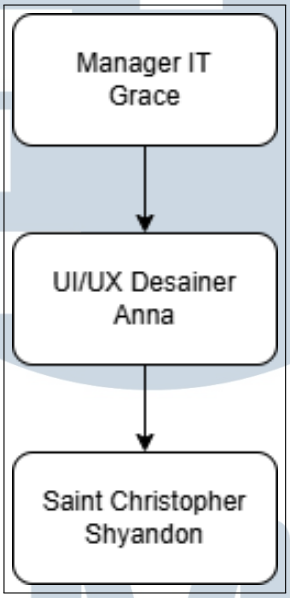
BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Dalam program magang di PT. Epluse Beautiful Gate, posisi yang ditempati berada dalam tim IT, yang berfokus pada perancangan aplikasi untuk coffeeshop perusahaan. Saya disini berposisi sebagai FrontEnd Sesuai dengan Gambar 3.1, tim ini dipimpin oleh Ibu Grace sebagai Manajer IT, dengan Ibu Anna sebagai UI/UX desainer.



Gambar 3.1. Struktur tim IT

Struktur tim IT yang tertera pada Gambar 3.1 terdiri dari satu orang Manager, satu orang UI/UX Desainer, dan satu orang Developer. Selama masa magang, bimbingan langsung diberikan di bawah Manager Ibu Grace, yang memberikan arahan terkait tugas serta memastikan pekerjaan yang dilakukan sudah benar atau belum. Koordinasi dalam tim dilakukan melalui grup chat setiap harinya dengan mengirimkan laporan dalam bentuk format yang diberikan oleh Manager.

3.2 Tugas yang Dilakukan

Selama menjalani program magang di PT. Epluse Beautiful Gate, terdapat beberapa tugas yang dikerjakan dalam tim IT. Tim berkolaborasi aktif dalam merancang aplikasi, dimulai dari riset aplikasi lain dan mengambil beberapa contoh aplikasi

untuk dijadikan patokan desain serta fitur. Bimbingan dari atasan pun membantu pengerjaan perancangan. Selesai melakukan riset, tim kami lanjut ke proses pengembangan UI/UX desain dan pembuatan front end. Tugas yang dikerjakan dalam program magang dijabarkan sebagai berikut:

- Melakukan Penelitian
  - Meneliti contoh aplikasi lain dan beberapa aplikasi coffeeshop lainnya serta mencatat kelebihan dan kekurangannya. Tools yang digunakan yaitu Microsoft Word, Google, Aplikasi coffeeshop lain.
- Perancangan UI/UX
  - Melakukan diskusi untuk desain UI/UX yang dibuat oleh desainer, serta membantu membuat beberapa halaman dan ikon. Tools yang digunakan yaitu WhatsApp, Figma.
- Melakukan Koding Front-End
  - Menggunakan React Native dan Nativewind untuk membuat front-end aplikasi coffeeshop sesuai dengan desain UI/UX yang telah dirancang. Tools yang digunakan yaitu React Native, Nativewind.

Pada awal bulan, tugas yang diberikan masih melakukan riset dan diskusi UI/UX untuk desain aplikasi. Waktu yang tersedia digunakan untuk mempelajari desain dan memberikan pendapat dari desain yang dibuat.

3.3 Uraian Pelaksanaan Magang

Pelaksanaan kerja magang di PT. Epluse Beautiful Gate sudah ditetapkan oleh manajer IT dengan tugas yang sudah direncanakan oleh manajer. Setiap tugas diberikan diawali dengan diskusi tugas untuk hari itu juga dan apa capaian yang ingin dicapai hari itu. Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

