Atelier 3 NextJS

CRUD articles

# Variable d’environnement

Modifier next.config.js pour ajouter une variable d’environnement :



/\*\* @type {import('next').NextConfig} \*/

const nextConfig = {

    images:{

        domains:["res.cloudinary.com"]

        },

    env:{

        API\_URL : "http://localhost:3001/api"

    }

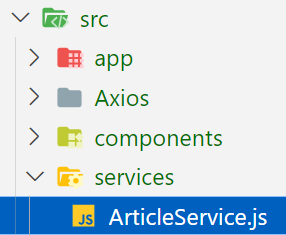
}

module.exports = nextConfig

# Les services

Ensuite, sous /src créer le dossier services.

Dans lequel créer ArticleService.js



const ARTICLE\_API="/articles/"

export const fetchArticles=async()=> {

const res = await fetch(process.env.API\_URL+ARTICLE\_API, { cache: 'no-store' })

if (!res.ok) {

  throw new Error('Failed to fetch data')

}

const response = await res.json();

return response;

}

export const fetchArticleById=async(articleId)=> {

    const res = await fetch(process.env.API\_URL+ARTICLE\_API+`${articleId}`,{

        method: 'GET'

    });

    const response = await res.json();

    return response

}

export const deleteArticle=async(articleId) =>{

    const res = await fetch(process.env.API\_URL+ARTICLE\_API+`${articleId}`,{

        method: 'DELETE'

    });

    const response = await res.json();

    return response;

}

export const addArticle=async(article)=> {

    const res = await fetch(process.env.API\_URL+ARTICLE\_API, {

        method: 'POST',

        headers: {

        'Content-Type': 'application/json',

        },

        body: JSON.stringify(article),

    });

    const response = await res.json();

    return response;

}

export const editArticle=async(article) =>{

    const res = await fetch(process.env.API\_URL+ARTICLE\_API+`${article.\_id}`, {

        method: 'PUT',

        headers: {

        'Content-Type': 'application/json',

        },

        body: JSON.stringify(article),

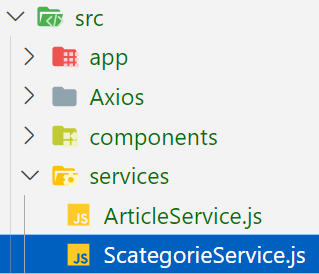
    });

    const response = await res.json();

    return response;

}

De même, créer ScategorieService.js



const SCATEGORIE\_API="/scategories/"

export const fetchSCategories=async()=> {

    const res = await fetch(process.env.API\_URL+SCATEGORIE\_API, { cache: 'no-store' })

    if (!res.ok) {

      throw new Error('Failed to fetch data')

    }

    const response = await res.json();

    return response;

    }

export const fetchSCategorieById=async(scategorieId)=> {

    const res = await fetch(process.env.API\_URL+SCATEGORIE\_API+`${scategorieId}`,{

        method: 'GET'

    });

    const response = await res.json();

    return response

}

export const deleteSCategorie=async(scategorieId) =>{

    const res = await fetch(process.env.API\_URL+SCATEGORIE\_API+`${scategorieId}`,{

        method: 'DELETE'

    });

    const response = await res.json();

    return response;

}

export const addSCategorie=async(scategorie)=> {

    const res = await fetch(process.env.API\_URL+SCATEGORIE\_API, {

        method: 'POST',

        headers: {

        'Content-Type': 'application/json',

        },

        body: JSON.stringify(scategorie),

    });

    const response = await res.json();

    return response;

}

export const editSCategorie=async(scategorie) =>{

    const res = await fetch(process.env.API\_URL+SCATEGORIE\_API+`${scategorie.\_id}`, {

        method: 'PUT',

        headers: {

        'Content-Type': 'application/json',

        },

        body: JSON.stringify(scategorie),

    });

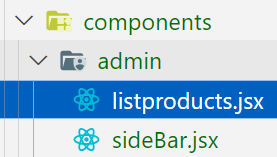
    const response = await res.json();

    return response;

}

# Affichage des articles

Sous /src/components/admin créer le fichier listproducts.jsx



Accéder au site<https://www.material-react-table.com/>

Puis Quick examples->Basic Example

Installer le composant material-react-table

npm install material-react-table

"use client"

import  { useMemo } from 'react';

import { MaterialReactTable } from 'material-react-table';

import { Box } from '@mui/material';

import Image from "next/image"

const Listproducts = ({produits}) => {

  const columns = useMemo(

    () => [

    {

    accessorKey: 'imageart', //access nested data with dot notation

    header: 'Image',

    Cell: ({ cell}) => (

    <Box

    sx={{

    display: 'flex',

    alignItems: 'center',

    gap: '1rem',

    }}

    >

    <Image

              src={cell.getValue()}

              alt="product anme"

              height="50"

              width="50"

            />

    </Box>),

    },

    {

    accessorKey: 'reference', //access nested data with dot notation

    header: 'Référence',

    size: 100,

    },

    {

    accessorKey: 'designation',

    header: 'Désignation',

    size: 100,

    },

    {

    accessorKey: 'marque', //normal accessorKey

    header: 'Marque',

    size: 100,

    },

    {

    accessorKey: 'prix',

    header: 'Prix',

    size: 100,

    },

    {

    accessorKey: 'qtestock',

    header: 'Stock',

    size: 100,

    },

    ],

    [produits],

    );

return (

<div>

<MaterialReactTable columns={columns} data={produits} />;

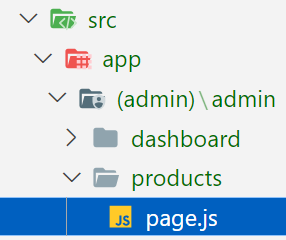
</div>

  )

}

export default Listproducts

On va créer un dossier appelé products sous /src/app/(admin)/admin ayant page.js



import Listproducts from '@/components/admin/listproducts';

import {fetchArticles} from "@/services/ArticleService"

const getProducts=async()=>{

const data=await fetchArticles()

return data;

}

const ProductPage = async() =>{

    const produits=await getProducts()

  return (

   <div className="container">

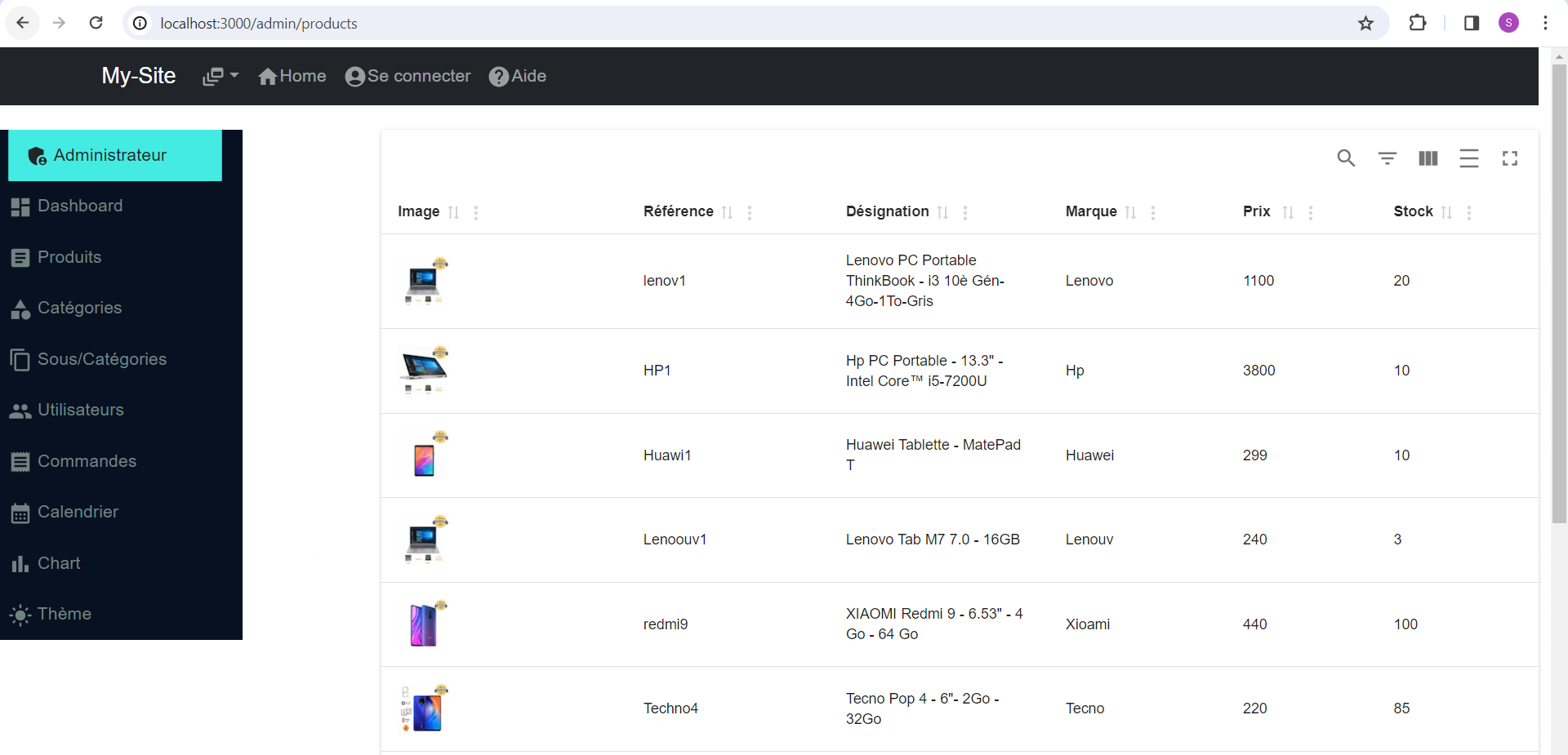
      <Listproducts produits={produits}/>

