Crud redux articles

article.route.js dans le backend Node.js

```
const express = require('express');
const router = express.Router();
const Article=require("../models/article")


// chercher un article par s/cat
router.get('/scat/:scategorieID',async(req, res)=>{
    try {
        const art = await Article.find({ scategorieID:
req.params.scategorieID}).exec();

        res.status(200).json(art);
    } catch (error) {
        res.status(404).json({ message: error.message });
    }
});

// afficher la liste des articles.
router.get('/', async (req, res, )=> {
    try {
        const articles = await Article.find({}, null, {sort: {'_id': -
1}}).populate("scategorieID").exec();

        res.status(200).json(articles);
    } catch (error) {
        res.status(404).json({ message: error.message });
    }

});
// afficher la liste des articles par page
router.get('/productspage', async(req, res) => {
    const { page, pagesize } = req.query;

    // Calculez le nombre d'éléments à sauter (offset)
    const offset = (page - 1) * pagesize;
    try {
    // Effectuez la requête à votre source de données en utilisant les paramètres
de pagination
    const articles = await Article.find( {}, null, {sort: {'_id': -1}})
      .skip(offset)
      .limit(pagesize)


      res.status(200).json(articles);
```

```
 } catch (error) {
   res.status(404).json({ message: error.message });
 }
 });

// créer un nouvel article
router.post('/', async (req, res) =>  {

 const nouvarticle = new Article(req.body)
  try {
    const response =await nouvarticle.save();
    const articles = await
Article.findById(response._id).populate("scategorieID").exec();

    res.status(200).json(articles);
  } catch (error) {
    res.status(404).json({ message: error.message });
  }


});
// chercher un article
router.get('/:articleId',async(req, res)=>{
  try {
    const art = await Article.findById(req.params.articleId);

    res.status(200).json(art);
  } catch (error) {
    res.status(404).json({ message: error.message });
  }
});
// modifier un article


router.put('/:articleId', async (req, res)=> {
  try {
  const art = await Article.findByIdAndUpdate(
    req.params.articleId,
    { $set: req.body },
   { new: true }
  );
  const articles = await
Article.findById(art._id).populate("scategorieID").exec();

  res.status(200).json(articles);
  } catch (error) {
  res.status(404).json({ message: error.message });
  }
});
```

```
// Supprimer un article
router.delete('/:articleId', async (req, res)=> {
    const  id  = req.params.articleId;
    try {
    await Article.findByIdAndDelete(id);

    res.status(200).json({ message: "article deleted successfully." });
   } catch (error) {
    res.status(404).json({ message: error.message });
    }
});




module.exports = router;
```

Déterminer l'URL du git repository

```
git config --get remote.origin.url
```

update git :

```
git add .

git commit -a -m "Commit new version"

git push --set-upstream origin main
```

Partie ReactJS

1- Installer fontAwesome

```
npm i @fortawesome/fontawesome-free
```

Importer fontawesome dans le fichier App.js

```
import '@fortawesome/fontawesome-free/css/all.min.css';
```

2- Créer sous le dossier src le dossier admin/components
   Créer le dossier articles sous le dossier articles
3- Créer le composant ProductsAppAdmin sous le dossier articles

```
import React,{useEffect} from 'react'
import { useDispatch } from "react-redux";
import {getArticles} from "../../../features/articleSlice";


import Createarticle from './Createarticle';
import AfficheAerticleTable from './AfficheAerticleTable';


const ProductsAppAdmin = () => {

  const dispatch = useDispatch();

   useEffect(() => {
      dispatch(getArticles());
        },[])



  return (
    <div>
      <Createarticle/>
 <AfficheAerticleTable />

 </div>
  )
}

export default ProductsAppAdmin
```

