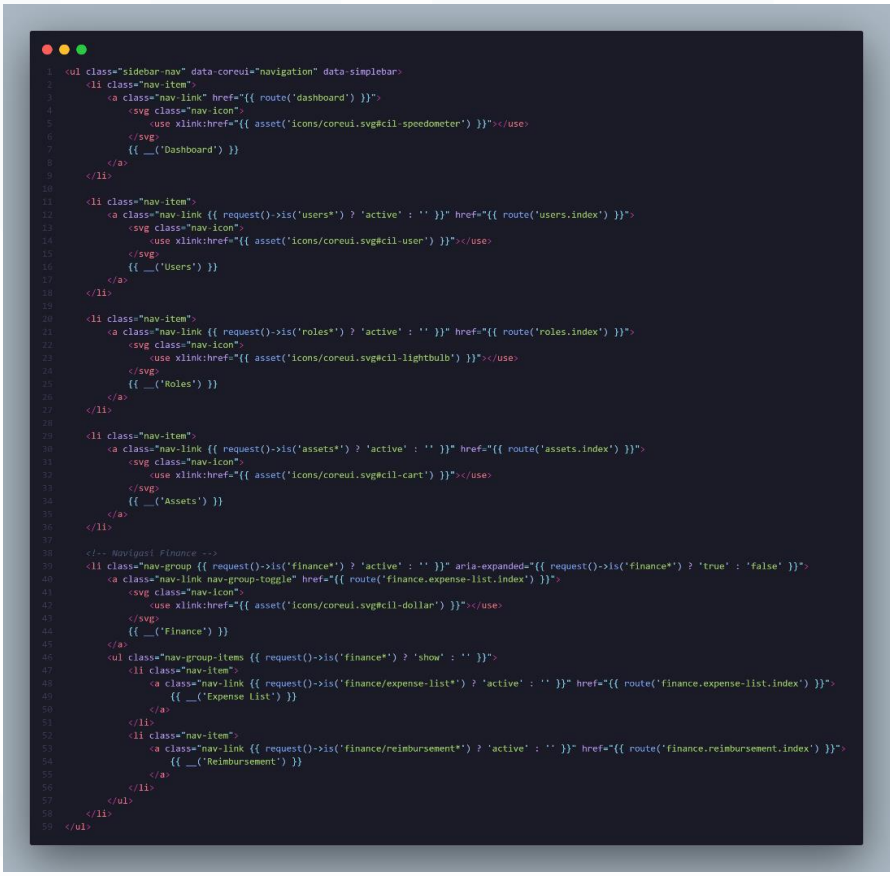
3.2.2 Menyempurnakan Login dan Menambahkan Fitur Logout, Redirect, serta Penggantian *Password*

Pada minggu kedua, pengembangan sistem dilanjutkan dengan menyempurnakan fitur login serta menambahkan beberapa komponen penting lainnya yaitu logout, redirect, dan fitur penggantian *password*. Fitur logout bertujuan untuk memastikan bahwa pengguna dapat keluar dari sistem dengan aman sehingga sesi tidak terbuka untuk disalahgunakan. Fitur redirect dirancang agar setelah login, pengguna secara otomatis diarahkan ke halaman yang sesuai dengan hak aksesnya misalnya admin diarahkan ke dashboard admin, sementara *user* diarahkan ke halaman utama mereka. Fitur penggantian *password* juga ditambahkan sebagai bagian dari peningkatan keamanan sistem. Dengan fitur ini, pengguna dapat mengganti *password* mereka secara mandiri jika diperlukan. Dalam proses implementasinya, perhatian khusus diberikan pada keamanan, seperti verifikasi *password* lama dan validasi *password* baru. Semua fitur ini diuji secara menyeluruh untuk memastikan integrasi berjalan lancar dan tidak menimbulkan konflik pada sesi pengguna atau database sistem.

3.2.3 Membuat dan Menguji Fitur Manajemen *User* (CRUD)

Minggu ketiga difokuskan pada pengembangan fitur manajemen *user* yang mendukung fungsi CRUD (*Create, Read, Update, Delete*). Fitur ini memungkinkan admin untuk mengelola data pengguna, termasuk menambahkan akun baru, melihat daftar pengguna yang

terdaftar, memperbarui data pengguna, serta menghapus akun yang tidak lagi digunakan. Fitur ini sangat penting karena menjadi alat utama dalam pengelolaan struktur pengguna sistem secara keseluruhan. Setiap operasi CRUD diimplementasikan dengan prosedur validasi untuk memastikan integritas data, seperti memastikan *username* unik, validasi email, dan pembatasan hak akses tertentu. Sistem dikembangkan menggunakan HTML5, CSS3, Bootstrap framework, Javascript, PHP, dan MySQL sebagai *database*.

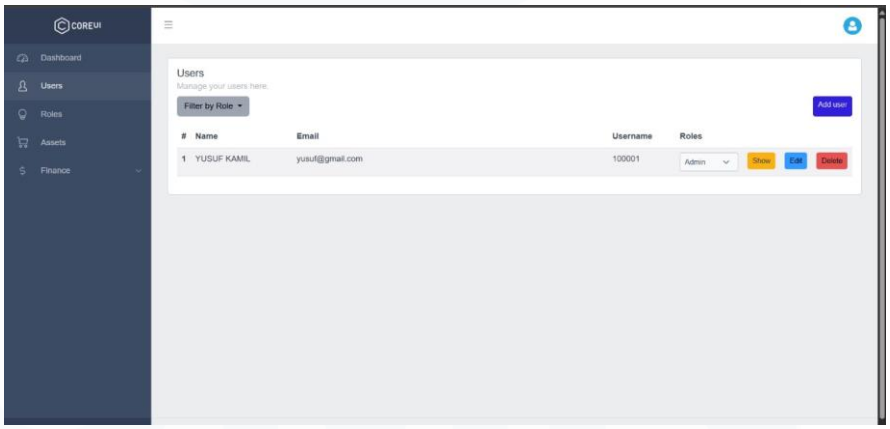


```
<ul class="sidebar-nav" data-coreui="navigation" data-simplebar>
    <li class="nav-item">
        <a class="nav-link" href="{{ route('dashboard') }}">
            <svg class="nav-icon">
                <use xlink:href="{{ asset('icons/coreui.svg#cil-speedometer') }}"></use>
            </svg>
            {{ __('Dashboard') }}
        </a>
    </li>

    <li class="nav-item">
        <a class="nav-link {{ request()->is('users*') ? 'active' : '' }}" href="{{ route('users.index') }}">
            <svg class="nav-icon">
                <use xlink:href="{{ asset('icons/coreui.svg#cil-user') }}"></use>
            </svg>
            {{ __('Users') }}
        </a>
    </li>

    <li class="nav-item">
        <a class="nav-link {{ request()->is('roles*') ? 'active' : '' }}" href="{{ route('roles.index') }}">
            <svg class="nav-icon">
                <use xlink:href="{{ asset('icons/coreui.svg#cil-lightbulb') }}"></use>
            </svg>
            {{ __('Roles') }}
        </a>
    </li>

    <li class="nav-item">
        <a class="nav-link {{ request()->is('assets*') ? 'active' : '' }}" href="{{ route('assets.index') }}">
            <svg class="nav-icon">
                <use xlink:href="{{ asset('icons/coreui.svg#cil-cart') }}"></use>
            </svg>
            {{ __('Assets') }}
        </a>
    </li>

    <!-- Navigasi Finance -->
    <li class="nav-group {{ request()->is('finance*') ? 'active' : '' }}" aria-expanded="{{ request()->is('finance*') ? 'true' : 'false' }}">
        <a class="nav-link nav-group-toggle" href="{{ route('finance.expense-list.index') }}">
            <svg class="nav-icon">
                <use xlink:href="{{ asset('icons/coreui.svg#cil-dollar') }}"></use>
            </svg>
            {{ __('Finance') }}
        </a>
        <ul class="nav-group-items {{ request()->is('finance*') ? 'show' : '' }}">
            <li class="nav-item">
                <a class="nav-link {{ request()->is('finance/expense-list*') ? 'active' : '' }}" href="{{ route('finance.expense-list.index') }}">
                    {{ __('Expense List') }}
                </a>
            </li>
            <li class="nav-item">
                <a class="nav-link {{ request()->is('finance/reimbursement*') ? 'active' : '' }}" href="{{ route('finance.reimbursement.index') }}">
                    {{ __('Reimbursement') }}
                </a>
            </li>
        </ul>
    </li>
</ul>
```

Gambar 3.3 *Coding Navigation Page*

Gambar 3.3 menampilkan *coding* pada file navigation.blade.php berfungsi untuk membangun menu navigasi sidebar pada sistem manajemen aset dan pelacakan pengeluaran berbasis web. Struktur navigasi ini dibuat menggunakan *Blade templating engine* dari *Laravel* dan memanfaatkan framework *CoreUI* untuk tampilan antarmuka yang responsif dan konsisten. Setiap item

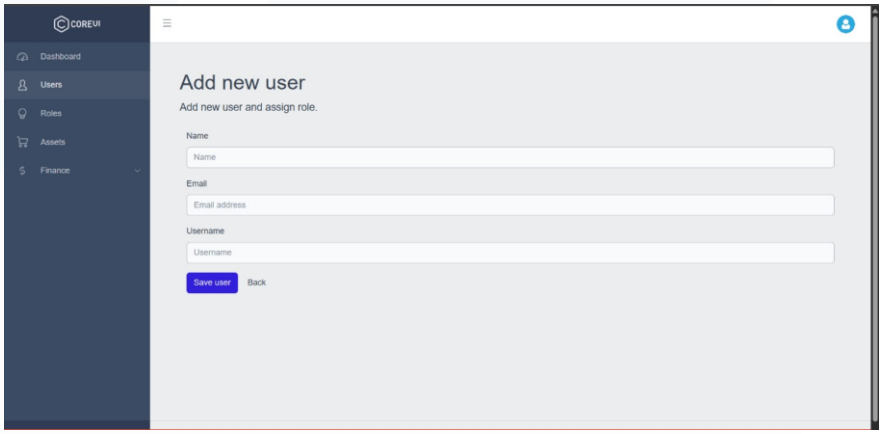
menu dibungkus dalam elemen <li class="nav-item"> yang berisi tautan dengan kelas dinamis *active* yang ditambahkan menggunakan request()->is() untuk menandai halaman yang sedang diakses oleh pengguna. Rute-rute navigasi didefinisikan menggunakan fungsi route() yang merujuk pada *named routes* seperti *dashboard*, *users.index*, *roles.index*, dan *assets.index* guna memudahkan pengelolaan dan pemanggilan halaman. Masing-masing item menu juga dilengkapi ikon SVG dari pustaka *CoreUI* yang dimuat menggunakan helper asset(), seperti ikon *cil-speedometer* untuk Dashboard, *cil-user* untuk *User*s, *cil-lightbulb* untuk Roles, dan *cil-cart* untuk Assets. Komponen sidebar ini berperan penting dalam meningkatkan pengalaman pengguna dengan menyediakan akses cepat ke setiap modul utama sistem, sekaligus menjaga konsistensi desain dan navigasi yang sesuai dengan peran pengguna.



Gambar 3.4 *User List Page*

Gambar 3.4 menunjukkan halaman daftar *user* yang menampilkan semua pengguna yang terdaftar dalam sistem. Halaman ini memiliki header navigation dan menu hamburger untuk navigasi, serta side navigation yang mencakup menu *Dashboard, Users, Roles, Assets, dan Finance*. Pada bagian utama halaman terdapat filter dropdown "*Filter by Role*" untuk menyaring pengguna berdasarkan peran, dan tombol "*Add user*" di kanan atas untuk menambahkan

pengguna baru. Tabel *user* menampilkan informasi lengkap pengguna meliputi nomor urut, nama pengguna, alamat email, *username* untuk login, dan peran pengguna dengan *dropdown* untuk mengubah peran. Setiap baris pengguna dilengkapi dengan tombol aksi Show untuk melihat detail, tombol Edit untuk mengedit data, dan *Delete* untuk menghapus pengguna. Halaman ini memungkinkan admin untuk melakukan operasi *Read* dalam CRUD, yaitu melihat dan memfilter daftar pengguna yang ada dalam sistem dengan interface yang clean dan intuitif menggunakan *framework* Bootstrap untuk responsivitas.



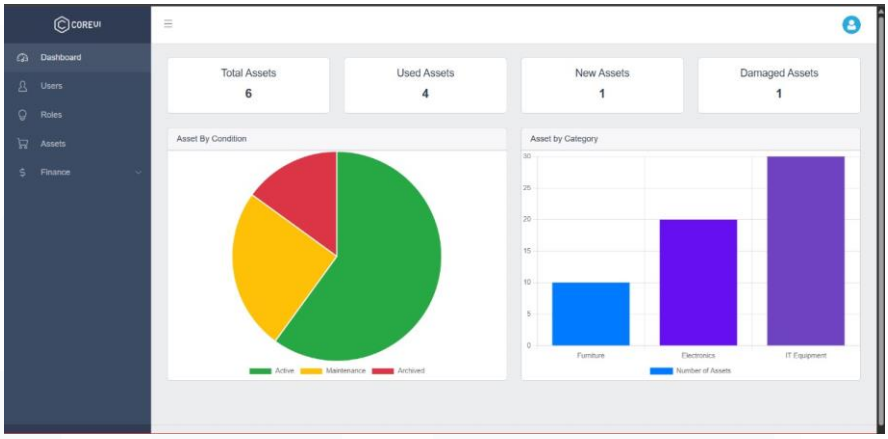
Gambar 3.5 *Add New User Page*

Gambar 3.5 menunjukkan halaman untuk menambahkan pengguna baru ke dalam sistem. Halaman ini mengimplementasikan operasi *Create* dalam CRUD dengan judul "*Add New user*" dan deskripsi "*Add New user and assign role*". *Form input* terdiri dari *field* Name untuk memasukkan nama lengkap pengguna, *field* Email untuk memasukkan alamat email pengguna, dan *field Username* untuk memasukkan *username* yang akan digunakan untuk *login*. Di bagian bawah *form* terdapat tombol "*Save user*" berwarna biru untuk menyimpan data pengguna baru dan tombol "*Back*" untuk kembali ke halaman sebelumnya. Halaman ini mempertahankan *side navigation* yang sama dengan halaman *user List* untuk konsistensi navigasi. Form dilengkapi dengan validasi untuk memastikan *username* bersifat unik dalam sistem, format email yang valid, semua *field* wajib diisi dengan

benar, dan integritas data sebelum disimpan ke database. Proses pengujian dilakukan pada setiap fungsi untuk memastikan bahwa tidak ada *error* atau data yang tidak tersimpan dengan baik. Proses ini memperkuat sistem dalam aspek manajemen data pengguna dan mempermudah pengelolaan peran pengguna oleh admin. Dengan diselesaikannya fitur manajemen *user* ini, sistem telah memiliki fondasi yang kuat untuk pengelolaan pengguna dengan antarmuka yang *user-friendly* dan proses validasi yang ketat.