    </div>

  )

}

export default ProductPage



# File conventions

## error.js

Un fichier d'erreurs définit une limite d'erreur d'interface utilisateur pour un segment de route.

Il est utile pour détecter les erreurs inattendues qui se produisent dans les composants serveur et les composants clients et pour afficher une interface utilisateur de secours.

Sous /src/app/(admin)/admin/products créer le fichier error.js



'use client' // Error components must be Client Components

import { useEffect } from 'react'

export default function Error({ error, reset }) {

  useEffect(() => {

    // Log the error to an error reporting service

    console.error(error)

  }, [error])

  return (

    <div>

      <h2>Something went wrong!</h2>

      <button

        onClick={

          // Attempt to recover by trying to re-render the segment

          () => reset()

        }

      >

        Try again

      </button>

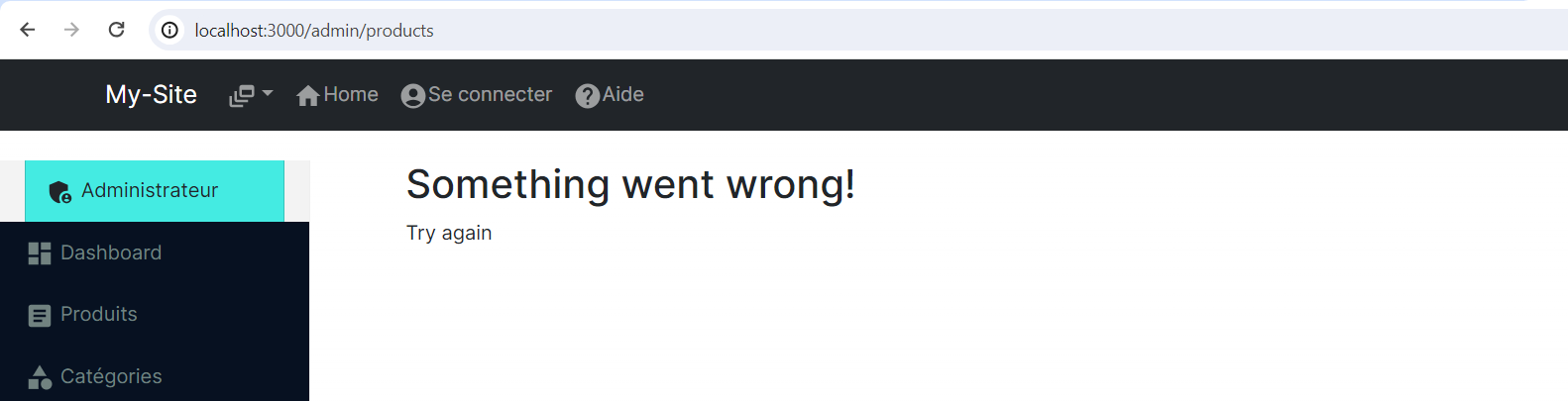
    </div>

  )

}

Un essai est de fermer le serveur backend puis d’essayer l’url : <http://localhost:3000/admin/products>

On aura une page personnalisée d’erreur.



## loading.js

Par défaut, ce fichier est un composant serveur - mais peut également être utilisé comme composant client via la directive "use client".

Sous /src/app/(admin)/admin/products créer le fichier loading.js



export default function Loading() {

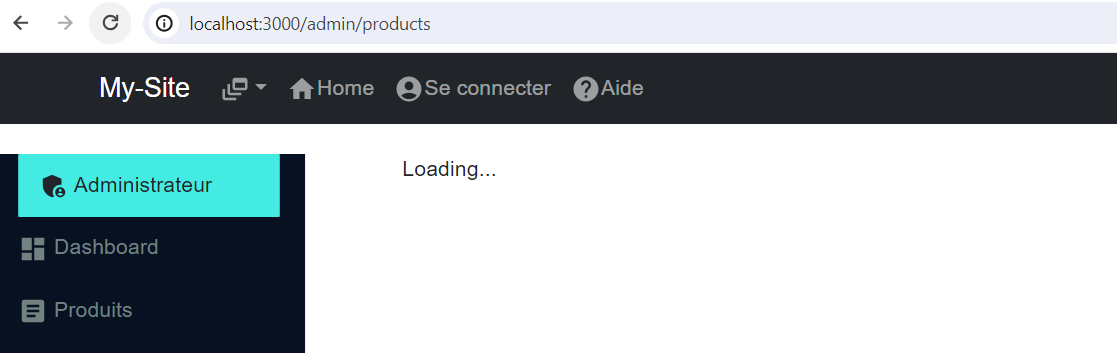
    // Or a custom loading skeleton component

    return <p>Loading...</p>

  }

Loading n'accepte aucun paramètre.

Comme essai le temps de charger la page on peut voir dans l’url : <http://localhost:3000/admin/products>



## not-found.js

Le fichier not-found est utilisé pour restituer l'interface utilisateur lorsque la fonction notFound est lancée dans un segment de route. En plus de proposer une interface utilisateur personnalisée, Next.js renverra un code d'état HTTP 200 pour les réponses diffusées en continu et 404 pour les réponses non diffusées.

En plus de détecter les erreurs notFound() attendues, le fichier racine app/not-found.js gère également toutes les URL sans correspondance pour l'ensemble de votre application. Cela signifie que les utilisateurs qui visitent une URL qui n'est pas gérée par votre application verront l'interface utilisateur exportée par le fichier app/not-found.js.