4- Créer le composant AfficheAerticleTable

```
import React,{useState,useMemo} from 'react'
import ReactLoading from 'react-loading';
import {useSelector} from "react-redux"
import Button from 'react-bootstrap/Button';
import {  toast } from 'react-toastify';
import EditArticle from './EditArticle';
import {delArticle} from "../../../features/articleSlice"
import {useDispatch} from "react-redux";
import { MaterialReactTable } from 'material-react-table';
import { Box } from '@mui/material';
const AfficheAerticleTable = () => {

    const dispatch=useDispatch()
    const [showModal, setShowModal] = useState(false);

      const {articles,isLoading,error} = useSelector((state)=>state.storearticles);
      const [selectedItem, setSelectedItem] = useState(null);
     const handleClose= () => {
        setShowModal(false);
        setSelectedItem(null);
      }

      const handleDelete=(id,ref)=>{
        if(window.confirm("supprimer Article O/N"))  {
           dispatch(delArticle(id));
           toast(`Article ${ref} Supprimé`, {
            position: "top-right",
            autoClose: 5000,
            hideProgressBar: false,
            closeOnClick: true,
            pauseOnHover: true,
            draggable: true,
            progress: undefined,
            })
         }
    }



      const handleEdit = (item) => {
        setShowModal(true);
        setSelectedItem(item);

      };

      const columns = useMemo(
        () => [
            {
            accessorKey: 'imageart', //access nested data with dot notation
            header: 'Image',
            Cell: ({ cell}) => (
                <Box
                  sx={{
                    display: 'flex',
                    alignItems: 'center',
                    gap: '1rem',
                  }}
                >
                  <img
                    alt=""
                    height={60}
                    src={cell.getValue()}
                    loading="lazy"
                    style={{ borderRadius: '20%' }}
                  />


                </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={() => handleEdit(cell.row.original)}
                    size="md"
                    className="text-warning btn-link edit"
                  >
                   <i class="fa-solid fa-pen-to-square"></i>
                  </Button>
                  <Button

                    onClick={(e) => {

                      handleDelete(cell.row.original._id,cell.row.original.referenc
e);
                      }}
                    size="md"
                    className="text-danger btn-link delete"
                  >
                  <i className="fa fa-trash" />
                  </Button>


                </div>
              ),
            },


        ],
        [articles],
      );

      if (isLoading) return  <center><ReactLoading type='spokes' color="red" 
height={'8%'} width={'8%'}  /></center>
      if (error) return <p>Impossible d'afficher la liste des articles...</p>
  return (
    <>
    <div>

   <MaterialReactTable columns={columns} data={articles} />;
    </div>
    {showModal && (
        <EditArticle
          show={showModal}
          handleClose={handleClose}
          art={selectedItem}
        />
      )}

    </>

  )
}

export default AfficheAerticleTable
```

Ajouter une route pour le composant ProductsAppAdmin dans le fichier App.js

```
import ProductsAppAdmin from './admin/components/articles/ProductsAppAdmin';
import './App.css';
import Listarticles from './components/articles/Listarticles';
import NavScroll from './components/NavScroll';
import { BrowserRouter as Router,Routes,Route } from 'react-router-dom';
function App() {
  return (
    <div className="App">
      <Router>
      <NavScroll/>
      <Routes>
        <Route path='/articlescard' element={<Listarticles/>}/>
        <Route path='/articlesadmin' element={<ProductsAppAdmin/>}/>


      </Routes>
      </Router>
      </div>
  );
}

export default App;
```

Dans le composant Navscroll.js on ajoute l'appel au composant ProductsAppAdmin

```
 <Nav.Link as={Link} to="/articlesadmin">Artiles Admin</Nav.Link>
```

Création d'un nouvel article

1- Installer filepond :

```
npm i filepond react-filepond filepond-plugin-image-exif-orientation filepond-plugin-image-preview
```