3.2.4 Implementasi Pencatatan Inventaris dan Pembelian Barang

Pada minggu keempat, sistem dikembangkan lebih lanjut dengan menambahkan fitur pencatatan inventaris dan pembelian barang. Fitur ini berfungsi untuk mendata dan mencatat barang-barang yang masuk ke dalam sistem, baik dari pembelian maupun penambahan manual. Proses pencatatan meliputi informasi nama barang, jumlah, tanggal pembelian, dan vendor pemasok. Fitur ini sangat penting dalam konteks manajemen stok dan pengelolaan aset perusahaan. Implementasi dilakukan dengan struktur tabel database yang memadai untuk menyimpan histori pembelian, serta tampilan antarmuka yang mudah dipahami oleh pengguna. Selain itu, fitur pencarian dan filter juga ditambahkan untuk memudahkan proses monitoring. Pengujian dilakukan untuk memastikan bahwa pencatatan data berjalan akurat dan data dapat ditampilkan kembali saat diperlukan.



Gambar 3.6 *Dashboard Assets*

Gambar 3.6 menampilkan tampilan *Dashboard Assets* yang berperan sebagai pusat kendali (*command center*) dari sistem manajemen aset. Tampilan ini memberikan gambaran menyeluruh tentang kondisi inventaris perusahaan dalam bentuk visual yang mudah dipahami. Di bagian atas *dashboard*, terdapat empat *metric cards* yang menunjukkan *Key Performance Indicators (KPIs)*, yaitu: *Total Assets* sebanyak 6, *Good Assets* sebanyak 4, *New Assets* sebanyak 1, dan *Broken Assets* sebanyak 1. Informasi ini memberikan ringkasan cepat tentang penyebaran dan kondisi aset perusahaan yang sangat berguna untuk pengambilan keputusan oleh manajemen serta pemantauan kondisi inventaris secara langsung (*real-time*). Sementara itu, bagian bawah *dashboard* menampilkan dua visualisasi data utama. Pertama, terdapat *pie chart* dengan judul *"Asset By Status"* yang menggambarkan distribusi aset berdasarkan kondisinya. Warna hijau mewakili aset yang masih aktif (*Active*) dan mendominasi grafik, warna kuning menandakan aset yang sedang dalam perawatan (*Maintenance*), dan warna merah menunjukkan aset yang sudah tidak digunakan atau diarsipkan (*Inactive*). Tampilan visual ini memudahkan dalam melihat kondisi keseluruhan inventaris; dominasi warna hijau menunjukkan bahwa sebagian besar aset dalam keadaan baik, sedangkan warna kuning dan merah menandakan aset yang perlu perhatian lebih lanjut.

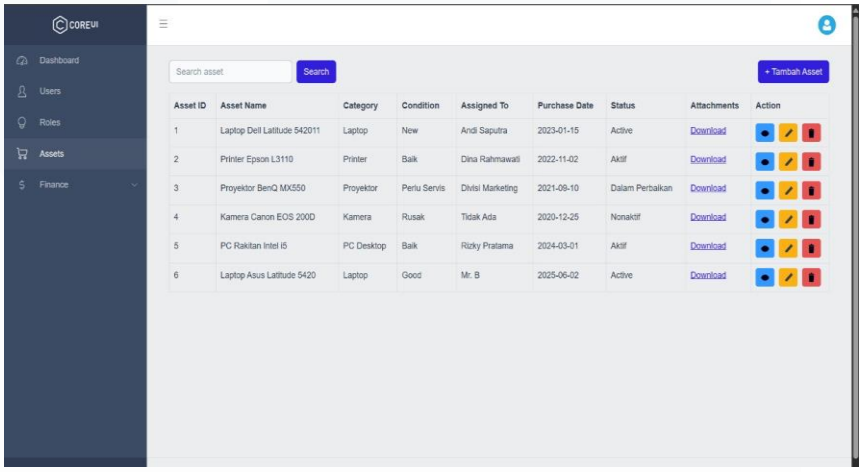
Kedua, terdapat *bar chart* dengan judul *"Asset by Category"* yang menunjukkan jumlah aset berdasarkan kategori, dengan tiga batang utama: *Furniture* sekitar 10 unit, *Electronics* sekitar 20 unit, dan *IT Equipment* dengan jumlah tertinggi sekitar 30 unit. Visualisasi ini menunjukkan bahwa aset perusahaan didominasi oleh perangkat IT, yang merupakan hal umum dalam perusahaan berbasis teknologi. Skema warna yang digunakan adalah gradasi biru yang tidak hanya menarik secara visual tetapi juga mempertahankan kesan profesional. Di bagian bawah grafik terdapat *legend* berwarna biru bertuliskan *"Number of Assets"* untuk menjelaskan makna tinggi masing-masing batang. Secara keseluruhan, *dashboard* ini dirancang dengan memperhatikan prinsip-prinsip *data visualization* yang baik, seperti penggunaan jenis grafik yang tepat untuk masing-masing data, konsistensi warna, label yang jelas, serta tata letak yang memungkinkan pemahaman cepat dan menghasilkan *insight* yang dapat ditindaklanjuti untuk pengambilan keputusan strategis dalam pengelolaan aset.

```
    @endsection

    @push('after-scripts')
    <!-- Sertakan Chart.js, jika belum termasuk di layout Anda -->
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script>
      // Pie Chart: Asset By Condition
      // Pie Chart: Asset By Condition
const ctxCondition = document.getElementById('assetByConditionChart').getContext('2d');
const assetByConditionChart = new Chart(ctxCondition, {
  type: 'pie',
  data: {
    labels: ['Active', 'Maintenance', 'Archived'],
    datasets: [{
      data: [60, 25, 15],
      backgroundColor: ['#28a745', '#ffc107', '#dc3545']
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    plugins: {
      legend: { position: 'bottom' }
    }
  }
});

// Bar Chart: Asset by Category
const ctxCategory = document.getElementById('assetByCategoryChart').getContext('2d');
const assetByCategoryChart = new Chart(ctxCategory, {
  type: 'bar',
  data: {
    labels: ['Furniture', 'Electronics', 'IT Equipment'],
    datasets: [{
      label: 'Number of Assets',
      data: [10, 20, 30],
      backgroundColor: ['#007bff', '#6610f2', '#6f42c1']
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    scales: {
      y: { beginAtZero: true }
    },
    plugins: {
      legend: { position: 'bottom' }
    }
  }
});

    </script>
@endpush
```

Gambar 3.7 *Coding Dashboard Page*

Gambar 3.7 merupakan *coding* dashboard.blade.php merupakan struktur halaman dashboard utama dari sistem manajemen aset yang dirancang menggunakan *Blade templating engine* dalam framework *Laravel*. Halaman ini menampilkan beberapa *widget card* yang berisi ringkasan data aset, seperti *Total Assets*, *Used Assets*, *New Assets*, dan *Damaged Assets*, dengan masing-masing nilai ditampilkan secara visual di dalam elemen *card* yang diatur dalam grid responsif. Selain itu, halaman ini juga menampilkan dua jenis grafik menggunakan *Chart.js*, yaitu *pie chart* untuk memvisualisasikan aset berdasarkan kondisi (*Active*,

Maintenance, *Archived*) dan *bar chart* untuk memvisualisasikan aset berdasarkan kategori (*Furniture*, *Electronics*, *IT Equipment*). Grafik ini memberikan insight cepat mengenai distribusi dan status aset perusahaan. Kode JavaScript di bagian bawah halaman mengatur visualisasi tersebut dengan konfigurasi label, data, warna, serta responsivitas tampilan. Dengan kombinasi antara *Bootstrap* untuk layout dan *Chart.js* untuk grafik, halaman dashboard ini dirancang agar informatif, mudah dipahami, dan mendukung pengambilan keputusan secara cepat oleh pengguna.



Gambar 3.8 Asset List Page

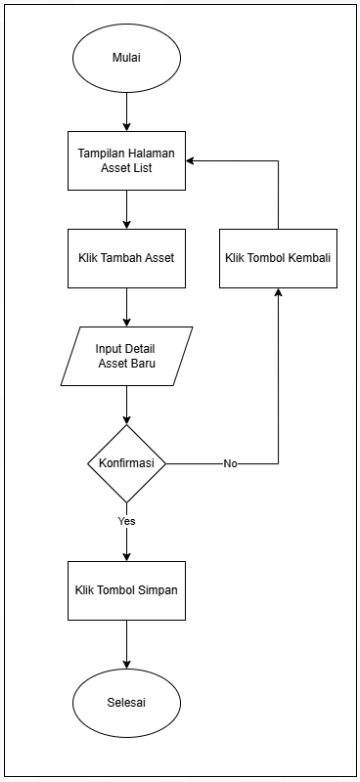
Gambar 3.8 menampilkan halaman *"Asset List"* yang berfungsi sebagai pusat data utama (*master data*) dari inventaris perusahaan secara menyeluruh. Halaman ini menyajikan semua aset perusahaan dalam format tabel yang terstruktur dengan baik dan mudah untuk dinavigasi. Terdapat enam aset yang sudah terdaftar di dalam sistem, mulai dari *Laptop Dell Latitude 542011* hingga *Laptop Asus Latitude 5420*, masing-masing disertai informasi lengkap seperti ID aset, nama, kategori, kondisi, penanggung jawab (*assignee*), tanggal pembelian, status, dan *attachments*. Di bagian atas tabel terdapat fitur pencarian (*search functionality*) yang memungkinkan pengguna menemukan aset tertentu dengan cepat, serta tombol *"+ Tambah Asset"* berwarna biru mencolok untuk menambahkan

aset baru ke dalam sistem. Struktur tabel ini menunjukkan penyusunan data yang sangat rapi dan informatif, dengan kolom-kolom yang tidak hanya memberikan informasi tetapi juga bisa ditindaklanjuti (*actionable*). Kolom *"Asset ID"* memberikan pengenal unik untuk setiap aset, sedangkan kolom *"Asset Name"* menampilkan nama lengkap dan spesifikasi dari masing-masing aset sehingga memudahkan proses identifikasi. Kolom *"Category"* mengelompokkan aset berdasarkan jenisnya seperti *Laptop*, *Printer*, *Proyektor*, *Kamera*, dan *PC Desktop*, sehingga memudahkan proses penyaringan (*filtering*) dan pelaporan (*reporting*). Sementara itu, kolom *"Condition"* memberikan informasi tentang kondisi fisik aset, dengan status seperti *"New"*, *"Baik"*, *"Good"*, hingga *"Rusak"*, yang memberikan gambaran mengenai kebutuhan perawatan atau penggantian aset.

Kolom *"Assigned To"* menunjukkan tanggung jawab terhadap aset dengan mencantumkan nama penanggung jawab seperti "Andi Saputra", "Dina Rahmawati", "Divisi Marketing", dan lainnya. Beberapa aset juga memiliki keterangan *"Tidak Ada"* yang menunjukkan bahwa aset tersebut belum atau tidak lagi dialokasikan kepada individu atau divisi tertentu. Kolom *"Purchase Date"* menampilkan informasi kronologis dari tahun 2020 hingga 2025, menunjukkan keberagaman usia aset dalam sistem inventaris perusahaan. Kolom *"Status"* menggambarkan kondisi operasional aset dengan berbagai status seperti *"Active"*, *"Aktif"*, *"Nonaktif"*, dan *"Dalam Perbaikan"*, yang mencerminkan keadaan terkini aset tersebut. Kolom *"Attachments"* berisi tautan *"Download"* yang mengarah pada dokumen pendukung tiap aset, mencerminkan adanya integrasi sistem manajemen dokumen dalam aplikasi.

Di bagian paling kanan tabel terdapat tombol-tombol *"Action"* yang terdiri dari tiga jenis: tombol biru untuk melihat detail (*view*), tombol kuning untuk mengedit (*edit*), dan tombol merah untuk menghapus

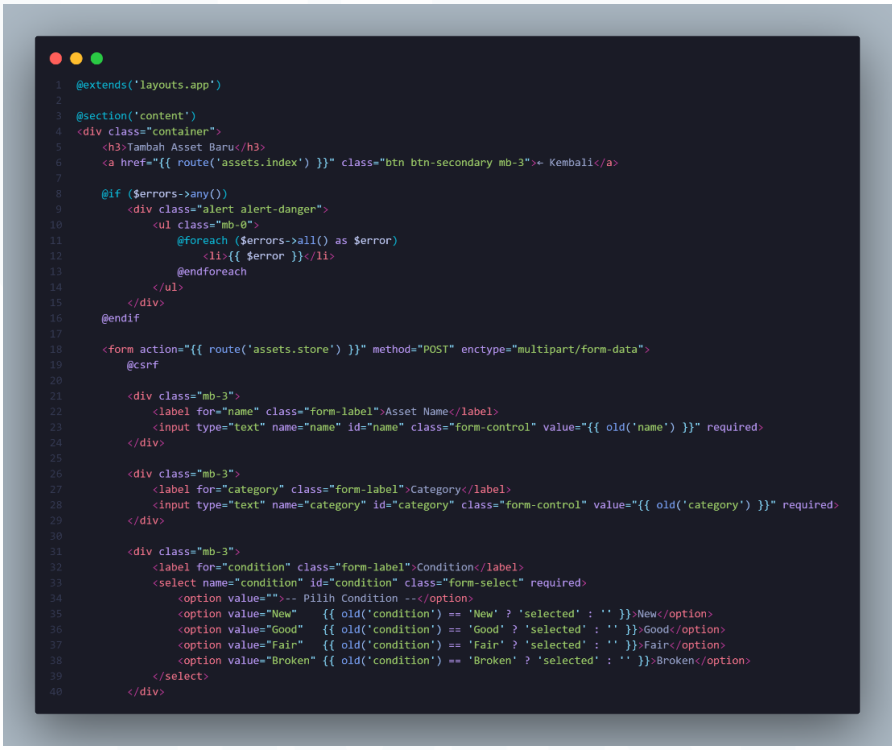
(*delete*), memberikan akses penuh terhadap operasi *CRUD* (Create, Read, Update, Delete) untuk masing-masing aset. Secara keseluruhan, desain tabel ini dirancang untuk mengoptimalkan kepadatan informasi (*information density*) namun tetap menjaga keterbacaan (*readability*) dan kemudahan penggunaan (*usability*), dengan penggunaan warna yang konsisten serta jarak antar elemen yang cukup sehingga nyaman dilihat dan digunakan.



