Keterangan:

File gambar produk ditambahkan ke folder public agar dapat digunakan di komponen ProdukPage.

Gambar yang ditambahkan meliputi:

- PCX.png: Gambar produk motor PCX.
- kws650.png: Gambar produk motor Kawasaki 650.
- vario.png: Gambar produk motor Vario.

Folder public digunakan untuk menyimpan aset statis seperti gambar, sehingga dapat diakses dengan mudah melalui URL relatif.

Program

page.tsx

app > modules > produk > page.tsx > ProdukPage > products

```tsx
export default function ProdukPage() {
  const products = [
    {
      id: 1,
      name: "PCX",
      price: "Rp 15.000.000",
      image: "/PCX.png",
    },
    {
      id: 2,
      name: "Kawasaki",
      price: "Rp 18.000.000",
      image: "/kws650.png",
    },
    {
      id: 3,
      name: "Vario",
      price: "Rp 20.000.000",
      image: "/vario.png",
    },
  ];
}
```

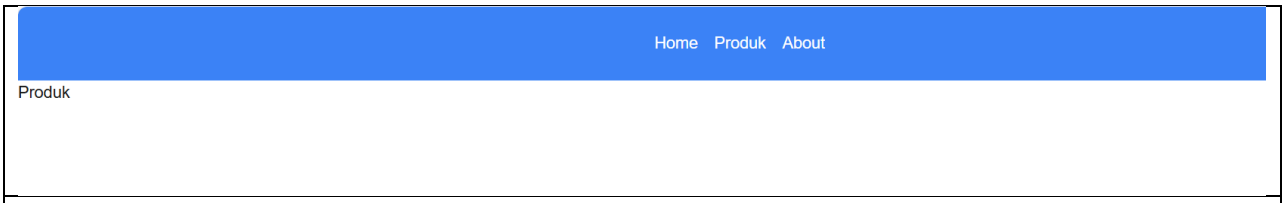
Keterangan :

- Mendeklarasikan komponen default bernama ProdukPage.
- Menginisialisasi array products yang berisi objek-objek data produk, termasuk id, name, price, dan image.

Program

```tsx
return (
  <main>
    <h1>Produk</h1>
  </main>
);
```

Output



Keterangan:

- Membungkus konten halaman di dalam elemen <main>.
- Menambahkan elemen <h1> sebagai judul halaman.

Program

```
return (
  <main>
    <h1>Produk</h1>
    <ul></ul>
  </main>
);
```

Keterangan:

- Membuka elemen HTML <ul> (unordered list), yang akan digunakan untuk menampilkan daftar produk dalam bentuk bullet point.

Program

```
return (
  <main>
    <h1>Produk</h1>
    <ul>
    {products.map((product) => (
    ))}
    </ul>
  </main>
);
```

Keterangan :

- products.map(...): Metode array JavaScript yang digunakan untuk membuat daftar elemen berdasarkan data dalam array products.
- Setiap item dalam array products diakses sebagai parameter product dalam callback function.

Program

```
return (
  <main>
    <h1>Produk</h1>
    <ul>
      {products.map((product) => (
        <li key={product.id}></li>
      ))}
    </ul>
  </main>
);
```

Keterangan:

- Elemen <li> dibuat untuk setiap item dalam array products.
- Properti key={product.id} digunakan oleh React untuk mengidentifikasi elemen secara unik, membantu meningkatkan performa saat melakukan rendering ulang komponen.

Program

```
{products.map((product) => (
  <li key={product.id}>
    <div>
      <img src={product.image} alt={product.name} />
    </div>
  </li>
))}
```

Output

Keterangan:

- Elemen <img> digunakan untuk menampilkan gambar produk.
- src={product.image}: Mengatur sumber gambar dari properti image di objek product. alt={product.name}: Memberikan teks alternatif (alt text) berdasarkan nama produk, dan membantu aksesibilitas.

Program

```
<ul>
  {products.map((product) => (
    <li key={product.id}>
      <div>
        <img src={product.image} alt={product.name} />
      </div>
      <div>
        <h2>{product.name}</h2>
      </div>
    </li>
  ))}
</ul>
```

Output

PCX

Keterangan:

- <h2>: Untuk menampilkan nama produk.
- {product.name}: Memasukkan nama produk dari array products.

Program

```
<div>
  <h2>{product.name}</h2>
  <p>{product.price}</p>
</div>
```

Output

PCX
Rp 15.000.000

Keterangan:

- Menambahkan tag <p> untuk Menampilkan harga produk.
- {product.price}: Memasukkan harga produk.

Program

```
<div>
  <h2>{product.name}</h2>
  <p>{product.price}</p>
  <button>Lihat Detail</button>
</div>
```

Output

PCX
Rp 15.000.000
Lihat Detail

Keterangan:

- Menambahkan tag <button>: Tombol aksi dengan teks "Lihat Detail".
  Berfungsi untuk memberikan interaksi lebih lanjut, seperti menavigasi ke halaman detail produk

Program

```
return (
  <main className="p-6 bg-gray-100 min-h-screen">
```

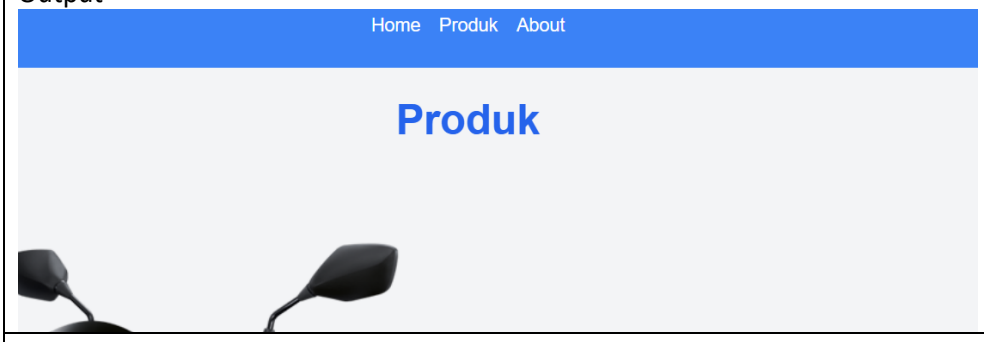
Keterangan:

- Pada elemen <main> menambahkan padding (p-6), latar belakang abu-abu muda (bg-gray-100), dan tinggi minimum 100% dari layar (min-h-screen).

