Nama : Rifki Abdullah

NPM : 22312080

DOKUMENTASI PENGERJAAN

Membuat Komponen Form

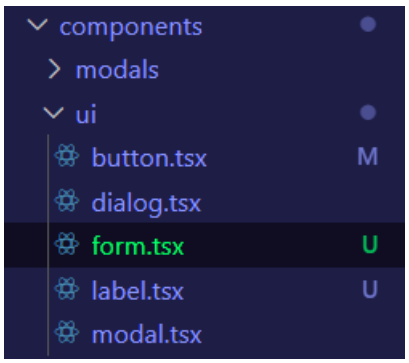
Tanggal : 07 Januari 2025

Waktu : 18.54

- Menginstall komponen **form** dengan perintah **npx shadcn@latest add form** dari Pustaka **snadcn/ui** kedalam proajek, dengan dokumentasi terminal berikut ini :

```
PS D:\ecommerce-visionvortex> npx shadcn@latest add form
✓ Checking registry.
  Installing dependencies.

It looks like you are using React 19.
Some packages may fail to install due to peer dependency issues in npm (see https://ui.shadcn.com/react-19).

√ How would you like to proceed? » Use --legacy-peer-deps
✓ Installing dependencies.
√ The file button.tsx already exists. Would you like to overwrite? ... yes
✓ Created 2 files:
  - components\ui\form.tsx
  - components\ui\label.tsx
i Updated 1 file:
  - components\ui\button.tsx

npm notice
npm notice New major version of npm available! 10.9.0 -> 11.0.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.0.0
npm notice To update run: npm install -g npm@11.0.0
npm notice
PS D:\ecommerce-visionvortex>
```

- Berikut ini hasil yang sudah terinstall dengan menghasilkan file baru berupa **form.tsx dan label.tsx** pada pada folder **ui** didalam **components :**



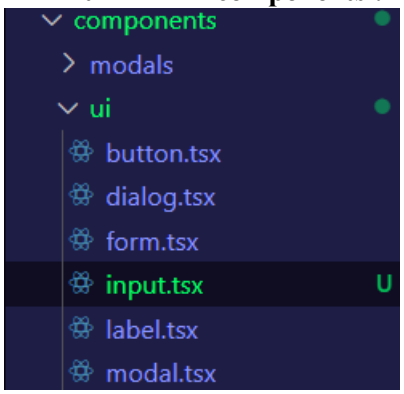
Tanggal : 07 Januari 2025

Waktu : 19.21

- Menambahkan komponen **input** dengan perintah **npx shadcn@latest add input** dari Pustaka **snadcn/ui** kedalam proajek, dengan dokumentasi terminal berikut ini :

```
PS D:\ecommerce-visionvortex> npx shadcn@latest add input
✓ Checking registry.
✓ Created 1 file:
  - components\ui\input.tsx
```

- Berikut ini hasil yang sudah terinstall dengan menghasilkan file baru berupa **input.tsx** pada pada folder **ui** didalam **components :**



Tanggal : 08 Januari 2025

Waktu : 00.02

Melanjutkan membuat form komponen

Pada file **store-modal.tsx**

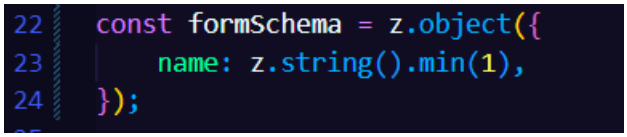
-
```
import * as z from "zod";
import { useForm } from "react-hook-form";
import { zodResolver } from "@hookform/resolvers/zod";
```

  Menambahkan import seperti diatas seperrti zod untuk membuat dan memvalidasi skema data, lalu import useForm yaitu hook dari Pustaka react hook Form, lalu menambahkan import zodResolver supaya zod dengan React Hook Form terintegrasikan.

- 

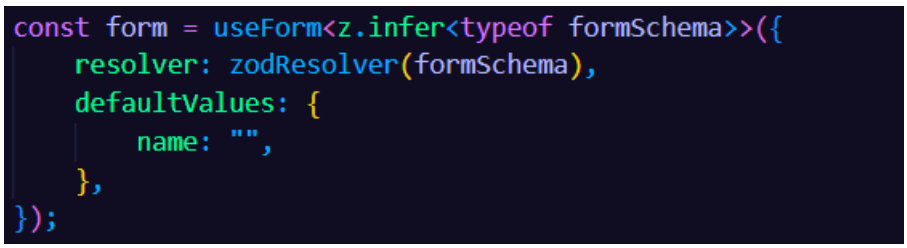
```
import {
    Form,
    FormControl,
    FormField,
    FormItem,
    FormLabel,
    FormMessage
} from "@/components/ui/form";
import { Input } from "@/components/ui/input";
import { Button } from "@/components/ui/button";
```

Menambahkan import berbagai komponen UI terkait (Form, FormControl, FormField, FormItem, FormLabel, FormMessage). Menginport komponen (Input), dan menginport komponen tombol (Button).

- 

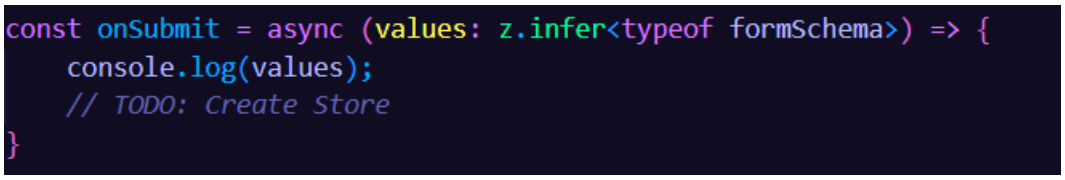
```
const formSchema = z.object({
    name: z.string().min(1),
});
```

Menambahkan const formSchema untuk mendefinisikan skema validasi formulir menggunakan Zod dan `name` adalah field teks wajib dengan minimal 1 karakter

- 

```
const form = useForm<z.infer<typeof formSchema>>({
    resolver: zodResolver(formSchema),
    defaultValues: {
        name: "",
    },
});
```

Menambahkan const baru untuk menginisialisasi hook `useForm` dengan resolver Zod untuk validasi skema dengan `defaultValues` menetapkan nilai default untuk input formulir, di sini `name` diinisialisasi sebagai string kosong.

- 

```
const onSubmit = async (values: z.infer<typeof formSchema>) => {
    console.log(values);
    // TODO: Create Store
}
```

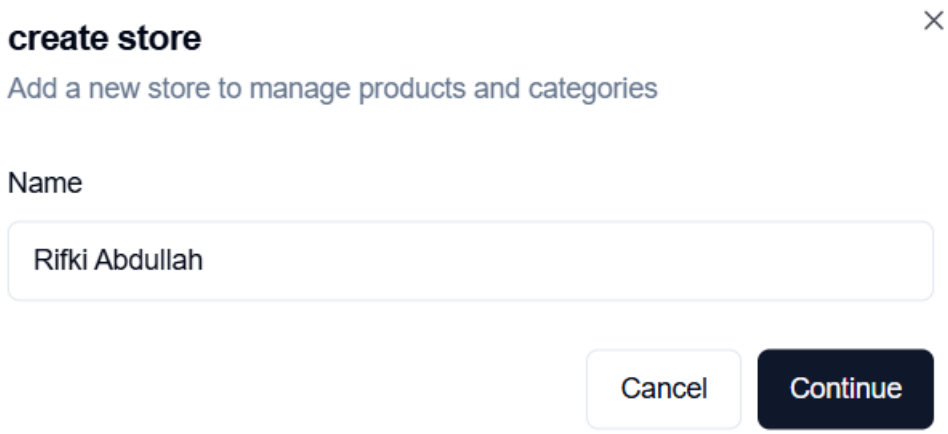
Menambahkan const baru untuk mendefinisikan fungsi `onSubmit`, yang akan dipanggil saat formulir dikirimkan.

```
<Modal
title="create store"
description="Add a new store to manage products and categories"
isOpen={storeModal.isOpen}
onClose={storeModal.onClose}
>
    <div>
        <div className="space-y-4 py-2 pb-4">
            <Form {...form}>
                <form onSubmit={form.handleSubmit(onSubmit)}>
                    <FormField
                        control={form.control}
                        name="name"
                        render={({ field }) => (
                            <FormItem>
                                <FormLabel>Name</FormLabel>
                                <FormControl>
                                    <Input placeholder="E-Commerce" {...field}/>
                                </FormControl>
                                <FormMessage />
                            </FormItem>
                        )}
                    />
                    <div className="pt-6 space-x-2 flex items-center justify-end w-full">
                        <Button
                            variant="outline"
                            onClick={storeModal.onClose}>Cancel
                        </Button>
                        <Button type="submit">Continue</Button>
                    </div>
                </form>
            </Form>
        </div>
    </div>
</Modal>
```

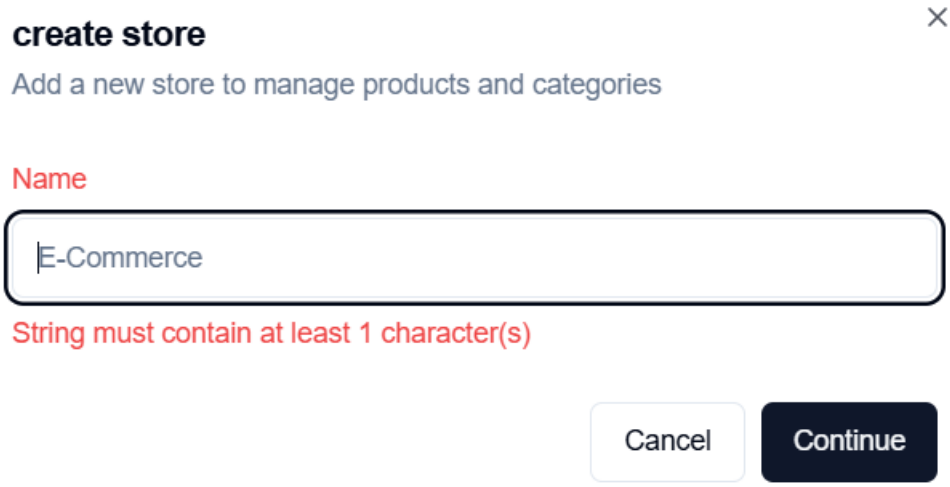
Menambahkan isi didalam Form yang terdapat :

- Membungkus formulir HTML dengan komponen `Form` untuk integrasi dengan React Hook Form dan `handleSubmit` menangani validasi dan pemrosesan data sebelum memanggil `onSubmit`.
- Komponen `FormField` agar terhubung dengan input form
- Menambahkan `render` supaya mengonfigurasi elemen input dan menampilkan pesan error jika ada.
- Dan menambahkan Input dengan isi placeholder "E-Commerce" dan dihubungkan ke field "name".
- Menambahkan tombol Button pada form yang terdiri dari Cancel dan Continue

- **Hasil form yang ditambahkan diatas**



- **Jika table tidak terisikan**

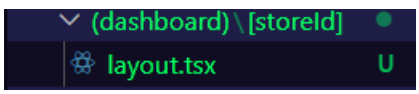


Tanggal : 09 Januari 2025

Waktu : 14.34 s/d …

Membuat HomePage pada Admin

- Membuat folder (dashboard) dan lalu didalamnya terdapat folder [storeId] lalu membuat file layout.tsx



- Mengerjakan isi layout.tsx

```
1   import db from "@/lib/db";
2   import { auth } from "@clerk/nextjs/server";
3   import { redirect } from "next/navigation";
4   import React from "react";
```

Mengimpor modul db, mengimpor fungsi auth dari clerk, mengimport fungsi redirect dari next.js, mengimport modul react.

```
export default async function DashboardLayout({
    children,
    params,
}: {
    children: React.ReactNode;
    params: {storeId: string}
}) {
```

Membuat komponen layout untuk dashboard. Menambahkan komponen children, params. Untuk memastikan hanya pengguna yang login.

```
}) {
    const {userId} = await auth();
    if (!userId) {
        redirect("sign-in")
    }
    const store = await db.store.findFirst({
        where: {
            id: params.storeId,
            userId
        }
    })
```

Mendapatkan data autentikasi

```
}) {
    const {userId} = await auth();
    if (!userId) {
        redirect("sign-in")
    }
    const store = await db.store.findFirst({
        where: {
            id: params.storeId,
            userId
        }
    })
```

Membuat supaya pengguna jika blm login diarahkan ke halaman sign-in

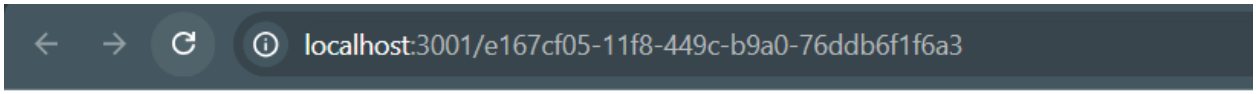
```
}) {
    const {userId} = await auth();
    if (!userId) {
        redirect("sign-in")
    }
    const store = await db.store.findFirst({
        where: {
            id: params.storeId,
            userId
        }
    })
```

Membuat query ke database untuk mencari data store pertama yang memenuhi kriteria.

```
if (!store) {
    redirect("/")
}
return (
    <>
        <div>This is Navbar</div>
        {children}
    </>
)
```

Melakukan pengecekan jika store tidak ada

- Dokumentasi hasil perubahan dari [storeId]



This is Navbar

This is Dashboard

Dia mendapatkan Id store di Alamat halaman diatas.