Sous **/src/app/** créer le fichier not-found.js



import Link from 'next/link'

export default function NotFound() {

  return (

    <div>

      <h2>Not Found</h2>

      <p>Could not find requested resource</p>

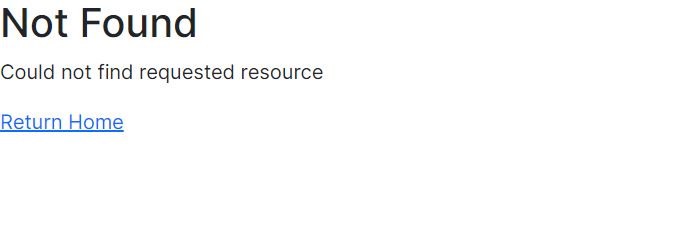
      <Link href="/">Return Home</Link>

    </div>

  )

}

Si on essaie n’importe quelle URL inexistante on aura :



# Suppression d’articles

On va ajouter l’action de suppression, modifier le contenu de listproducts.jsx



"use client"

import  { useMemo } from 'react';

import { MaterialReactTable } from 'material-react-table';

import { Box } from '@mui/material';

import Image from "next/image"

import Button from 'react-bootstrap/Button';

import {deleteArticle} from "@/services/ArticleService"

import { useRouter } from "next/navigation";

import DeleteForeverIcon from '@mui/icons-material/DeleteForever';

const Listproducts = ({produits}) => {

    const router = useRouter();

    const deletearticle=(id)=>{

      if(window.confirm("supprimer le produit O/N")) {

        deleteArticle(id)

        .then((res)=>{ console.log(res)

         router.refresh()

        })

        .catch(error=>{

            console.log(error)

           })

  }

        }

  const columns = useMemo(

    () => [

    {

    accessorKey: 'imageart', //access nested data with dot notation

    header: 'Image',

    Cell: ({ cell}) => (

    <Box

    sx={{

    display: 'flex',

    alignItems: 'center',

    gap: '1rem',

    }}

    >

    <Image

              src={cell.getValue()}

              alt="product anme"

              height="50"

              width="50"

            />

    </Box>),

    },

    {

    accessorKey: 'reference', //access nested data with dot notation

    header: 'Référence',

    size: 100,

    },

    {

    accessorKey: 'designation',

    header: 'Désignation',

    size: 100,

    },

    {

    accessorKey: 'marque', //normal accessorKey

    header: 'Marque',

    size: 100,

    },

    {

    accessorKey: 'prix',

    header: 'Prix',

    size: 100,

    },

    {

    accessorKey: 'qtestock',

    header: 'Stock',

    size: 100,

    },

    {

        accessorKey: '\_id',

        header: 'actions',

        size: 100,

        Cell: ({ cell, row }) => (

        <div >

        <Button

        onClick={(e) => {

        deletearticle(cell.row.original.\_id,e);

        }}

        variant="danger"

        size="md"

        className="text-danger btn-link delete"

        >

        <DeleteForeverIcon />

        </Button>

        </div>

        ),

        },

    ],

    [produits],

    );

return (

<div className="container">

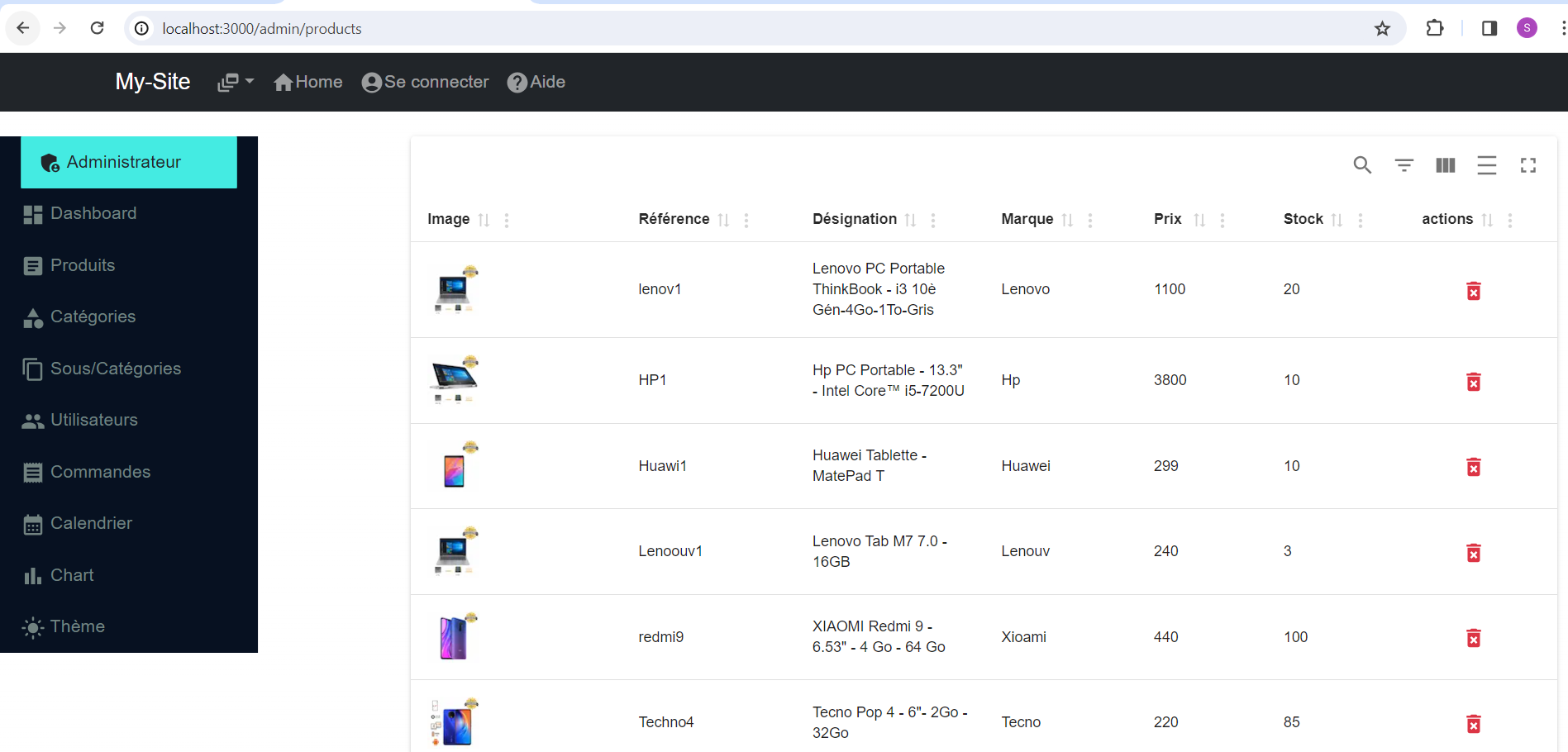
<MaterialReactTable columns={columns} data={produits} />;

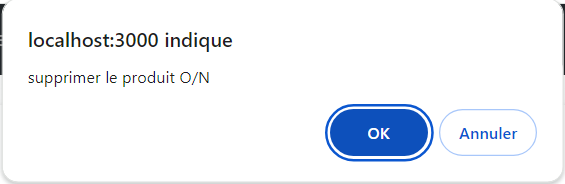
</div>

  )

}

export default Listproducts





# Ajout d’articles

On va ajouter le formulaire d’ajout.

Tout d’abord modifier lisproducts pour ajouter le bouton de lien vers le formulaire.



"use client"

import  { useMemo } from 'react';

import { MaterialReactTable } from 'material-react-table';

import { Box } from '@mui/material';

import Image from "next/image"

import Button from 'react-bootstrap/Button';

import {deleteArticle} from "@/services/ArticleService"

import { useRouter } from "next/navigation";

import DeleteForeverIcon from '@mui/icons-material/DeleteForever';

import Link from 'next/link';

import AddCircleOutlineIcon from '@mui/icons-material/AddCircleOutline';

const Listproducts = ({produits}) => {

    const router = useRouter();

    const deletearticle=(id)=>{

      if(window.confirm("supprimer le produit O/N")) {

        deleteArticle(id)

        .then((res)=>{ console.log(res)

         router.refresh()

        })

        .catch(error=>{

            console.log(error)

           })

  }

        }

  const columns = useMemo(

    () => [

    {

    accessorKey: 'imageart', //access nested data with dot notation

    header: 'Image',

    Cell: ({ cell}) => (

    <Box

    sx={{

    display: 'flex',

    alignItems: 'center',

    gap: '1rem',

    }}

    >

    <Image

              src={cell.getValue()}

              alt="product anme"

              height="50"

              width="50"

            />

    </Box>),

    },

    {

    accessorKey: 'reference', //access nested data with dot notation

    header: 'Référence',

    size: 100,

    },

    {

    accessorKey: 'designation',

    header: 'Désignation',

    size: 100,

    },

    {

    accessorKey: 'marque', //normal accessorKey

    header: 'Marque',

    size: 100,

    },

    {

    accessorKey: 'prix',

    header: 'Prix',

    size: 100,

    },

    {

    accessorKey: 'qtestock',

    header: 'Stock',

    size: 100,

    },

    {

        accessorKey: '\_id',

        header: 'actions',

        size: 100,

        Cell: ({ cell, row }) => (

        <div >

        <Button

        onClick={(e) => {

        deletearticle(cell.row.original.\_id,e);

