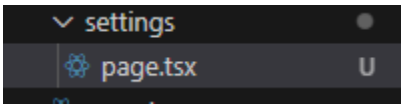
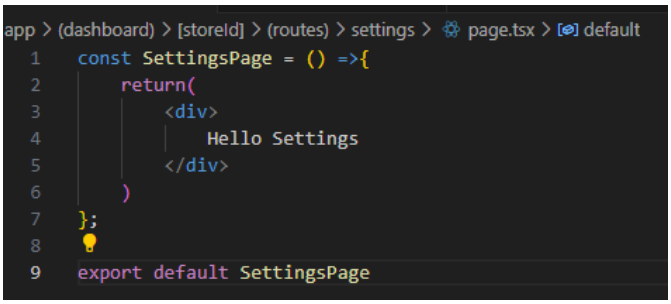
Nama: Rico Decaprio Esa Ananta
Npm:22312081

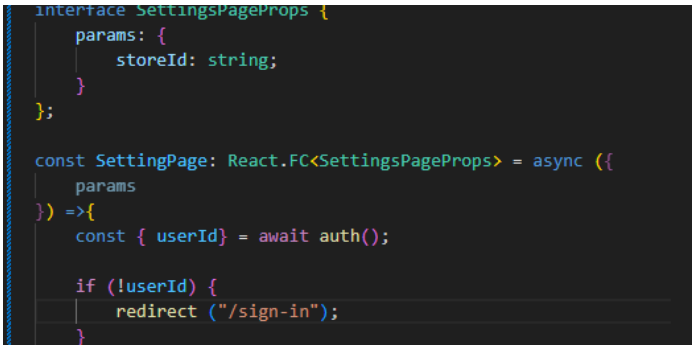
1.Membuat folder setting dan membuat file di dalam folder app



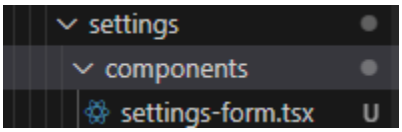
2.membuat hello setting di page.tsx



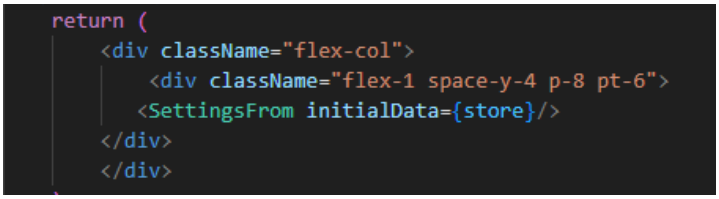
3.membuat sign –in



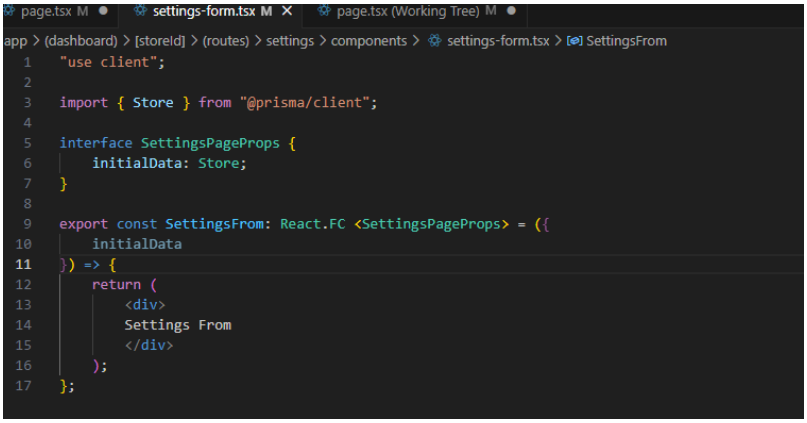
4.membuat folder components dan membuat file settings-form.tsx



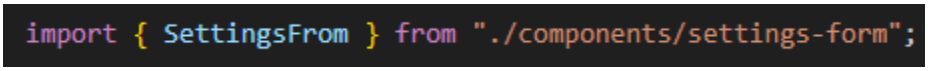
5. Menampilkan halaman yang memiliki tata letak vertikal dengan menggunakan flexbox.



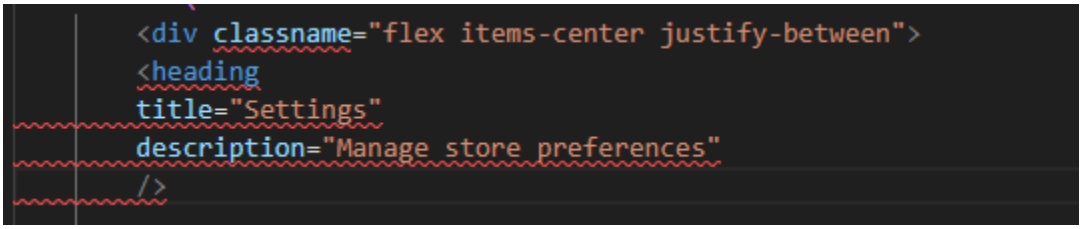
6. menambah program pada settings- from.tsx berfungsi untuk menampilkan atau memproses data terkait pengaturan (settings).