Program

```
<h1 className="text-4xl font-bold text-center mb-8 text-blue-600">
  Produk
</h1>
```

Output



Keterangan:

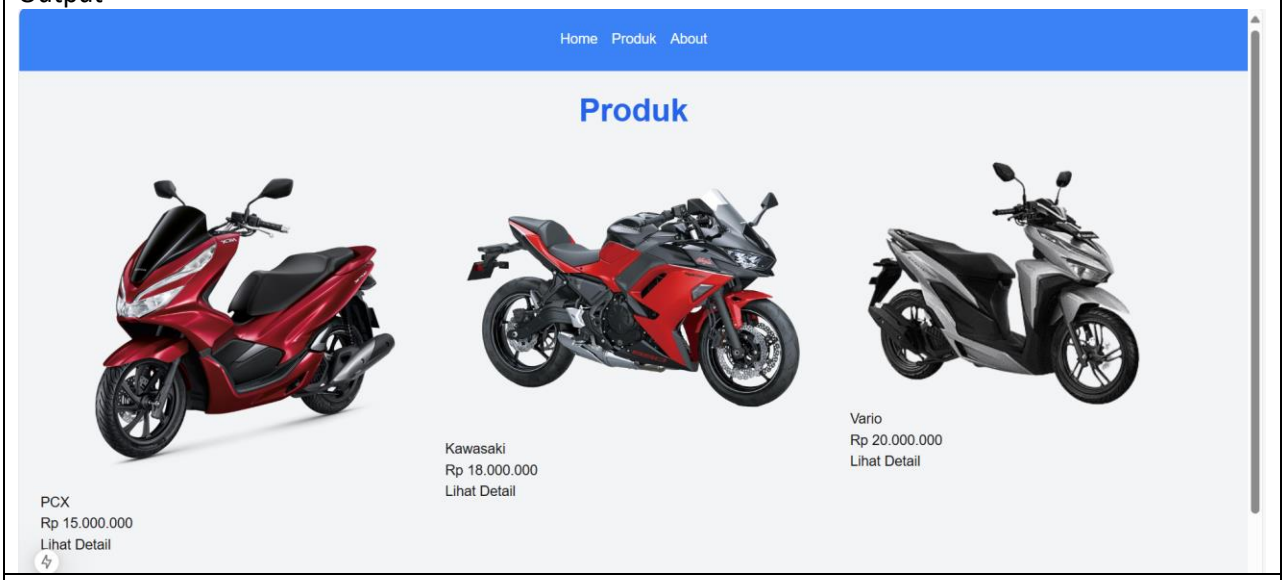
- Judul halaman menggunakan elemen <h1>. Kelas CSS:
- text-4xl: Ukuran teks besar.
- font-bold: Teks tebal.

- text-center: Menyusun teks di tengah.
- mb-8: Memberikan margin bawah 2rem.
- text-blue-600: Mengatur warna teks menjadi biru.

Program

```
<ul className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 gap-8">
```

Output



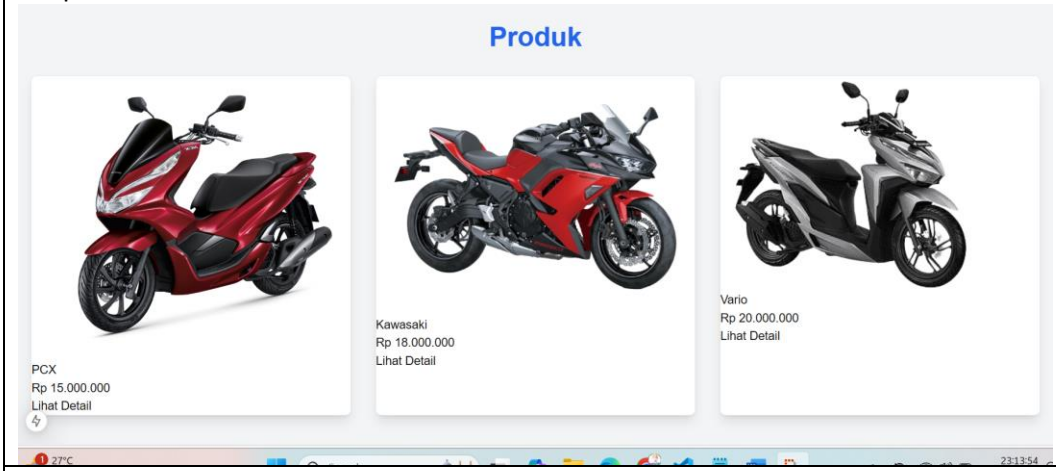
Keterangan:

- grid: Menggunakan grid layout.
- grid-cols-1: Satu kolom untuk ukuran layar kecil.
- sm:grid-cols-2: Dua kolom untuk ukuran layar menengah.
- md:grid-cols-3: Tiga kolom untuk ukuran layar besar.
- gap-8: Memberikan jarak antar elemen grid sebesar 2rem.

Program

```
<li
  key={product.id}
  className="border rounded-lg shadow-lg bg-white overflow-hidden hover:shadow-2xl transition-shadow duration-300"
>
```

Output



Keterangan :

- border: Memberikan border pada setiap item.
- rounded-lg: Membuat sudut elemen melengkung.
- shadow-lg: Memberikan bayangan besar pada elemen.
- bg-white: Memberikan latar belakang putih.
- overflow-hidden: Mencegah konten yang meluber keluar dari batas elemen.
- hover:shadow-2xl: Memberikan efek bayangan lebih besar saat hover.
- transition-shadow duration-300: Menambahkan transisi halus pada bayangan dengan durasi 300ms.

Program

```
<div className="w-full h-48 bg-gray-200 flex items-center justify-center">
  <img
    src={product.image}
    alt={product.name}
    className="max-w-full max-h-full object-contain"
  />
</div>
```

Output

Keterangan:
Div digunakan untuk menampilkan gambar produk dengan styling:

- w-full: Lebar penuh.
- h-48: Tinggi tetap 12rem.
- bg-gray-200: Latar belakang abu-abu muda.
- flex items-center justify-center: Menggunakan Flexbox untuk men-center-kan gambar.
- className="max-w-full max-h-full object-contain": Memastikan gambar menyesuaikan dengan ukuran kontainer tanpa terdistorsi.