        }}

        variant="danger"

        size="md"

        className="text-danger btn-link delete"

        >

        <DeleteForeverIcon />

        </Button>

        </div>

        ),

        },

    ],

    [produits],

    );

return (

<div className="container">

<Button

variant='dark'

size="sm"

>

<Link

href="/admin/products/newProduct"

style={{

  textDecoration: 'none',

  color: 'aqua',

  fontSize: 14,

}}

>

<AddCircleOutlineIcon/> Nouveau

</Link>

</Button>

<MaterialReactTable columns={columns} data={produits} />;

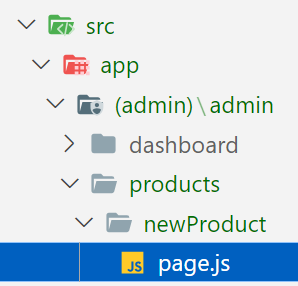
</div>

  )

}

export default Listproducts

Créer sous /src/app/(admin)/admin/products/ le dossier newProducts incluant page.js



import NewProduct from "@/components/admin/newProductComponent";

import {fetchSCategories} from "@/services/ScategorieService"

const getscategories=async()=>{

    const data=await fetchSCategories()

    return data;

}

const NewProductPage = async() => {

    const scategories=await getscategories()

  return (

    <div>

     <NewProduct scategories={scategories}/>

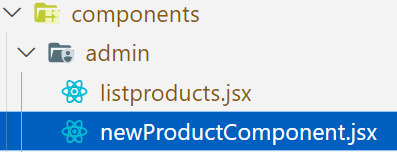
    </div>

  )

}

export default NewProductPage

Par la suite, créer sous /src/components/admin/newProductComponent.jsx



"use client"

import React, { useState } from 'react';

import Button from 'react-bootstrap/Button';

import Form from 'react-bootstrap/Form';

import Col from 'react-bootstrap/Col';

import InputGroup from 'react-bootstrap/InputGroup';

import Row from 'react-bootstrap/Row';

import { useRouter } from "next/navigation";

import { addArticle } from '@/services/ArticleService';

const NewProduct = ({scategories}) => {

  const router = useRouter();

  const [reference, setReference] = useState("");

const [designation, setDesignation] = useState("");

const [prix, setPrix] = useState("");

const [marque, setMarque] = useState("");

const [qtestock, setQtestock] = useState("");

const [imageart, setImageart] = useState("");

const [scategorieID, setScategorieID] = useState("");

const [validated, setValidated] = useState(false);

const handleSubmit = (e) => {

e.preventDefault();

const form = e.currentTarget;

if (form.checkValidity() === true) {

const newProduct = {

reference,

designation,

prix,

marque,

qtestock,

imageart,

scategorieID

};

//faire le add dans la BD

addArticle(newProduct)

.then(res => {

  router.push('/admin/products')

    router.refresh()

})

.catch(error=>{

alert("Erreur ! Insertion non effectuée")

})

}

setValidated(true);

}

return (

<div>

 <Form noValidate validated={validated} onSubmit={handleSubmit}>

 <h2>Ajout Produit</h2>

<div className="container w-100 d-flex justify-content-center">

<div>

<div className='form mt-3'>

<Row className="mb-2">

<Form.Group as={Col} md="6" >

<Form.Label >Référence \*</Form.Label>

<Form.Control

required

type="text"

placeholder="Référence"

value={reference}

onChange={(e)=>setReference(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Saisir Référence Article

</Form.Control.Feedback>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Désignation \*</Form.Label>

<Form.Control

required

type="text"

placeholder="Désignation"

value={designation}

onChange={(e)=>setDesignation(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Saisir Désignation

</Form.Control.Feedback>

</Form.Group>

</Row>

<Row className="mb-2">

<Form.Group className="col-md-6">

<Form.Label>Marque \*</Form.Label>

<InputGroup hasValidation>

<Form.Control

type="text"

required

placeholder="Marque"

value={marque}

onChange={(e)=>setMarque(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Marque Incorrecte

</Form.Control.Feedback>

</InputGroup>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Prix</Form.Label>

<Form.Control

type="number"

placeholder="Prix"

value={prix}

onChange={(e)=>setPrix(e.target.value)}

/>

</Form.Group>

</Row>

<Row className="mb-3">

<Form.Group className="col-md-6 ">

<Form.Label>

Qté stock<span className="req-tag">\*</span>

</Form.Label>

<Form.Control

required

type="number"

value={qtestock}

onChange={(e)=>setQtestock(e.target.value)}

placeholder="Qté stock"

/>

<Form.Control.Feedback type="invalid">

Qté stock Incorrect

</Form.Control.Feedback>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Image</Form.Label>

<Form.Control

type="text"

placeholder="Image"

value={imageart}

onChange={(e)=>setImageart(e.target.value)}

/>

</Form.Group>

<Form.Group as={Col} md="12">

<Form.Label>S/Catégorie</Form.Label>

<Form.Control

as="select"

type="select"

value={scategorieID}

onChange={(e)=>setScategorieID(e.target.value)}

>

<option></option>

{scategories.map((scat)=><option key={scat.\_id}

value={scat.\_id}>{scat.nomscategorie}</option>

)}

</Form.Control>

</Form.Group>

</Row>

</div>

</div>

</div>

<Button type="submit">Enregistrer</Button>

<Button type="button" className="btn btn-warning"