- Membuat Folder baru didalam [storeId] dengan nama (routes) dan lalu membuat file dengan nama page.tsx
- Isi page.tsx

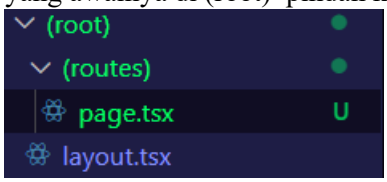
```tsx
const DasboardPage = () => {
    return (
        <div>
            This is Dashboard
        </div>
    );
}

export default DasboardPage;
```

- Membuat file baru didalam folder (root) yaitu file layout.tsx

- Membuat isi nya :

Pada isinya hamper sama dengan file layout.tsx pada (dashboard)/[storeId] hanya saja menambahkan isi yang ditandakan untuk memastikan store yang diminta oleh pengguna benar-benar ada dan dapat diakses.

- Membuat folder baru didalam folder (root) dengan nama (routes) dan memindahkan file page.tsx yang awalnya di (root) pindah ke (routes).



- Melakukan **npx prisma migrate reset** untuk mereset database ke keadaan awal berdasarkan migrasi prisma.
- 

Pada bash tersebut melakukan perintah tersebut untuk menghasilkan client prisma yang akan digunakan.

- 

Push skema Prisma langsung ke database

```
const onSubmit = async (values: z.infer<typeof formSchema>) => {
    try {
        setLoading(true)
        const response = await axios.post('/api/stores', values)
        console.log(response.data);
        toast.success("Berhasil Membuat Toko");
        window.location.assign(`/${response.data.id}`)
    } catch (error) {
        toast.error("Gagal Membuat Toko")
    } finally{
        setLoading(false)
    }

}
```

- Menambahkan kode untuk mengarahkan pengguna ke URL tertentu berdasarkan data yang diterima dari respons server.
- Memodifikasi pada file page.tsx pada folder (dashboard)/[storeId]/(routes)

```
interface DashboardPageProps {
    params: {storeId: string}
}
```

Menambahkan interface tersebut untuk mendeskripsikan tipe properti yang diterima oleh komponen.

```
const DashboardPage = async ({params} : DashboardPageProps) => {
```

Menambahkan properti params (dari tipe DashboardPageProps)

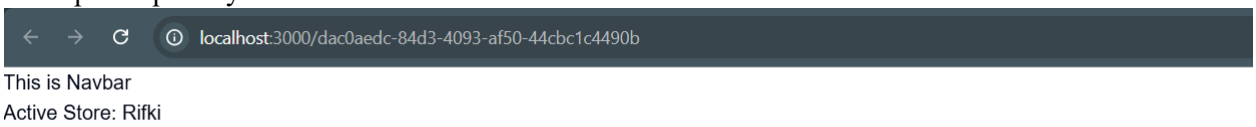
```
    const store = await db.store.findFirst({
        where: {
            id: params.storeId
        }
    })
```

Menambahkan const baru untuk mencari data di tabel store. Query ini mencari satu entri dalam tabel store yang memiliki id sesuai dengan nilai params.storeId.

```
return (
    <div>
        Active Store: {store?.name}
    </div>
);
```

Menambahkan untuk menampilkan keterangan berdasarkan nama storenya.

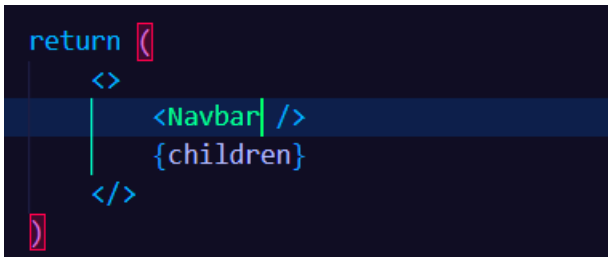
Hasil penampilannya:



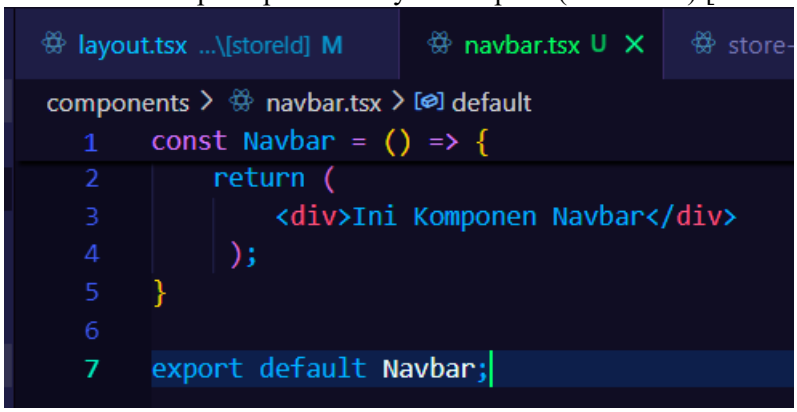
Tanggal : 10 Januari 2025

Waktu : 07.23 s/d …

Membuat Navbar pada Admin

- 

  Merubah isi tampilan pada file layout.tsx pada (dashboard)/[storeId]

- 

  Lalu membuat file baru pada folder components dengan nama file navbar.tsx dan serta memberikan isinya untuk dipanggil Navbar pada layout.tsx sebelumnya.

- ```
  layout.tsx ...\[storeId]   navbar.tsx M ×   store-modal.tsx   page.tsx
  components > navbar.tsx > default
  ```
  ```tsx
  import { UserButton } from "@clerk/nextjs";

  const Navbar = () => {
      return (
          <div className="border-b">
              <div className="flex h-16 items-center px-4">
                  <div>Store Switcher</div>
                  <div>main nav</div>
                  <div className="ml-auto flex items-center space-x-4">
                      <UserButton afterSignOutUrl="/"/>
                  </div>
              </div>
          </div>
      );
  }

  export default Navbar;
  ```

  Membuat isi navbar berupa bordernya, mengatur ukurannya, membuat userbutton-nya.

- ```tsx
  <MainNav />
  ```

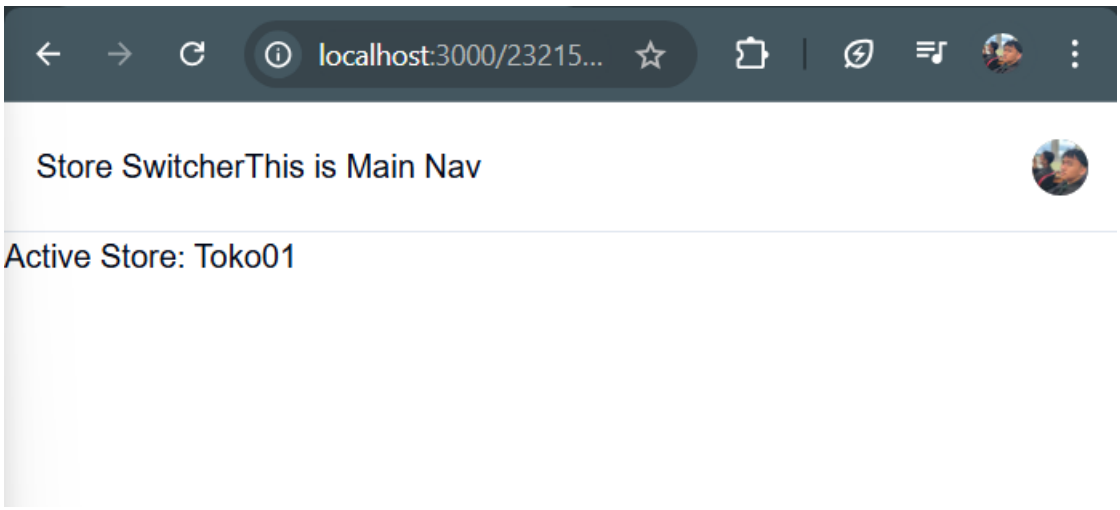
  Mengubah main nav seblumnya menjadi seperti itu untuk memanggil hasil dari main-nav.tsx

- ```
  layout.tsx ...\[storeId]   navbar.tsx M   main-nav.tsx U ×   sto
  components > main-nav.tsx > MainNav
  ```
  ```tsx
  'use client'

  export function MainNav({
      className,

  }: React.HTMLAttributes<HTMLElement>) {
      return (
          <nav>
              This is Main Nav
          </nav>
      )
  }
  ```

  Mendefinisikan MainNav untuk dipanggil dan di render.
  ```

- 

Hasilnya.

```
const pathname = usePathname();
const params = useParams();

const routes = [
    {
        href: `/${params.storeId}/settings`,
        label: 'settings',
        active: pathname === `/${params.storeId}/settings`,
    }
]
```

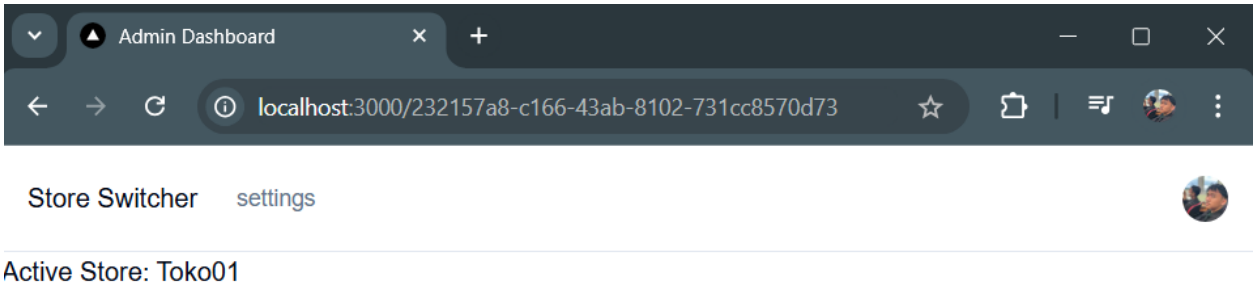
Menambahkan code pada main-nav.tsx untuk mendefinisikan const pathname dan const params. Dan membuat const routes.

```
{routes.map((route) => (
    <Link
    key={route.href}
    href={route.href}
    className={cn(
        "text-sm font-medium transition-colors hover:text-primary",
        route.active ? "text-black dark:text-white" : "text-muted-foreground"
    )}
    >
        {route.label}
    </Link>
))}
```

Merender navigasi dinamis menggunakan data dari array routes pada main-nav.tsx

```
<MainNav className="mx-6"/>
```

Dan memberikan jarak nya.



-

Dan ini hasilnya.

-
```
PS D:\Universitas Teknokrat Indonesia\Semester 5\WEB 2\ecommerce-visionvortex> npx shadcn@latest add popover
```
Menginstall popover

-
```
PS D:\Universitas Teknokrat Indonesia\Semester 5\WEB 2\ecommerce-visionvortex>
 npx shadcn@latest add command
```
Menginstall command

-
```
import db from "@/lib/db";

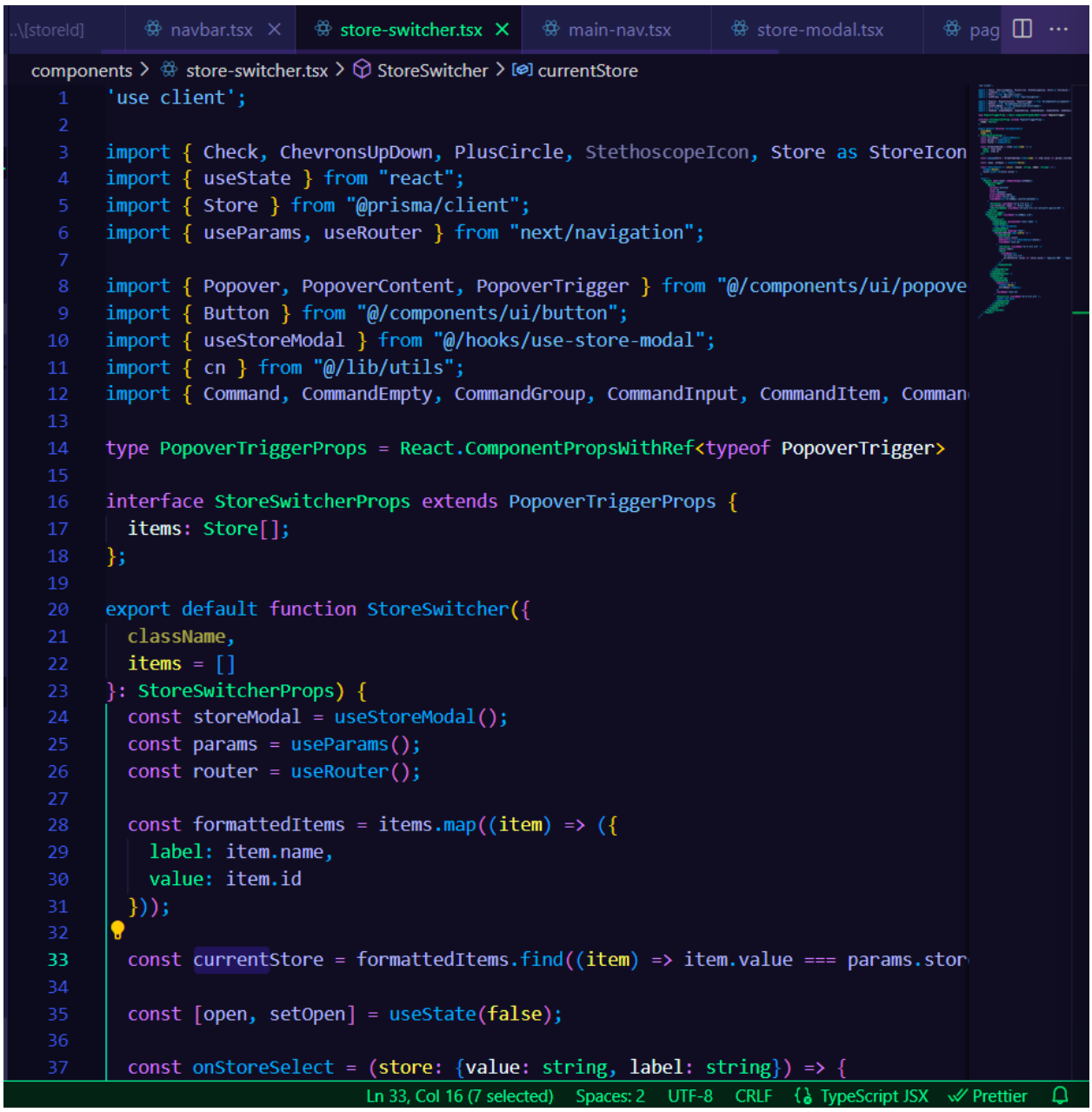
const Navbar = async () => {
    const {userId} = await auth();