Program

```
<div className="p-4">
  <h2>{product.name}</h2>
  <p>{product.price}</p>
  <button>Lihat Detail</button>
</div>
```

Keterangan:
p-4 :Menambahkan padding pada kontainer

Program

```
<h2 className="font-semibold text-xl text-gray-800">
  {product.name}
</h2>
<p className="text-gray-500">{product.price}</p>
```

Output

Keterangan:
Nama produk menggunakan <h2> dengan:

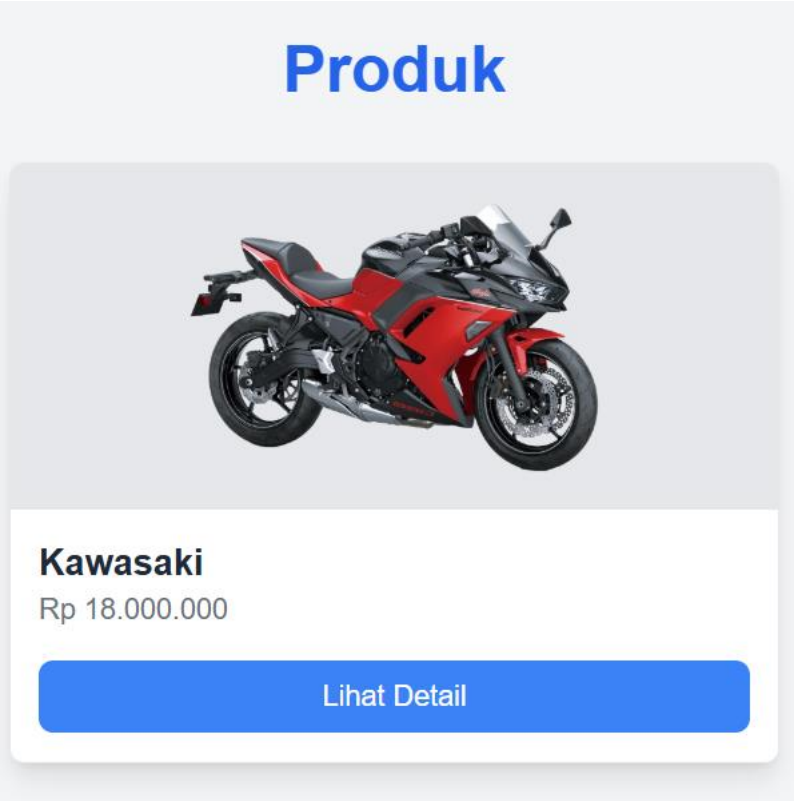
- font-semibold: Menebalkan teks.
- text-xl: Ukuran teks besar.
- text-gray-800: Warna teks abu-abu gelap.

Harga produk menggunakan <p> dengan kelas text-gray-500 untuk warna abu-abu muda.

Program

```
<button className="mt-4 w-full bg-blue-500 text-white py-2 rounded-lg hover:bg-blue-600 transition-colors">
  Lihat Detail
</button>
```

Output



Keterangan:

- mt-4: Margin atas 1rem.
- w-full: Lebar tombol penuh.
- bg-blue-500: Warna latar belakang biru.
- text-white: Warna teks putih.
- py-2: Padding vertikal 0.5rem.
- rounded-lg: Sudut tombol melengkung.
- hover:bg-blue-600: Efek hover dengan warna biru lebih gelap.
- transition-colors: Transisi halus pada perubahan warna.

Program

```
1  import { useState } from "react";
2
```

Keterangan:

menambahkan useState dari React, yang digunakan untuk menyimpan dan memperbarui nilai pencarian yang dimasukkan oleh pengguna. Fungsi useState ini memungkinkan kita untuk membuat state yang dapat diubah selama aplikasi berjalan. Di sini, kita menggunakan useState untuk menyimpan nilai input pencarian produk (misalnya, kata yang diketik oleh pengguna).

Program

```
      id: 3,
      name: "Vario",
      price: "Rp 20.000.000",
      image: "/vario.png",
    },
  ];
  const [searchQuery, setSearchQuery] = useState("");

  return (
    <main className="p-6 bg-gray-100 min-h-screen">
      <h1 className="text-4xl font-bold text-center mb-8 text-blue-600">
```

Keterangan:

- useState digunakan untuk mendeklarasikan state baru yang bernama searchQuery. State ini akan menyimpan nilai dari input pencarian yang dimasukkan oleh pengguna.
- searchQuery menyimpan teks yang diketik oleh pengguna dalam kolom pencarian.
- setSearchQuery adalah fungsi yang digunakan untuk memperbarui nilai searchQuery. Ketika pengguna mengetik sesuatu di kolom pencarian, nilai searchQuery akan diperbarui dengan teks baru yang dimasukkan.

Program

```
  ];
  const [searchQuery, setSearchQuery] = useState("");

  const filteredProducts = products.filter((product) =>
    product.name.toLowerCase().includes(searchQuery.toLowerCase())
  );
```

Keterangan:

- Produk yang ditampilkan difilter berdasarkan nama produk yang mengandung teks pencarian. Hal ini dilakukan dengan menggunakan fungsi filter pada array products. Filter ini membandingkan nama produk dengan searchQuery, yang disesuaikan dengan huruf kecil agar pencarian tidak sensitif terhadap kapitalisasi.
- Fungsi filter akan menghasilkan array baru yang hanya berisi produk yang nama produknya mencocokkan (dengan cara yang tidak sensitif terhadap kapitalisasi) query pencarian.
- Misalnya, jika pengguna mengetikkan "pcx", maka produk yang memiliki nama "PCX" akan tetap muncul, meskipun berbeda kapitalisasi hurufnya.