onClick={()=>handleReset()}>Annuler</Button>

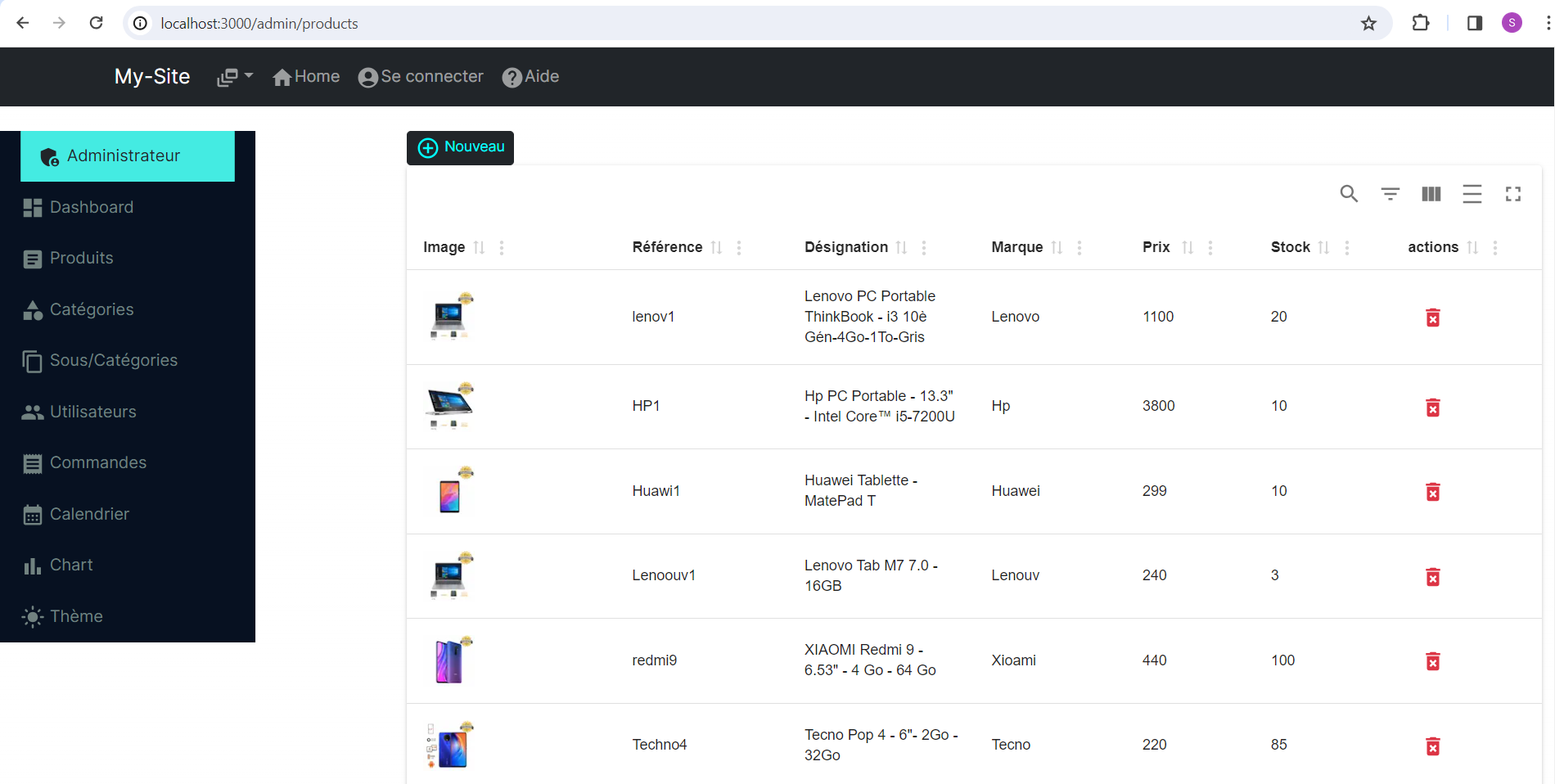
</Form>

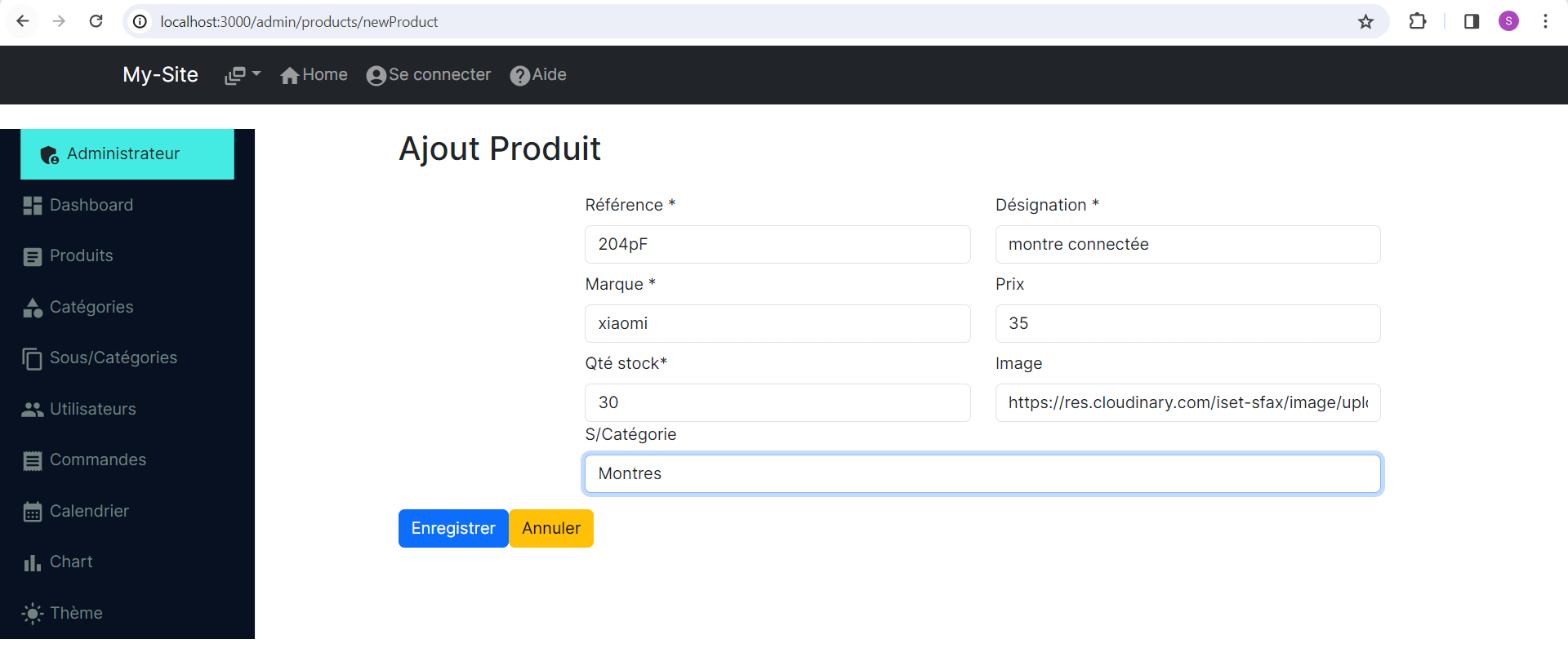
</div>

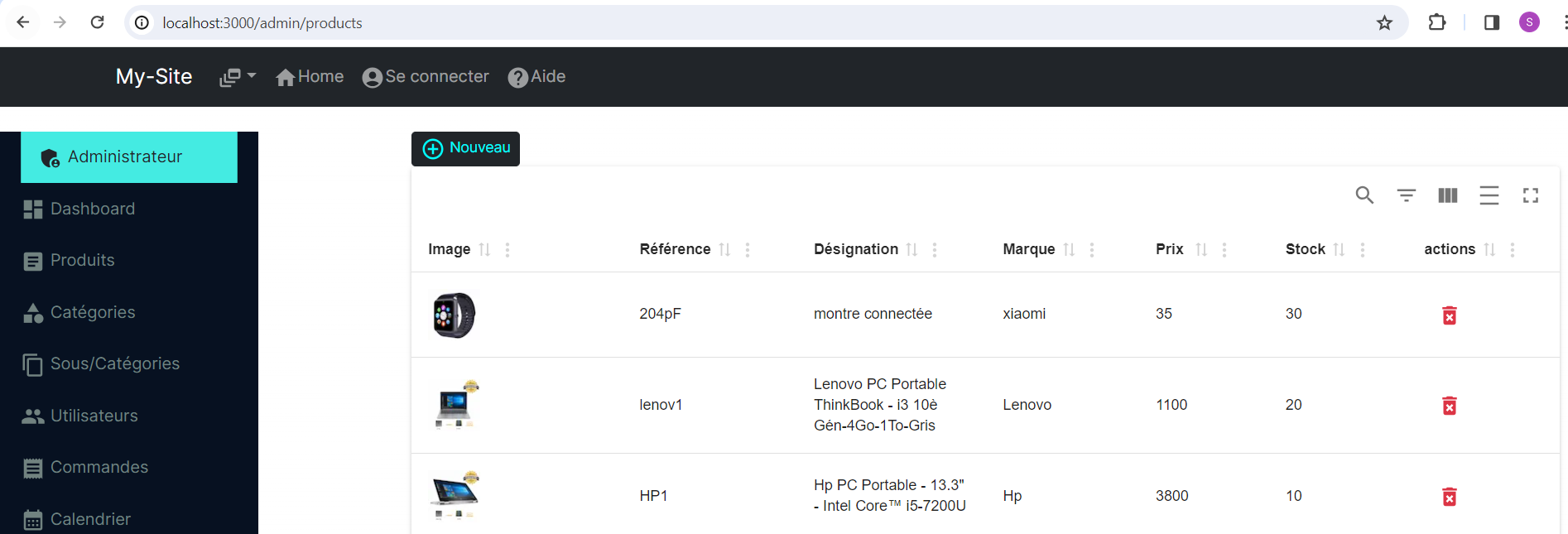
);

};

export default NewProduct







# Upload d’images

Maintenant on va intégrer l’upload vers Cloudinary. Il s’agit d’une API d'images et de vidéos pour des expériences visuelles puissantes. Vous pourrez y stocker, transformer, optimiser et diffuser tous vos actifs multimédias avec des API, des widgets ou une interface utilisateur faciles à utiliser.

La progression de l’upload sera effectuée avec Filepond qui est une bibliothèque JavaScript qui peut télécharger tout ce que vous lui lancez, optimise les images pour des téléchargements plus rapides et offre une expérience utilisateur exceptionnelle, accessible et fluide.