    if (!userId) {
        redirect("/sign-in");
    }

    const stores = await db.store.findMany({
        where: {
            userId,
        }
    })

    return (
        <div className="border-b">
            <div className="flex h-16 items-center px-4">
                <StoreSwitcher items={stores} />
```

Menambahkan const baru untuk mengaitkan StoreSwitcher dan menambahkan const baru

- 

  Mendefinisikan StoreSwitcher didalamnya membuat button took berupa searcing took, nama toko, tambah toko.

Tanggal : 12 Januari 2025

Waktu : 14.44 s/d …

-

settings-form.tsx M | layout.tsx | use-origin.tsx U

hooks > use-origin.tsx > useOrigin

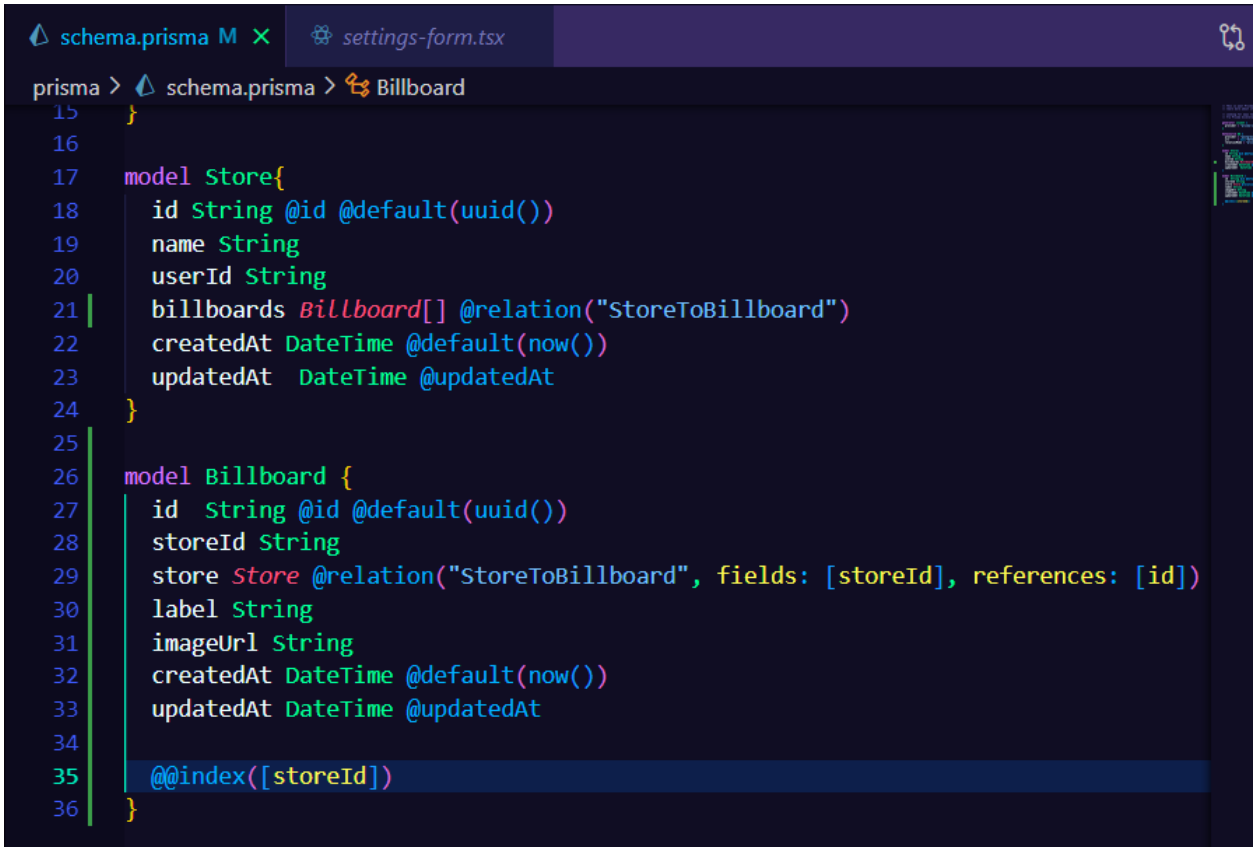
```tsx
import { useState, useEffect } from "react";

export const useOrigin = () => {
    const [mounted, setMounted] = useState(false);
    const origin = typeof window !== "undefined" && window.location.origin ?  window.location.origin : '';

    useEffect(() => {
        setMounted(true);
    }, []);

    if (!mounted) {
        return '';
    }

    return origin;
}
```

Menambahkan custom React hook bernama useOrigin. Untuk tampilan settings pada admin.

-

```tsx
import { useOrigin } from "@/hooks/use-origin";



interface SettingsFormProps {
    initialData: Store;
}

const formSchema = z. object ({
    name: z.string().min(1)
});
type SettingsFormValues = z. infer<typeof formSchema>;

export const SettingsForm: React.FC<SettingsFormProps> =
    initialData
}) => {
const params = useParams();
const router = useRouter();
const origin = useOrigin();
```

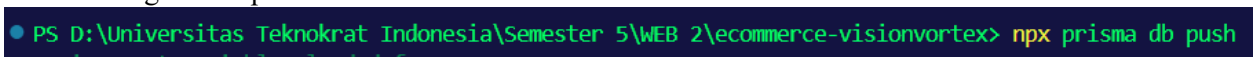
Menambahkan const origin dan mengimport dari useOrigin yang sebelumnya dibuat.

- 

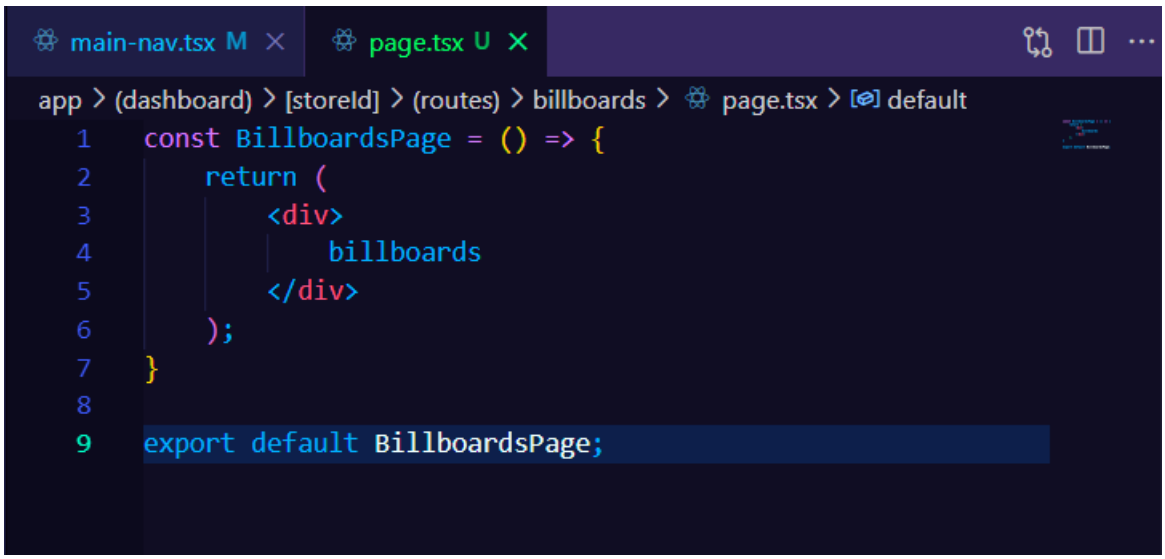
Membuat model baru Billboard dan mendefinisikannya

- 

Melakukan generate prisma

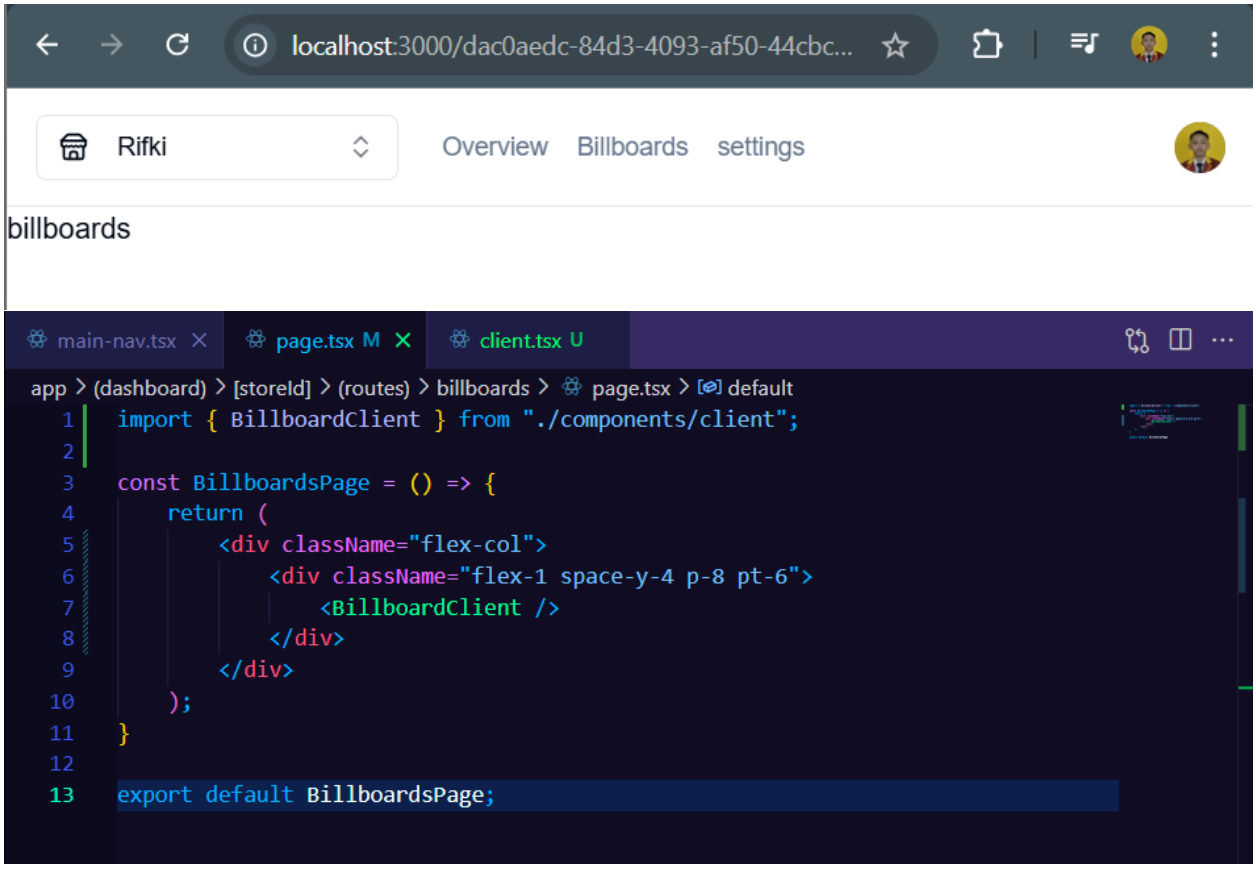
- 

Melakukan push prisma

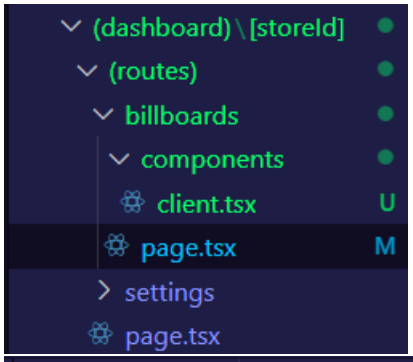
- 

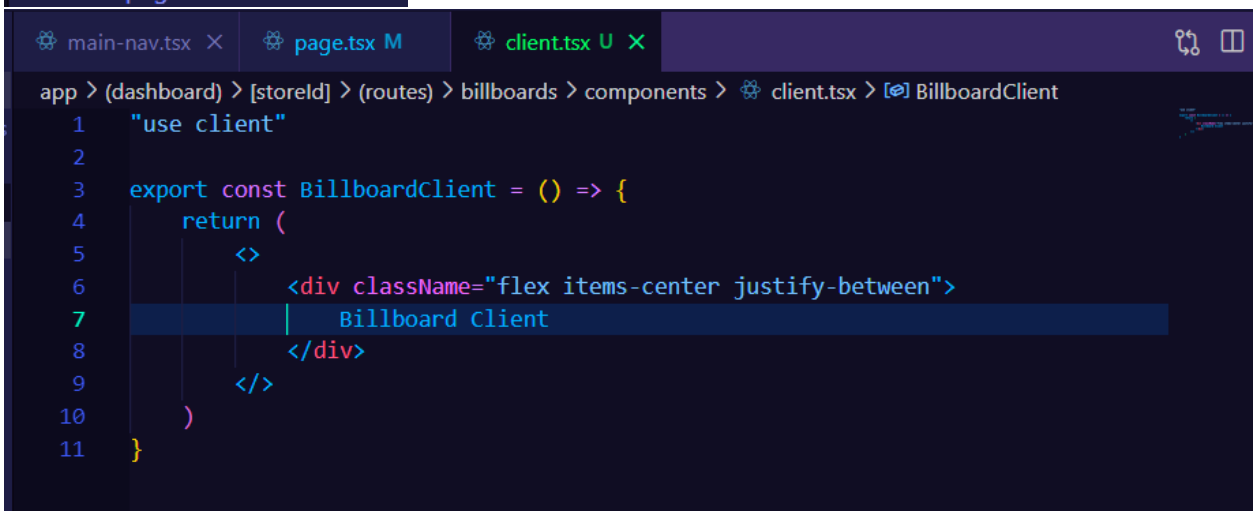
Membuat Lembah papan iklan/Billboards.

hasil lembar

- 

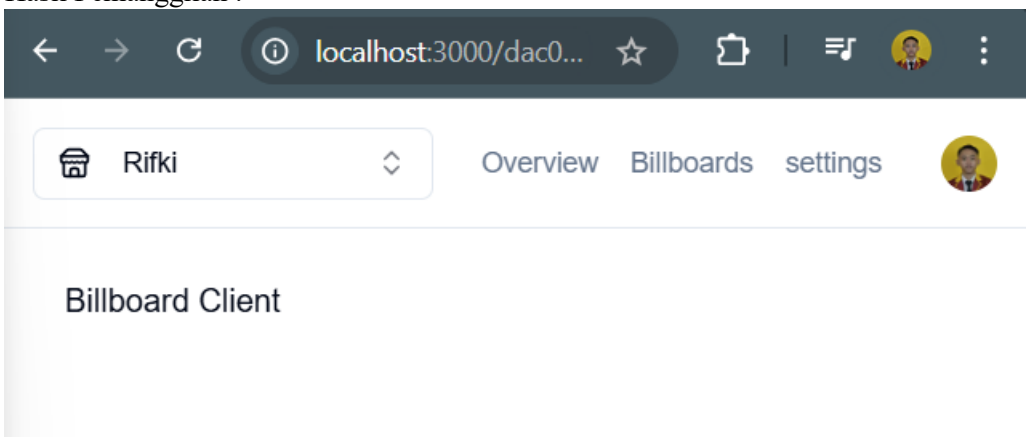
Memperbarui isi pada halaman page billboards dan lalu membuat folder baru didalam folder billboards Bernama folder components dan berisi file client.tsx



- 

Pada file client berisikan isi untuk dipanggil pada BillboardClient yang dibuat pada file BillboardsPage sebelumnya.

Hasil Pemanggilan :

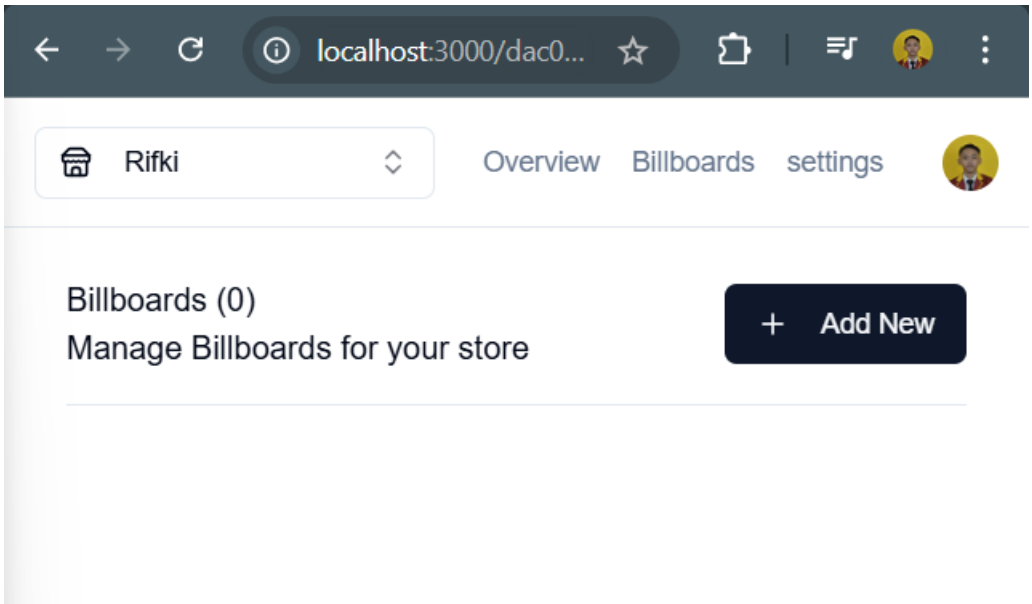