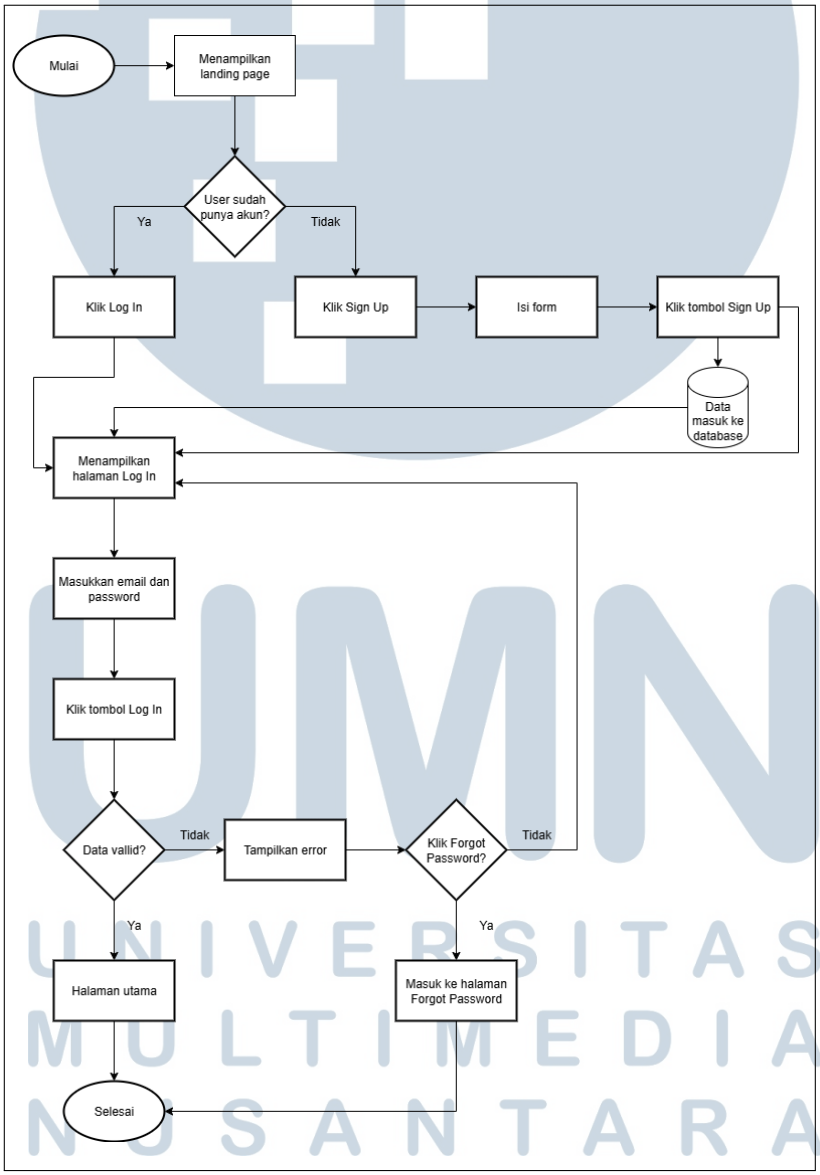
Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Minggu Ke -	Pekerjaan yang dilakukan
1	Melakukan meeting tentang aplikasi yang ingin dibuat
2	Melakukan riset untuk pembuatan aplikasi dimulai dari riset desain aplikasi lain seperti : Janji Jiwa, Kopi nako, Kopi tuku, Kopi kenangan, Fore coffee.
3	Membuat desain UI/UX dan berdiskusi dengan UI/UX desainer mengenai desain yang sudah dibuat.
4	Membuat sejumlah icon untuk desain yang akan dipilih oleh desainer untuk desain aplikasinya
5	Mulai mensetup project menggunakan React Native untuk pembuatan aplikasi dan mulai ke bagian coding frontend
6	Mengerjakan coding frontend sesuai desain UI/UX yang sudah dibuat
7	Meeting mengenai hasil coding front end dan melakukan revisi sesuai dengan hasil meeting
8	Mulai mengerjakan revisi desain karna ada perbuahan dari atasan
9	Selesai mengerjakan desain UI/UX dilanjut pengecekan dari atasan
10	Atasan sudah oke dengan hasil UI/UX dan diminta untuk membantu divisi data untuk memperbaiki data store
11	Melakukan meeting dan mulai mengerjakan untuk perbaikan data store
12	Selesai melakukan perbaikan data

Secara keseluruhan, alur kerja yang diterapkan selama magang ini mengikuti salah satu model pengembangan perangkat lunak, yaitu *Prototyping Model* [2]. Model ini sangat sesuai untuk proyek yang berfokus pada antarmuka pengguna, di mana proses dimulai dengan tahap perancangan (riset dan desain *UI/UX*) untuk menghasilkan sebuah prototipe visual. Setelah prototipe tersebut dianggap final, barulah tahap implementasi atau *coding frontend* dimulai berdasarkan rancangan yang sudah ada.