Program

app > modules > produk > page.tsx > ProdukPage

```
1   "use client";
2
3   import { useState } from "react";
```

Keterangan:

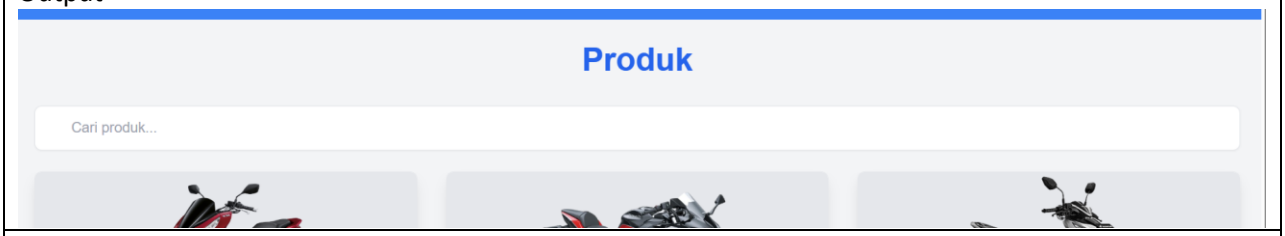
- Dengan menambahkan use client di bagian atas komponen atau file, Next.js memastikan bahwa komponen ini hanya akan dirender di sisi klien.
- Ini berguna menggunakan fitur JavaScript yang hanya tersedia di sisi klien, seperti penggunaan state, efek samping (useEffect), dan event handler di React, yang tidak bisa dijalankan di sisi server karena tidak ada akses langsung ke DOM.

Program

```
return (
  <main className="p-6 bg-gray-100 min-h-screen">
    <h1 className="text-4xl font-bold text-center mb-8 text-blue-600">
      Produk
    </h1>

    <div className="mb-6 flex items-center">
      <input
        type="text"
        value={searchQuery}
        onChange={(e) => setSearchQuery(e.target.value)}
        placeholder="Cari produk..."
        className="w-full p-3 pl-10 border rounded-lg shadow-sm focus:outline-none focus:ring-2 focus:ring-blue-400"
      />
    </div>
```

Output



Keterangan:

- : Membuat sebuah kontainer untuk menampung elemen input dengan margin bawah (mb-6). flex digunakan untuk menata elemen secara horizontal, dan items-center memastikan elemen-elemen di dalam kontainer terpusat secara vertikal.
- <input>: Elemen input untuk menerima input dari pengguna. Atribut:
  - value={searchQuery}: Nilai input diikat ke variabel state searchQuery. Jadi, saat input berubah, state searchQuery akan diperbarui.
  - onChange={(e) => setSearchQuery(e.target.value)}: Fungsi ini akan dijalankan setiap kali pengguna mengetik di input. Fungsi ini memperbarui nilai state searchQuery dengan nilai yang diketik pengguna.
  - placeholder="Cari produk...": Teks placeholder yang muncul di dalam input ketika kosong, memberikan petunjuk kepada pengguna untuk mencari produk.
  - w-full: Membuat input mengisi lebar penuh kontainer.

- p-3: Memberikan padding (ruang) di dalam input agar teks tidak terlalu dekat dengan batas.
- pl-10: Memberikan padding ekstra di sisi kiri untuk memberi ruang bagi ikon pencarian.
- border: Menambahkan garis batas di sekitar input.
- rounded-lg: Membuat sudut input membulat agar terlihat lebih halus.
- shadow-sm: Menambahkan bayangan kecil untuk memberi efek kedalaman.
- focus:outline-none: Menghilangkan outline default ketika input fokus.
- focus:ring-2 focus:ring-blue-400: Menambahkan efek ring biru saat input fokus untuk menandakan elemen aktif.

Program

```
<div className="mb-6 flex items-center">
  <span className="text-gray-600 mr-2">
    <i className="fas fa-search"></i>
  </span>
  <input
```

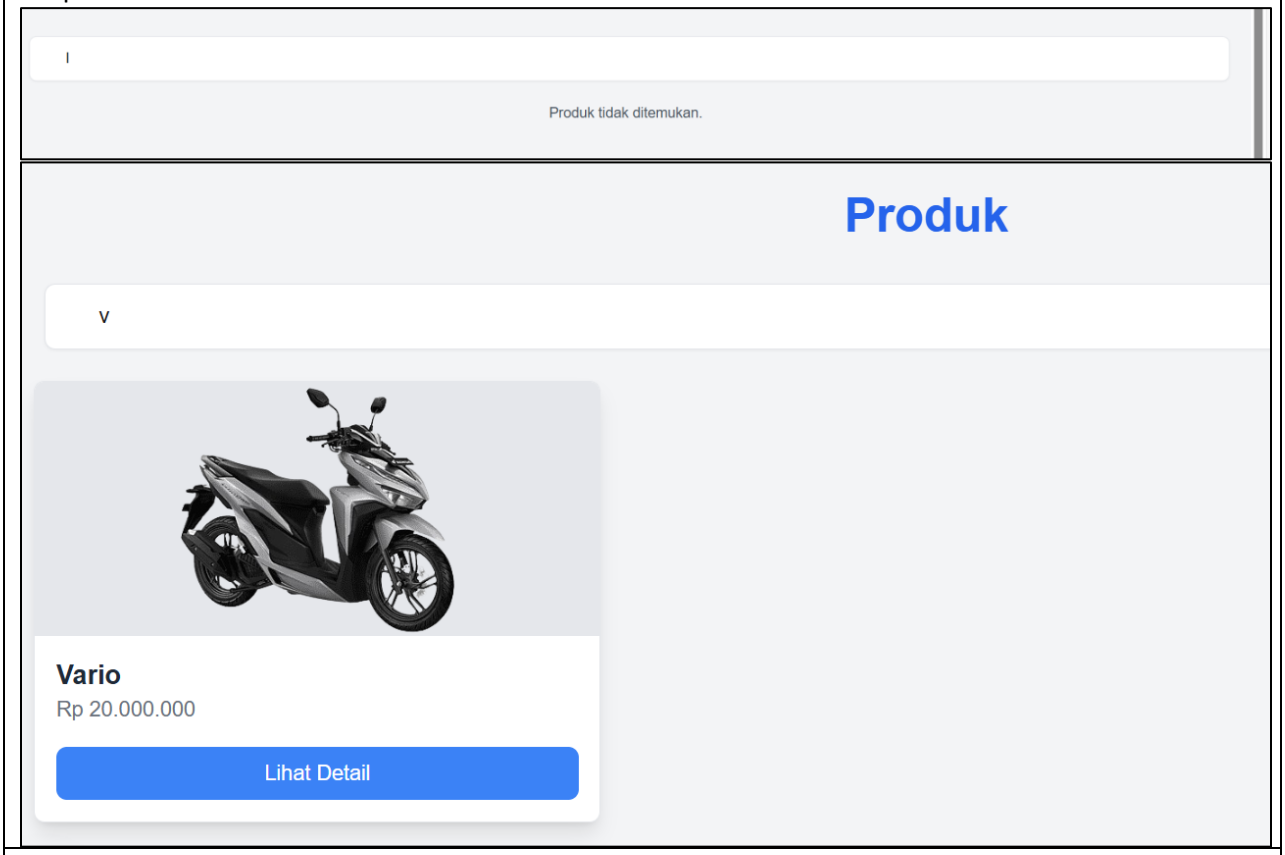
Keterangan:

- : Kontainer untuk ikon pencarian. Kelas text-gray-600 memberikan warna abu-abu pada ikon, dan mr-2 menambahkan margin kanan agar ada jarak antara ikon dan input.
- : Menggunakan FontAwesome untuk menambahkan ikon pencarian (search icon). Ini memberikan petunjuk visual kepada pengguna untuk mengetikkan sesuatu yang ingin mereka cari.

Program

```
<ul className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 gap-8">
  {filteredProducts.length > 0 ? (
    filteredProducts.map((product) => (
      <li
        key={product.id}
        className="border rounded-lg shadow-lg bg-white overflow-hidden hover:shadow-2xl transition-shadow duration-300"
      >
        <div className="w-full h-48 bg-gray-200 flex items-center justify-center">
          <img
            src={product.image}
            alt={product.name}
            className="max-w-full max-h-full object-contain"
          />
        </div>
        <div className="p-4">
          <h2 className="font-semibold text-xl text-gray-800">
            {product.name}
          </h2>
          <p className="text-gray-500">{product.price}</p>
          <button className="mt-4 w-full bg-blue-500 text-white py-2 rounded-lg hover:bg-blue-600 transition-colors">
            Lihat Detail
          </button>
        </div>
      </li>
    ))
  ) : (
    <li className="col-span-3 text-center text-gray-600">
      Produk tidak ditemukan.
    </li>
  )}
```

Output



Keterangan:

- Bagian ini menampilkan produk berdasarkan hasil penyaringan menggunakan filteredProducts. Jika ada produk yang cocok dengan query pencarian, produk tersebut akan ditampilkan dalam daftar. Jika tidak ada produk yang cocok, maka pesan "Produk tidak ditemukan" akan muncul.
- filteredProducts.length > 0: Mengecek apakah ada produk yang cocok dengan pencarian.
- Jika ada, maka produk tersebut akan dipetakan menggunakan .map() untuk menampilkan setiap produk dalam elemen <li>.
- Jika tidak ada produk yang ditemukan (array kosong), maka elemen <li> akan menampilkan teks "Produk tidak ditemukan".
- col-span-3: Mengatur elemen untuk melintang (span) 3 kolom dalam grid layout. Ini biasanya digunakan dalam konteks grid untuk menentukan berapa banyak kolom yang harus ditempati elemen tersebut.
- text-center: Menyelaraskan teks di dalam elemen ke tengah.
- text-gray-600: Memberikan warna teks abu-abu dengan intensitas sedang, sesuai dengan skema warna Tailwind.

Program

```
export default function ProdukPage() {
  const products = [
    {
      id: 1,
      name: "PCX",
      price: "Rp 15.000.000",
      image: "/PCX.png",
      category: "matic",
    },
```

Keterangan :

Penambahan properti category pada data produk dilakukan untuk mengklasifikasikan produk berdasarkan jenis atau kategori tertentu, seperti matic, trail, dan sport. Hal ini memungkinkan pengelompokan dan pencarian produk menjadi lebih mudah, serta dapat meningkatkan fleksibilitas dalam pengelolaan data produk, misalnya untuk fitur filter atau pencarian berdasarkan kategori.

Program

```
];
const [searchQuery, setSearchQuery] = useState("");
const [selectedCategory, setSelectedCategory] = useState("all");

const filteredProducts = products.filter((product) =>
  product.name.toLowerCase().includes(searchQuery.toLowerCase())
);
```

Keterangan:

Penambahan state selectedCategory bertujuan untuk menyimpan kategori yang dipilih oleh pengguna. Nilai awal state diatur ke "all" untuk menampilkan semua kategori secara default. Dengan state ini, aplikasi dapat secara dinamis memfilter dan menampilkan produk sesuai dengan kategori yang dipilih, memberikan pengalaman yang lebih interaktif bagi pengguna.

Program

```
const [searchQuery, setSearchQuery] = useState("");
const [selectedCategory, setSelectedCategory] = useState("all");

const filteredProducts = products.filter((product) => {
  const matchesSearch = product.name
    .toLowerCase()
    .includes(searchQuery.toLowerCase());
  const matchesCategory =
    selectedCategory === "all" || product.category === selectedCategory;
  return matchesSearch && matchesCategory;
});
```

Keterangan:

Logika filter diperbarui untuk mendukung fitur pencarian sekaligus kategori. Produk akan difilter berdasarkan dua kondisi:

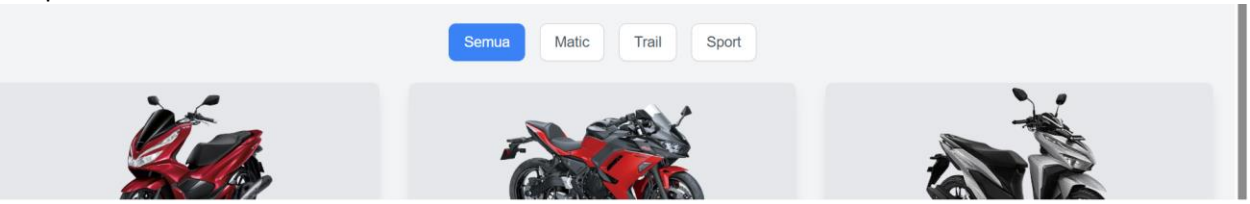
- Pencarian: Produk cocok dengan kata kunci pencarian (searchQuery).
- Kategori: Produk cocok dengan kategori yang dipilih (selectedCategory), atau semua kategori ditampilkan jika kategori yang dipilih adalah "all".