-

```tsx
"use client"

import { Plus } from "lucide-react"

import { Button } from "@/components/ui/button"
import { Heading } from "@/components/ui/heading"
import { Separator } from "@/components/ui/separator"

export const BillboardClient = () => {
    return (
        <>
            <div className="flex items-center justify-between">
                <Heading
                    title="Billboards (0)"
                    description="Manage Billboards for your store"
                />
                <Button>
                    <Plus className="mr-2 h-4 w-4"/>
                    Add New
                </Button>
            </div>
            <Separator />
        </>
    )
}
```

Menambahkan heading dan button tambah pada tampilan.

Hasil Menambahkan tampilan :



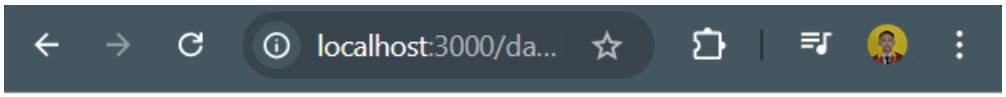
main-nav.tsx | page.tsx | client.tsx M

app > (dashboard) > [storeId] > (routes) > billboards > components > client.tsx > BillboardClient

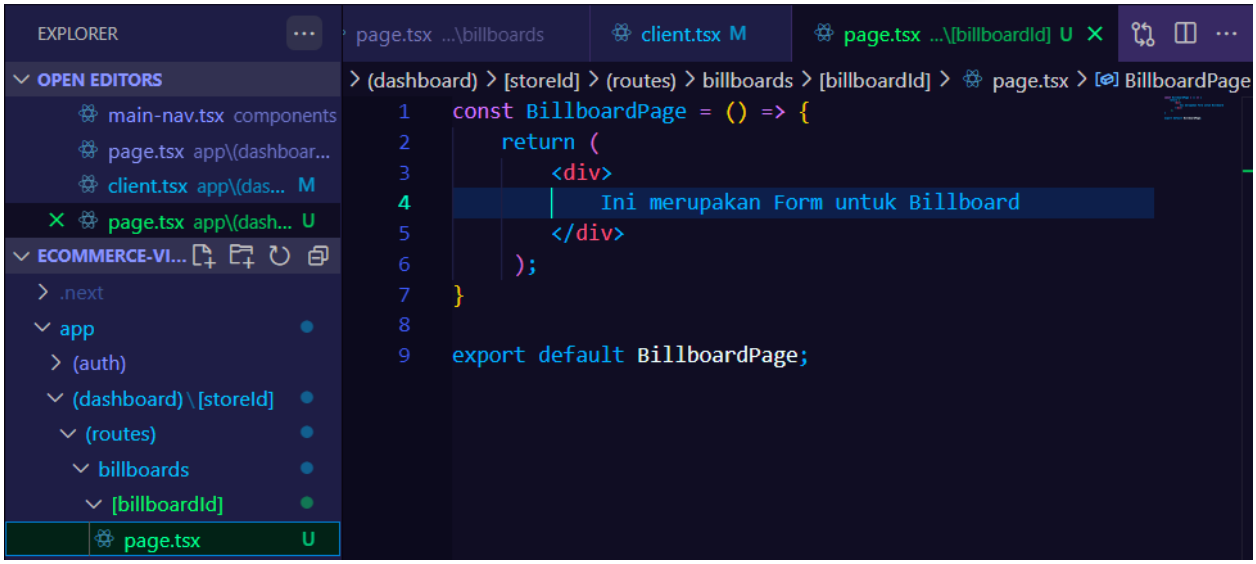
```tsx
"use client"

import { Plus } from "lucide-react"
import { useParams, useRouter } from "next/navigation"

import { Button } from "@/components/ui/button"
import { Heading } from "@/components/ui/heading"
import { Separator } from "@/components/ui/separator"

export const BillboardClient = () => {
    const router = useRouter();
    const params = useParams();


    return (
        <>
            <div className="flex items-center justify-between">
                <Heading
                    title="Billboards (0)"
                    description="Kelola Billboard untuk toko Anda"
                />
                <Button onClick={() => router.push(`/${params.storeId}/billboards/new`)}>
                    <Plus className="mr-2 h-4 w-4"/>
                    Add New
                </Button>
            </div>
            <Separator />
        </>
    )
}
```

- Membuat onClick pada button tambah billboards

  hasil :



404 | This page could not be found.

- 

Membuat folder baru Bernama [billboardId] didalam folder billboards dan membuat lembar baru Bernama page.tsx untuk membuat lembar form

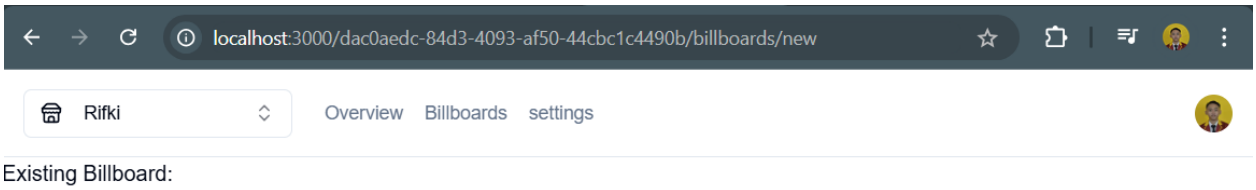
page.tsx ...\billboards | client.tsx | page.tsx ...\[billboardId] M

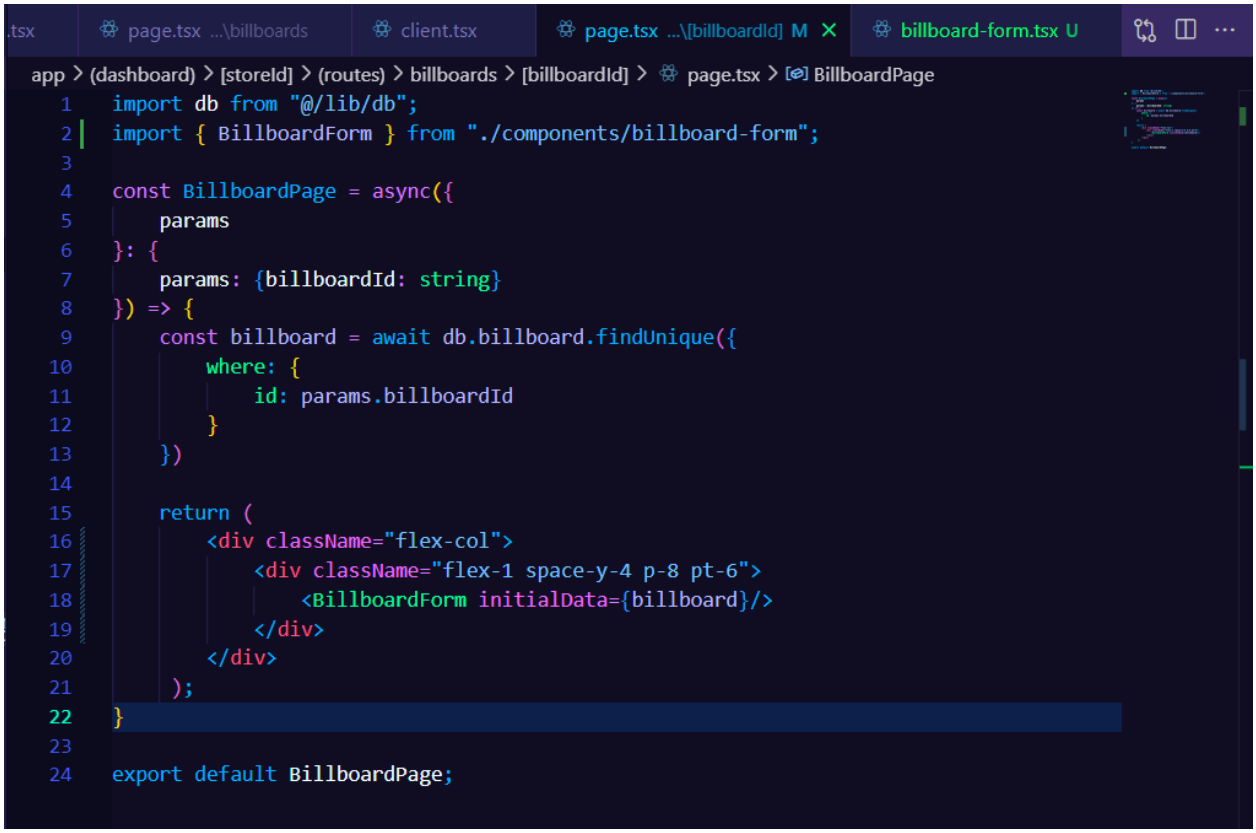
(dashboard) > [storeId] > (routes) > billboards > [billboardId] > page.tsx > BillboardPage