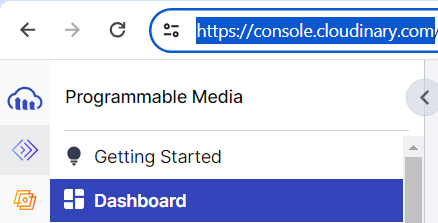
Tout d’abord, il faut créer un compte dans le site Cloudinary en suivant cette URL :

<https://cloudinary.com/>

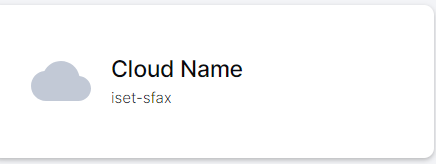
Une fois le compte est créé, accédez à la console https://console.cloudinary.com/

Nous avons 2 données précises à noter qu’on va en avoir besoin par la suite dans le code.

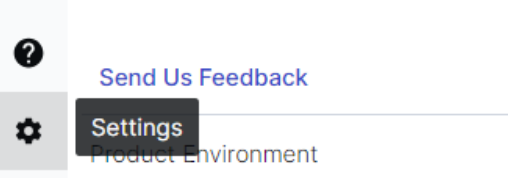
D’abord dans la rubrique Dashboard



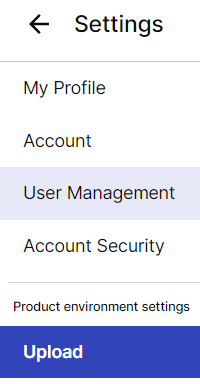
Copiez la valeur du cloudName. Exemple :



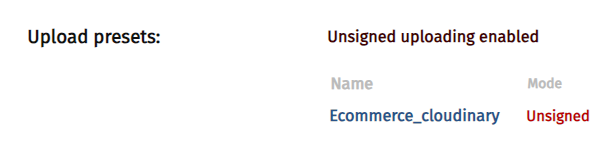
Par la suite, accédez à Settings (en bas à gauche)



Puis Upload (Menu à gauche)



Puis en bas de la page, on retrouvera, « Upload presets », il faut aussi copier le nom. Exemple :



Ensuite, dans le projet, installer les packages suivants :

npm i filepond react-filepond filepond-plugin-file-encode filepond-plugin-file-validate-type filepond-plugin-image-exif-orientation filepond-plugin-image-preview filepond-plugin-pdf-preview

Modifier le contenu de NewProduct.jsx

**N.B.** : Veuillez changer les valeurs ci-dessous dans le code par les paramètres de votre compte.

data.append('upload\_preset', 'Ecommerce\_cloudinary');

      data.append('cloud\_name', 'iset-sfax');

      data.append('public\_id', file.name);

      axios.post('https://api.cloudinary.com/v1\_1/iset-sfax/image/upload', data)

**Remarque** : Remplacer par le cloud-name cette zone, fourni par Cloudinary.

  axios.post('https://api.cloudinary.com/v1\_1/iset-sfax/image/upload', data)



"use client"

import React, { useState } from 'react';

import Button from 'react-bootstrap/Button';

import Form from 'react-bootstrap/Form';

import Col from 'react-bootstrap/Col';

import InputGroup from 'react-bootstrap/InputGroup';

import Row from 'react-bootstrap/Row';

import { useRouter } from "next/navigation";

import { addArticle } from '@/services/ArticleService';

import axios from 'axios';

import { FilePond,registerPlugin } from 'react-filepond'

import 'filepond/dist/filepond.min.css';

import FilePondPluginImageExifOrientation from 'filepond-plugin-image-exif-orientation'

import FilePondPluginImagePreview from 'filepond-plugin-image-preview'

import 'filepond-plugin-image-preview/dist/filepond-plugin-image-preview.css'

registerPlugin(FilePondPluginImageExifOrientation, FilePondPluginImagePreview)

const NewProduct = ({scategories}) => {

  const router = useRouter();

**const [files, setFiles] = useState([]);**

const [reference, setReference] = useState("");

const [designation, setDesignation] = useState("");

const [prix, setPrix] = useState("");

const [marque, setMarque] = useState("");

const [qtestock, setQtestock] = useState("");

const [imageart, setImageart] = useState("");

const [scategorieID, setScategorieID] = useState("");

const [validated, setValidated] = useState(false);

const handleSubmit = (e) => {

e.preventDefault();

const form = e.currentTarget;

if (form.checkValidity() === true) {

const newProduct = {

reference,

designation,

prix,

marque,

qtestock,

imageart,

scategorieID

};

//faire le add dans la BD

addArticle(newProduct)

.then(res => {

  router.push('/admin/products')

    router.refresh()

})

.catch(error=>{

alert("Erreur ! Insertion non effectuée")

})

}

setValidated(true);

}

const serverOptions = () => { console.log('server pond');

  return {

    process: (fieldName, file, metadata, load, error, progress, abort) => {

        console.log(file)

      const data = new FormData();

      data.append('file', file);

      data.append('upload\_preset', 'Ecommerce\_cloudinary');

      data.append('cloud\_name', 'iset-sfax');

      data.append('public\_id', file.name);

      axios.post('https://api.cloudinary.com/v1\_1/iset-sfax/image/upload', data)

        .then((response) => response.data)

        .then((data) => {

          console.log(data);

         setImageart(data.url) ;

          load(data);

        })

        .catch((error) => {

          console.error('Error uploading file:', error);

          error('Upload failed');

          abort();

        });

    },

  };

};

return (

<div>

 <Form noValidate validated={validated} onSubmit={handleSubmit}>

 <h2>Ajout Produit</h2>

<div className="container w-100 d-flex justify-content-center">

<div>

<div className='form mt-3'>

<Row className="mb-2">

<Form.Group as={Col} md="6" >

<Form.Label >Référence \*</Form.Label>

<Form.Control

required

type="text"

placeholder="Référence"

value={reference}

onChange={(e)=>setReference(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Saisir Référence Article

</Form.Control.Feedback>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Désignation \*</Form.Label>

<Form.Control

required

type="text"

placeholder="Désignation"

value={designation}

onChange={(e)=>setDesignation(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Saisir Désignation

</Form.Control.Feedback>

</Form.Group>

</Row>

<Row className="mb-2">

<Form.Group className="col-md-6">

<Form.Label>Marque \*</Form.Label>

<InputGroup hasValidation>

<Form.Control

type="text"

required

placeholder="Marque"

value={marque}

onChange={(e)=>setMarque(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Marque Incorrecte

</Form.Control.Feedback>

</InputGroup>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Prix</Form.Label>

<Form.Control

type="number"

placeholder="Prix"

value={prix}

onChange={(e)=>setPrix(e.target.value)}

/>

</Form.Group>

</Row>

<Row className="mb-3">

<Form.Group className="col-md-6 ">

<Form.Label>

Qté stock<span className="req-tag">\*</span>

</Form.Label>

<Form.Control

required

type="number"

value={qtestock}

onChange={(e)=>setQtestock(e.target.value)}

placeholder="Qté stock"

/>

<Form.Control.Feedback type="invalid">

Qté stock Incorrect

</Form.Control.Feedback>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Image</Form.Label>

<div style={{ width: "80%", margin: "auto", padding: "1%" }}>

     <FilePond

                   files={files}

                   acceptedFileTypes="image/\*"

                   onupdatefiles={setFiles}

                   allowMultiple={false}

                   server={serverOptions()}

                   name="file"