Gambar 3.9 *Flowchart Create New Asset*

Gambar 3.9 diatas menggambarkan alur pengguna dalam menambahkan data aset baru ke dalam sistem manajemen aset. Proses dimulai dengan pengguna mengakses halaman *Asset List*, yaitu halaman utama yang menampilkan daftar semua aset yang dimiliki perusahaan. Dari halaman ini, pengguna dapat memilih opsi *Tambah Asset* untuk memulai proses pencatatan aset baru. Setelah tombol tersebut diklik, pengguna akan diarahkan ke halaman form input untuk mengisi detail aset yang akan ditambahkan, seperti nama aset, kategori, kondisi fisik, penanggung jawab, tanggal pembelian, status

aset, dan dokumen pendukung. Setelah semua data terisi, sistem akan meminta konfirmasi dari pengguna. Jika pengguna memilih “No”, maka akan diarahkan untuk kembali dengan menekan tombol *Kembali* untuk memperbaiki atau membatalkan input. Sebaliknya, jika pengguna menyetujui pengisian data dengan memilih “Yes”, maka langkah selanjutnya adalah menekan tombol *Simpan*. Setelah data disimpan ke dalam sistem, proses pencatatan aset dianggap selesai.

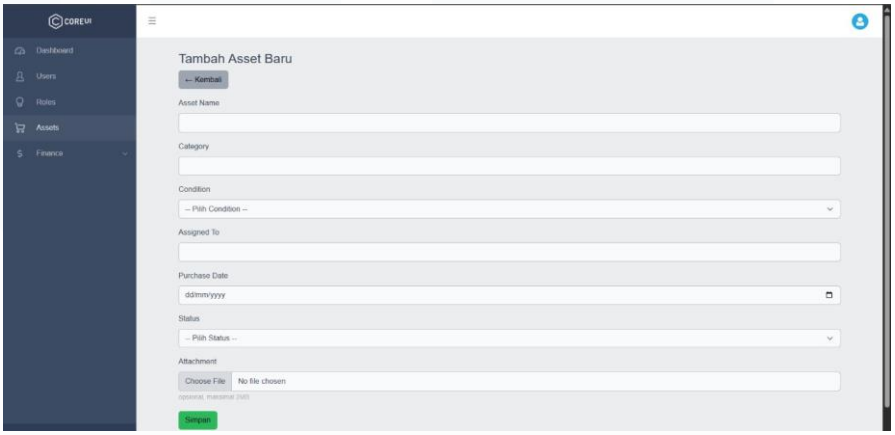


```
@extends('layouts.app')

@section('content')
<div class="container">
    <h3>Tambah Asset Baru</h3>
    <a href="{{ route('assets.index') }}" class="btn btn-secondary mb-3">← Kembali</a>

    @if ($errors->any())
        <div class="alert alert-danger">
            <ul class="mb-0">
                @foreach ($errors->all() as $error)
                    <li>{{ $error }}</li>
                @endforeach
            </ul>
        </div>
    @endif

    <form action="{{ route('assets.store') }}" method="POST" enctype="multipart/form-data">
        @csrf

        <div class="mb-3">
            <label for="name" class="form-label">Asset Name</label>
            <input type="text" name="name" id="name" class="form-control" value="{{ old('name') }}" required>
        </div>

        <div class="mb-3">
            <label for="category" class="form-label">Category</label>
            <input type="text" name="category" id="category" class="form-control" value="{{ old('category') }}" required>
        </div>

        <div class="mb-3">
            <label for="condition" class="form-label">Condition</label>
            <select name="condition" id="condition" class="form-select" required>
                <option value="">-- Pilih Condition --</option>
                <option value="New"    {{ old('condition') == 'New' ? 'selected' : '' }}>New</option>
                <option value="Good"   {{ old('condition') == 'Good' ? 'selected' : '' }}>Good</option>
                <option value="Fair"   {{ old('condition') == 'Fair' ? 'selected' : '' }}>Fair</option>
                <option value="Broken" {{ old('condition') == 'Broken' ? 'selected' : '' }}>Broken</option>
            </select>
        </div>
```

Gambar 3.10 *Coding Create New Asset List*

Kode pada Gambar 3.10 merupakan halaman *Create New Asset* (create.blade.php) yang digunakan untuk menambahkan data aset baru ke dalam sistem. Halaman ini dibangun menggunakan *Blade templating engine* dari *Laravel* dan memperluas layout utama (layouts.app). Terdapat form input yang mengarah ke route assets.store dengan metode POST serta enctype="multipart/form-data", memungkinkan pengguna mengirim data aset secara aman. Fitur @csrf disertakan sebagai perlindungan dari serangan *CSRF*. Formulir ini terdiri dari beberapa isian penting seperti *Asset Name*,

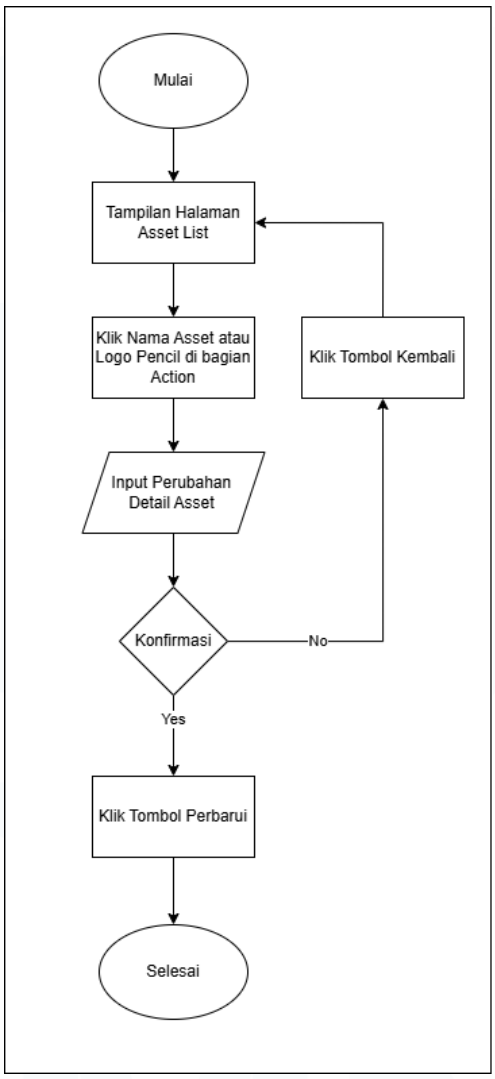
Category, dan *Condition*, di mana *Condition* ditampilkan dalam bentuk *dropdown menu* yang menyediakan beberapa pilihan seperti *New*, *Good*, *Fair*, dan *Broken*. Validasi error juga telah diimplementasikan, ditampilkan secara dinamis jika terdapat kesalahan input. Halaman ini berfungsi sebagai antarmuka utama dalam proses input data aset dan memastikan semua informasi penting tercatat dengan baik sesuai standar yang ditentukan dalam sistem manajemen aset.



Gambar 3.11 *Add New Asset Page*

Gambar 3.11 menampilkan halaman *"Tambah Asset Baru"* yang berfungsi sebagai formulir pendaftaran untuk memasukkan aset baru ke dalam sistem inventaris perusahaan. Halaman ini menyajikan sebuah *form* kosong yang telah dirancang dengan *layout* yang bersih dan terstruktur agar proses pengisian data menjadi lebih mudah dan efisien. *Form* ini memiliki susunan *field* yang sama seperti yang terdapat pada halaman *edit*, namun dalam kondisi kosong, menunjukkan konsistensi *design pattern* yang diterapkan dalam aplikasi secara keseluruhan. Setiap *field* dilengkapi dengan *placeholder text* atau *dropdown options* sebagai panduan bagi pengguna untuk mengetahui jenis data yang perlu diisikan. Struktur *form* dimulai dari *"Asset Name"* sebagai kolom utama yang menjadi identitas utama aset di dalam sistem, diikuti dengan *"Category"* yang menggunakan *dropdown* untuk memastikan aset diklasifikasikan

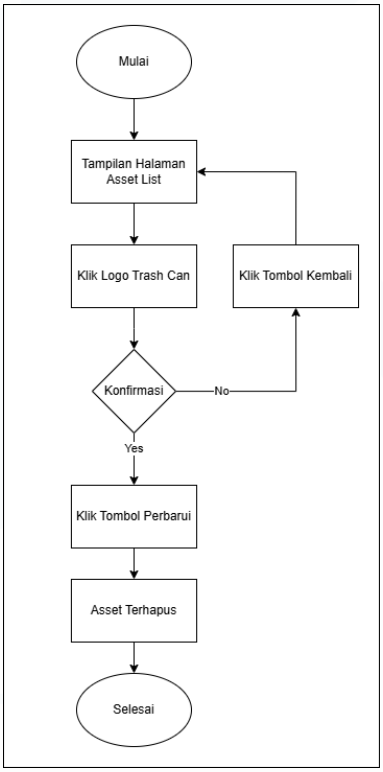
sesuai dengan kategori standar perusahaan. *Field "Condition"* yang juga menggunakan *dropdown* dengan pilihan *"-- Pilih Condition --"* memudahkan pengguna dalam menentukan kondisi awal aset ketika pertama kali didaftarkan, yang sangat penting untuk kebutuhan pelacakan siklus hidup aset (*lifecycle tracking*). Selanjutnya, *field "Assigned To"* memungkinkan pengguna langsung menetapkan siapa yang akan menjadi penanggung jawab aset tersebut. Sementara itu, *"Purchase Date"* menggunakan format *"dd/mm/yyyy"*, yang bertujuan untuk mencatat tanggal pembelian secara akurat demi kepentingan pelacakan garansi (*warranty tracking*) dan perhitungan depresiasi (*depreciation calculation*). Bagian *"Status"* dilengkapi dengan *dropdown "-- Pilih Status --"* untuk menetapkan status operasional dari aset baru, apakah langsung digunakan (*aktif*) atau masih dalam proses penyiapan (*setup*). Di bagian bawah *form*, terdapat bagian *"Attachment"* yang menunjukkan kemampuan sistem dalam mengelola dokumen-dokumen pendukung, seperti *invoice*, *warranty certificate*, atau *manual book* yang berkaitan dengan aset yang bersangkutan. Pesan "Optional, maksimal file harus untuk manajemen" memberikan informasi tambahan bahwa fitur *attachment* ini bersifat opsional serta terdapat batasan ukuran file yang dapat diunggah. Terakhir, tombol *"Simpan"* berwarna hijau yang mencolok menunjukkan *primary action* atau tindakan utama untuk menyimpan seluruh data aset baru ke dalam sistem secara permanen.



Gambar 3.12 *Flowchart Edit Asset*

Flowchart pada Gambar 3.12 ini menggambarkan alur proses pengelolaan aset yang dimulai dari tahap awal hingga selesai. Proses dimulai ketika pengguna mengakses tampilan halaman daftar aset (*Asset List*) dari menu utama. Pada halaman ini, pengguna dapat memilih untuk melakukan dua tindakan utama: menambah aset baru dengan mengklik nama aset atau logo pensil di bagian *action*, atau kembali ke menu sebelumnya dengan mengklik tombol kembali. Jika pengguna memilih untuk menambah atau mengedit aset, sistem akan mengarahkan ke halaman input perubahan detail aset. Di halaman ini, pengguna dapat memasukkan atau mengubah informasi detail

mengenai aset yang bersangkutan, seperti nama aset, kategori, kondisi, lokasi, dan data relevan lainnya. Setelah pengguna selesai mengisi atau mengubah data, sistem akan meminta konfirmasi untuk memastikan bahwa perubahan yang dilakukan sudah benar. Pada tahap konfirmasi, pengguna akan dihadapkan pada pilihan "Ya" atau "No". Jika pengguna memilih "No", sistem akan mengarahkan kembali ke tombol kembali, memberikan kesempatan untuk membatalkan atau meninjau ulang perubahan yang telah dibuat. Namun jika pengguna memilih "Ya", sistem akan memproses perubahan tersebut dengan mengklik tombol perbarui, yang kemudian akan menyimpan semua perubahan data aset ke dalam database. Setelah proses penyimpanan berhasil, alur kerja akan berakhir dan pengguna dapat melanjutkan ke aktivitas lainnya atau kembali ke menu utama sistem.