2- créer le fichier firebaseConfig.js dans le dossier src

```
import { initializeApp } from "firebase/app";
import { getStorage } from "firebase/storage";
 // Initialize Firebase
```

```
const app = initializeApp ({
  apiKey: "AIzaSyBlWs2Aqc7t9xIHdjsTyddFlC2nXs76tU4",
  authDomain: "gestcom-4a752.firebaseapp.com",
  projectId: "gestcom-4a752",
  storageBucket: "gestcom-4a752.appspot.com",
  messagingSenderId: "676737216571",
  appId: "1:676737216571:web:6655dcd90e681d36f3ac1e"

});
 // Firebase storage reference
const storage = getStorage(app);
export default storage;
```

3- Créer le dossier utils sous le dossier src
4- Créer sous le dossier src le fichier UploadFirebase

```
import  storage from "../firebaseConfig";
import { ref, uploadBytesResumable, getDownloadURL } from "firebase/storage";

export const UploadFirebase=async(file)=>{
    let imageurl=""

    const storageRef = ref(storage, `/files/${file.name}`);

    const uploadTask = uploadBytesResumable(storageRef, file);

    await new Promise(resolve => {
    uploadTask.on(
        "state_changed",
        (snapshot) => {
            const percent = Math.round(
                (snapshot.bytesTransferred / snapshot.totalBytes) * 100
            );

           console.log(percent);
        },
        (err) => console.log(err),
        async () => {
                await getDownloadURL(uploadTask.snapshot.ref).then((url) => {

                imageurl=url;
                console.log(imageurl);
                resolve();
            });
        }
    );
}
)
    return imageurl;

}
```

5- Ajouter le composant articles/Createarticle.js sous le dossier admin

```
import React, { useState, useEffect } from 'react';
import Button from 'react-bootstrap/Button';
import Form from 'react-bootstrap/Form';
import Modal from 'react-bootstrap/Modal';
import Col from 'react-bootstrap/Col';
import InputGroup from 'react-bootstrap/InputGroup';
import Row from 'react-bootstrap/Row';
import {useDispatch,useSelector} from "react-redux";
import {createArticle} from "../../../features/articleSlice"
import {getScategories} from "../../../features/scategorieSlice";

import {UploadFirebase} from '../../../utils/UploadFirebase';

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 Createarticle = () => {
    const [file, setFile] = useState("");

const [show, setShow] = useState(false);
const handleClose = () => setShow(false);
const handleShow = () => setShow(true);
const [validated, setValidated] = useState(false);
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 dispatch = useDispatch();
const {scategories,isLoading} = useSelector((state) =>state.storescategories);

useEffect(() => {
dispatch(getScategories());
},[dispatch]);

const handleSubmit = (url) => {

  const article={
  reference: reference,
  designation: designation,
  prix: prix,
  marque: marque,
  qtestock: qtestock,
  imageart: url,
  scategorieID: scategorieID
  }
  dispatch(createArticle(article))
  .then(res=>{
  console.log("Insert OK",res);
  setReference("");
  setDesignation("");
  setPrix("");
  setMarque("");
  setQtestock("");
  setImageart("");
  setScategorieID("");
  setValidated(false);

  setFile("")

  handleClose()

})
.catch(error=>{
  console.log(error)
  alert("Erreur ! Insertion non effectuée")
  })

}

  const handleUpload = (event) => {
    event.preventDefault();
    const form = event.currentTarget;
   if (form.checkValidity() === true) {

       if(file){
          if (!file[0].file) {
              alert("Please upload an image first!");
          }
          else {
            console.log(file[0].file)
            resultHandleUpload(file[0].file,event);
        }
       } else {
          alert("Please upload an image first!");
      }

   setValidated(true);
  };
  }

  const resultHandleUpload = async(file) => {

    try {

    const url =  await UploadFirebase(file);


    handleSubmit(url)
   } catch (error) {
      console.log(error);
   }

  }

  return (
    <>
    <Button


        onClick={handleShow}
        variant="success"
        size="sm"
         style={{'margin':10,'left':10,fontFamily:'Arial'}}
      >
      <i className="fa-solid fa-circle-plus"></i>
      &nbsp;
      Nouveau
      </Button>
    <Modal show={show} onHide={handleClose}>
    <Form noValidate validated={validated} onSubmit={handleUpload}>
    <Modal.Header closeButton>
    <Modal.Title> <h1 align="center">Ajout Article</h1></Modal.Title>
    </Modal.Header>
    <Modal.Body>
    <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>
  <FilePond
                files={file}
                allowMultiple={false}
                onupdatefiles={setFile}
                labelIdle='<span className="filepond--label-action">Browse
One</span>'

              />
  </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>
  {!isLoading? scategories.map((scat)=><option key={scat._id}
  value={scat._id}>{scat.nomscategorie}</option>):null}
  </Form.Control>
  </Form.Group>
  </Row>
  </div>
  </div>
  </div>
  </Modal.Body>
  <Modal.Footer>
  <Button variant="secondary" onClick={handleClose}>
  Fermer
  </Button>
  <Button type="submit">Enregistrer</Button>
  </Modal.Footer>
  </Form>
  </Modal>
  </>

  )
}
export default Createarticle
```