Pembaruan ini memungkinkan pengguna untuk mempersempit hasil pencarian berdasarkan kategori tertentu, memberikan fleksibilitas yang lebih besar dalam menampilkan produk.

Program

```
{/* Filter kategori */}
<div className="mb-6 flex justify-center gap-4">
  {[
    { label: "Semua", value: "all" },
    { label: "Matic", value: "matic" },
    { label: "Trail", value: "trail" },
    { label: "Sport", value: "sport" },
  ].map((category) => (
    <button
      key={category.value}
      onClick={() => setSelectedCategory(category.value)}
      className={`px-4 py-2 rounded-lg border transition-colors shadow-sm ${
        selectedCategory === category.value
          ? "bg-blue-500 text-white border-blue-500"
          : "bg-white text-gray-600 border-gray-300 hover:bg-gray-100"
      }`}
    >
      {category.label}
    </button>
  ))}
</div>
```

Output



Keterangan:

1. Container
   Elemen digunakan sebagai wadah tombol-tombol kategori.
   - className="mb-6 flex justify-center gap-4":
   - mb-6: Margin bawah sebesar 1.5rem (24px).
   - flex: Mengaktifkan layout fleksibel untuk membuat tombol berada dalam satu baris.
   - justify-center: Menengahkan tombol secara horizontal.
   - gap-4: Memberikan jarak sebesar 16px antara tombol-tombol.

2. Sebuah array berisi daftar kategori (label untuk teks tombol, value untuk nilai filter).
3. Array kategori di-map untuk membuat tombol kategori.
   - key={category.value}: Nilai unik untuk setiap tombol.
   - onClick={() => setSelectedCategory(category.value)}: Mengubah state selectedCategory sesuai nilai tombol yang diklik.
   - {category.label}: Menampilkan teks tombol.
4. Tombol memiliki gaya yang berubah berdasarkan kondisi selectedCategory.
   - Gaya Dasar:
     - px-4 py-2: Padding horizontal 16px dan vertikal 8px.
     - rounded-lg: Membuat sudut tombol melengkung.
     - border: Menambahkan garis tepi tombol.
     - transition-colors: Animasi transisi warna saat tombol berubah.
     - shadow-sm: Memberikan efek bayangan kecil pada tombol.

   - Class Kondisional:

     - Ketika Kategori Dipilih (selectedCategory === category.value):
       - bg-blue-500: Latar belakang biru.
       - text-white: Warna teks putih.
       - border-blue-500: Garis tepi biru.
     - Ketika Kategori Tidak Dipilih:
       - bg-white: Latar belakang putih.
       - text-gray-600: Warna teks abu-abu gelap.
       - border-gray-300: Garis tepi abu-abu muda.
       - hover:bg-gray-100: Latar belakang berubah menjadi abu-abu muda saat di-hover.

Program

```
"use client";

import { useState, useEffect } from "react";
import { useSearchParams } from "next/navigation";

interface Specification {
  label: string;
  value: string;
}
```

Keterangan :

- Interface Specification:
  - Sebuah kontrak atau struktur data yang mendefinisikan properti label dan value.
  - Interface digunakan untuk memastikan bahwa setiap objek yang diimplementasikan memiliki properti yang sesuai.

- Properti label:
  - Bertipe string.
  - Digunakan untuk menyimpan informasi deskriptif, seperti nama spesifikasi.

- Properti value:
  - Bertipe string.
  - Digunakan untuk menyimpan nilai dari spesifikasi.

Program

```tsx
interface Specification {
  label: string;
  value: string;
}
interface Product {
  id: number;
  name: string;
  price: string;
  image: string;
  category: string;
  description: string;
  specifications: Specification[];
}
```

Keterangan:

- id (number):
  Merupakan identitas unik untuk setiap produk.
- name (string):
  Nama produk.
- price (string):
  Harga produk dalam bentuk string.
- image (string):
  URL atau path gambar produk.
- category (string):
  Kategori produk.
- description (string):
  Deskripsi singkat atau panjang mengenai produk.
- specifications (Specification[]):
  Array dari spesifikasi produk, yang menggunakan interface Specification yang telah didefinisikan sebelumnya.

Program

```tsx
import { useState, useEffect } from "react";
import { useSearchParams, useRouter } from "next/navigation";
```

```tsx
  ];
  const router = useRouter();
  const searchParams = useSearchParams();
  const [searchQuery, setSearchQuery] = useState("");
  const [selectedCategory, setSelectedCategory] = useState("all");
```

Keterangan:
useRouter digunakan untuk mempermudah manipulasi URL. Misalnya, memperbarui URL sesuai pencarian atau kategori tanpa melakukan refresh halaman.

Program

```
});

const handleSearch = () => {
  router.push(`?search=${searchQuery}&category=${selectedCategory}`);
};

return (
  <main className="p-6 bg-gray-100 min-h-screen">
    <h1 className="text-4xl font-bold text-center mb-8 text-blue-600">
      Produk
    </h1>
```

Keterangan :

- handleSearch:
  - Fungsi ini bertugas menangani aksi pencarian (misalnya, saat pengguna menekan tombol "Search").
  - Mengarahkan pengguna ke URL baru dengan parameter pencarian dan kategori.

- router.push:
  - Metode ini digunakan untuk mengarahkan pengguna ke rute atau URL tertentu.
  - Dalam hal ini, router.push akan memperbarui URL dengan menambahkan query string.

- Template String:
  - ?search=${searchQuery}: Menambahkan parameter query bernama search yang nilainya berasal dari variabel searchQuery.
  - &category=${selectedCategory}: Menambahkan parameter query kedua, yaitu category, yang nilainya berasal dari variabel selectedCategory.

- Variabel Dinamis:
  - searchQuery: Nilai input dari pengguna yang mewakili kata kunci pencarian.
  - selectedCategory: Nilai kategori yang dipilih pengguna.