3.3.1 Perancangan Sistem

Pada tahap perancangan sistem, alur kerja atau *flowchart* aplikasi didefinisikan untuk memastikan pengalaman pengguna yang logis dan aman, dimulai dari proses autentikasi pengguna. Perancangan ini menjadi landasan utama sebelum proses implementasi coding frontend dilakukan. Berdasarkan diagram alir yang telah dibuat pada gambar **??**, sistem pertama kali akan menampilkan halaman login sebagai gerbang utama akses ke dalam aplikasi.



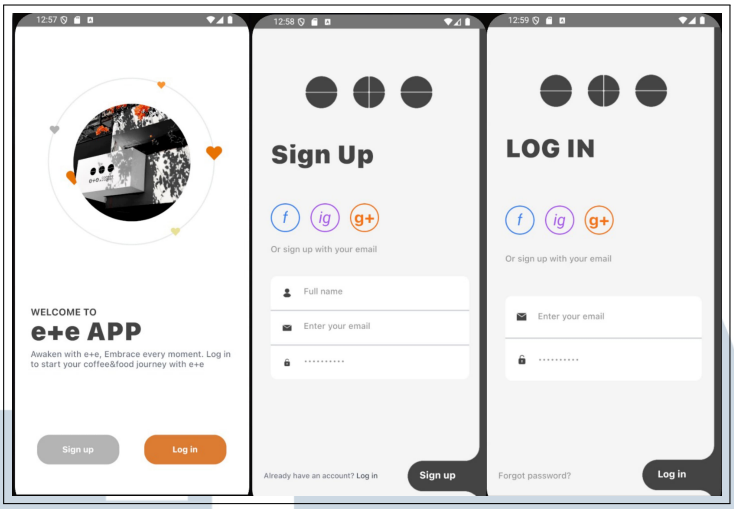
Gambar 3.2. Login dan Register Flow

Pada halaman ini, alur dibagi menjadi dua skenario utama berdasarkan apakah

pengguna sudah memiliki akun atau belum. Skenario pertama adalah untuk pengguna baru. Pengguna akan memilih opsi untuk mendaftar (register), yang akan mengarahkannya ke formulir pendaftaran. Setelah pengguna mengisi data yang diperlukan dan menekan tombol 'Daftar', sistem akan memproses dan menyimpan informasi tersebut ke dalam basis data (*database*). Setelah pendaftaran berhasil, pengguna akan dikembalikan ke halaman login untuk masuk ke aplikasi.
Skenario kedua adalah untuk pengguna yang sudah terdaftar. Pengguna akan memasukkan email dan password mereka, lalu menekan tombol 'Login'. Sistem kemudian akan melakukan validasi kredensial tersebut. Jika data yang dimasukkan valid dan sesuai dengan yang ada di database, pengguna akan berhasil masuk dan diarahkan ke halaman utama aplikasi. Namun, jika data tidak valid, sistem akan menampilkan pesan *error*, dan pengguna akan tetap berada di halaman login untuk mencoba kembali atau melakukan pendaftaran jika belum memiliki akun. Alur perancangan ini menjadi dasar untuk implementasi antarmuka (UI) dan logika frontend dengan React Native, memastikan bahwa proses masuk dan pendaftaran pengguna dapat berjalan dengan baik dan intuitif.

