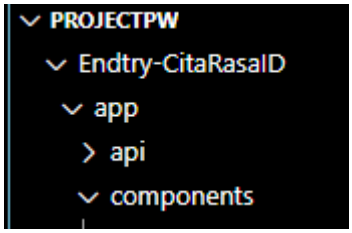
Nama : Muflikhatun Laila Fadilah

Npm : 22312053

1. **Membuat folder Components di dalam folder APP**



2. **Membuat file Banner.js, Categories.js, ProductList.js, HomeModule.css**

3. **Membuat file Banner.js**

```
1   export default function Banner() {
```

Ini mendefinisikan sebuah fungsi bernama Banner yang mengembalikan sebuah elemen JSX. Kata kunci export default membuat komponen ini dapat diimpor dan digunakan di komponen lain.

```
3       <section
```

- section: Elemen HTML section digunakan sebagai wadah utama untuk komponen banner.
- style: Atribut style digunakan untuk mengatur tampilan dari elemen section dan elemen-elemen di dalamnya.

```
4       style={{
```

- **Styling**

```
    backgroundImage: 'url(/img/bg.jpg)', // Menentukan gambar sebagai background
    backgroundSize: 'cover', // Mengatur gambar agar menutupi seluruh elemen
    backgroundPosition: 'center', // Memusatkan posisi gambar
    display: 'flex',
    alignItems: 'center',
    justifyContent: 'space-between',
    padding: '4rem 2rem',
    borderBottom: '2px solid #ccc',
    borderRadius: '8px',
    height: '70vh',
    boxShadow: '0 4px 10px rgba(0, 0, 0, 0.1)',
    color: '#fff', // Memberikan warna teks yang cocok dengan gambar
    textShadow: '0 2px 4px rgba(0, 0, 0, 0.5)', // Memberikan bayangan pada teks untuk keterbacaan
  }
```

- **Background:**
    - backgroundImage: Mengatur gambar latar belakang.
    - backgroundSize: Mengatur ukuran gambar latar belakang agar menutupi seluruh area.
    - backgroundPosition: Memposisikan gambar latar belakang di tengah.
- **Layout:**
    - display: flex: Mengatur tata letak elemen anak menjadi fleksibel.
    - alignItems: center: Menyelaraskan elemen anak secara vertikal di tengah.
    - justifyContent: space-between: Membagi ruang secara merata di antara elemen anak dengan jarak di antara mereka.
- **Padding, Border, dan Shadow:** Menambahkan padding, border, dan shadow untuk memberikan tampilan yang lebih baik.
- **Warna dan Teks:**
    - color: Mengatur warna teks.
    - textShadow: Memberikan efek bayangan pada teks.
- **Dimensi:**
    - height: Mengatur tinggi dari banner.

- **Konten Banner**

- **Div Kiri:**
    - Mengandung judul "Selamat Datang di CitaRasaID" dan deskripsi singkat.
    - Menggunakan gaya yang sesuai dengan desain keseluruhan banner.
- **Div Kanan:**
    - Mengandung gambar produk.
    - Mengatur ukuran, posisi, dan gaya gambar.

4. Membuat file Categories.js

```
1   export default function Categories({ onSelectCategory }) {
```

export default function Categories({ onSelectCategory }): Ini mendefinisikan sebuah fungsi bernama Categories yang akan menjadi komponen React. Komponen ini menerima sebuah prop bernama onSelectCategory sebagai fungsi.

```
const categories = [
  { name: 'Semua', img: '/img/all.jpg' },
  { name: 'Makanan', img: '/img/makanan.jpg' },
  { name: 'Minuman', img: '/img/minuman.jpg' },
  { name: 'Kerajinan', img: '/img/kerajinan.jpg' },
  { name: 'Pakaian', img: '/img/pakaian.jpg' },
];
```

Baris ini membuat sebuah array konstan bernama categories yang berisi objek-objek. Setiap objek mewakili satu kategori produk.
Setiap objek di dalam array memiliki properti:

- name: Nama kategori yang akan ditampilkan (misalnya, "Semua", "Makanan").
- img: Path ke gambar yang mewakili kategori tersebut (misalnya, "/img/all.jpg").

```
return (
  <section
    id="kategori"
    style={{
      textAlign: 'center',
      padding: '6rem 2rem',
      backgroundColor: '#f3f4f6',

      borderBottom: '2px solid #ddd',
      borderRadius: '16px',
      boxShadow: '0 8px 16px rgba(0, 0, 0, 0.1)',
    }}
```

- return: Keyword ini menandakan kode JSX yang akan dirender oleh komponen.
- <section id="kategori" style={{ ... }}>: Ini membuat elemen <section> dengan ID "kategori". Atribut style digunakan untuk mengatur gaya elemen tersebut menggunakan CSS inline.

```
    <h2
      style={{
        fontSize: '3rem',
        fontWeight: '700',
        color: '#222',
        marginBottom: '3rem',
        letterSpacing: '1.5px',
      }}
    >
      Pilih Kategori Oleh-Oleh
    </h2>
```

Baris ini membuat elemen <h2> yang menampilkan judul "Pilih Kategori Oleh-Oleh". Anda bisa menyesuaikan gaya judul menggunakan CSS inline.

```
<ul
  style={{
    display: 'flex',
    justifyContent: 'center',
    gap: '3rem',
    listStyle: 'none',
    padding: 0,
    flexWrap: 'wrap',
  }}
```

: Ini membuat elemen <ul> yang akan digunakan untuk menampilkan daftar kategori secara tidak berurutan (unordered list). Anda bisa mengatur gaya daftar ini menggunakan CSS inline.

```
>
  {categories.map((category, index) => (
    <li
      key={index}
      style={{
        textAlign: 'center',
        width: '200px',
        cursor: 'pointer',
        transition: 'all 0.4s ease',
        borderRadius: '20px',
      }}
      onClick={() => onSelectCategory(category.name)}
    >
```

categories.map: Ini menggunakan fungsi map untuk mengiterasi setiap objek di dalam array categories.

- (category, index): Parameter yang diterima fungsi map untuk setiap iterasi:
- category: Objek yang mewakili kategori saat ini (misalnya, { name: "Semua", img: "/img/all.jpg" }).
- index: Indeks dari kategori saat ini di dalam array (mulai dari 0).
- <li key={index} style={{ ... }}>: Ini membuat elemen <li> untuk setiap kategori. Atribut key penting untuk React dalam mengidentifikasi item mana yang berubah, ditambahkan, atau dihapus. Anda bisa mengatur gaya item kategori ini menggunakan CSS inline.

5. Membuat file ProductList.js

```
import { useState, useEffect } from 'react';
```

Baris ini mengimpor hook useState dan useEffect dari library React. Hook ini digunakan untuk mengelola state (keadaan) komponen.

```
3   export default function ProductList({ selectedCategory }) {
4     const [products, setProducts] = useState([]);
5     const [loading, setLoading] = useState(true);
6     const [error, setError] = useState(null);
7     const [editingProduct, setEditingProduct] = useState(null);  // Track product being edited
8     const [newProduct, setNewProduct] = useState({ name: '', category: '', price: '' });  // New product form state
9     const [showModal, setShowModal] = useState(false);  // Control modal visibility
10
```

- products: Array yang berisi objek-objek produk. Setiap objek produk memiliki properti id, name, category, dan price.
- loading: State untuk menunjukkan apakah data produk sedang dimuat (true) atau sudah selesai dimuat (false).
- error: State untuk menyimpan pesan error jika terjadi kesalahan saat mengambil data produk.
- editingProduct: State untuk menyimpan objek produk yang sedang diedit (null jika tidak ada produk yang diedit).
- newProduct: State untuk menyimpan data produk baru yang sedang diinputkan dalam form.
- showModal: State untuk mengontrol visibilitas modal (true untuk menampilkan modal, false untuk menyembunyikan modal).

```
11    // Fetch products when component mounts
12    useEffect(() => {
13      const fetchProducts = async () => {
14        try {
15          const response = await fetch('/api/products');
16          if (!response.ok) {
17            throw new Error('Failed to fetch products');
18          }
19          const data = await response.json();
20          setProducts(data);
21        } catch (err) {
22          setError(err.message);
23        } finally {
24          setLoading(false);
25        }
26      };
27
28      fetchProducts();
29    }, []);
```