Program

```
const router = useRouter();
const searchParams = useSearchParams();
const [searchQuery, setSearchQuery] = useState("");
const [selectedCategory, setSelectedCategory] = useState("all");
const [selectedProduct, setSelectedProduct] = useState<Product | null>(null);
```

Keterangan:
selectedProduct:

- State: Menyimpan produk yang dipilih oleh pengguna.
- Tipe Data:
  - Product | null: Produk dapat berupa objek bertipe Product atau null jika belum ada produk yang dipilih.
  - Tipe Product harus sudah didefinisikan sebelumnya
- Inisialisasi Awal: null (tidak ada produk yang dipilih saat komponen pertama kali dirender).
- Fungsi Setter: setSelectedProduct digunakan untuk mengubah nilai produk yang dipilih.

Program

```
const router = useRouter();
const searchParams = useSearchParams();
const [searchQuery, setSearchQuery] = useState("");
const [selectedCategory, setSelectedCategory] = useState("all");
const [selectedProduct, setSelectedProduct] = useState<Product | null>(null);
const [isModalOpen, setIsModalOpen] = useState(false);
```

Keterangan:
isModalOpen:

- State: Menyimpan status modal (pop-up), apakah sedang terbuka (true) atau tertutup (false).
- Tipe Data: boolean.
- Inisialisasi Awal: false (modal dalam keadaan tertutup).
- Fungsi Setter: setIsModalOpen untuk membuka atau menutup modal.

Program

```
const router = useRouter();
const searchParams = useSearchParams();
const [searchQuery, setSearchQuery] = useState("");
const [selectedCategory, setSelectedCategory] = useState("all");
const [selectedProduct, setSelectedProduct] = useState<Product | null>(null);
const [isModalOpen, setIsModalOpen] = useState(false);
const [activeTab, setActiveTab] = useState<"description" | "specifications">(
  "description"
);
```

Keterangan:
activeTab:

- State: Menyimpan tab yang sedang aktif di antarmuka.
- Tipe Data: "description" | "specifications" (pilihan hanya bisa berupa salah satu dari kedua string ini).
- Inisialisasi Awal: "description" (tab deskripsi aktif secara default).
- Fungsi Setter: setActiveTab untuk mengubah tab yang aktif.

Program

```
const handleOpenModal = (product: Product) => {
  setSelectedProduct(product);
  setActiveTab("description");
  setIsModalOpen(true);
};
```

Keterangan:

- Parameter:
  product: Product: Parameter ini adalah objek produk yang akan digunakan untuk menampilkan detail produk. Tipe datanya adalah Product, yang didefinisikan sebelumnya sebagai sebuah interface

- Fungsi setSelectedProduct(product):
  - Mengatur state selectedProduct dengan produk yang dipilih.
  - Produk ini akan digunakan untuk menampilkan detail di modal.

- Fungsi setActiveTab("description"):
  - Mengatur state activeTab ke "description".
  - Artinya, saat modal terbuka, tab "Description" akan aktif secara default.

- Fungsi setIsModalOpen(true):
  - Membuka modal dengan mengatur state isModalOpen menjadi true.

Program

```
const handleCloseModal = () => {
  setIsModalOpen(false);
  setSelectedProduct(null);
};
```

Keterangan:
handleCloseModal:

- Tujuan Fungsi:
  - Menutup modal yang sedang terbuka.
  - Membersihkan produk yang sebelumnya dipilih.

- Fungsi setIsModalOpen(false):
  - Mengatur state isModalOpen menjadi false.
  - Ini memastikan modal tidak ditampilkan lagi di antarmuka.

- Fungsi setSelectedProduct(null):
  - Mengatur state selectedProduct kembali ke null.
  - Membersihkan informasi produk yang dipilih sebelumnya, sehingga modal tidak menyimpan data lama.

Program

```
{isModalOpen && selectedProduct && (
  <div className="fixed inset-0 bg-black bg-opacity-50 flex items-center justify-center z-50">
    <div className="bg-white rounded-lg p-6 shadow-lg max-w-lg w-full">
      <div className="flex justify-between items-center mb-4">
        <h2 className="text-xl font-bold">{selectedProduct.name}</h2>
        <button onClick={handleCloseModal} className="text-gray-600">
          ×
        </button>
      </div>
    </div>
  </div>
)}
```

Keterangan :

1. Kondisi Ternary untuk Render Modal
   {isModalOpen && selectedProduct && (...)}
   - Modal hanya akan dirender jika kedua kondisi berikut terpenuhi:
   - isModalOpen: Variabel boolean yang menunjukkan apakah modal dalam keadaan terbuka (true) atau tertutup (false).
   - selectedProduct: Variabel yang berisi data produk yang dipilih. Modal hanya akan muncul jika ada produk yang dipilih.

2. Struktur Modal

   - fixed inset-0: Modal menggunakan posisi tetap dan mencakup seluruh viewport (top: 0, right: 0, bottom: 0, left: 0).
   - bg-black bg-opacity-50: Background overlay berwarna hitam dengan transparansi 50% untuk memberikan efek fokus ke modal.
   - flex items-center justify-center: Menggunakan Flexbox untuk menempatkan modal di tengah layar, baik secara horizontal maupun vertikal.
   - z-50: Memberikan prioritas tinggi pada modal dengan z-index 50.

3. Kontainer Utama Modal

   - bg-white: Background modal berwarna putih.
   - rounded-lg: Sudut modal dibuat melengkung dengan radius besar.
   - p-6: Memberikan padding sebesar 1.5rem (24px) di dalam modal.
   - shadow-lg: Menambahkan efek bayangan untuk memberikan kesan mengambang.

- max-w-lg w-full: Modal memiliki lebar maksimal setara lg (32rem atau 512px) dengan mengambil seluruh lebar kontainer jika kurang dari itu.

4. 4. Header Modal

   - flex justify-between items-center: Mengatur header modal dengan Flexbox sehingga:
   - justify-between: Elemen di dalamnya dipisah ke ujung kiri dan kanan.
   - items-center: Elemen di tengah vertikal secara sejajar.
   - mb-4: Memberikan margin bawah sebesar 1rem (16px).

5. Judul Produk
   <h2 className="text-xl font-bold">{selectedProduct.name}</h2>
   - Menampilkan nama produk yang dipilih (selectedProduct.name).
   - text-xl: Ukuran teks judul besar.
   - font-bold: Menambahkan ketebalan pada teks.