          />

    </div>

</Form.Group>

<Form.Group as={Col} md="12">

<Form.Label>S/Catégorie</Form.Label>

<Form.Control

as="select"

type="select"

value={scategorieID}

onChange={(e)=>setScategorieID(e.target.value)}

>

<option></option>

{scategories.map((scat)=><option key={scat.\_id}

value={scat.\_id}>{scat.nomscategorie}</option>

)}

</Form.Control>

</Form.Group>

</Row>

</div>

</div>

</div>

<Button type="submit">Enregistrer</Button>

<Button type="button" className="btn btn-warning"

onClick={()=>handleReset()}>Annuler</Button>

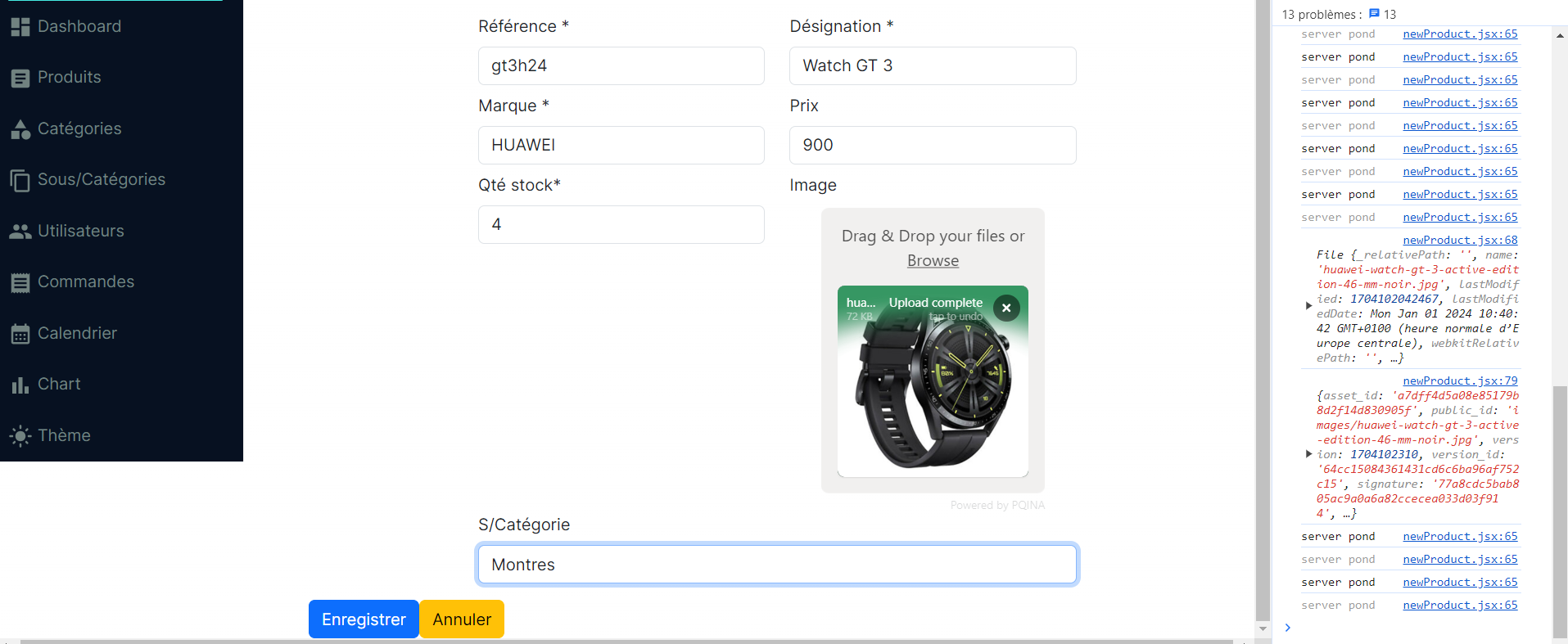
</Form>

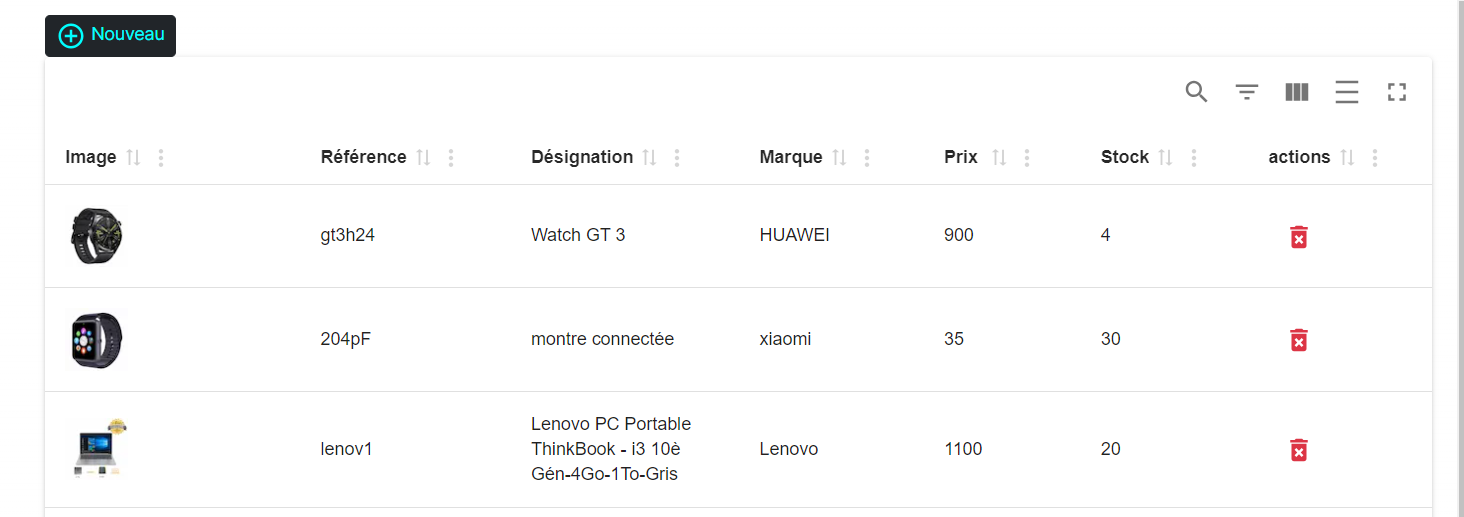
</div>

);

};

export default NewProduct





# Modification des articles

Maintenant, on va procéder à la création du formulaire de la modification.

D’abord, modifier listProducts pour ajouter le bouton Link vers la page edit.

"use client"

import  { useMemo } from 'react';

import { MaterialReactTable } from 'material-react-table';

import { Box } from '@mui/material';

import Image from "next/image"

import Button from 'react-bootstrap/Button';

import {deleteArticle} from "@/services/ArticleService"

import { useRouter } from "next/navigation";

import DeleteForeverIcon from '@mui/icons-material/DeleteForever';

import Link from 'next/link';

import AddCircleOutlineIcon from '@mui/icons-material/AddCircleOutline';

import EditOutlinedIcon from '@mui/icons-material/EditOutlined';

const Listproducts = ({produits}) => {

    const router = useRouter();

    const deletearticle=(id)=>{

      if(window.confirm("supprimer le produit O/N")) {

        deleteArticle(id)

        .then((res)=>{ console.log(res)

         router.refresh()

        })

        .catch(error=>{

            console.log(error)

           })

  }

        }

  const columns = useMemo(

    () => [

    {

    accessorKey: 'imageart', //access nested data with dot notation

    header: 'Image',

    Cell: ({ cell}) => (

    <Box

    sx={{

    display: 'flex',

    alignItems: 'center',

    gap: '1rem',

    }}

    >

    <Image

              src={cell.getValue()}

              alt="product anme"

              height="50"

              width="50"

            />

    </Box>),

    },

    {

    accessorKey: 'reference', //access nested data with dot notation

    header: 'Référence',

    size: 100,

    },

    {

    accessorKey: 'designation',

    header: 'Désignation',

    size: 100,

    },

    {

    accessorKey: 'marque', //normal accessorKey

    header: 'Marque',

    size: 100,

    },

    {

    accessorKey: 'prix',

    header: 'Prix',

    size: 100,

    },

    {

    accessorKey: 'qtestock',

    header: 'Stock',

    size: 100,

    },

    {

        accessorKey: '\_id',

        header: 'actions',

        size: 100,

        Cell: ({ cell, row }) => (

        <div >

      <Button

       size="md"

       className="text-primary btn-link edit"

      >

        <Link href={`/admin/products/updateProduct/${cell.row.original.\_id}`}>

        <EditOutlinedIcon/>

    </Link>

      </Button>

        <Button

        onClick={(e) => {

        deletearticle(cell.row.original.\_id,e);