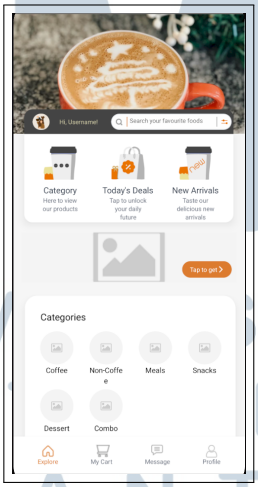
3.3.2 Perancangan Tampilan

Perancangan tampilan ini mengadopsi pendekatan *User-Centered Design* (UCD), sebuah filosofi yang menekankan pada kebutuhan dan kemudahan pengguna akhir dalam menggunakan sebuah produk [3]. Prinsip ini tercermin dalam alur autentikasi yang terlihat pada Gambar 3.3, di mana setiap halaman dirancang untuk memberikan pengalaman yang jelas dan kohesif bagi pengguna.
Halaman Pendaftaran (*Sign Up Page*) dirancang untuk mempermudah pembuatan akun baru dengan menawarkan opsi via media sosial dan formulir konvensional. Desainnya menjaga konsistensi visual dengan halaman sebelumnya, menggunakan gaya ikon dan font yang seragam. Tata letaknya yang terstruktur memastikan proses pengisian data berjalan intuitif dan mudah diikuti.
Halaman Masuk (*Log In Page*) menyediakan akses bagi pengguna terdaftar dengan mengadopsi struktur yang familier seperti halaman pendaftaran untuk menciptakan pengalaman yang kohesif. Penyertaan tautan `Forgot password?` merupakan fitur penting untuk usabilitas, sementara penekanan visual pada tombol `Log in` menegaskan fungsinya sebagai aksi final untuk masuk ke aplikasi.



Gambar 3.3. Welcome Page, SignUp and Login Page

Halaman Utama (*Home Page*), seperti yang terlihat pada Gambar 3.4, berfungsi sebagai pusat navigasi dan penemuan produk, menyambut pengguna dengan sapaan personal dan menyorot berbagai fitur. Tatanan visualnya yang terstruktur, mulai dari bar pencarian hingga *banner* promosi, dirancang untuk memandu alur perhatian pengguna secara alami. Produk-produk dikelompokkan secara visual untuk kemudahan penelusuran, dan penggunaan warna oranye yang konsisten pada elemen interaktif berfungsi sebagai penanda visual yang jelas sekaligus memperkuat identitas merek.



Gambar 3.4. Home Page

Halaman Keranjang Saya (*My Cart*), seperti yang dilampirkan pada Gambar **??**, berfungsi sebagai tahap akhir sebelum pembayaran, di mana pengguna dapat

meninjau dan mengelola pesanan mereka. Di bagian atas, pengguna diberikan kontrol yang jelas melalui pilihan metode pesanan (`Pick-Up` atau `Delivery`) dan informasi lokasi gerai. Setiap produk disajikan dalam format kartu yang terstruktur dan informatif, menampilkan detail harga, gambar, serta fitur penyesuaian jumlah (*quantity adjuster*) yang mudah digunakan. Tampilan diskon yang transparan dirancang untuk menonjolkan nilai yang didapat pengguna. Bagian ringkasan di bawah secara efektif merangkum total belanja, yang mengarah pada tombol `PAYMENT`. Tombol ini sengaja dibuat menonjol dengan warna dan ukuran untuk menciptakan hierarki visual yang kuat, mengarahkan pengguna ke aksi terpenting di halaman ini.

Di balik antarmuka yang interaktif tersebut, terdapat logika manajemen *state* yang menangani perubahan data secara dinamis. Logika inti untuk menambah dan mengurangi jumlah produk pada keranjang diimplementasikan seperti yang terlihat pada Kode 3.1.

```
// Fungsi untuk menambah jumlah produk
const incrementQuantity = (index: number) => {
  const updatedProducts = [...products];
  updatedProducts[index].quantity += 1;
  setProducts(updatedProducts);
};

// Fungsi untuk mengurangi jumlah produk
const decrementQuantity = (index: number) => {
  const updatedProducts = [...products];
  if (updatedProducts[index].quantity > 0) {
    updatedProducts[index].quantity -= 1;
    setProducts(updatedProducts);
  }
};
```

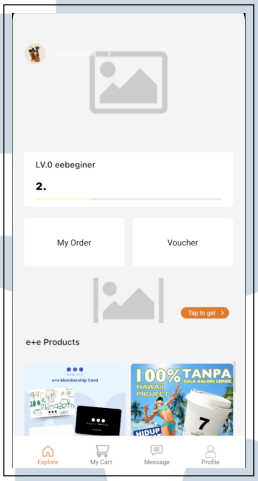
Kode 3.1: Contoh Kode Logika untuk Manajemen Jumlah Produk