```tsx
import db from "@/lib/db";

const BillboardPage = async({
    params
}: {
    params: {billboardId: string}
}) => {
    const billboard = await db.billboard.findUnique({
        where: {
            id: params.billboardId
        }
    })

    return (
        <div>
            Existing Billboard: {billboard?.label}
        </div>
     );
}

export default BillboardPage;
```

Membeikan id halaman berdasarkan label pada billboard

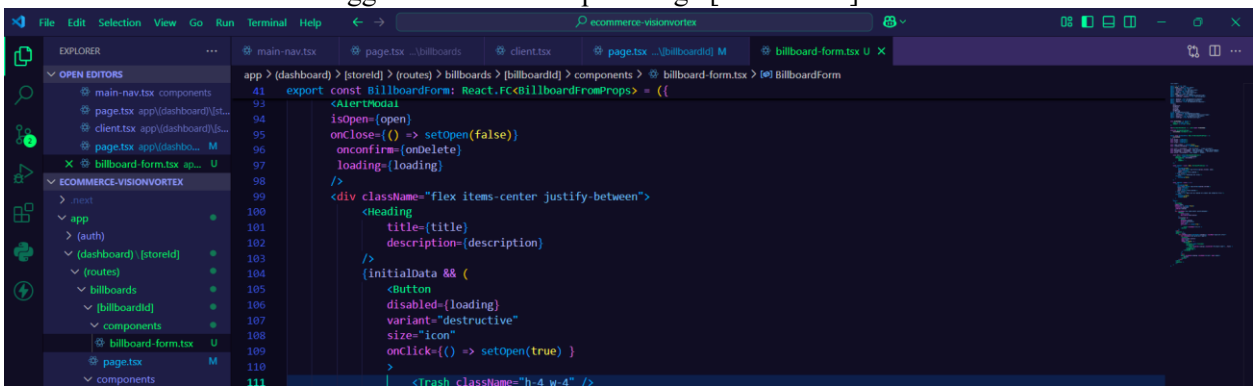
hasil:



- 

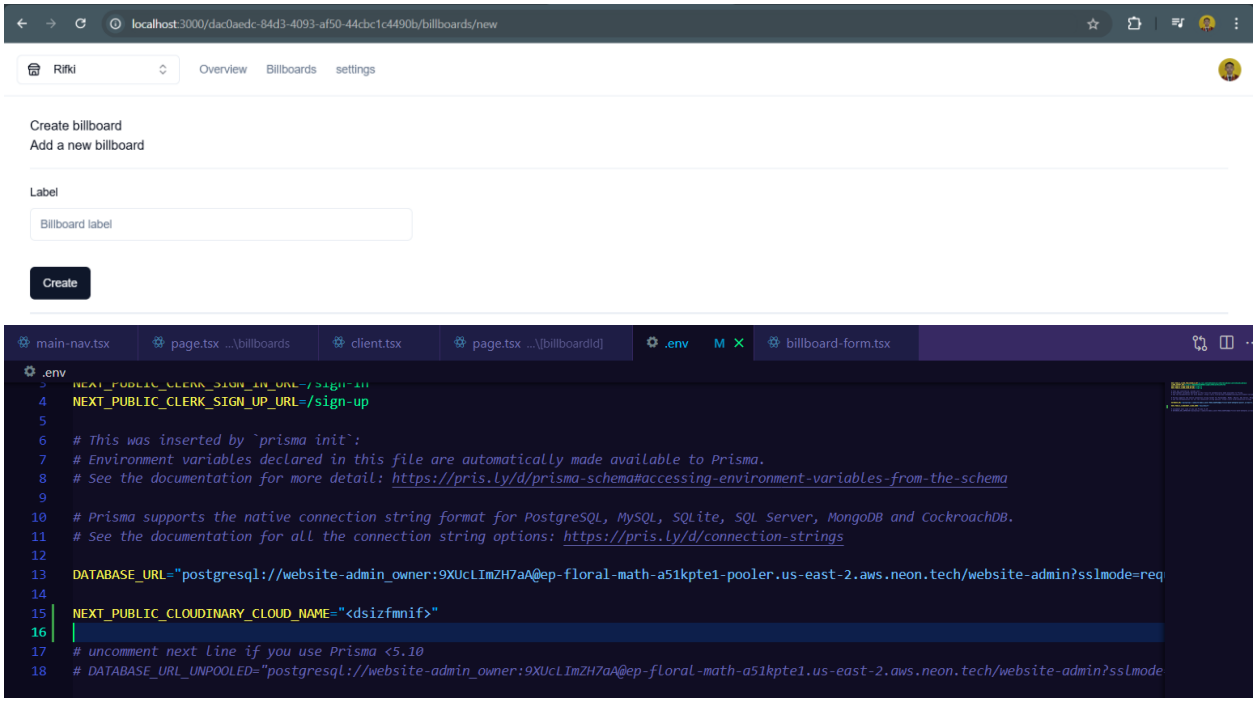
```tsx
import db from "@/lib/db";
import { BillboardForm } from "./components/billboard-form";

const BillboardPage = async({
    params
}: {
    params: {billboardId: string}
}) => {
    const billboard = await db.billboard.findUnique({
        where: {
            id: params.billboardId
        }
    })

    return (
        <div className="flex-col">
            <div className="flex-1 space-y-4 p-8 pt-6">
                <BillboardForm initialData={billboard}/>
            </div>
        </div>
     );
}

export default BillboardPage;
```

Menambahkan untuk memanggil BillboardForm pada Page/[billboardId]

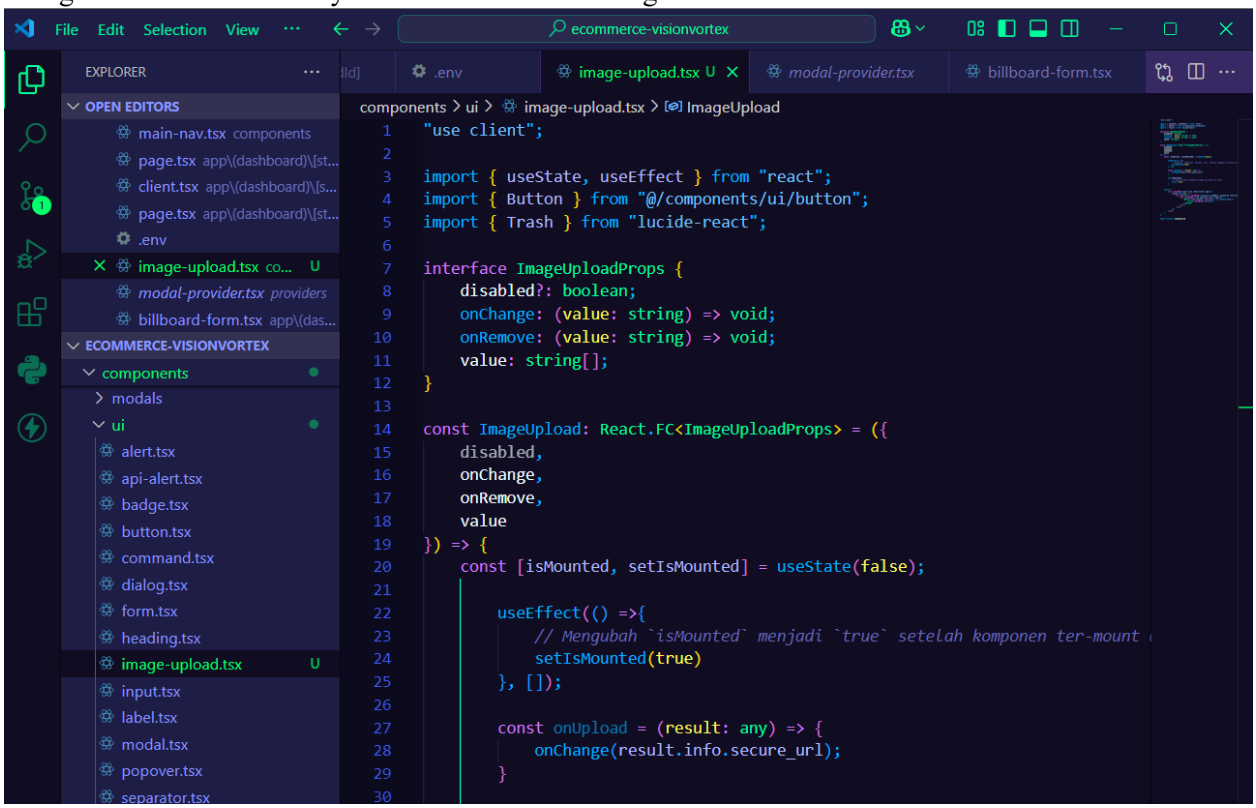
- 

Membuat component baru dari folder [billboardId] dan lalu mengcopy file settings-form.tsx dan di paste di dalam folder components dan memberikan nama file baru billboard-form.tsx dan modifikasi isi dalam form tersebut.

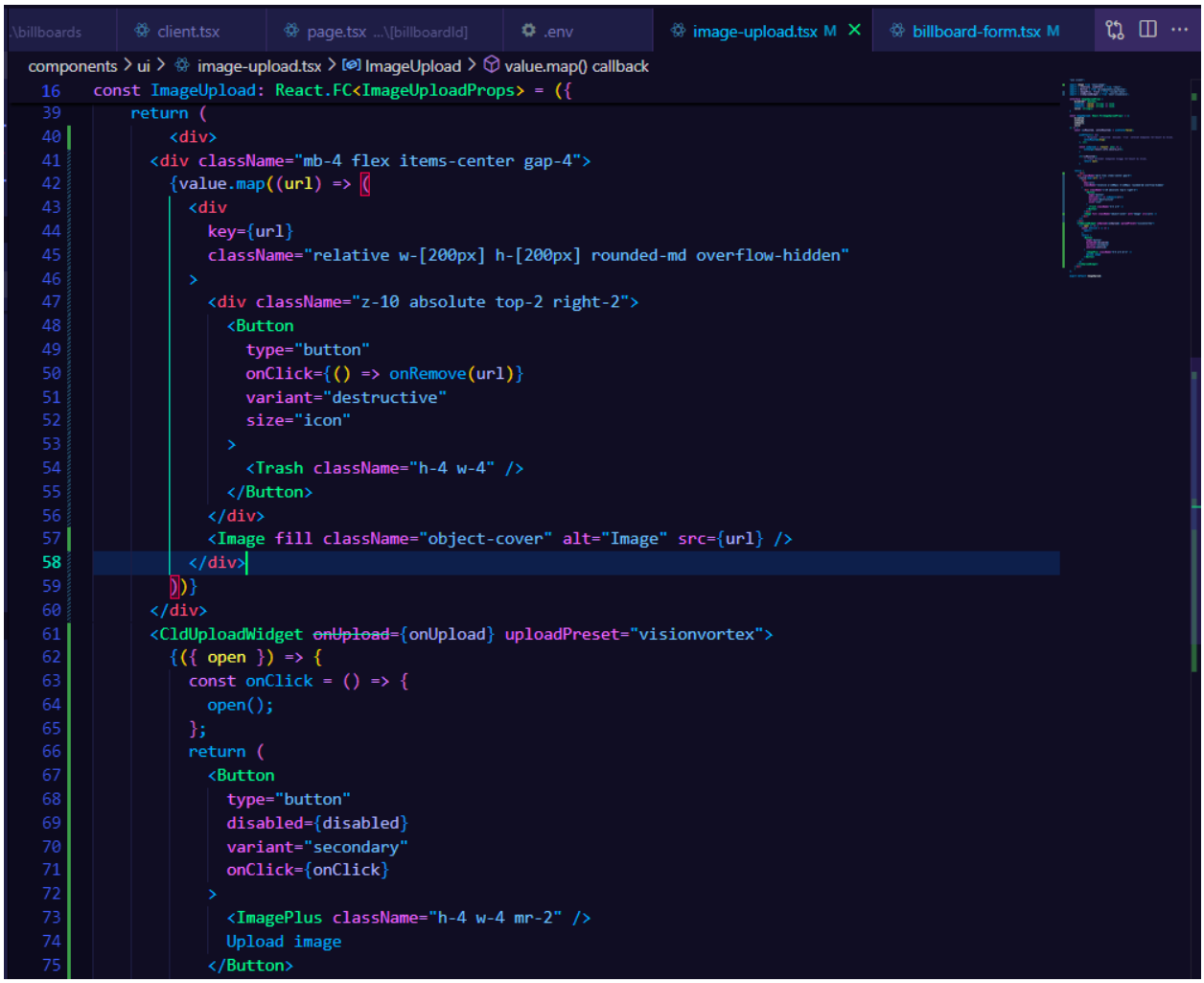
hasil:

- 

Meng install next-cloudinary dan menambahkan konfigurasi di .env

- 

Membuat file baru didalam components/ui/ yang diberi nama image-upload.tsx yang didalamnya kita isikan untuk meng upload image pada billboards.

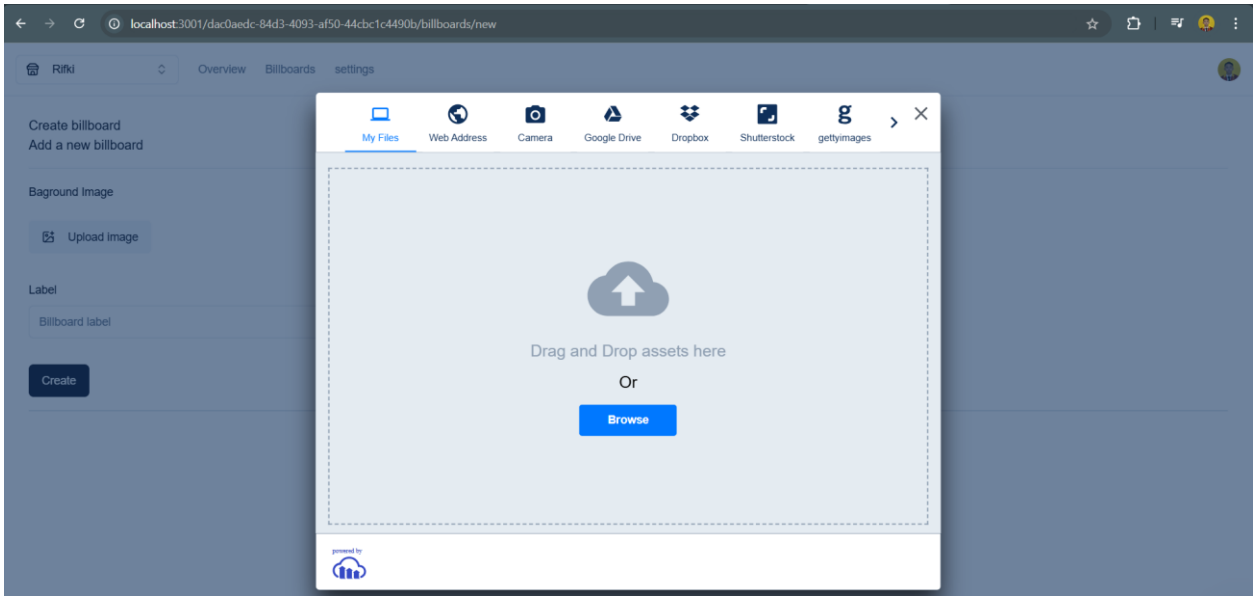
- 

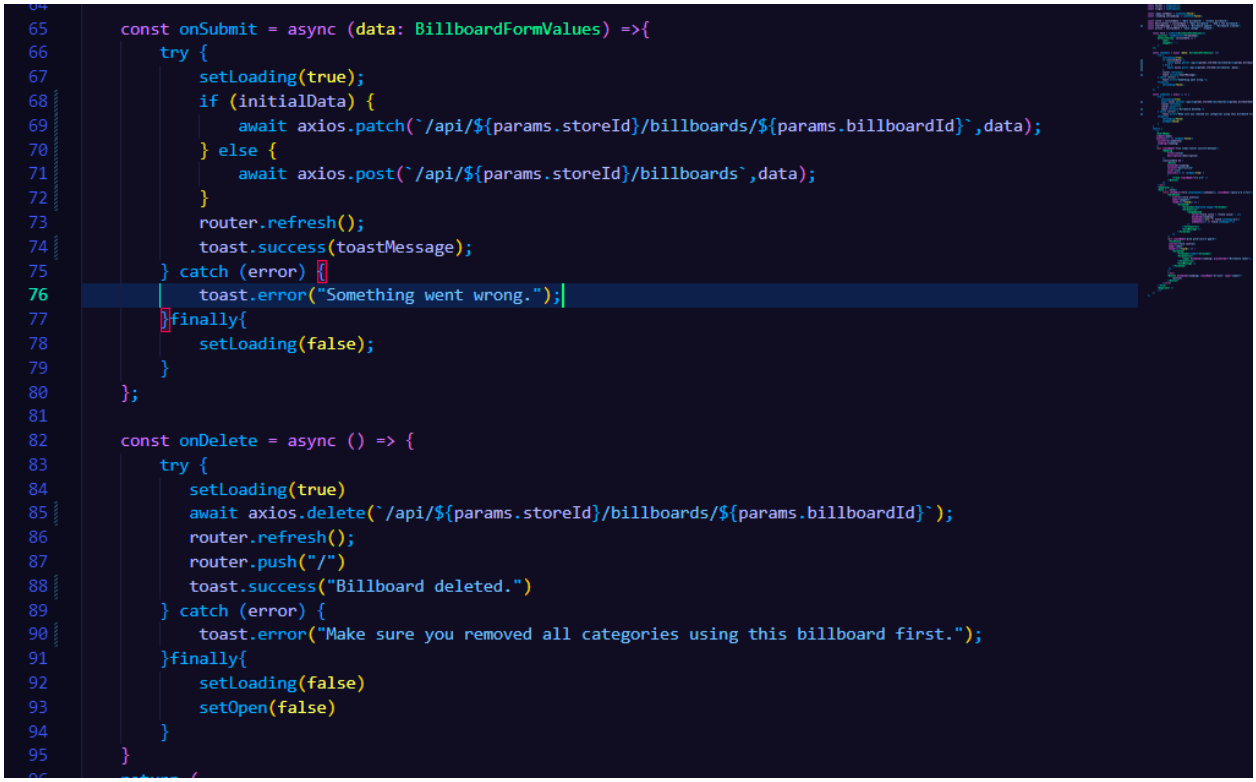
Menampilkan daftar gambar yang dapat dihapus oleh pengguna serta menyediakan tombol untuk mengunggah gambar baru menggunakan widget upload Cloudinary.