Gambar 3.13 *Flowchart Delete Asset*

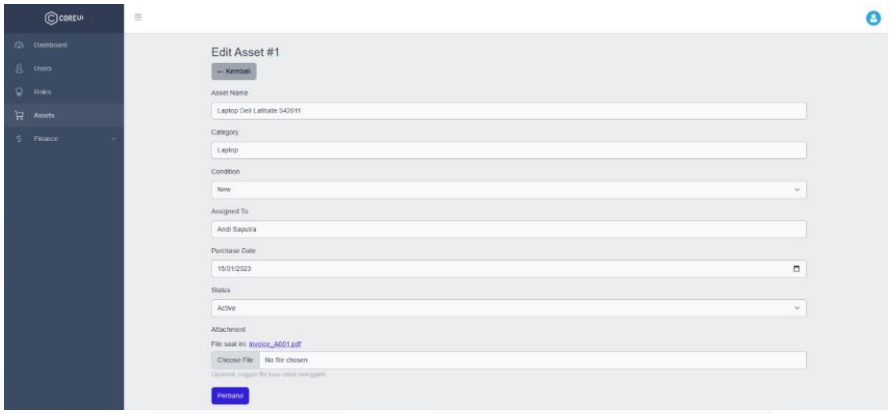
Flowchart pada Gambar 3.13 ini menggambarkan alur proses penghapusan aset yang dimulai dari tahap awal hingga selesai. Proses

dimulai ketika pengguna mengakses tampilan halaman daftar aset (*Asset List*) dari menu utama. Pada halaman ini, pengguna dapat memilih untuk melakukan dua tindakan utama: menghapus aset dengan mengklik logo *trash can* (tempat sampah), atau kembali ke menu sebelumnya dengan mengklik tombol kembali. Jika pengguna memilih untuk menghapus aset dengan mengklik logo *trash can*, sistem akan langsung menampilkan dialog konfirmasi untuk memastikan bahwa pengguna benar-benar ingin menghapus aset tersebut. Tahap konfirmasi ini sangat penting untuk mencegah penghapusan data yang tidak disengaja atau kesalahan operasional. Pada tahap konfirmasi, pengguna akan dihadapkan pada pilihan "Yes" atau "No". Jika pengguna memilih "No", sistem akan membatalkan proses penghapusan dan mengarahkan kembali ke tombol kembali, memberikan kesempatan untuk kembali ke halaman daftar aset tanpa melakukan perubahan apapun. Namun jika pengguna memilih "Yes", sistem akan melanjutkan proses penghapusan dengan mengklik tombol perbarui untuk mengeksekusi perintah hapus. Setelah tombol perbarui diklik, sistem akan memproses penghapusan aset dari database dan menampilkan status "Aset Terhapus" sebagai konfirmasi bahwa operasi penghapusan telah berhasil dilakukan. Setelah proses penghapusan selesai, alur kerja akan berakhir dan pengguna dapat melanjutkan ke aktivitas lainnya atau kembali ke menu utama sistem untuk mengelola aset lainnya.

```
@extends('layouts.app')

@section('content')
<div class="container">
    <h3>Edit Asset #{{ $asset->id }}</h3>
    <a href="{{ route('assets.index') }}" class="btn btn-secondary mb-3">← Kembali</a>

    @if ($errors->any())
        <div class="alert alert-danger">
            <ul class="mb-0">
                @foreach ($errors->all() as $error)
                    <li>{{ $error }}</li>
                @endforeach
            </ul>
        </div>
    @endif

    <form action="{{ route('assets.update', $asset->id) }}" method="POST" enctype="multipart/form-data">
        @csrf
        @method('PUT')

        <!-- Asset Name -->
        <div class="mb-3">
            <label for="name" class="form-label">Asset Name</label>
            <input type="text" name="name" id="name" class="form-control"
                   value="{{ old('name', $asset->name) }}" required>
        </div>

        <!-- Category -->
        <div class="mb-3">
            <label for="category" class="form-label">Category</label>
            <input type="text" name="category" id="category" class="form-control"
                   value="{{ old('category', $asset->category) }}" required>
        </div>
```

Gambar 3.14 Coding Edit List Asset

Kode yang ditampilkan pada Gambar 3.14 merupakan halaman *Edit Asset* (edit.blade.php) yang digunakan untuk memperbarui data aset yang sudah terdaftar dalam sistem. Dibangun menggunakan *Blade templating engine* dari *Laravel*, halaman ini mewarisi tampilan utama dari layouts.app dan menyediakan formulir yang ditujukan ke route assets.update dengan parameter ID aset. Meskipun menggunakan metode POST, terdapat @method('PUT') yang menunjukkan bahwa operasi ini adalah permintaan pembaruan data sesuai standar RESTful. Formulir ini memuat input untuk *Asset Name*, *Category*, dan *Condition*, yang telah di-*pre-filled* dengan data sebelumnya menggunakan fungsi old() dan fallback ke $asset->nama_kolom. Validasi error juga ditampilkan secara otomatis jika pengguna menginput data yang tidak sesuai. Elemen select pada *Condition* juga disesuaikan dengan kondisi aset saat ini.



Gambar 3.15 *Edit Asset Page*

Gambar 3.15 menampilkan halaman *"Edit Asset #1"* yang merupakan *interface* untuk memodifikasi data aset yang telah terdaftar dalam sistem. Halaman ini menampilkan *form editing* yang telah terisi sebelumnya dengan data yang sudah ada dari aset bernama *"Laptop Dell Latitude 542011"*, yang dikategorikan sebagai *"Laptop"* dengan kondisi *"New"*. *Form* ini memiliki struktur yang sangat terorganisir, dengan *field-field* yang sudah *pre-populated*, sehingga memudahkan pengguna dalam melakukan perubahan data secara efisien. Aset yang sedang diedit ini tercatat dimiliki oleh *"Andi Saputra"* dengan tanggal pembelian *"15/01/2023"* dan status *"Active"*, yang menunjukkan bahwa aset ini masih aktif digunakan dalam kegiatan operasional perusahaan. Tampilan *interface* untuk mengedit aset ini dirancang dengan pendekatan yang *user-friendly*, di mana setiap *field* dilengkapi dengan *label* yang jelas dan jenis input yang sesuai. *Field "Asset Name"* menggunakan *text input* agar pengguna dapat mengubah nama aset dengan leluasa. Sedangkan *field "Category"* dan *"Condition"* menggunakan *dropdown*, untuk memastikan keseragaman dan konsistensi data yang dimasukkan. *Field "Assigned To"* memungkinkan pengguna untuk memindahkan kepemilikan aset dengan mudah, yang penting dalam manajemen aset perusahaan apabila terjadi perubahan struktur organisasi atau perpindahan karyawan. Tanggal pembelian menggunakan *date picker*,

memudahkan pemilihan tanggal yang akurat. Sementara itu, status aset juga menggunakan *dropdown* untuk menjamin pilihan status yang valid.

Bagian *attachment* pada *form* ini menunjukkan adanya integrasi dengan sistem manajemen dokumen, di mana aset tersebut sudah memiliki file *"saat ini: invoice_A001.pdf"* yang dapat diakses melalui tautan yang tersedia. Fitur *"Choose File"* memungkinkan pengguna untuk mengganti atau menambahkan dokumen baru sebagai pendukung, dengan batas ukuran maksimal 2MB. Informasi tambahan juga diberikan untuk menjelaskan bahwa fitur ini bersifat opsional. Tombol *"Perbarui"* berwarna biru mencolok menjadi *call-to-action* utama untuk menyimpan perubahan yang telah dilakukan oleh pengguna. Secara keseluruhan, desain *interface* ini mengikuti prinsip-prinsip *UX* yang baik, dengan hierarki informasi yang jelas, skema warna yang konsisten dengan identitas visual *CoreUI*, serta navigasi yang intuitif melalui keberadaan *breadcrumb "← Kembali"* yang membantu pengalaman pengguna secara menyeluruh. Keempat komponen yang ditampilkan dari gambar-gambar sebelumnya membentuk sebuah *ecosystem* yang saling terintegrasi dalam manajemen aset perusahaan. *Dashboard* berfungsi sebagai titik awal yang memberikan gambaran umum dan mengarahkan perhatian pengguna ke area yang memerlukan tindakan. *Asset List Page* menyediakan tampilan data yang lebih rinci dan akses penuh terhadap seluruh aset, sementara *Add New Asset Page* memfasilitasi penambahan aset baru ke dalam sistem. *Edit Asset Page* memberikan kemampuan untuk memperbarui dan menjaga akurasi data aset secara berkelanjutan. Alur kerja (*workflow*) yang dimulai dari *Dashboard* sebagai pusat informasi dan *insight*, kemudian mengarahkan pengguna ke *Asset List* untuk penelusuran lebih detail, dan dilanjutkan ke halaman *Add* atau *Edit* untuk manipulasi data, menciptakan sistem yang efisien dan logis. Konsistensi pada *design pattern*, skema warna,

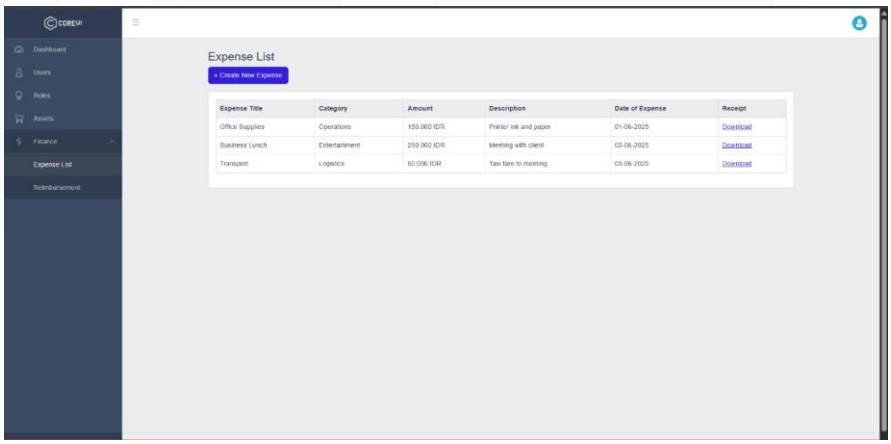
serta elemen *user interface* di seluruh halaman menciptakan pengalaman pengguna (*user experience*) yang terpadu, sehingga mempermudah adopsi sistem serta penggunaan sehari-hari oleh tim operasional.

Implementasi sistem pencatatan inventaris dan pembelian barang pada minggu keempat ini memberikan berbagai manfaat yang signifikan bagi perusahaan. Dari sisi efisiensi operasional, sistem ini mengotomatisasi proses pencatatan dan pelacakan aset yang sebelumnya dilakukan secara manual, mengurangi risiko kesalahan manusia dan meningkatkan akurasi data. Dari sisi visibilitas manajemen, *dashboard* menyediakan *real-time insights* yang sangat membantu dalam pengambilan keputusan mengenai penggunaan aset, penjadwalan perawatan, dan perencanaan pengadaan. Dari segi kepatuhan (*compliance*) dan kesiapan audit, sistem ini menyediakan *audit trail* yang lengkap serta dokumentasi yang dibutuhkan untuk kepatuhan terhadap regulasi dan keperluan audit internal. Sementara itu, dari sisi pengelolaan biaya, pelacakan aset yang lebih baik serta pemantauan kondisi secara berkala memungkinkan optimalisasi siklus hidup aset, penjadwalan perawatan secara prediktif, serta perhitungan *ROI* (*Return on Investment*) yang lebih akurat untuk perencanaan investasi di masa depan.

3.2.5 Penyempurnaan dan Pengujian Fitur Pembelian dan Permintaan Barang

Pada tahap ini, dilakukan penyempurnaan dan pengujian menyeluruh terhadap fitur *Expense Management System* yang telah dikembangkan pada minggu-minggu sebelumnya. Seluruh komponen sistem manajemen pengeluaran telah memasuki fase *refinement* dan *quality assurance*, di mana setiap aspek mulai dari *fungsionalitas*, *user interface*, hingga *user experience* ditinjau secara mendalam untuk memastikan sistem dapat beroperasi secara optimal dalam lingkungan produksi. Tim melakukan serangkaian pengujian

menyeluruh, mulai dari *unit testing*, *integration testing*, hingga *user acceptance testing* untuk memastikan bahwa semua fitur dalam sistem pengeluaran berjalan sesuai dengan *requirement* yang telah ditentukan. Proses *debugging* dan *optimasi* dilakukan secara intensif untuk memperbaiki *bug* minor yang ditemukan selama proses pengujian, sementara *performance tuning* dilakukan untuk menjamin sistem mampu menangani volume data besar dengan *response time* yang tetap dapat diterima.



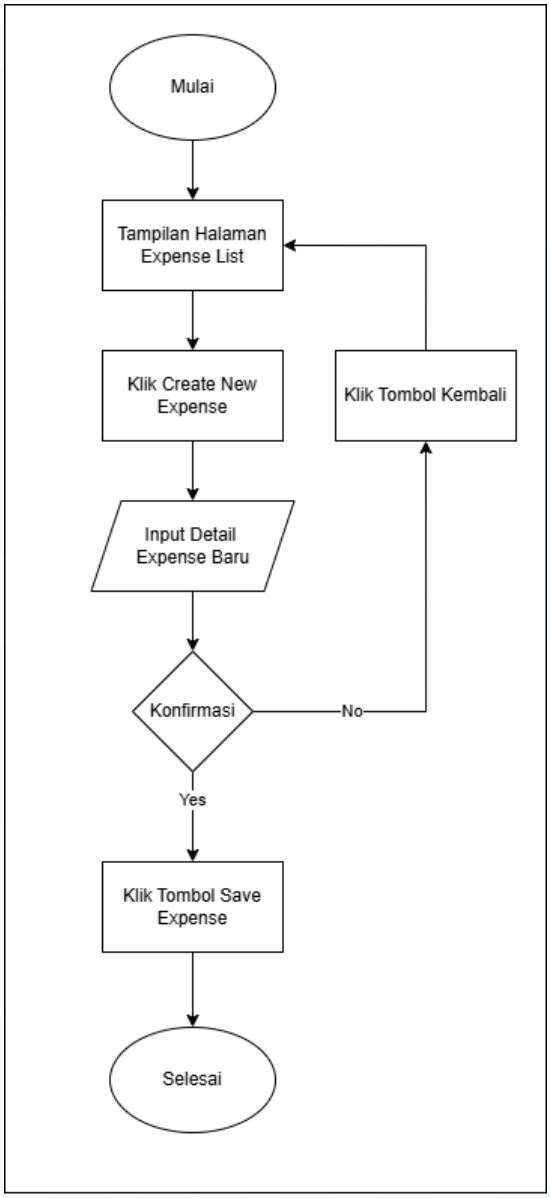
Gambar 3.16 *Expense List Page*

Gambar 3.16 diatas merupakan halaman *Expense List* merupakan tampilan utama yang menampilkan daftar lengkap seluruh pengeluaran yang telah direkam dalam sistem berbasis *CoreUI*. Halaman ini berfungsi sebagai *dashboard central* untuk melakukan pemantauan dan pengelolaan aktivitas keuangan perusahaan. *Interface* ini dirancang dengan menggunakan *framework CoreUI* yang menghadirkan tampilan yang bersih, profesional, dan mudah digunakan, mengikuti prinsip desain yang terbukti efektif dalam aplikasi berskala *enterprise*. Pada bagian atas halaman terdapat judul *"Expense List"* yang ditampilkan secara menonjol untuk memberikan konteks yang jelas kepada pengguna mengenai fungsi halaman tersebut, dengan tipografi yang konsisten menggunakan *font family* sesuai *design system* milik CoreUI. Di sisi kanan judul, terdapat tombol *"+ Create New Expense"* berwarna biru mencolok, dengan

gaya sesuai panduan *CoreUI button component*, yang memudahkan pengguna menambahkan entri pengeluaran baru tanpa harus melakukan navigasi yang rumit. *Layout* halaman ini dibangun dengan sistem *responsive grid* yang memastikan tampilan optimal pada berbagai perangkat, mulai dari desktop hingga *mobile device*. Tabel yang disajikan memiliki struktur yang rapi dan informatif, dengan kolom-kolom penting seperti *Expense Title*, *Category*, *Amount*, *Description*, *Date of Expense*, dan *Receipt*. Setiap kolom telah diatur dengan lebar proporsional agar informasi dapat ditampilkan secara lengkap tanpa memerlukan *horizontal scrolling* yang berlebihan. Header tabel menggunakan *styling* yang selaras dengan *theme CoreUI*, dengan *background color* yang cukup kontras untuk meningkatkan keterbacaan.