Kode pada Kode 3.1 mendemonstrasikan logika inti manajemen *state* di *React*. Fungsi tersebut secara aman memperbarui jumlah produk dengan membuat salinan *array* terlebih dahulu (prinsip *immutability*) dan menyertakan validasi untuk mencegah nilai kuantitas menjadi negatif. Pemanggilan fungsi `setProducts` pada akhirnya akan memicu pembaruan antarmuka pengguna secara otomatis, sehingga perubahan jumlah langsung terlihat di layar.

Gambar 3.5 merupakan halaman Profil (*Profile Page*) yang berfungsi sebagai dasbor personal bagi pengguna, menyajikan informasi akun seperti nama pengguna

dan status keanggotaan. Penggunaan elemen gamifikasi seperti level (`LV.0 eebeginer`) dan bar kemajuan dirancang untuk meningkatkan keterlibatan dan loyalitas pengguna. Tata letaknya yang terstruktur menyediakan akses cepat ke riwayat pesanan (`My Order`) dan `Voucher` melalui tombol yang jelas. Di bawahnya, terdapat area untuk promosi dan produk terkait yang disajikan secara visual untuk menarik minat pengguna. Halaman ini menjaga konsistensi desain aplikasi, dengan ikon `Profile` yang aktif pada *navigation bar* sebagai penanda lokasi yang jelas, melengkapi pengalaman pengguna yang terpadu.

Implementasi untuk menampilkan area ”e+e Products” tersebut menggunakan komponen *FlatList* dari *React Native* untuk efisiensi, seperti yang ditunjukkan pada Kode 3.2.



Gambar 3.5. Profile Page

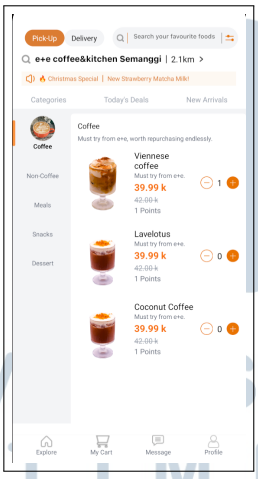
```
<Text className="font-sf-medium mr-[65%]">e+e Products</Text>
<View className="w-[92%] h-[25%]">
  <FlatList
    data={images}
    keyExtractor={(item) => item.id}
    numColumns={2}
    renderItem={({ item }) => (
      <Image
        source={item.source}
        style={{ flex: 1, margin: 4, aspectRatio: 1 }}
        className="rounded-md"
      />
    )}
  />
```

```
15 </View>
```

Kode 3.2: Contoh Kode FlatList untuk Menampilkan Produk

Kode pada Kode 3.2 menunjukkan penggunaan komponen *FlatList* dari *React Native* untuk menampilkan daftar produk secara efisien. Properti `numColumns={2}` digunakan untuk mengubah tampilan daftar standar menjadi format grid dua kolom. Komponen ini secara dinamis me-*render* setiap item dari *array* data menjadi sebuah komponen *Image*, menjadikannya solusi yang optimal untuk menampilkan galeri produk.

Halaman Kategori (*Categories Page*), yang diakses dari halaman utama dan desainnya terlihat pada Gambar 3.6, berfungsi sebagai etalase produk utama dalam aplikasi. Halaman ini menggunakan tata letak dua kolom yang efisien: kolom kiri sebagai menu navigasi vertikal untuk memilih kategori, dan kolom kanan untuk menampilkan daftar produk dari kategori yang sedang aktif. Adanya penanda visual yang jelas pada kategori terpilih (`Coffee`) memberikan umpan balik langsung kepada pengguna. Desain kartu produk yang digunakan di sini konsisten dengan bagian lain aplikasi, menampilkan informasi lengkap beserta penyesuai jumlah, sehingga menciptakan pengalaman yang familier dan intuitif saat menjelajahi menu. Implementasi lengkap dari sistem navigasi kategori ini, mulai dari data hingga tampilan visual yang dinamis, ditunjukkan pada Kode 3.3.