```
<FormField
    control={form.control}
    name="imageUrl"
    render={({field}) => (
        <FormItem>
            <FormLabel>Baground Image</FormLabel>
            <FormControl>
                <ImageUpload
                    value={field.value ? [field.value] : []}
                    disabled={loading}
                    onChange={(url) => field.onChange(url)}
                    onRemove={() => field.onChange("")}
                />
            </FormControl>
            <FormMessage />
        </FormItem>
    )}
/>
```

Membuat form untuk mengunggah dan mengatur URL gambar latar (background image)

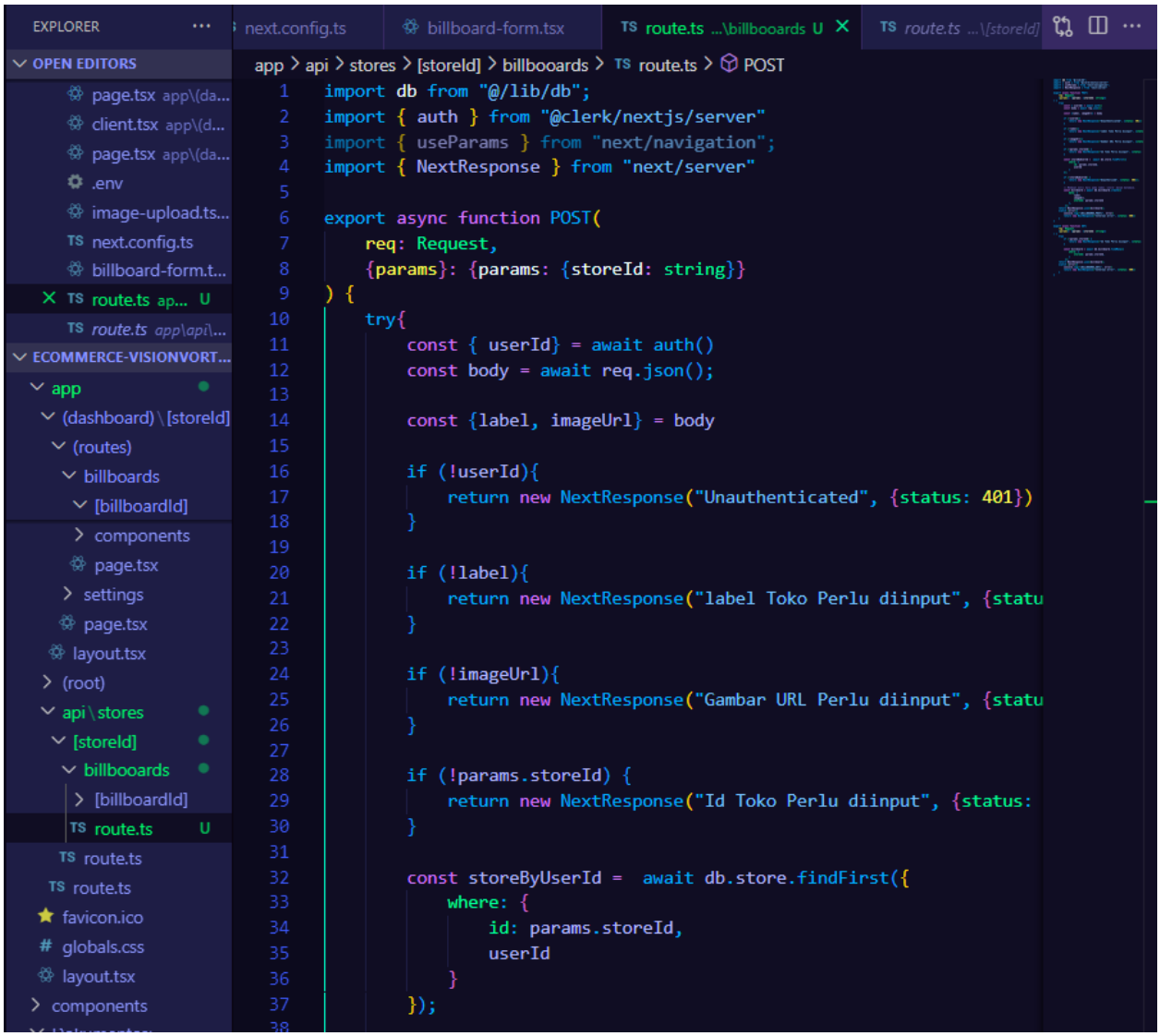
Hasil :



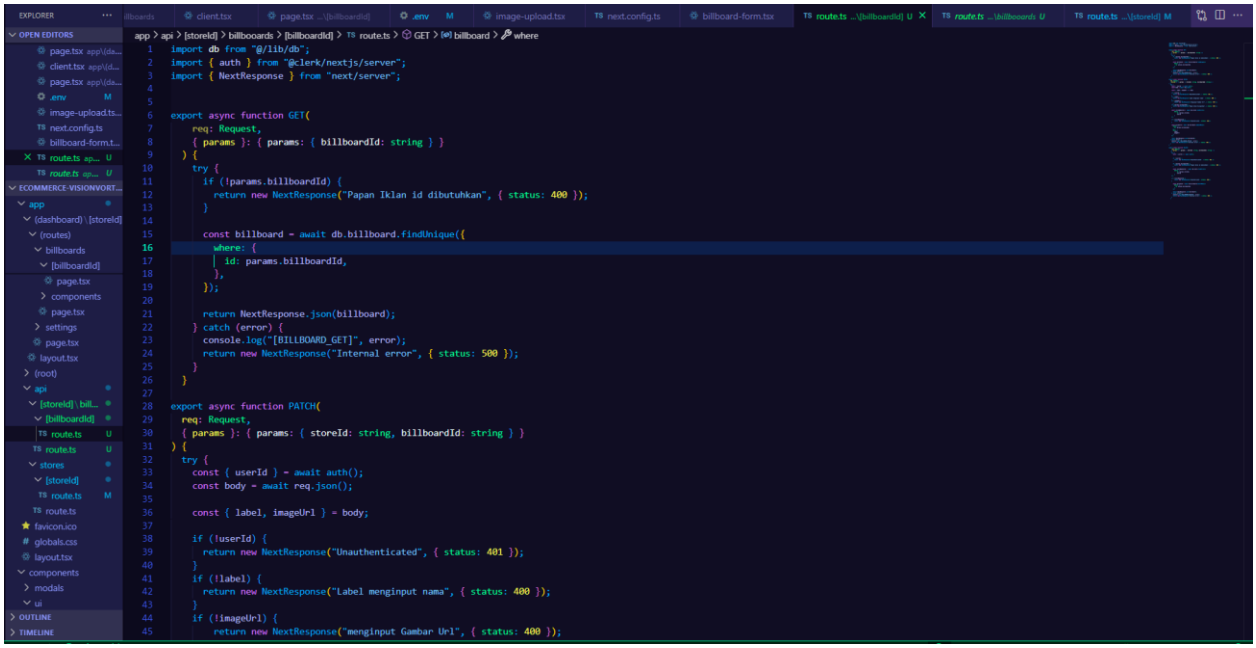
- 