7. Kode tersebut adalah pernyataan **import** dalam JavaScript atau TypeScript

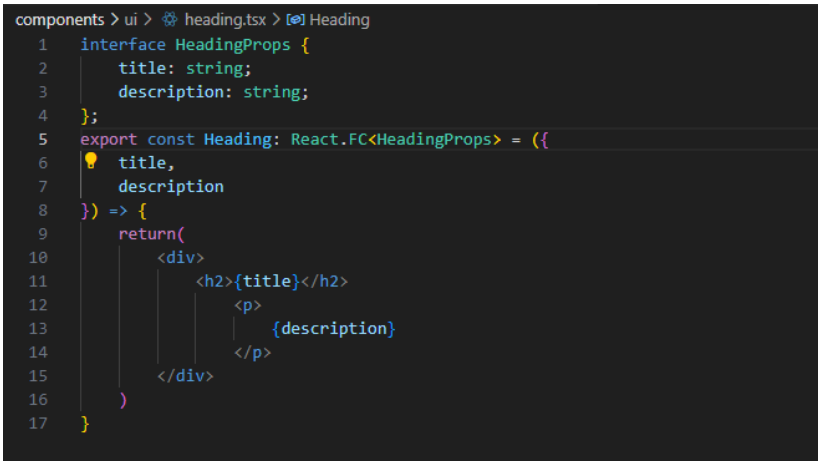


8. potongan JSX (JavaScript XML) yang berfungsi untuk membuat struktur HTML dengan bantuan React atau Next.js.



9.

definisi sebuah komponen React bernama Heading. Komponen ini dirancang untuk menampilkan judul dan deskripsi.



10.fungsi kode program tersebut komponen React yang mendefinisikan sebuah tombol dengan ikon bergambar tong sampah (trash icon).

```
<Button
    variant="destructive"
    size="icon"
    onClick={() => {}}
>
<Trash className="h-4 w-4" />
</Button>
```

11. menginstall separator

```
vortex> npx shadcn@latest add separator
```

12. Dengan Zod, Anda dapat membuat skema (schemas) yang mendefinisikan struktur

```
import * as z from "zod";
```

13. useFrom: Mengelola status

zodResolver: Mengintegrasikan Zod untuk validasi form dengan React Hook Form.

```
import { useForm } from "react-hook-form";
import { zodResolver } from "@hookform/resolvers/zod";
```

14. useState: dalah hook bawaan dari React yang digunakan untuk mengelola state di dalam komponen fungsi React.

Form:Menyediakan fitur seperti validasi, pengiriman data, atau integrasi dengan pustaka form lain (misalnya React Hook Form).

```
import { useState } from "react";
import { Form } from "@/components/ui/form";
```

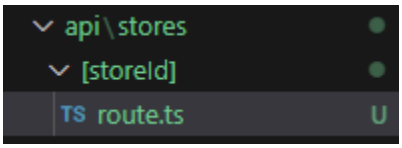
15. lemen-elemen input seperti `<input>`, `<textarea>`, atau `<select>` untuk menangani logika pengelolaan form.

```
import {
    Form,
    FormControl,
    FormField,
    FormItem,
    FormLabel
} from "@/components/ui/form";
```

16. mengirimkan formulir (submit), dan ketika aplikasi dalam kondisi loading, tombol tersebut akan dinonaktifkan sehingga pengguna tidak bisa mengkliknya dua kali atau mengirim formulir lagi.

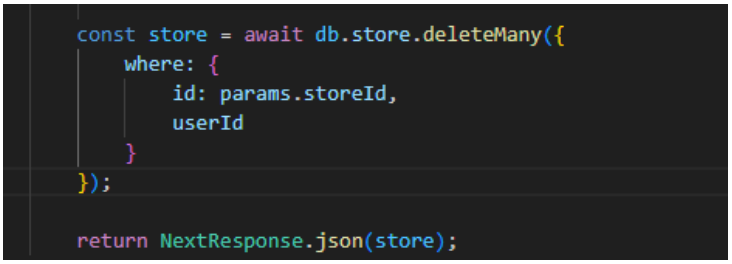
```
</div>
<button disabled={loading} className="ml-auto" type="submit">
    save changes
</button>
</form>
```

17.buat folder di folder api nama folder storeId dan buat file route.ts



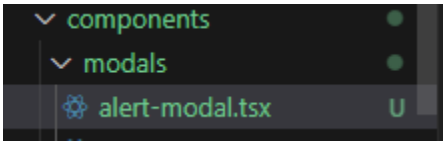
18. kode ini memastikan bahwa hanya pengguna yang terautentikasi yang bisa mengupdate data, dan juga memastikan bahwa parameter `name` disediakan dalam request.