Setiap baris pada tabel mewakili satu entri pengeluaran yang berbeda, memberikan gambaran lengkap tentang aktivitas finansial perusahaan dalam format yang mudah dibaca dan dianalisis. Contoh data yang ditampilkan mencakup “Office Supplies” dengan kategori *Operations* senilai 150.000 IDR untuk pembelian tinta dan kertas printer pada tanggal 01-06-2025, “Business Lunch” dengan kategori *Entertainment* senilai 250.000 IDR untuk pertemuan dengan klien pada 02-06-2025, serta “Transport” dengan kategori *Logistics* senilai 60.000 IDR untuk biaya taksi menuju lokasi rapat pada 03-06-2025. Data tersebut menggambarkan variasi jenis pengeluaran yang dikelola oleh sistem, mulai dari kebutuhan operasional hingga aktivitas pengembangan bisnis. Fitur pemformatan pada kolom *Amount* menggunakan *currency formatter* yang sesuai dengan standar mata uang Rupiah, dengan pemisah ribuan untuk meningkatkan keterbacaan angka. Kolom *Date of Expense* menggunakan format tanggal *DD-MM-YYYY* yang konsisten dan sesuai dengan kebiasaan regional di Indonesia. Setiap entri juga dilengkapi dengan tautan *"Download"* pada kolom *Receipt*, yang memungkinkan pengguna

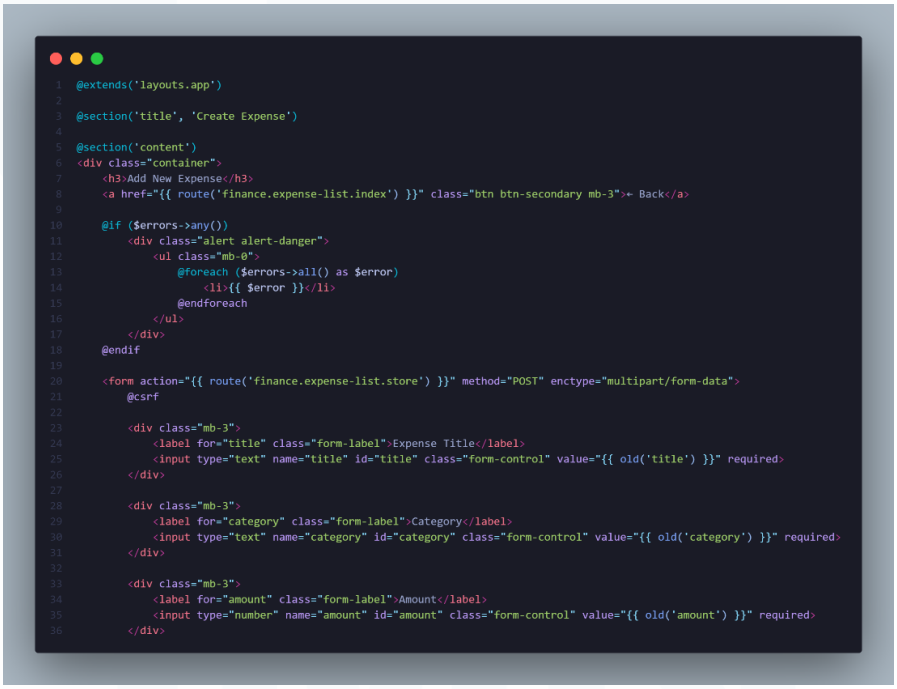
mengakses dokumen pendukung seperti nota, kwitansi, atau bukti pembayaran lainnya. Fitur ini sangat penting untuk keperluan audit, pelaporan keuangan, dan kepatuhan terhadap regulasi perpajakan di Indonesia. Desain tabel mengikuti *styling* yang konsisten dengan tema CoreUI, dengan header berwarna abu-abu terang serta baris data yang menggunakan pola warna selang-seling (*alternating row colors*) untuk meningkatkan keterbacaan dan mempermudah pelacakan data saat melakukan *scrolling*. *Hover effects* diterapkan pada setiap baris sebagai umpan balik visual yang responsif ketika pengguna berinteraksi. Navigasi *sidebar* di sisi kiri menampilkan menu lengkap seperti *Dashboard*, *Users*, *Roles*, *Assets*, dan *Finance*, dengan sub-menu *Expense List* dan *Reimbursement*, menunjukkan bahwa halaman ini merupakan bagian integral dari sistem manajemen keuangan yang lebih luas dan komprehensif. Sistem juga dilengkapi dengan fitur *sorting* dan *filtering* yang memungkinkan pengguna menyusun data berdasarkan kriteria seperti tanggal, kategori, atau jumlah pengeluaran. Kontrol *pagination* tersedia di bagian bawah tabel untuk mengelola dataset besar tanpa mengganggu performa sistem. Tak lupa, *search functionality* juga diintegrasikan agar pengguna dapat mencari data pengeluaran tertentu berdasarkan kata kunci pada *Expense Title* maupun *Description*.



Gambar 3.17 *Flowchart Create New Expense List*

Flowchart pada Gambar 3.17 diatas menjelaskan alur aktivitas pengguna saat melakukan pencatatan pengeluaran baru dalam sistem. Proses dimulai dengan pengguna membuka halaman *Expense List*, yaitu halaman utama yang menampilkan seluruh data pengeluaran perusahaan. Dari halaman ini, pengguna dapat memilih untuk menambahkan data baru dengan menekan tombol *Create New Expense*. Setelah itu, sistem akan menampilkan form untuk mengisi detail pengeluaran baru, seperti judul pengeluaran, kategori, jumlah

nominal, deskripsi, tanggal pengeluaran, dan dokumen pendukung. Setelah seluruh informasi diinput, sistem akan meminta konfirmasi dari pengguna. Jika pengguna belum yakin dan memilih “No”, maka sistem akan memberikan opsi untuk kembali dengan menekan tombol *Kembali*. Namun, jika pengguna menyetujui input tersebut dengan memilih “Yes”, maka pengguna akan diarahkan untuk menekan tombol *Save Expense* guna menyimpan data pengeluaran ke dalam sistem. Setelah data disimpan, proses pencatatan pengeluaran dianggap selesai.

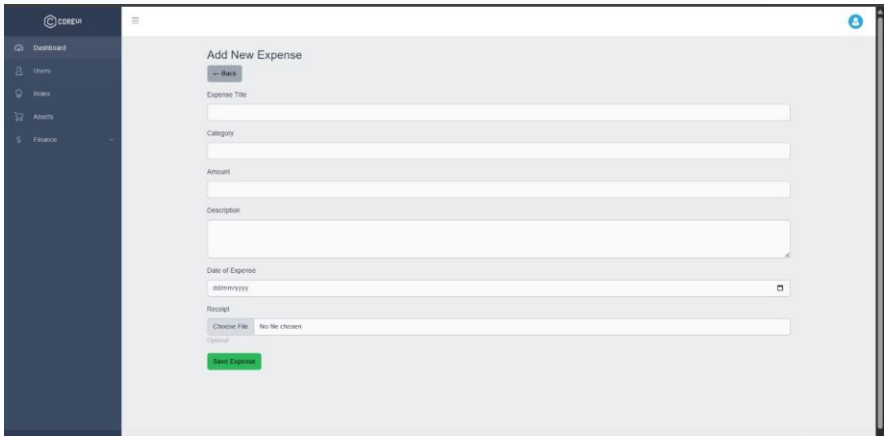


```
@extends('layouts.app')

@section('title', 'Create Expense')

@section('content')
<div class="container">
    <h3>Add New Expense</h3>
    <a href="{{ route('finance.expense-list.index') }}" class="btn btn-secondary mb-3">← Back</a>

    @if ($errors->any())
        <div class="alert alert-danger">
            <ul class="mb-0">
                @foreach ($errors->all() as $error)
                    <li>{{ $error }}</li>
                @endforeach
            </ul>
        </div>
    @endif

    <form action="{{ route('finance.expense-list.store') }}" method="POST" enctype="multipart/form-data">
        @csrf

        <div class="mb-3">
            <label for="title" class="form-label">Expense Title</label>
            <input type="text" name="title" id="title" class="form-control" value="{{ old('title') }}" required>
        </div>

        <div class="mb-3">
            <label for="category" class="form-label">Category</label>
            <input type="text" name="category" id="category" class="form-control" value="{{ old('category') }}" required>
        </div>

        <div class="mb-3">
            <label for="amount" class="form-label">Amount</label>
            <input type="number" name="amount" id="amount" class="form-control" value="{{ old('amount') }}" required>
        </div>
```

Gambar 3.18 *Coding Create New Expense List*

Gambar 3.18 merupakan halaman *Create Expense* (create.blade.php) yang digunakan untuk menambahkan data pengeluaran baru pada sistem manajemen keuangan berbasis *Laravel*. File ini menggunakan *Blade templating engine* dengan memperluas layout utama (layouts.app) dan menetapkan konten pada bagian title dan content. Di dalamnya terdapat sebuah form yang mengirim data ke route bernama *finance.expense-list.store* menggunakan metode POST dan mendukung pengiriman file dengan

enctype="multipart/form-data". Fitur keamanan *CSRF token* juga disertakan untuk mencegah serangan *Cross-Site Request Forgery*. Formulir ini terdiri dari beberapa input field, antara lain: *Expense Title*, *Category*, *Amount*, dan *Description*. Masing-masing input dilengkapi dengan validasi untuk memastikan data wajib diisi. Jika terdapat kesalahan saat pengisian form, sistem akan menampilkan notifikasi kesalahan secara otomatis menggunakan $errors->any() dan foreach untuk menampilkan setiap pesan error. Dengan tampilan yang sederhana namun fungsional, halaman ini memudahkan pengguna untuk mencatat pengeluaran secara efisien dan aman.



Gambar 3.19 *Add New Expense Page*

Gambar 3.19 merupakan halaman *Add New Expense* merupakan sebuah *form interface* yang dirancang secara intuitif dan *user-friendly* untuk memfasilitasi proses penambahan data pengeluaran baru ke dalam sistem dengan tingkat akurasi dan kelengkapan data yang tinggi. Halaman ini dapat diakses melalui tombol *"Create New Expense"* yang terdapat pada halaman *Expense List*, dan menampilkan sebuah *formulir komprehensif* berisi berbagai *field* yang diperlukan untuk mencatat pengeluaran secara akurat, lengkap, serta sesuai dengan standar akuntansi dan audit yang berlaku. Pada bagian atas halaman terdapat tombol *"← Back"* dengan ikon dan teks yang jelas, memberikan kemudahan bagi pengguna untuk kembali ke halaman sebelumnya tanpa kehilangan konteks atau data

yang telah diisi. *Formulir* ini menggunakan *single-column layout* yang optimal baik untuk tampilan *desktop* maupun *mobile*, dengan *spacing* yang tepat antar *field* untuk menciptakan *visual hierarchy* yang jelas. *Design system* yang digunakan mengikuti *framework CoreUI*, memanfaatkan *form controls* yang telah terbukti efektif dan familiar bagi pengguna aplikasi web modern. Setiap *field* dilengkapi dengan *label* yang deskriptif dan posisi yang mengikuti *best practices* dalam desain *form*, sehingga meningkatkan *usability* dan mengurangi tingkat kesalahan saat pengguna mengisi data.

Field pertama adalah *"Expense Title"* yang berupa *text input* untuk memberikan nama atau judul pengeluaran. *Field* ini dilengkapi dengan aturan *validasi* agar tidak kosong dan memiliki panjang karakter yang sesuai dengan kebutuhan pelaporan serta penyimpanan di *database*. Tersedia juga *placeholder text* yang menjadi panduan bagi pengguna terkait format atau isi informasi yang diharapkan. Selanjutnya, *field "Category"* dirancang sebagai *dropdown* atau *select input* yang berisi *pre-defined categories* yang telah ditetapkan dalam sistem, seperti *Operations*, *Entertainment*, *Logistics*, *Marketing*, *Travel*, *Equipment*, dan kategori lainnya sesuai kebutuhan bisnis perusahaan. Sistem ini juga mendukung penambahan kategori baru seiring berkembangnya kebutuhan operasional. *Field "Amount"* menggunakan *numeric input* yang dilengkapi dengan *validation* agar hanya angka yang dapat diinput. Format otomatis menambahkan pemisah ribuan untuk meningkatkan keterbacaan angka. *Validation rules* memastikan nilai yang dimasukkan tidak negatif dan berada dalam rentang yang masuk akal. Selain itu, simbol mata uang dan jumlah desimal dapat dikonfigurasi sesuai standar perusahaan. *Field "Description"* merupakan *textarea* yang luas, memungkinkan pengguna menjelaskan rincian pengeluaran, tujuan pembelian, informasi vendor, atau konteks lain yang relevan. *Field* ini mendukung fitur *character counter* sebagai panduan panjang teks

optimal, serta *auto-resize functionality* untuk menyesuaikan tinggi bidang input berdasarkan banyaknya isi yang diketik. Untuk tanggal, *field "Date of Expense"* menggunakan *date picker component* yang canggih dengan tampilan kalender, memudahkan pemilihan tanggal secara akurat. Komponen ini dilengkapi dengan kontrol navigasi untuk berpindah bulan atau tahun dan *validation* untuk mencegah pengguna memilih tanggal di masa depan secara tidak sengaja. Format tanggal disesuaikan dengan preferensi regional, yaitu *DD-MM-YYYY*.