6. Tombol Close
   <button onClick={handleCloseModal} className="text-gray-600">
   ×
   </button>
   - onClick={handleCloseModal}: Fungsi yang dipanggil ketika tombol "×" diklik untuk menutup modal.
   - className="text-gray-600": Menambahkan warna teks abu-abu sedang untuk ikon tombol "×".

Program

```
<div className="mb-4">
  <button
    onClick={() => setActiveTab("description")}
    className={`px-4 py-2 rounded-t-lg ${
      activeTab === "description"
        ? "bg-blue-500 text-white"
        : "bg-gray-100"
    }`}
  >
    Deskripsi
  </button>
  <button
    onClick={() => setActiveTab("specifications")}
    className={`px-4 py-2 rounded-t-lg ${
      activeTab === "specifications"
        ? "bg-blue-500 text-white"
        : "bg-gray-100"
    }`}
  >
    Spesifikasi
  </button>
</div>
```

Keterangan:

1. Struktur Utama

   - mb-4: Memberikan margin bawah sebesar 1rem (16px) untuk memberikan jarak antara elemen ini dengan elemen di bawahnya.

2. Tombol Tab Deskripsi
   - onClick={() => setActiveTab("description")}:
     - Ketika tombol ini diklik, fungsi setActiveTab dipanggil dengan argumen "description".
     - Fungsi ini bertugas mengubah state activeTab menjadi "description", yang menandakan tab "Deskripsi" sedang aktif.

   - className={...}:

     - px-4 py-2: Padding horizontal sebesar 1rem (16px) dan padding vertikal sebesar 0.5rem (8px).
     - rounded-t-lg: Sudut atas tombol dibuat melengkung dengan radius besar.
     - Tampilan tombol bergantung pada kondisi activeTab === "description":
       - Jika tab "Deskripsi" aktif, tombol akan memiliki kelas bg-blue-500 text-white:
         - bg-blue-500: Latar belakang tombol berwarna biru.
         - text-white: Teks tombol berwarna putih.
       - Jika tidak aktif, tombol akan memiliki kelas bg-gray-100, yang membuat tombol

memiliki latar belakang abu-abu terang.

3. Tombol tab spesifikasi
    - onClick={() => setActiveTab("specifications")}:
      Mengubah state activeTab menjadi "specifications" ketika tombol ini diklik.
    - Kondisi activeTab === "specifications":
        - Jika tab "Spesifikasi" aktif, tombol mendapatkan kelas bg-blue-500 text-white.
        - Jika tidak aktif, tombol mendapatkan kelas bg-gray-100.

Program

```
{activeTab === "description" ? (
  <p>{selectedProduct.description}</p>
) : (
  <ul>
    {selectedProduct.specifications.map((spec) => (
      <li key={spec.label}>
        <strong>{spec.label}:</strong> {spec.value}
      </li>
    ))}
  </ul>
)}
```

Keterangan:

- activeTab === "description":
    - Jika state activeTab bernilai "description", maka konten deskripsi produk akan ditampilkan.
    - Jika tidak, maka spesifikasi produk akan ditampilkan.

- {selectedProduct.description}
    - Menampilkan deskripsi produk dalam elemen HTML (paragraf).
    - selectedProduct.description:
        - Mengambil data deskripsi dari objek selectedProduct.
        - Misalnya, jika selectedProduct.description = "Produk ini berkualitas tinggi.", maka teks tersebut akan ditampilkan di dalam paragraf.

- : Membungkus daftar spesifikasi dalam elemen daftar tidak berurutan (unordered list).
- selectedProduct.specifications.map((spec) => ...):
    - Mengiterasi array selectedProduct.specifications untuk menghasilkan daftar spesifikasi.
    - Setiap elemen array spec harus memiliki properti label dan value.
- <li key={spec.label}>:
    - Setiap spesifikasi dirender dalam elemen daftar <li>.
    - key={spec.label}:
        - Properti key adalah atribut unik untuk setiap elemen dalam daftar (dibutuhkan oleh React untuk optimasi rendering).
        - label dari spesifikasi digunakan sebagai key karena unik untuk setiap item.
- Tampilan Format:
    - {spec.label}:

- Menampilkan label spesifikasi dengan teks tebal.
  - {spec.value}:
    - Menampilkan nilai spesifikasi setelah label.

Program

```
  category: "matic",
  description: "PCX adalah motor matic terbaik di kelasnya.",
  specifications: [
    { label: "Mesin", value: "150cc" },
    { label: "Tipe", value: "Matic" },
    { label: "Warna", value: "Hitam" },
  ],
},
```

Keterangan :

- description:
  - Properti ini berfungsi untuk memberikan deskripsi singkat mengenai produk tersebut.
  - Deskripsi ini memberikan informasi singkat yang membantu pengguna memahami produk secara lebih detail.

- Specifications:
  - specifications adalah array yang berisi objek-objek yang memberikan detail teknis atau fitur utama dari produk.
  - Setiap objek dalam array specifications memiliki dua properti:
  - label: Menyebutkan nama dari spesifikasi atau fitur produk. Contohnya "Mesin", "Tipe", dan "Warna".
  - value: Nilai atau informasi terkait dari spesifikasi tersebut. Contohnya, untuk label

Program

```
const router = useRouter();
const searchParams = useSearchParams();
const initialSearchQuery = searchParams.get("search") || "";
const initialCategory = searchParams.get("category") || "all";
```

Keterangan:

- searchParams.get("search"):
  - Mengambil nilai dari parameter query string search yang ada pada URL.
  - Jika parameter search ada dalam URL (misalnya: ?search=PCX), maka searchParams.get("search") akan mengembalikan nilai PCX.
  - Jika parameter search tidak ada, maka searchParams.get("search") akan mengembalikan null. Dalam hal ini, operator || "" digunakan untuk memberikan nilai default berupa string kosong "".
- searchParams.get("category"):
  - Mengambil nilai dari parameter query string category yang ada pada URL.
  - Jika parameter category ada dalam URL (misalnya: ?category=matic), maka

searchParams.get("category") akan mengembalikan nilai matic.

- Jika parameter category tidak ada, maka searchParams.get("category") akan mengembalikan null. Dalam hal ini, operator || "all" memberikan nilai default berupa string "all".