Gambar 3.6. Categories Page

```
1 // 1. Struktur data untuk daftar kategori
2 const categories = [
3   { name: "Coffee", image: require("../assets/coffee-rounded.png")
      },
```

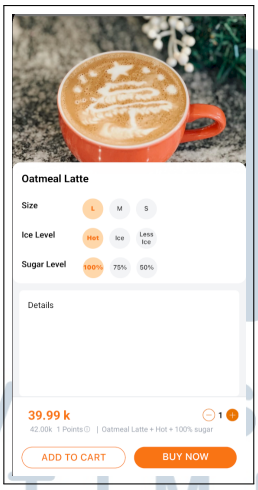
```
  { name: "Non-Coffee", image: require("../assets/coffee-rounded.
    png") },
  // ... dan seterusnya
];

// ------------------------------------------------

// 2. Implementasi komponen untuk setiap tombol kategori
// (Kode ini berada di dalam sebuah perulangan/map dari array '
    categories')
<TouchableOpacity
  key={index}
  onPress={() => setActiveCategory(category.name)}
  className="flex-row mb-[6%] items-center py-[4%]"
>
  {/* Konten visual di dalam tombol */}
  <View className="items-center flex-1 ml-[4%]">
    {/* Gambar hanya muncul jika kategori aktif */}
    {activeCategory === category.name && (
      <Image
        source={
          typeof category.image === "string"
            ? { uri: category.image }
            : category.image
        }
        className="w-12 h-12 mb-[2%] rounded-full"
      />
    )}
    {/* Gaya teks berubah sesuai kondisi aktif/tidak aktif */}
    <Text
      className={`text-xs ${
        activeCategory === category.name
          ? "text-black font-semibold"
          : "text-gray-500"
      }`}
    >
      {category.name}
    </Text>
  </View>
</TouchableOpacity>
```

Kode 3.3: Implementasi Lengkap Navigasi Kategori (Data, Interaksi, dan Tampilan)

Kode pada Kode 3.3 menunjukkan implementasi lengkap dari navigasi kategori.

Sebuah *array* ‘categories‘ menjadi sumber data, yang kemudian di-*render* menjadi komponen ‘TouchableOpacity‘ untuk setiap itemnya. Bagian terpenting adalah **logika kondisional** di dalamnya: sebuah ‘Image‘ (ikon) hanya akan muncul jika kategori tersebut aktif. Selain itu, gaya pada komponen ‘Text‘ juga berubah secara dinamis—menjadi tebal dan berwarna hitam saat aktif, dan abu-abu saat tidak aktif. Ketika ditekan, ‘onPress‘ akan memperbarui *state* ‘activeCategory‘, yang secara reaktif mengubah tampilan visual untuk memberikan umpan balik yang jelas kepada pengguna.

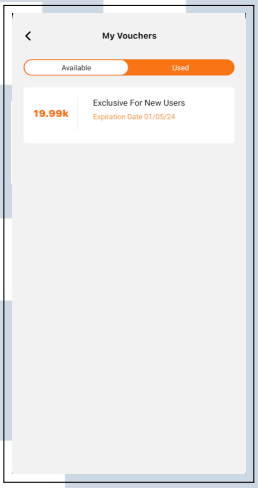
Halaman Detail Produk (*Detail Page*), dengan antarmuka seperti pada Gambar 3.7, memberikan informasi mendalam mengenai satu item dan memungkinkan pengguna melakukan kustomisasi pesanan. Halaman ini menggunakan gambar produk yang besar dan menarik sebagai fokus visual utama. Di bawahnya, pengguna diberikan kontrol penuh untuk menyesuaikan pesanannya melalui pilihan-pilihan seperti ukuran, level es, dan gula, dengan penanda visual yang jelas pada opsi yang terpilih. Bagian bawah halaman secara efektif merangkum total harga dan ringkasan kustomisasi, memberikan konfirmasi akhir sebelum pengguna mengambil tindakan. Desain dua tombol aksi `BUY NOW` sebagai aksi utama yang menonjol dan `ADD TO CART` sebagai aksi sekunder—secara cerdas memandu pengguna sesuai dengan niat mereka.