18. berfungsi untuk menghapus satu atau lebih entri toko dari database berdasarkan storeId dan userId yang disertakan dalam request, kemudian mengembalikan hasilnya dalam format JSON.



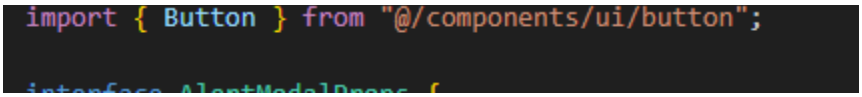
19. mempermudah pembuatan form yang modular, terstruktur, dan mudah dibaca.

20.membuat file



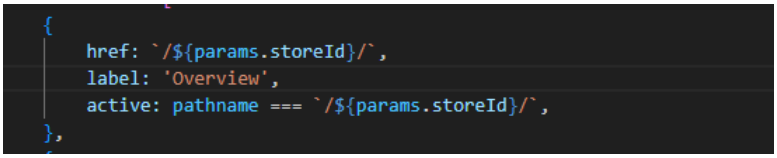
21.d igunakan untuk menampilkan modal konfirmasi dengan pesan peringatan. Modal ini meminta konfirmasi pengguna sebelum melakukan tindakan yang tidak dapat dibatalkan.

22. digunakan untuk mengimpor komponen

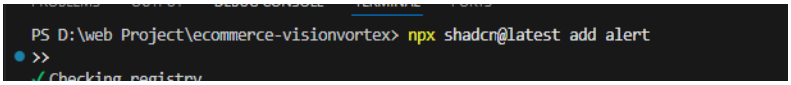


23.

digunakan untuk mendeskripsikan sebuah link atau item navigasi di dalam aplikasi.

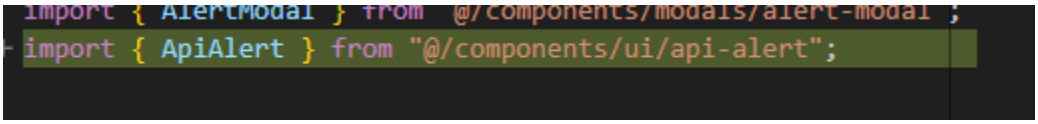


24.menginstal add alert



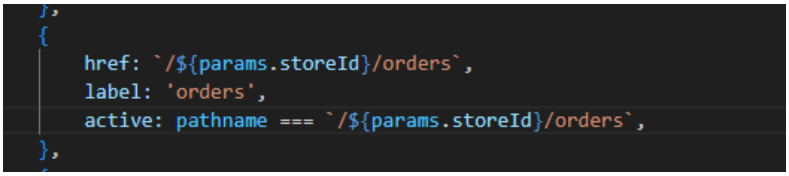
25 menampilkan pesan alert dengan beberapa properti dan styling yang disesuaikan berdasarkan variannya

26. mengimpor komponen **`ApiAlert`** yang kemungkinan besar digunakan untuk menampilkan pemberitahuan atau alert terkait interaksi dengan API di aplikasi.



27.membuat model order dan model orderItem

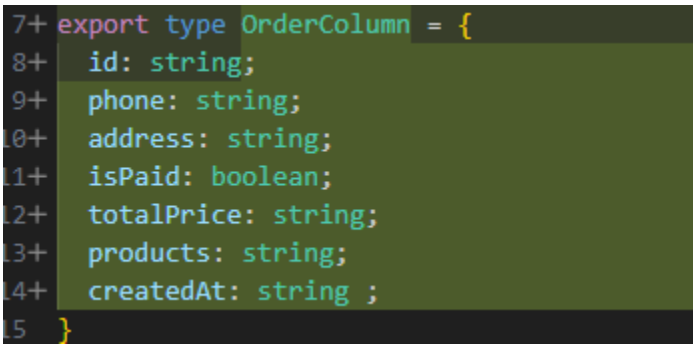
28.membuat orders di tampilan



29. mengambil data dari database menggunakan Prisma ORM

```
import { format, formatDate } from "date-fns"

import db from "@/lib/db";
import { BillboardClient } from "./components/client";
import { BillboardColumn } from "./components/columns";

const BillboardsPage = async ({
    params
}: {
    params: {storeId: string}
}) => {
    const billboards= await db.billboard.findMany({
        where: {
            storeId: params.storeId
        },
        orderBy: {
            createdAt: 'desc'
        }
    });

    const formattedBillboards: BillboardColumn[] = billboards.
        id: item.id,
        label: item.label,
        createdAt: formatDate(item.createdAt,"MMMM do, yyyy" )
    }));

    return (
        <div className="flex-col">
            <div className="flex-1 space-y-4 p-8 pt-6">
                <BillboardClient data={formattedBillboards}/>
            </div>
        </div>
```

30.membuat ordercolumn



```
export type OrderColumn = {
  id: string;
  phone: string;
  address: string;
  isPaid: boolean;
  totalPrice: string;
  products: string;
  createdAt: string ;
}
```