```
const onSubmit = async (data: BillboardFormValues) =>{
    try {
        setLoading(true);
        if (initialData) {
            await axios.patch(`/api/${params.storeId}/billboards/${params.billboardId}`,data);
        } else {
            await axios.post(`/api/${params.storeId}/billboards`,data);
        }
        router.refresh();
        toast.success(toastMessage);
    } catch (error) {
        toast.error("Something went wrong.");
    }finally{
        setLoading(false);
    }
};

const onDelete = async () => {
    try {
       setLoading(true)
       await axios.delete(`/api/${params.storeId}/billboards/${params.billboardId}`);
       router.refresh();
       router.push("/")
       toast.success("Billboard deleted.")
    } catch (error) {
        toast.error("Make sure you removed all categories using this billboard first.");
    }finally{
        setLoading(false)
        setOpen(false)
    }
}
```

  mengupdate untuk file billboard-form.tsx

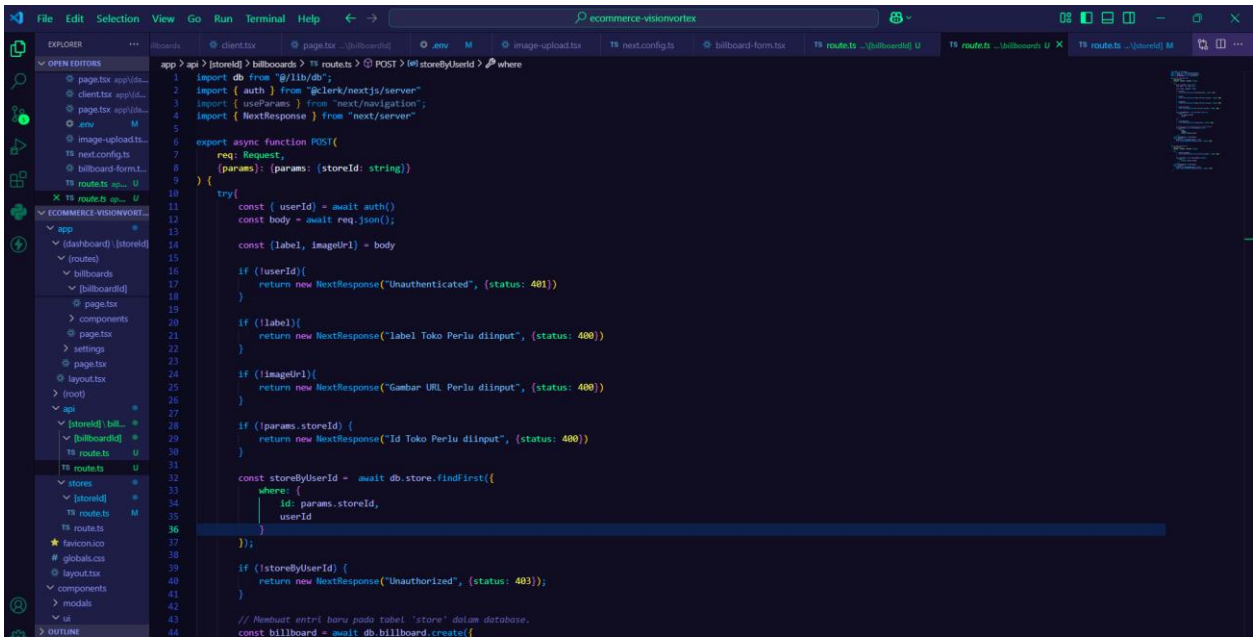
- 

membuat folder baru berupa billboards dan file baru route.ts dan memberikan fungsi billboards

- 

```
import db from "@/lib/db";
import { auth } from "@clerk/nextjs/server";
import { NextResponse } from "next/server";


export async function GET(
    req: Request,
    { params }: { params: { billboardId: string } }
  ) {
    try {
      if (!params.billboardId) {
        return new NextResponse("Papan Iklan id dibutuhkan", { status: 400 });
      }

      const billboard = await db.billboard.findUnique({
        where: {
          id: params.billboardId,
        },
      });

      return NextResponse.json(billboard);
    } catch (error) {
      console.log("[BILLBOARD_GET]", error);
      return new NextResponse("Internal error", { status: 500 });
    }
  }

export async function PATCH(
  req: Request,
  { params }: { params: { storeId: string, billboardId: string } }
) {
  try {
    const { userId } = await auth();
    const body = await req.json();

    const { label, imageUrl } = body;

    if (!userId) {
      return new NextResponse("Unauthenticated", { status: 401 });
    }
    if (!label) {
      return new NextResponse("Label menginput nama", { status: 400 });
    }
    if (!imageUrl) {
        return new NextResponse("menginput Gambar Url", { status: 400 });
```

Membuat fungsi GET, PATCH, DELETE pada billboardId

- 

```
import db from "@/lib/db";
import { auth } from "@clerk/nextjs/server"
import { useParams } from "next/navigation";
import { NextResponse } from "next/server"

export async function POST(
    req: Request,
    {params}: {params: {storeId: string}}
) {
    try{
        const { userId } = await auth()
        const body = await req.json();

        const {label, imageUrl} = body

        if (!userId){
            return new NextResponse("Unauthenticated", {status: 401})
        }

        if (!label){
            return new NextResponse("label Toko Perlu diinput", {status: 400})
        }

        if (!imageUrl){
            return new NextResponse("Gambar URL Perlu diinput", {status: 400})
        }

        if (!params.storeId) {
            return new NextResponse("Id Toko Perlu diinput", {status: 400})
        }

        const storeByUserId =  await db.store.findFirst({
            where: {
                id: params.storeId,
                userId
            }
        });

        if (!storeByUserId) {
            return new NextResponse("Unauthorized", {status: 403});
        }

        // Membuat entri baru pada tabel 'store' dalam database.
        const billboard = await db.billboard.create({
```

Memperbaiki route pada billboard

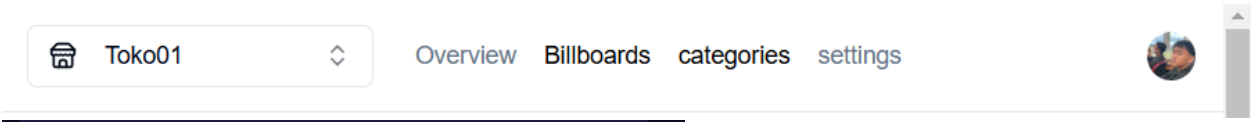
```
39
40  model Category {
41    id String @id @default(uuid())
42    storeId String
43    store Store @relation("StoreToCategory", fields: [storeId], references: [id])
44    billboardId String
45    billboard Billboard @relation(fields: [billboardId], references: [id])
46    name String
47    createdAt DateTime @default(now())
48    updatedAt DateTime @updatedAt
49
50    @@index([storeId])
51    @@index([billboardId])
52  }
```

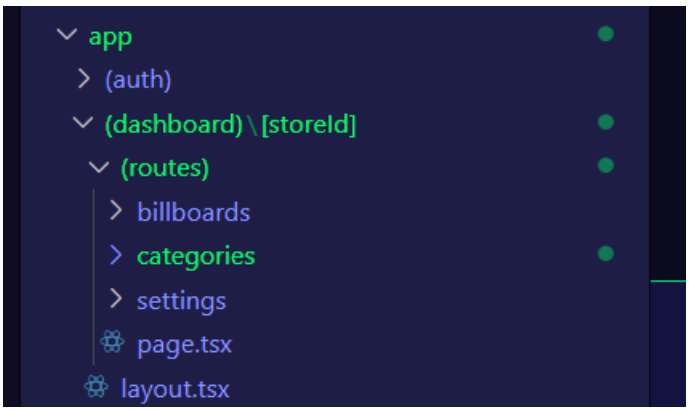
- Menambahkan model prisma baru berupa model Category.

```
{
    href: `/${params.storeId}/categories`,
    label: 'categories',
    active: pathname === `/${params.storeId}/billboards`,
},
```

- Menambahkan navbar berupa menu baru yaitu categories.

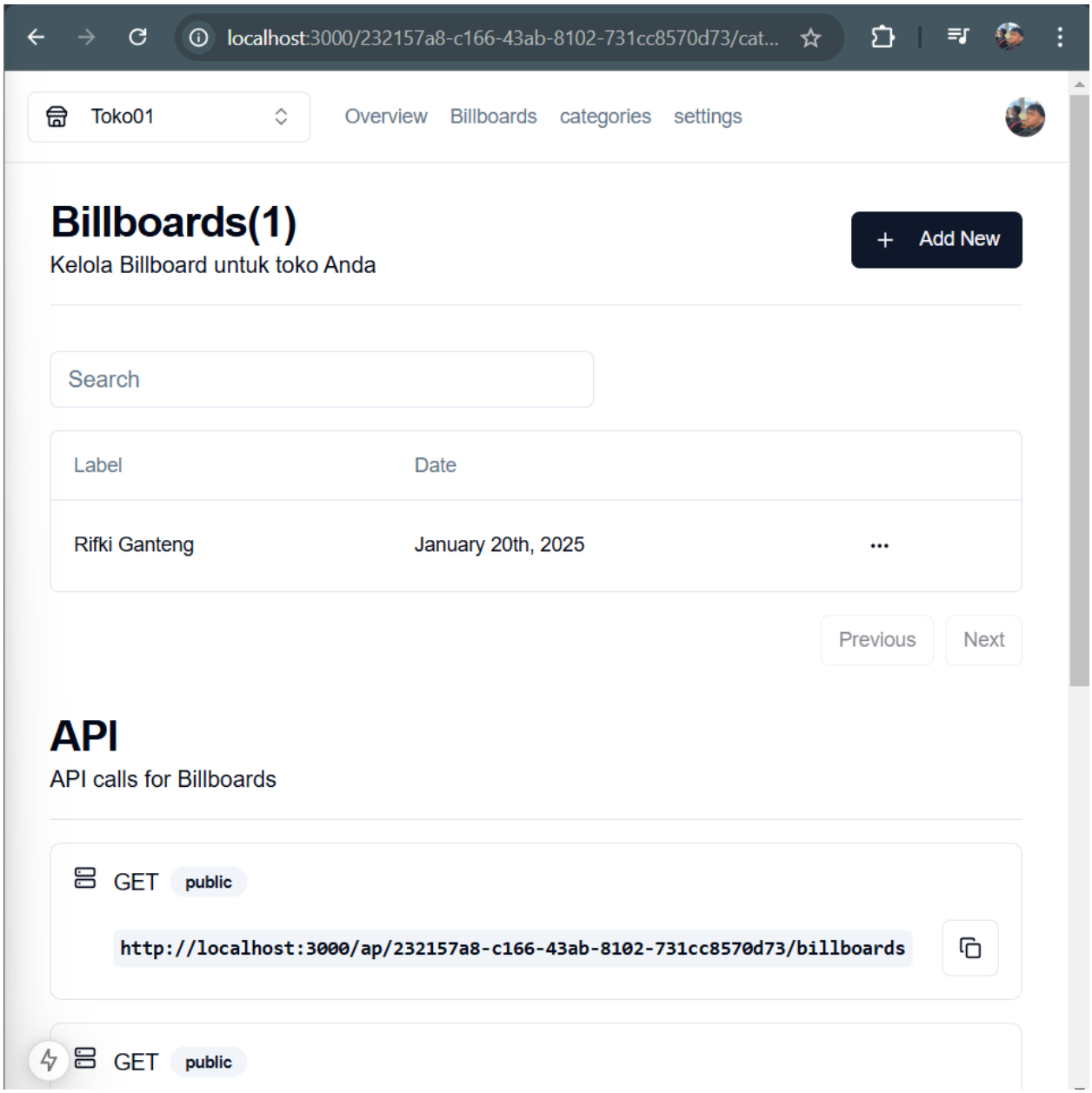
Hasil :





- Mengcopy folder billboards dan merename menjadi categories hingga menu categories pun bisa terpanggil.

hasil :


localhost:3000/232157a8-c166-43ab-8102-731cc8570d73/cat...
Toko01
Overview
Billboards
categories
settings
Billboards(1)
Kelola Billboard untuk toko Anda
Add New
Search
Label
Date
Rifki Ganteng
January 20th, 2025
Previous
Next
API
API calls for Billboards
GET
public
http://localhost:3000/ap/232157a8-c166-43ab-8102-731cc8570d73/billboards
GET
public

```tsx
const CategoriesPage = async ({
    params
}: {
    params: {storeId: string}
}) => {
    const categories= await db.category.findMany({
        where: {
            storeId: params.storeId
        },
        include: {
            billboard: true,
        },
        orderBy: {
            createdAt: 'desc'
        }
    });

    const formattedCategories: CategoryColumn[] = categories.map((item) => ({
        id: item.id,
        name: item.name,
        billboardLabel: item.billboard.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={formattedCategories}/>
            </div>
        </div>
    );
}

export default CategoriesPage;
```

Pada page.tsx folder categories kita memakai file lama dan lalu menyesuaikan Namanya menjadi CategoriesPage dan menyesuaikan lainnya dan menambahkan include pada const categories yang sudah saya kotakin orange, dan merubah dan menambahkan isi const pada formattedCategories berupa name dan billboardLabel yang saya kotakin warna yellow.