Gambar 3.7. DetailItem Page

Halaman Voucher Saya (*My Vouchers*), seperti yang terlihat pada Gambar 3.8, dirancang untuk membantu pengguna mengelola kupon promosi mereka dengan efisien. Halaman ini menggunakan navigasi tab yang jelas untuk memisahkan antara voucher yang tersedia (`Available`) dan yang sudah terpakai (`Used`),

sehingga memberikan struktur informasi yang rapi. Setiap kartu voucher dirancang dengan hierarki visual yang baik; nominal voucher ditonjolkan sebagai informasi utama, diikuti oleh detail deskripsi dan tanggal kedaluwarsa. Desainnya yang minimalis dan bersih membantu pengguna untuk fokus pada informasi penting, memberikan akses terpusat untuk melihat keuntungan yang mereka miliki.



Gambar 3.8. Voucher Page

Secara keseluruhan, Halaman Voucher Saya dirancang sebagai fitur strategis untuk mendorong loyalitas pengguna. Tampilan voucher yang jelas berfungsi sebagai insentif langsung yang dapat memotivasi pengguna untuk melakukan transaksi kembali. Dengan demikian, desain halaman ini secara efektif mendukung tujuan aplikasi dalam meningkatkan frekuensi pembelian dan membangun hubungan jangka panjang dengan pelanggan.

3.4 Kendala dan Solusi yang Ditemukan

Berikut adalah pemaparan mengenai berbagai kendala yang ditemukan selama periode magang beserta solusi yang telah diimplementasikan untuk mengatasi setiap permasalahan tersebut.

1. Kesulitan komunikasi akibat perbedaan bahasa dengan *desainer* dan *manajer*, yang berisiko menimbulkan salah tafsir terhadap detail desain yang diinginkan.

2. Adanya ekspektasi tinggi dari pimpinan untuk menghasilkan produk dengan kualitas visual premium, sementara jumlah anggota tim dan sumber daya terbatas.

3. Tantangan dalam menerjemahkan hasil riset pasar (fitur aplikasi lain, skema diskon) menjadi sebuah desain antarmuka (*UI/UX*) yang konkret, fungsional, dan menarik.

4. Kesulitan teknis dalam menerjemahkan desain *UI/UX* yang sudah jadi ke dalam kode *frontend* secara presisi (*pixel-perfect*) menggunakan *React Native* dan *Nativewind*.

5. Menghadapi masalah teknis terkait konfigurasi lingkungan pengembangan (*development environment*) *React Native* yang terkadang kompleks atau menemukan *bug* yang spesifik pada salah satu platform (*iOS* atau *Android*).

Dalam menghadapi setiap tantangan yang telah diuraikan, tim tidak hanya berhenti pada identifikasi masalah. Langkah selanjutnya yang krusial adalah merumuskan dan menerapkan berbagai solusi yang proaktif dan strategis untuk memastikan proyek tetap berjalan sesuai tujuan. Berikut adalah rincian solusi yang diimplementasikan untuk mengatasi masing-masing kendala tersebut.

1. Menggunakan alat bantu penerjemah, memaksimalkan komunikasi berbasis visual dengan merujuk langsung pada *mockup*, dan melakukan konfirmasi berulang untuk memastikan pemahaman yang selaras.

2. Menerapkan strategi *MVP (Minimum Viable Product)* untuk fokus pada fitur inti terlebih dahulu dan mengelola ekspektasi pimpinan dengan memberikan laporan progres yang transparan dan realistis.

3. Membuat alur pengguna (*user flow*) dan *wireframe* sebelum mendesain *mockup* detail untuk memvalidasi logika dan fungsi sebelum fokus pada aspek visual.

4. Memecah desain menjadi komponen-komponen kecil yang dapat digunakan kembali (*reusable components*) dan proaktif berdiskusi dengan *desainer* untuk mengklarifikasi detail interaksi yang tidak terlihat pada gambar statis.

5. Mengandalkan dokumentasi resmi, aktif mencari solusi di forum komunitas *developer* (seperti *Stack Overflow*), dan meminta bimbingan dari anggota tim yang lebih senior saat menghadapi masalah teknis.