Fungsi fetchProducts melakukan:

- Fetch data produk dari endpoint /api/products menggunakan fetch.
- Jika request berhasil (response.ok bernilai true), data produk yang berupa JSON di-parse dan disimpan ke state products menggunakan setProducts.
- Jika terjadi kesalahan (catch), pesan error disimpan ke state error menggunakan setError.
- Terakhir (finally), state loading diubah menjadi false untuk menunjukkan bahwa data sudah selesai dimuat.

```
32    const filteredProducts = selectedCategory === 'Semua'
33      ? products
34      : products.filter((product) => product.category === selectedCategory);
35
```

Variabel filteredProducts menyimpan hasil filter dari array products berdasarkan selectedCategory.

- Jika selectedCategory adalah 'Semua', maka semua produk ditampilkan (filteredProducts berisi semua produk dari products).
- Jika selectedCategory adalah kategori tertentu (misalnya "Makanan"), maka hanya produk yang memiliki kategori tersebut yang ditampilkan (filteredProducts berisi produk hasil filter).

```
128
129         {loading && <p style={{ textAlign: 'center' }}>Loading...</p>}
130         {error && <p style={{ color: 'red', textAlign: 'center' }}>Error: {error}</p>}
131
132         <div style={{ display: 'flex', flexWrap: 'wrap', gap: '1.5rem', justifyContent: 'center' }}>
133           {filteredProducts.length > 0 ? (
134             filteredProducts.map((product) => (
135               <div
136                 key={product.id}
137                 style={{
138                   border: '1px solid #e0e0e0',
139                   borderRadius: '8px',
140                   padding: '1.5rem',
141                   width: '250px',
142                   textAlign: 'center',
143                   backgroundColor: '#fff',
144                   boxShadow: '0 4px 8px rgba(0, 0, 0, 0.1)',
145                   transition: 'transform 0.2s ease-in-out',
146                 }}
```

➢ **Memeriksa Kondisi dan Menampilkan Data:**

- Blok kode if pertama memeriksa apakah filteredProducts memiliki elemen atau tidak (filteredProducts.length > 0).
  - Jika true (ada produk hasil filter), maka kode di dalam blok if akan dijalankan untuk menampilkan daftar produk.
  - Jika false (tidak ada produk hasil filter), maka pesan "No products found for this category." akan ditampilkan.

➢ **Iterasi Produk:**

- Kode filteredProducts.map((product) => (...)) digunakan untuk mengiterasi setiap produk dalam array filteredProducts.
- Di dalam blok fungsi panah, kode akan dieksekusi untuk setiap produk.

➢ **Menampilkan Detail Produk:**

- <h3>{product.name}</h3> menampilkan nama produk.
- Kategori: {product.category} menampilkan kategori produk.
- <p style={{ fontWeight: 'bold' }}>Harga: Rp {product.price} menampilkan harga produk dalam format Rupiah.

➢ **Menampilkan Tombol Edit dan Delete:**

- Blok kode <div style={{ marginTop: '1rem' }}> ... membuat container untuk tombol Edit dan Delete.
- Tombol Edit:

    - <button onClick={() => handleEdit(product)} ... >Edit</button> membuat tombol Edit. Ketika diklik, fungsi handleEdit akan dipanggil dengan argumen product yang berisi informasi produk tersebut.
- Tombol Delete:
    - <button onClick={() => handleDelete(product.id)} ... >Delete</button> membuat tombol Delete. Ketika diklik, fungsi handleDelete akan dipanggil dengan argumen product.id yang berisi ID produk tersebut.