Salah satu fitur penting dalam *formulir* ini adalah *field "Receipt"* yang memungkinkan pengguna mengunggah dokumen pendukung dalam format seperti PDF, JPG, PNG, atau jenis file lainnya sesuai konfigurasi sistem. *File upload component* yang digunakan memiliki fitur modern seperti *drag-and-drop*, *upload progress indicator*, dan *image preview*. Sistem juga menjalankan *validation* terhadap ukuran dan jenis file, serta mendukung *virus scanning* untuk menjaga keamanan dokumen. Fitur unggah bukti pembayaran ini sangat penting untuk *compliance* dan membangun *audit trail*, karena memungkinkan perusahaan menyimpan bukti fisik pengeluaran dalam bentuk digital yang mudah diakses dan dicari kembali. Meskipun bersifat opsional—sebagaimana dicantumkan dalam label "*Optional*"—pengguna tetap didorong untuk mengunggah dokumen demi meningkatkan akurasi dan kelengkapan dokumentasi keuangan. Di bagian bawah *formulir*, terdapat tombol *"Save Expense"* berwarna hijau mencolok, dirancang sesuai dengan *CoreUI button component guidelines*, yang menjadi *call-to-action* utama untuk menyimpan data ke sistem. Tombol ini dilengkapi dengan *loading state* dan *confirmation dialog* untuk memberikan *feedback* yang sesuai kepada pengguna selama proses penyimpanan. Selain itu, sistem menyediakan *client-side validation* secara *real-time* yang memberikan notifikasi langsung jika ada *field* yang belum lengkap atau salah format. Desain keseluruhan *formulir* ini mengikuti

prinsip-prinsip *UX* yang baik, dengan *spacing* yang nyaman, *label* yang jelas dan informatif, ukuran *field* yang proporsional, dan *visual hierarchy* yang memandu pengguna secara intuitif. Konsistensi terhadap tema *CoreUI* dijaga melalui penggunaan *color scheme*, *typography*, dan *styling* yang seragam dengan komponen lain dalam aplikasi. Penanganan kesalahan dan pesan validasi juga disusun dengan baik agar pengguna mendapatkan panduan yang membantu dalam mengatasi masalah saat mengisi data.

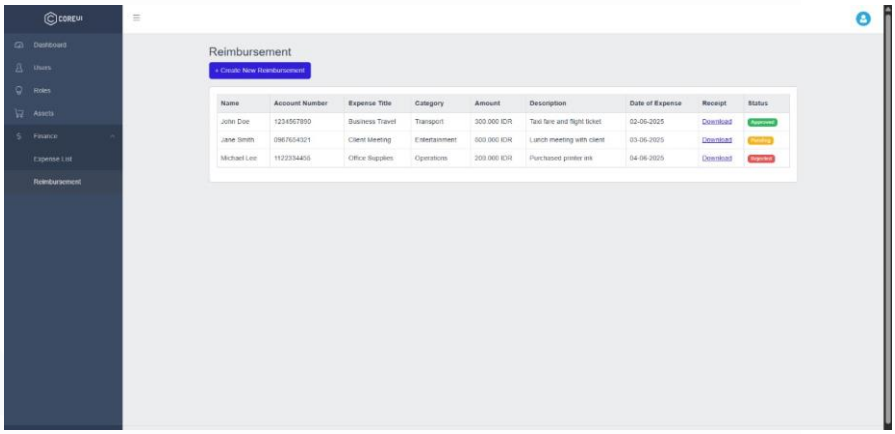
3.2.6 Penambahan Fitur Persetujuan Pembelian dan Reimbursement

Memasuki minggu keenam dan ketujuh dari roadmap pengembangan sistem CoreUI, tim development mulai mengimplementasikan salah satu fitur paling krusial dan kompleks dalam sistem manajemen keuangan perusahaan, yaitu fitur Reimbursement Management System. Fase ini menandai perluasan besar dari sistem pencatatan pengeluaran sederhana menjadi sebuah sistem manajemen keuangan yang comprehensive, melibatkan banyak pemangku kepentingan (multiple stakeholders) serta proses persetujuan (approval process) yang kompleks dan terstruktur. Tim melakukan analisis mendalam terhadap kebutuhan bisnis dalam proses reimbursement, termasuk studi terhadap prosedur manual yang telah ada, identifikasi hambatan dalam workflow saat ini, dan perancangan hierarki persetujuan yang mampu mengakomodasi berbagai tingkat otorisasi sesuai dengan struktur organisasi perusahaan.

Pengembangan fitur reimbursement ini menuntut integrasi yang mulus (seamless) dengan sistem Expense Management yang telah dikembangkan sebelumnya, agar data consistency dan referential integrity tetap terjaga. Tim architect melakukan tinjauan dan modifikasi menyeluruh terhadap skema database untuk menambahkan entities baru seperti permintaan reimbursement, alur

persetujuan, informasi akun pengguna, dan pelacakan status untuk membentuk audit trail yang lengkap. Pertimbangan keamanan juga menjadi prioritas utama, mengingat fitur ini menangani data keuangan dan informasi pribadi karyawan yang sensitif, sehingga penerapan role-based access control dan data encryption menjadi syarat wajib.

Selama dua minggu ini, tim juga melakukan parallel development untuk user interface components yang akan mendukung alur kerja reimbursement, termasuk perancangan approval dashboard untuk manajer, sistem notifikasi untuk real-time status updates, dan modul pelaporan yang dapat menghasilkan reimbursement report berdasarkan berbagai kriteria seperti rentang waktu, departemen, karyawan, atau status persetujuan. Fokus utama pada tahap ini adalah integration testing untuk memastikan fitur reimbursement yang baru dapat berjalan selaras dengan fungsi-fungsi pengelolaan pengeluaran yang telah ada sebelumnya, tanpa menyebabkan regression issues atau penurunan performa (performance degradation).



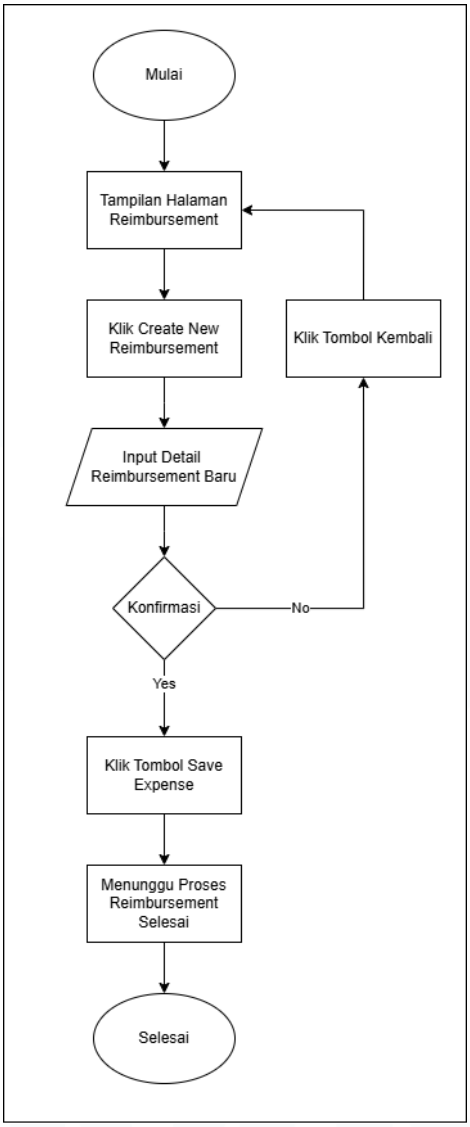
Gambar 3.20 *Reimburse List Page*

Gambar 3.20 merupakan halaman *Reimbursement List* merupakan *central dashboard* yang dirancang untuk memberikan *comprehensive overview* dari seluruh permintaan *reimbursement* yang telah diajukan oleh karyawan dalam organisasi. Halaman ini berfungsi sebagai *command center* untuk departemen HR, tim keuangan, dan

manajer yang bertanggung jawab atas *approval process*. *Interface* halaman ini mengadopsi *advanced table design* dengan kemampuan *enhanced filtering* dan *sorting* yang memungkinkan pengguna dengan mudah menavigasi volume besar data *reimbursement* berdasarkan berbagai kriteria dan parameter sesuai kebutuhan bisnis. Struktur tabel pada halaman ini secara signifikan lebih kompleks dibandingkan halaman *Expense List*, dengan kolom tambahan yang secara spesifik dirancang untuk mendukung manajemen alur kerja *reimbursement*. Kolom utama meliputi *Name* (nama karyawan), *Account Number* (untuk *direct transfer*), *Expense Title* (deskripsi pengeluaran), *Category* (jenis pengeluaran), *Amount* (jumlah penggantian dalam IDR), *Description* (penjelasan rinci), *Date of Expense* (tanggal terjadinya pengeluaran), *Receipt* (dokumen pendukung), dan kolom *Status* yang menunjukkan status terkini dari permintaan *reimbursement* dalam *workflow* persetujuan. Bagian *header* menampilkan judul *"Reimbursement"* dengan *typography* yang mencolok, memberikan konteks yang jelas kepada pengguna. Tombol *"+ Create New Reimbursement"* ditempatkan secara strategis di sebelah kanan dengan *blue accent color* yang konsisten dengan *CoreUI design system*, memungkinkan pengguna yang diotorisasi untuk mengajukan permintaan baru hanya dengan sekali klik. Penempatan dan gaya tombol mengikuti *UI patterns* yang familiar, mengurangi *cognitive load* dan meningkatkan pengalaman pengguna secara keseluruhan.

Data yang ditampilkan pada tabel mencerminkan *realistic business scenarios*, seperti John Doe yang mengajukan *reimbursement* untuk *Business Travel* dengan kategori *Transport* senilai 500.000 IDR pada tanggal 02-06-2025 dengan status *"Approved"* (ditandai dengan *green badge*), Jane Smith untuk *Client Meeting* kategori *Entertainment* senilai 300.000 IDR pada 03-06-2025

dengan status *"Pending"* (*yellow badge*), dan Michael Lee untuk *Office Supplies* kategori *Operations* senilai 200.000 IDR pada 04-06-2025 dengan status *"Rejected"* (*red badge*). Indikator status ini menggunakan sistem *color-coded badges* yang secara langsung mengkomunikasikan status permintaan kepada pengguna. *Green badges* menunjukkan status *"Approved"* dan menandakan bahwa permintaan telah disetujui dan siap diproses pembayarannya. *Yellow badges* untuk *"Pending"* menciptakan rasa urgensi bahwa permintaan masih dalam proses tinjauan. *Red badges* untuk *"Rejected"* menunjukkan hasil negatif dan biasanya memerlukan dokumentasi tambahan sebelum dapat diajukan ulang. Kolom *Account Number* menampilkan informasi rekening bank dengan tampilan yang aman dan format konsisten, di mana sebagian angka disembunyikan untuk perlindungan privasi. *Amounts* ditampilkan menggunakan format mata uang dengan denominasi IDR dan pemisah ribuan untuk keterbacaan yang lebih baik. Format tanggal menggunakan standar *DD-MM-YYYY* yang familiar bagi pengguna di Indonesia. Tautan *Download* di kolom *Receipt* memungkinkan personel yang berwenang mengakses dokumen pendukung yang diunggah bersamaan dengan permintaan *reimbursement*. Dokumen ini sangat penting untuk proses persetujuan dan audit, sebagai bukti dan justifikasi jumlah yang diminta. *File access control* memastikan hanya pengguna tertentu yang dapat melihat dokumen keuangan sensitif, menjaga keamanan data dan kepatuhan terhadap kebijakan perusahaan. Navigasi *sidebar* menampilkan struktur menu hierarkis dengan bagian *Finance* yang terbuka menampilkan *Expense List* dan *Reimbursement* sebagai *sub-menus*, menandakan integrasi yang erat antar fitur dalam sistem manajemen keuangan perusahaan. Desain navigasi ini memudahkan pengguna untuk berpindah antar modul keuangan tanpa kehilangan konteks atau harus melalui jalur navigasi yang rumit.