        }}

        variant="danger"

        size="md"

        className="text-danger btn-link delete"

        >

        <DeleteForeverIcon />

        </Button>

        </div>

        ),

        },

    ],

    [produits],

    );

return (

<div className="container">

<Button

variant='dark'

size="sm"

>

<Link

href="/admin/products/newProduct"

style={{

  textDecoration: 'none',

  color: 'aqua',

  fontSize: 14,

}}

>

<AddCircleOutlineIcon/> Nouveau

</Link>

</Button>

<MaterialReactTable columns={columns} data={produits} />;

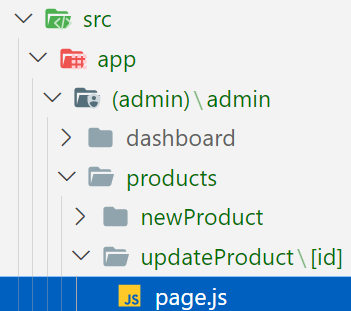
</div>

  )

}

export default Listproducts

Ensuite, sous /src/app/(admin)/admin/products créer le dossier updateProduct puis le dossier [id]. Dans lequel créer page.js



import {fetchSCategories} from "@/services/ScategorieService"

import {fetchArticleById} from "@/services/ArticleService"

import UpdateProduct from '@/components/admin/updateProductComponent';

const getscategories=async()=>{

    const data=await fetchSCategories()

    return data;

}

const getarticle=async(id)=>{

  const data=await fetchArticleById(id)

  return data;

}

const ProductUpdatePage = async({params}) => {

  const scategories = await getscategories();

  const article = await getarticle(params.id)

  return (

    <div>

       <UpdateProduct article={article} scategories={scategories} />

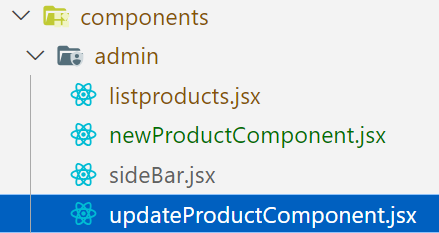
    </div>

  )

}

export default ProductUpdatePage

Ensuite, créer le composant updateProductComponent.jsx sous /src/components/admin



'use client';

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

import { useRouter } from 'next/navigation';

import Button from 'react-bootstrap/Button';

import Form from 'react-bootstrap/Form';

import Col from 'react-bootstrap/Col';

import InputGroup from 'react-bootstrap/InputGroup';

import Row from 'react-bootstrap/Row';

import axios from 'axios';

import { FilePond,registerPlugin } from 'react-filepond'

import 'filepond/dist/filepond.min.css';

import FilePondPluginImageExifOrientation from 'filepond-plugin-image-exif-orientation'

import FilePondPluginImagePreview from 'filepond-plugin-image-preview'

import 'filepond-plugin-image-preview/dist/filepond-plugin-image-preview.css'

registerPlugin(FilePondPluginImageExifOrientation, FilePondPluginImagePreview)

import {editArticle} from "@/services/ArticleService"

const UpdateProduct = ({article,scategories}) => {

  const [files, setFiles] = useState([]);

  const router = useRouter()

  const [reference, setReference] = useState("");

  const [designation, setDesignation] = useState("");

  const [prix, setPrix] = useState("");

  const [marque, setMarque] = useState("");

  const [qtestock, setQtestock] = useState("");

  const [imageart, setImageart] = useState("");

  const [scategorieID, setScategorieID] = useState("");

  const [validated, setValidated] = useState(false);

  useEffect(() => {

      setReference(article.reference)

      setDesignation(article.designation)

      setPrix(article.prix)

      setMarque(article.marque)

      setQtestock(article.qtestock)

      setImageart(article.imageart)

      setScategorieID(article.scategorieID)

      setFiles( [

        {

          source: article.imageart,

          options: { type: 'local' }

        }

        ])

   }, [article]);

  const handleSubmit = async(event) => {

    event.preventDefault();

    const form = event.currentTarget;

    if (form.checkValidity() === true) {

      const articleEdited={

        \_id:article.\_id,

        reference,

        designation,

        marque,

        prix,

        qtestock,

        imageart,

        scategorieID

     }

      await editArticle(articleEdited)

      .then(res=>{

      router.push("/admin/products")

      router.refresh()

      })

      .catch(error=>{

      console.log(error)

      alert("Erreur ! Modification non effectuée")

      })

      }

      setValidated(true);

      };

      const serverOptions = () => { console.log('server pond');

      return {

        load: (source, load, error, progress, abort, headers) => {

            var myRequest = new Request(source);

            fetch(myRequest).then(function(response) {

              response.blob().then(function(myBlob) {

                load(myBlob);

              });

            });

          },

        process: (fieldName, file, metadata, load, error, progress, abort) => {

            console.log(file)

          const data = new FormData();

          data.append('file', file);

          data.append('upload\_preset', 'Ecommerce\_cloudinary');

          data.append('cloud\_name', 'iset-sfax');

          data.append('public\_id', file.name);

          axios.post('https://api.cloudinary.com/v1\_1/iset-sfax/image/upload', data)

            .then((response) => response.data)

            .then((data) => {

              console.log(data);

             setImageart(data.url) ;

              load(data);

            })

            .catch((error) => {

              console.error('Error uploading file:', error);

              error('Upload failed');

              abort();

            });

        },

      };

    };

  return (

    <div>

    <Form noValidate validated={validated} onSubmit={handleSubmit}>

    <h2>Modification Produit</h2>

<div className="container w-100 d-flex justify-content-center">

<div>

<div className='form mt-3'>

<Row className="mb-2">

<Form.Group as={Col} md="6" >

<Form.Label >Référence \*</Form.Label>

<Form.Control

required

type="text"

placeholder="Référence"

value={reference}

onChange={(e)=>setReference(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Saisir Référence Article

</Form.Control.Feedback>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Désignation \*</Form.Label>

<Form.Control

required

type="text"

placeholder="Désignation"

value={designation}

onChange={(e)=>setDesignation(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Saisir Désignation

</Form.Control.Feedback>

</Form.Group>

</Row>

<Row className="mb-2">

<Form.Group className="col-md-6">

<Form.Label>Marque \*</Form.Label>

<InputGroup hasValidation>

<Form.Control

type="text"

required

placeholder="Marque"

value={marque}

onChange={(e)=>setMarque(e.target.value)}

/>

<Form.Control.Feedback type="invalid">

Marque Incorrecte

</Form.Control.Feedback>

</InputGroup>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Prix</Form.Label>

<Form.Control

type="number"

placeholder="Prix"

value={prix}

onChange={(e)=>setPrix(e.target.value)}

/>

</Form.Group>

</Row>

<Row className="mb-3">

<Form.Group className="col-md-6 ">

<Form.Label>

Qté stock<span className="req-tag">\*</span>

</Form.Label>

<Form.Control

required

type="number"

value={qtestock}

onChange={(e)=>setQtestock(e.target.value)}

placeholder="Qté stock"

/>

<Form.Control.Feedback type="invalid">

Qté stock Incorrect

</Form.Control.Feedback>

</Form.Group>

<Form.Group as={Col} md="6">

<Form.Label>Image</Form.Label>

<div style={{ width: "80%", margin: "auto", padding: "1%" }}>

     <FilePond

                   files={files}

                   acceptedFileTypes="image/\*"

                   onupdatefiles={setFiles}

                   allowMultiple={false}

                   server={serverOptions()}

                   name="file"

          />

    </div>

</Form.Group>

<Form.Group as={Col} md="12">

<Form.Label>S/Catégorie</Form.Label>

<Form.Control

as="select"

type="select"

value={scategorieID}

onChange={(e)=>setScategorieID(e.target.value)}

>

<option></option>

{scategories?.map((scat)=><option key={scat.\_id}

value={scat.\_id}>{scat.nomscategorie}</option>)}

</Form.Control>

</Form.Group>

</Row>

</div>

</div>

</div>

<Button type="submit">Valider</Button>

</Form>

 </div>

  )

}

export default UpdateProduct