```tsx
export type CategoryColumn = {
  id: string
  name: string
  billboardLabel: string
  createdAt: string
}

export const columns: ColumnDef<CategoryColumn>[] = [
  {
    accessorKey: "name",
    header: "Name",
  },
  {
    accessorKey: "billboard",
    header: "Billboard",
    cell: ({ row }) => row.original.billboardLabel,
  },
  {
    accessorKey: "createdAt",
    header: "Date",
  },
  {
    id: "Actions",
    cell: ({row}) => <CellAction data={row.original}/>
  }
]
```

Lalu pada columns.tsx menyesuaikan namasebelumnya menjadi CategoryColumn. Menambahkan strring baru berupa name dan billboardLabel seperti yang saya kotakin Orange. Dan lalu menyesuaikan isi const columns seperti yang saya kotakin Yellow.

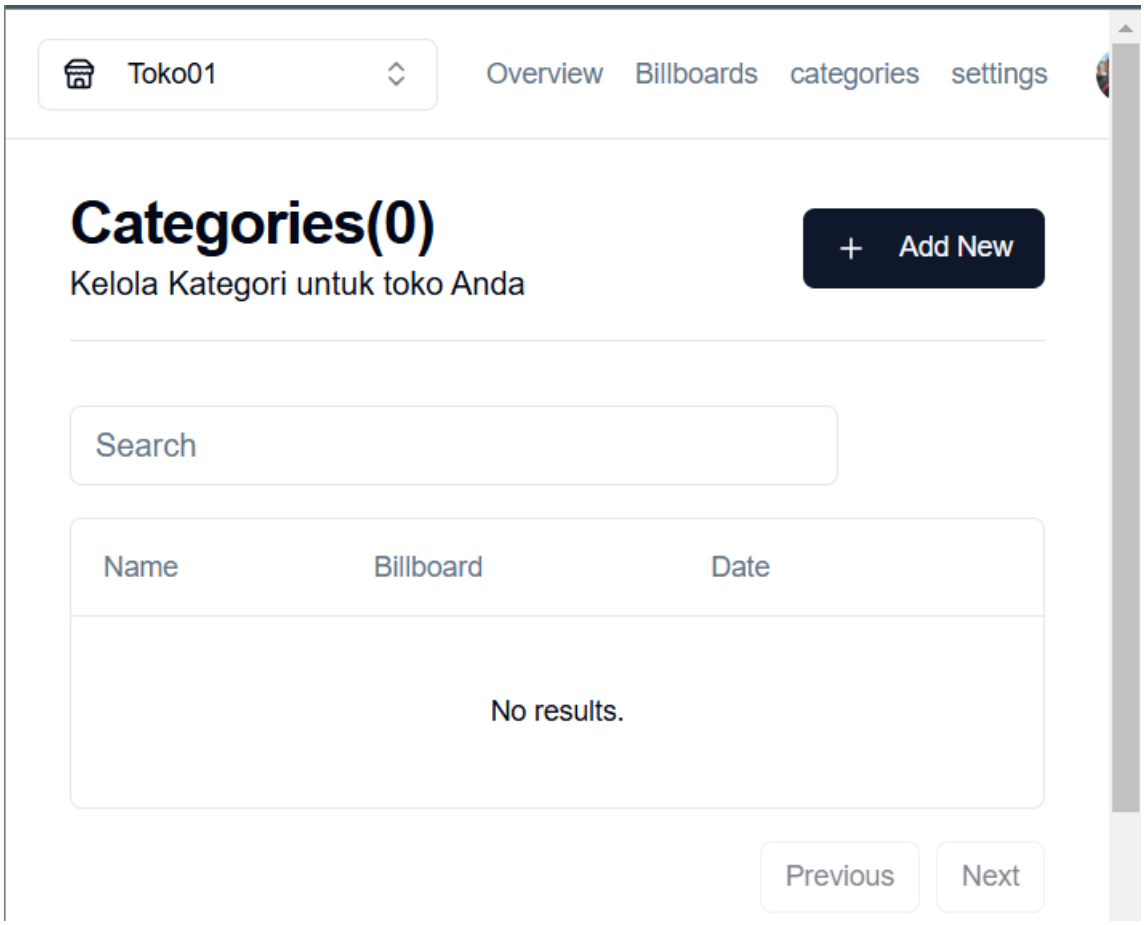
```tsx
import { DataTable } from "@/components/ui/data-table"
import { ApiList } from "@/components/ui/api-list"
import { CategoryColumn, columns } from "./columns"

interface CategoryClientProps{
    data: CategoryColumn[]
}

export const CategoryClient: React.FC<CategoryClientProps> = ({
    data
```

Melakukan penyesuaian pada client.tsx dan mengganti menjadi Category.

```tsx
return (
    <>
        <div className="flex items-center justify-between">
            <Heading
                title={`Categories(${data.length})`}
                description="Kelola Kategori untuk toko Anda"
            />
            <Button onClick={() => router.push(`/${params.storeId}/categories/new`)}>
                <Plus className="mr-2 h-4 w-4"/>
                Add New
            </Button>
        </div>
        <Separator />
        <DataTable searchKey="label" columns={columns} data={data}/>
        <Heading title="API" description="API calls for Categories"/>
        <Separator/>
        <ApiList entityName="categories" entityIdName="categoryId"/>
    </>
)
```

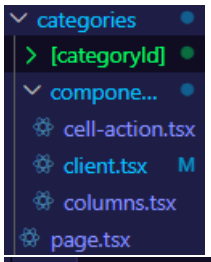
Melakukan perbaikan pada return dari yang sebelumnya billboard menjadi categories.

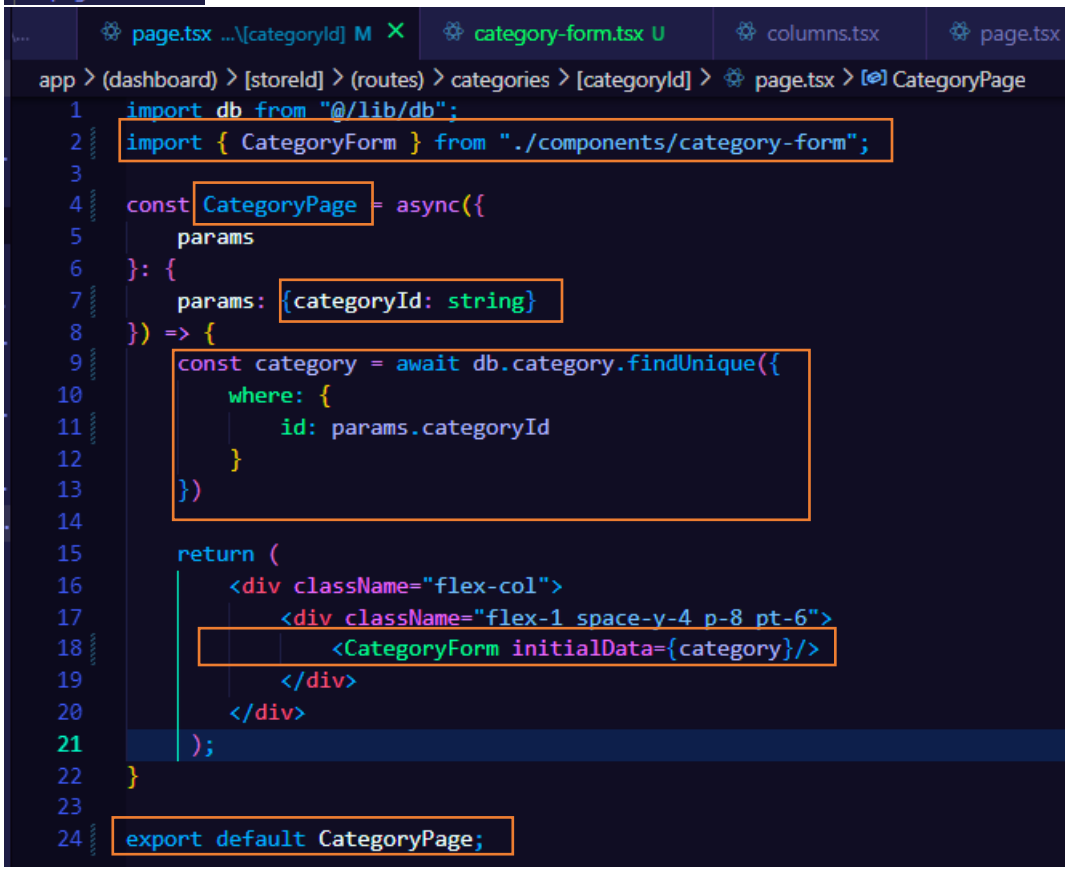
Hasil :



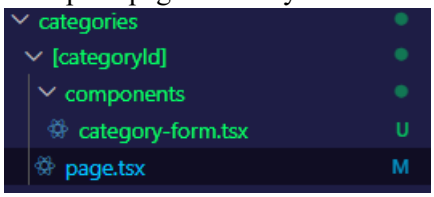
- ```
  <DataTable searchKey="name" columns={columns} data={data}/>
  ```

  Merubah searchKey menjadi name dan lalu merename folder [billboardId] menjadi [categoryId] seperti dibawah ini :



- 

  Lalu pada page.tsx menyesuaikan menjadi category seperti kotak warna orange.

- 

  Merename file sebelumnya menjadi category-form.tsx
  ```

```tsx
const formSchema = z. object ({
    name: z.string().min(1),
    billboardId: z.string().min(1)
});

type CategoryFormValues = z. infer<typeof formSchema>;

interface CategoryFromProps {
    initialData: Category | null;
}

export const CategoryForm: React.FC<CategoryFromProps> = ({
    initialData
}) => {
const params = useParams();
const router = useRouter();


const [open,setOpen] = useState(false);
const [loading,setLoading] = useState(false);

const title = initialData ? "Edit category" : "Create category";
const description = initialData ? "Edit category" : "Add a new category";
const toastMessage = initialData ? "Category update" : "Category created";
const action = initialData ? "Save change" : "Create";

    const form = useForm<CategoryFormValues>({
        resolver: zodResolver(formSchema),
        defaultValues: initialData || {
            name: '',
            billboardId: '',
        }
    });
```

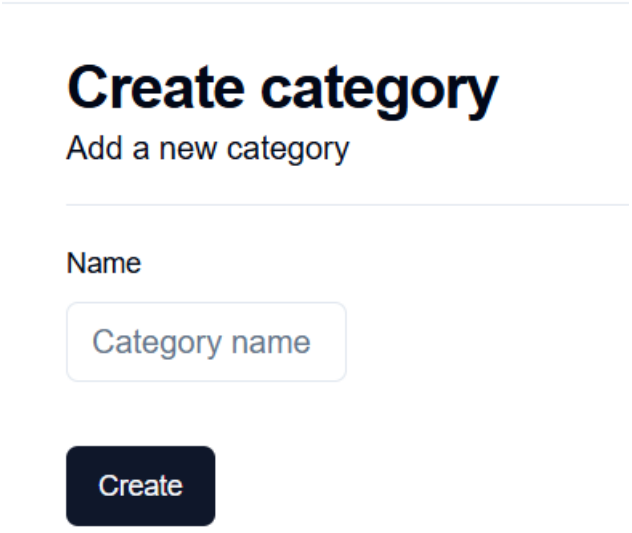
- Pada const formSchema diganti menjadi const name dan billboardId. Dan initialData diganti menjadi category. Lalu pada const form diganti menjadi name dan billboardId, seperti warna orange.

  Merubah nama billboard menjadi category pada kotak yellow.

```
</div>
<Separator />
<Form {...form}>
    <form onSubmit={form.handleSubmit(onSubmit)} className="space-y-8 w-full">
        <div className="grid grid-cols-3 gap-8">
         <FormField
         control={form.control}
         name="name"
         render={({field}) => (
            <FormItem>
                <FormLabel>Name</FormLabel>
                <FormControl>
                    <Input disabled={loading} placeholder="Category name"{...fi
                </FormControl>
                <FormMessage />
            </FormItem>
         )}
        />
        </div>
        <Button disabled={loading} className="ml-auto" type="submit">
            {action}
        </Button>
    </form>
</Form>
```

- Membuat form categorynya.

  Hasil :




```
PS D:\Universitas Teknokrat Indonesia\Semester 5\WEB 2\ecommerce-visionvortex> npx shadcn@latest add select
✓ Checking registry.
✓ Installing dependencies.
✓ Created 1 file:
  - components\ui\select.tsx
```

Menginsttall ui select.

- ```
  <FormField
  control={form.control}
  name="billboardId"
  render={({field}) => (
      <FormItem>
          <FormLabel>Billboard</FormLabel>
          <Select
              disabled={loading}
              onValueChange={field.onChange}
              value={field.value}
              defaultValue={field.value}
          >
              <FormControl>
                  <SelectTrigger>
                      <SelectValue
                          defaultValue={field.value}
                          placeholder="Select a Billboard"
                      />
                  </SelectTrigger>
              </FormControl>
              <SelectContent>

              </SelectContent>
          </Select>
          <FormMessage />
      </FormItem>
  )}
  />
  ```

Membuat FormField baru yang berisikan select pada form category.

- ```
  const CategoryPage = async({
      params
  }: {
      params: {categoryId: string, storeId: string}
  }) => {
  ```

Menambahkan string baru berupa storeId, pada page.tsx

- ```
  return (

              <CategoryForm
                  billboards={billboards}
                  initialData={category}
              />

  );
  ```

Menambahkan categoryForm berupa billboards pada page.tsx.
  ```

- 

```
interface CategoryFromProps {
    initialData: Category | null;
    billboards: Billboard[];
}
```

Pada categoryForm menambahkan interface baru yaitu billboards.