Gambar 3.21 *Flowchart Create New Reimbursement*

Gambar 3.21 diatas menggambarkan *Flowchart* aktivitas pengguna saat melakukan pengajuan reimbursement pada sistem. Proses dimulai dari pengguna yang membuka halaman Reimbursement. Setelah halaman terbuka, pengguna akan menekan tombol *Create New Reimbursement* untuk memulai proses pengajuan. Sistem kemudian akan menampilkan form input untuk mengisi detail reimbursement yang baru, seperti judul pengeluaran, kategori, jumlah dana yang diminta, deskripsi, tanggal pengeluaran, serta dokumen bukti pembayaran. Setelah seluruh data diisi, sistem akan meminta

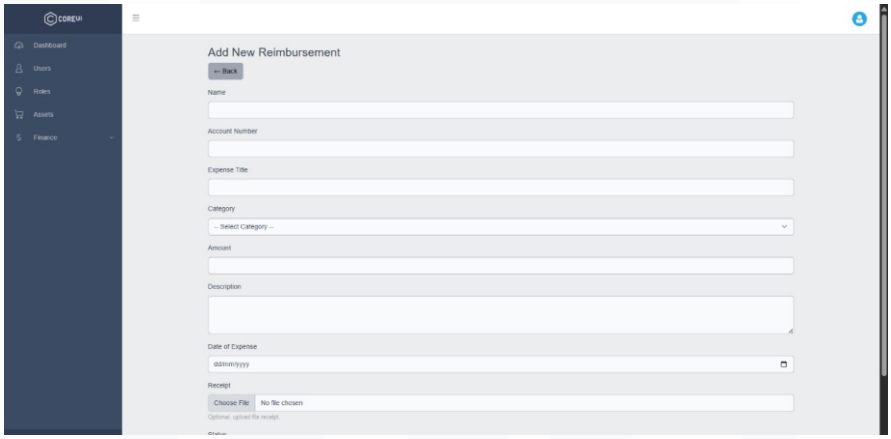
konfirmasi dari pengguna. Jika pengguna menekan “No” pada tahap konfirmasi, maka sistem akan memberikan opsi untuk kembali ke halaman sebelumnya dengan menekan tombol Kembali. Namun, jika pengguna menekan “Yes”, maka sistem akan melanjutkan proses dengan menyimpan data melalui tombol *Save Expense*. Setelah tombol ini diklik, sistem akan memproses permintaan reimbursement dan menampilkannya dalam daftar permintaan yang menunggu proses persetujuan. Setelah proses selesai, aktivitas dianggap berakhir dan pengguna kembali ke alur utama sistem. *Flowchart* ini menunjukkan alur logis dan sederhana yang dapat memandu pengguna dalam proses pengajuan reimbursement secara efisien dan terstruktur.

```
@extends('layouts.app')

@section('title', 'Create Reimbursement')

@section('content')
<div class="container">
    <h3>Add New Reimbursement</h3>
    <a href="{{ route('finance.reimbursement.index') }}" class="btn btn-secondary mb-3">← Back</a>

    @if ($errors->any())
        <div class="alert alert-danger">
            <ul class="mb-0">
                @foreach ($errors->all() as $error)
                    <li>{{ $error }}</li>
                @endforeach
            </ul>
        </div>
    @endif

    <form action="{{ route('finance.reimbursement.store') }}" method="POST" enctype="multipart/form-data">
        @csrf

        <!-- Name -->
        <div class="mb-3">
            <label for="name" class="form-label">Name</label>
            <input type="text" name="name" id="name" class="form-control" value="{{ old('name') }}" required>
        </div>

        <!-- Account Number -->
        <div class="mb-3">
            <label for="account_number" class="form-label">Account Number</label>
            <input type="text" name="account_number" id="account_number" class="form-control" value="{{ old('account_number') }}" required>
        </div>

        <!-- Expense Title -->
        <div class="mb-3">
            <label for="title" class="form-label">Expense Title</label>
            <input type="text" name="title" id="title" class="form-control" value="{{ old('title') }}" required>
        </div>
```

Gambar 3.22 *Coding Create New Reimbursement*

Gambar 3.22 diatas merupakan halaman *Create Reimbursement* (create.blade.php) yang digunakan untuk mencatat permintaan penggantian biaya (*reimbursement*) pada sistem keuangan. Halaman ini dibuat menggunakan *Blade templating engine* dari *Laravel*, dan memperluas template utama (layouts.app) untuk menjaga konsistensi tampilan. Formulir ini mengarah ke route *finance.reimbursement.store* dan menggunakan metode POST serta

enctype="multipart/form-data" untuk memungkinkan pengiriman data kompleks. Token @csrf digunakan untuk keamanan dari serangan *Cross-Site Request Forgery*. Formulir terdiri dari beberapa input field penting seperti *Name*, *Account Number*, dan *Expense Title*, yang semuanya wajib diisi dan menggunakan fungsi old() untuk menjaga data tetap tersimpan jika terjadi kesalahan input. Jika terdapat error validasi, akan ditampilkan secara otomatis dalam kotak notifikasi berwarna merah menggunakan struktur @if ($errors->any()). Dengan desain yang sederhana namun fungsional, halaman ini memudahkan pengguna untuk melakukan permintaan reimbursement secara terstruktur dan aman.



Gambar 3.23 *Add New* Reimbursement *Page*

Gambar 3.23 merupakan halaman *Add New Reimbursement* merupakan *sophisticated form interface* yang secara khusus dirancang untuk *facilitate submission of new reimbursement requests* dengan mekanisme pengumpulan dan validasi data yang *comprehensive* guna memastikan keakuratan dan kelengkapan dari informasi yang dikirimkan. Formulir ini merupakan titik krusial dalam *reimbursement workflow*, di mana karyawan dapat secara resmi mengajukan penggantian dana atas *business-related expenses* yang telah mereka keluarkan dari dana pribadi, memulai *formal approval process* yang akan *route* permintaan tersebut melalui *appropriate management hierarchy*. *Form design* mengikuti prinsip *progressive*

disclosure dengan urutan isian yang logis, membimbing pengguna melalui alur alami pengisian data, dimulai dari informasi identitas pribadi dan secara bertahap menuju rincian pengeluaran dan dokumen pendukung. Bagian *header* menampilkan judul yang jelas "*Add New Reimbursement*" yang secara langsung mengkomunikasikan tujuan halaman ini, disertai tombol navigasi "← *Back*" yang memungkinkan pengguna kembali ke halaman *Reimbursement List* tanpa kehilangan konteks isian atau secara tidak sengaja memicu pengiriman formulir.

Bagian utama dari formulir dimulai dengan isian *"Name"* yang menangkap identitas karyawan yang mengajukan *reimbursement request*. Isian ini dapat dikonfigurasi untuk *auto-populate* dengan informasi pengguna yang saat ini *logged-in*, mengurangi entri data manual dan potensi kesalahan dalam identifikasi pribadi. *Input validation* memastikan bahwa isian nama mengandung jenis karakter dan panjang teks yang sesuai dengan *database schema* dan kebutuhan bisnis. Isian *"Account Number"* merupakan komponen krusial yang menangkap informasi rekening bank untuk keperluan *direct transfer* saat *reimbursement* disetujui. Isian ini menggunakan *specialized input formatting* yang dapat mengakomodasi berbagai format nomor rekening bank yang digunakan dalam sistem perbankan Indonesia, dengan *built-in validation* untuk memverifikasi konsistensi format dan potensi integrasi dengan *banking APIs* untuk keperluan verifikasi rekening. Pertimbangan *security* untuk isian ini mencakup *encryption* terhadap data finansial sensitif dan *access controls* yang membatasi visibilitas hanya kepada pihak yang berwenang. Isian *"Expense Title"* menyediakan ruang untuk judul deskriptif yang merangkum sifat pengeluaran yang diajukan untuk *reimbursement*. Isian ini menggunakan *text input* dengan batas karakter agar tidak terlalu panjang namun cukup jelas. *Input suggestions* atau fungsi *auto-complete* dapat diimplementasikan berdasarkan data historis untuk meningkatkan konsistensi dan mengurangi pengetikan manual. Isian

"Category" menggunakan antarmuka *dropdown selection* dengan placeholder "-- *Select Category* --" yang menunjukkan bahwa pilihan diperlukan. *Dropdown* ini diisi dengan kategori yang telah ditentukan sebelumnya sesuai dengan sistem kategorisasi pengeluaran dalam modul manajemen pengeluaran, memastikan konsistensi dalam pelaporan dan analisis keuangan. Kategori seperti *Transport*, *Entertainment*, *Operations*, *Travel*, *Equipment*, dan lainnya dapat dikonfigurasi sesuai kebutuhan organisasi dan standar akuntansi. Isian *"Amount"* menggunakan *numeric input* dengan format mata uang yang secara otomatis menambahkan *thousand separators* dan simbol mata uang untuk keterbacaan yang lebih baik. Isian ini juga dilengkapi *validation rules* yang mencegah nilai negatif, jumlah yang terlalu besar (yang bisa jadi akibat kesalahan input), dan *formatting* yang konsisten dengan standar keuangan. Integrasi dengan aturan kebijakan pengeluaran memungkinkan validasi otomatis terhadap batas maksimum penggantian berdasarkan kategori atau level karyawan.

Isian *"Description"* menggunakan *large textarea* yang mendorong penjelasan detail mengenai alasan bisnis dari pengeluaran tersebut, konteks terjadinya pengeluaran, dan informasi tambahan lainnya yang relevan untuk proses persetujuan. Fungsi *character counting* memberikan panduan tentang panjang deskripsi yang optimal, sementara *auto-resize capability* memastikan area teks menyesuaikan dengan panjang isi secara otomatis.

Isian *"Date of Expense"* menggunakan komponen *sophisticated date picker* dengan antarmuka kalender yang memungkinkan pengguna memilih tanggal pengeluaran dengan mudah. *Validation rules* mencegah pemilihan tanggal di masa depan (karena pengeluaran harus terjadi sebelum diajukan), serta dapat mencakup aturan bisnis seperti jangka waktu maksimum pengajuan (misalnya, pengeluaran lebih dari 90 hari memerlukan persetujuan tambahan atau tidak memenuhi syarat). Isian *"Receipt"* menyediakan

mekanisme untuk melampirkan dokumen pendukung yang membuktikan klaim pengeluaran. Isian ini mendukung berbagai format file seperti PDF untuk nota formal, gambar untuk nota yang difoto, dan jenis dokumen lainnya. Fungsi *upload* mencakup *progress indicators*, validasi ukuran file, pembatasan jenis file, serta kemampuan *preview* untuk file gambar. *Security scanning* memastikan bahwa file yang diunggah tidak mengandung konten berbahaya yang dapat membahayakan sistem. Fitur lanjutan dalam unggahan *receipt* dapat mencakup *automatic OCR (Optical Character Recognition)* yang mengekstrak informasi penting dari gambar nota seperti jumlah, tanggal, dan nama vendor, untuk *automatic form population* dan validasi. Integrasi dengan basis data kebijakan pengeluaran memungkinkan pengecekan otomatis terhadap kepatuhan terhadap kebijakan perusahaan, termasuk batas pengeluaran dan dokumen yang diwajibkan. Isian *Status* (yang mungkin tidak terlihat dalam tampilan formulir namun diproses di *backend*) secara otomatis akan diatur menjadi "*Pending*" saat permintaan diajukan, memicu *approval workflow* yang akan *route* permintaan ke pihak yang berwenang sesuai dengan batas jumlah, hierarki karyawan, atau aturan persetujuan berdasarkan kategori. Pengiriman formulir menggunakan teknologi *AJAX* untuk memberikan pengalaman pengguna yang *seamless*, dengan *error handling* yang baik, konfirmasi keberhasilan, serta *loading states* yang memberikan umpan balik jelas selama proses pengiriman data. *Validation messaging* memberikan panduan spesifik untuk perbaikan ketika ada isian yang salah, sementara *success messages* dapat mencantumkan *reference numbers* atau informasi pelacakan agar pengguna dapat memantau status permintaan mereka.

3.2.7 Revisi Fitur Persetujuan dan Pengajuan Reimbursement

Pada minggu ketujuh hingga kedelapan, fokus kegiatan adalah melakukan revisi terhadap fitur persetujuan dan pengajuan *reimbursement* berdasarkan hasil pengujian dan masukan dari pembimbing maupun pengguna awal. Revisi dilakukan terutama untuk meningkatkan kejelasan alur proses, mengurangi *redundancy input*, serta memperbaiki respons sistem terhadap berbagai kondisi input. Proses revisi dimulai dengan analisis mendalam terhadap *feedback* yang diterima dari berbagai *stakeholder*. Masukan utama yang diterima mencakup kompleksitas *navigation* yang membingungkan pengguna baru, adanya duplikasi *field* yang tidak perlu dalam *form* pengajuan, serta lambatnya respons sistem ketika memproses *file upload* berukuran besar. Berdasarkan analisis ini, tim pengembang menyusun prioritas perbaikan yang dibagi menjadi tiga kategori utama: perbaikan *critical* (yang dapat mengganggu fungsi sistem), perbaikan *mayor* (yang mempengaruhi *user experience* secara signifikan), dan perbaikan *minor* (peningkatan kenyamanan penggunaan). Perbaikan *critical* yang dilakukan meliputi optimalisasi *query database* untuk mempercepat *loading* data pengajuan, perbaikan validasi *server-side* untuk mencegah *submission* data yang tidak valid, serta penambahan *error handling* yang lebih komprehensif untuk menangani berbagai skenario kegagalan sistem. Untuk perbaikan *mayor*, dilakukan *redesign* sebagian *user interface* dengan menerapkan prinsip *progressive disclosure*, di mana informasi kompleks disajikan secara bertahap sesuai kebutuhan pengguna. Selain itu, implementasi *breadcrumb navigation* dan *status indicator* yang lebih jelas membantu pengguna memahami posisi mereka dalam alur proses pengajuan.

Penyempurnaan tampilan *user interface* menjadi fokus utama dalam tahap revisi ini. Berdasarkan hasil *usability testing*, ditemukan bahwa pengguna sering mengalami kesulitan dalam memahami

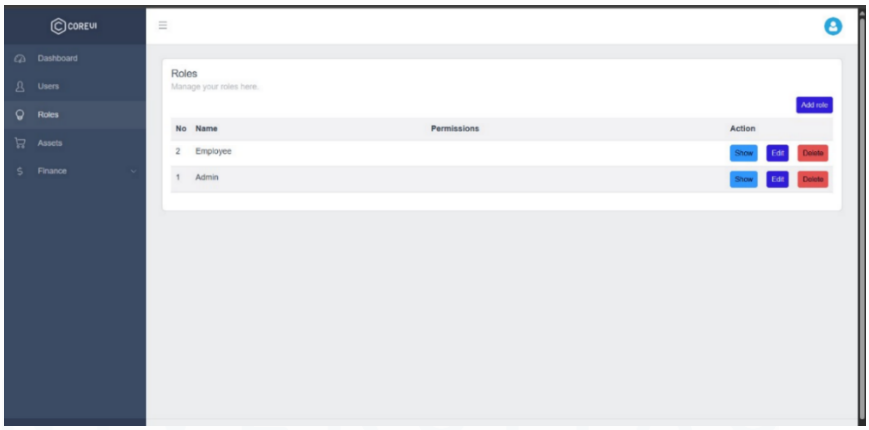
tahapan proses pengajuan *reimbursement*. Untuk mengatasi hal ini, dikembangkan *progress bar* yang interaktif, menampilkan tidak hanya tahapan saat ini tetapi juga estimasi waktu penyelesaian dan *action items* yang diperlukan dari pengguna. *Redesign form* pengajuan dilakukan dengan menerapkan konsep *multi-step form* yang memecah proses input menjadi beberapa tahap yang lebih *manageable*. Tahap pertama fokus pada informasi dasar pengajuan (jumlah, kategori, tanggal), tahap kedua untuk detail pembelian dan justifikasi, dan tahap ketiga untuk *upload* dokumen pendukung. Setiap tahap dilengkapi dengan *auto-save functionality* untuk mencegah kehilangan data jika terjadi *interruption* koneksi. Implementasi *responsive design* juga diperbaiki untuk memastikan sistem dapat diakses dengan baik melalui berbagai perangkat, mengingat banyak karyawan yang menggunakan *smartphone* untuk mengajukan *reimbursement* saat sedang dalam perjalanan dinas. Penyesuaian dilakukan pada ukuran tombol, *spacing* elemen, dan optimalisasi *touch interface* untuk meningkatkan *usability* pada perangkat *mobile*. Penambahan validasi pada *form* pengajuan *reimbursement* menjadi prioritas penting untuk memastikan integritas data yang masuk ke sistem. Validasi yang diimplementasikan mencakup *client-side validation* untuk memberikan *feedback* secara *real-time* kepada pengguna, serta *server-side validation* sebagai lapisan keamanan tambahan. Validasi *client-side* meliputi pengecekan format *file upload* (hanya menerima *PDF*, *JPG*, *PNG* dengan ukuran maksimal 5MB), validasi format mata uang dan angka, serta pengecekan kelengkapan *field* yang bersifat *mandatory*. *Server-side validation* dikembangkan lebih komprehensif dengan implementasi *business rule validation*, seperti pengecekan batas maksimal pengajuan per bulan sesuai dengan level jabatan, validasi tanggal pembelian yang tidak boleh lebih dari 30 hari sebelum pengajuan, serta *cross-reference* dengan *database vendor* untuk memastikan konsistensi data *merchant*. Selain itu, ditambahkan juga

audit trail yang mencatat setiap perubahan data pengajuan beserta *timestamp* dan *user* yang melakukan perubahan. *Security enhancement* dilakukan dengan implementasi *CSRF protection* pada semua *form*, *sanitasi input* untuk mencegah *SQL injection* dan *XSS attacks*, serta *enkripsi* data sensitif sebelum disimpan ke *database*. *Rate limiting* juga diterapkan untuk mencegah *spam submission* dan potensi *DDoS attacks* pada *endpoint API* yang terkait dengan fitur *reimbursement*.

3.2.8 Penyelesaian dan Pengujian Menyeluruh Fitur Reimbursement

Pada minggu kesembilan hingga kesepuluh, kegiatan magang difokuskan sepenuhnya pada penyelesaian akhir dan pengujian menyeluruh terhadap fitur *reimbursement* yang telah dibangun sebelumnya. Fitur *reimbursement* ini merupakan salah satu fitur penting dalam sistem karena berkaitan langsung dengan pengelolaan keuangan internal, khususnya dalam hal penggantian biaya atas pembelian barang atau layanan yang dilakukan oleh karyawan menggunakan dana pribadi. Fitur *reimbursement* yang dikembangkan menerapkan arsitektur *modular* dengan pemisahan yang jelas antara *presentation layer*, *business logic layer*, dan *data access layer*. *Presentation layer* menangani semua interaksi dengan pengguna melalui *web interface* yang *responsive*, *business logic layer* mengimplementasikan semua aturan bisnis dan *workflow approval*, sedangkan *data access layer* mengelola komunikasi dengan *database* dan *external services*. Komponen utama sistem mencakup modul pengajuan yang memungkinkan *user* untuk membuat *request* baru dengan *upload* dokumen pendukung, modul *approval workflow* yang mengatur alur persetujuan berdasarkan hierarki organisasi dan nilai pengajuan, modul *notification* yang memberikan *update real-time* kepada *stakeholder* terkait, dan modul *reporting* yang menyediakan *dashboard* dan *analytics* untuk *management review*. Integrasi dengan

sistem *existing* dilakukan melalui *API middleware* yang memastikan konsistensi data antara sistem *reimbursement* dengan sistem *HR* untuk validasi *employee data*, sistem *accounting* untuk *journal entry automation*, dan sistem *procurement* untuk *vendor verification*. *Message queue system* diimplementasikan untuk menangani *async processing*, terutama untuk proses yang membutuhkan waktu lama seperti *file processing* dan *email notification*. Salah satu aspek penting yang diselesaikan adalah implementasi sistem *authorization* yang *granular*. Sistem dibagi menjadi beberapa *role* utama dengan *permission* yang berbeda-beda. *Employee role* memiliki akses untuk membuat pengajuan baru, melihat status pengajuan milik sendiri, dan melakukan *edit* pada pengajuan yang masih dalam status *draft*. Pada tahap ini, sistem otorisasi pengguna mulai diimplementasikan secara komprehensif. Salah satu hasil konkret dari tahapan ini adalah penyelesaian fitur *Role Management*, yang ditampilkan pada *Gambar 3.13*. Dalam sistem ini, pengguna diklasifikasikan ke dalam beberapa *role* utama, yaitu *Admin* dan *Employee*, seperti terlihat pada halaman manajemen *role*.



Gambar 3.24 Halaman *Role Management*

Gambar 3.24 di atas memperlihatkan tampilan dari halaman *Role Management* dalam sistem berbasis web yang dikembangkan selama kegiatan magang. Halaman ini merupakan salah satu fitur penting yang berfungsi untuk mengelola peran (*role*) pengguna dalam

sistem. Fitur ini dikembangkan guna memastikan bahwa setiap pengguna hanya memiliki hak akses yang sesuai dengan tanggung jawab dan posisinya di dalam organisasi. Dari segi desain, tampilan halaman ini sederhana dan intuitif, dengan tata letak yang disusun secara rapi menggunakan *framework CoreUI*. Bilah navigasi di sebelah kiri menampilkan menu utama seperti Dashboard, Users, Roles, Assets, dan Finance, sehingga memudahkan admin berpindah antar fitur dengan cepat tanpa kembali ke halaman utama. Pada tabel yang ditampilkan di halaman *Role Management*, terdapat beberapa kolom utama:

1. No: Menunjukkan nomor urut peran dalam sistem.
2. Name: Menampilkan nama dari masing-masing *role*, seperti *Admin* dan *Employee*.
3. Permissions: Dirancang untuk menampilkan daftar hak akses yang dimiliki oleh masing-masing *role*. Namun, pada tahap ini, kolom ini masih kosong karena fitur manajemen izin masih dalam proses integrasi lanjutan.
4. Action: Berisi tiga tombol utama, yaitu Show, Edit, dan Delete. Masing-masing tombol berfungsi untuk:

   a. Show: Menampilkan rincian hak akses atau deskripsi lengkap dari *role* tersebut.

   b. *Edit*: Mengarahkan ke halaman formulir untuk mengubah nama *role* atau izin yang melekat pada *role* tersebut.

   c. *Delete*: Menghapus *role* dari sistem jika sudah tidak diperlukan lagi.

Selain itu, terdapat tombol *Add Role* berwarna ungu di sisi kanan atas tabel. Tombol ini memungkinkan admin untuk menambahkan *role* baru ke dalam sistem. Fitur ini sangat bermanfaat jika di kemudian hari organisasi mengalami perubahan struktur atau pembagian tugas, sehingga perlu ditambahkan *role* seperti *Manager*, *Supervisor*, atau peran fungsional lainnya. Sistem membedakan peran pengguna berdasarkan fungsinya dalam organisasi. Pada tahap awal,

role yang telah didefinisikan dan diuji meliputi *Admin* (dengan akses penuh terhadap konfigurasi sistem, manajemen pengguna, dan pemantauan semua fitur), serta *Employee* (dengan akses terbatas, hanya untuk membuat dan memantau pengajuan *reimbursement* milik sendiri). Masing-masing *role* ditampilkan dalam bentuk tabel yang memungkinkan administrator untuk:

1. Menambahkan *role* baru melalui tombol *Add Role*.
2. Melihat rincian hak akses tiap *role* dengan tombol *Show*.
3. Mengubah nama atau pengaturan *role* melalui *Edit*.
4. Menghapus *role* yang tidak dibutuhkan melalui *Delete*, dengan konfirmasi terlebih dahulu agar tidak terhapus secara tidak sengaja.

Pengaturan *role* menggunakan pendekatan berbasis *middleware*, di mana setiap permintaan yang masuk akan divalidasi berdasarkan hak akses yang dimiliki oleh *role* pengguna. Pendekatan ini tidak hanya meningkatkan keamanan sistem, tetapi juga meminimalkan kesalahan akses dan potensi penyalahgunaan wewenang. Hak akses juga divalidasi pada antarmuka pengguna (*front-end*), sehingga pengguna hanya melihat tombol atau menu yang sesuai dengan perannya. Untuk memperkuat keamanan dan *audit* sistem, setiap tindakan penting seperti perubahan *role*, penghapusan pengguna, dan persetujuan *reimbursement* dicatat dalam *audit log* sistem yang dilengkapi dengan alamat IP, jenis *browser* yang digunakan, waktu tindakan dilakukan, serta parameter yang diubah. *Team Leader* memiliki akses untuk melihat dan menyetujui pengajuan dari anggota timnya, serta memiliki izin tambahan untuk melihat *summary report* tingkat tim. *Manager* memiliki akses yang lebih luas, dapat melihat semua pengajuan di departemennya, menyetujui secara massal pengajuan dengan nilai di bawah batas tertentu, serta mengakses *dashboard analytics* lanjutan. *Finance* memiliki hak akses khusus untuk memproses pembayaran, melakukan *reconciliation* dengan *bank statement*, serta menghasilkan *financial report*. *Admin*

memiliki akses penuh terhadap sistem, termasuk manajemen pengguna, pengaturan sistem, dan *audit log*. Implementasi dilakukan dengan pendekatan otorisasi berbasis *middleware* yang dilengkapi dengan mekanisme *caching* untuk meningkatkan kinerja. Pemeriksaan hak akses dilakukan baik di tingkat *controller* maupun pada tingkat tampilan (*view*), untuk memastikan bahwa elemen antarmuka yang tidak sesuai dengan peran pengguna tidak ditampilkan. Pencatatan *audit* diterapkan untuk semua tindakan yang bersifat sensitif, tidak hanya mencatat apa yang dilakukan, tetapi juga konteks seperti alamat IP, informasi *browser*, dan *business justification* jika diperlukan. Penyelesaian fitur *reimbursement* dan *role management* menjadi tonggak utama dalam siklus pengembangan sistem selama kegiatan magang. Sistem yang dibangun tidak hanya berfungsi dengan baik secara teknis, tetapi juga telah memenuhi standar perusahaan dalam aspek keamanan, fleksibilitas, dan skalabilitas. Adanya halaman *Role Management* mempermudah pengelolaan hak akses dalam jangka panjang, serta memberikan fleksibilitas bagi admin untuk menyesuaikan sistem dengan struktur organisasi yang terus berkembang. Sistem telah siap digunakan dalam kondisi produksi yang stabil, optimal, dan terdokumentasi dengan baik. Tahapan ini mencerminkan keberhasilan pendekatan yang sistematis, kolaborasi tim yang solid, serta pemahaman yang mendalam terhadap kebutuhan bisnis perusahaan.

3.3 Kendala yang Ditemukan

Selama pelaksanaan magang sebagai web developer di PT. *Back*slash Creative Nusantara, terdapat beberapa kendala yang dihadapi, yaitu:

1. Kesulitan dalam memahami serta mempelajari templating dari Laravel yang sebelumnya belum pernah dipelajari selama masa perkuliahan di kampus.
2. Kesulitan dalam menggunakan bahasa pemograman seperti Blade, Javascript dan PHP yang masih belum mahir dikuasai sepenuhnya dan di

gabungkan dengan Blade yang merupakan bahasa templating yang dimiliki oleh Laravel.

3. Kesulitan ketika ingin melakukan debugging ketika terjadi error pada file Blade yang terjadi karena pesan error yang muncul tidak langsung mendeskripsikan detail dari masalahnya.
4. Beberapa fitur pada website memerlukan integrasi yang cukup kompleks antara front-end dan back-end, sementara anggota tim pengembang memiliki tingkat pengalaman yang berbeda-beda sehingga proses diskusi teknis terkadang memerlukan waktu lebih lama.

3.4 Solusi atas Kendala yang Ditemukan

Solusi yang dilakukan untuk mengatasi kendala-kendala yang dihadapi selama magang adalah:

1. Untuk mengatasi atas kendala terhadap kesulitan dalam memahami Blade, dapat dimulai dari memahami konsep dasar dari Blade dan Laravel dari video online, lalu mencoba tutorial interaktif di website kegiatan ini dapat melatih kebiasaan belajar sehingga akan cepat memahami pola dan struktur templating dari Laravel.
2. Dalam mengatasi masalah kesulitan dalam menggunakan bahasa pemograman yang terlalu dikuasai dapat dimulai dari, memecahkan masalah masalah yang besar menjadi bagian bagian yang lebih kecil sehingga masalah tersebut dapat lebih mudah untuk di selesaikan.
3. Dalam menghadapi masalah ketika melakukan debugging, yang pertama adalah menganalisis penyebabnya dan jika sudah benar benar errornya tidak di ketahui jelas, dapat menggunakan alat bantu dari AI sebagai analisa yang lebih jelas terkait error yang terjadi di file yang mana.
4. Dilakukan pembagian tugas yang jelas dan terstruktur berdasarkan keahlian masing-masing anggota tim. Selain itu, rutin diadakan diskusi harian untuk saling bertukar informasi dan memecahkan kendala teknis secara bersama-sama. Dokumentasi teknis juga diperbanyak untuk memudahkan koordinasi antar bagian.