Remarque: pour pouvoir afficher le nouvel article au début du tableau, changer l'action `createArticle` `dans` le fichier articleSlice.js

```
.addCase(createArticle.fulfilled, (state, action) => {


  //state.articles.push(action.payload);
  state.articles =
    [ action.payload,...state.articles];
  state.isLoading=false;
  state.error=null;
  state.success=action.payload;
})
```

Une autre solution consiste à remplacer push par unshift.

6- Appel du composant Createarticle.js dans le composant ProductAppAdmin

```
import Createarticle from './Createarticle';
```

```
return (
    <div>
 <Createarticle/>
 <AffichearticlesAdmin />
</div>
```

```
  )
}
```

React-toastify

https://www.npmjs.com/package/react-toastify

installer react-toastify

```
npm I react-toastify
```

fichier index.js importer le fichier css

```
import 'react-toastify/dist/ReactToastify.css';
```

dans le fichier App.js

```
import ProductsAppAdmin from './admin/components/articles/ProductsAppAdmin';
import './App.css';
import Listarticles from './components/articles/Listarticles';
import NavScroll from './components/NavScroll';
import { ToastContainer } from 'react-toastify';
import { BrowserRouter as Router,Routes,Route } from 'react-router-dom';
function App() {
  return (
    <div className="App">
      <ToastContainer/>
      <Router>

      <NavScroll/>

      <Routes>

        <Route path='/articlescard' element={<Listarticles/>}/>
        <Route path='/articlesadmin' element={<ProductsAppAdmin/>}/>


      </Routes>

      </Router>
      </div>
  );
}

export default App;
```

dans le composant Createarticle.js

```
import { ToastContainer } from 'react-toastify';
```

Dans la méthode hundlesubmit, on ajoute toast

```
const handleSubmit = (url) => {

  const article={
  reference: reference,
  designation: designation,
  prix: prix,
  marque: marque,
  qtestock: qtestock,
  imageart: url,
  scategorieID: scategorieID
  }
  dispatch(createArticle(article))

  .then(res=>{


  setReference("");
  setDesignation("");
  setPrix("");
  setMarque("");
  setQtestock("");
  setImageart("");
  setScategorieID("");
  setValidated(false);

  setFile("")

  handleClose()
  toast("Article ajouté", {
    position: "top-right",
    autoClose: 5000,
    hideProgressBar: false,
    closeOnClick: true,
    pauseOnHover: true,
    draggable: true,
    progress: undefined,
    })
})
.catch(error=>{
  console.log(error)
  alert("Erreur ! Insertion non effectuée")
  })

}
```

Modification d’un article

Créer le composant EditArticle.js dans le dossier admin/articles

```
import React, { useState,useEffect } 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 Modal from 'react-bootstrap/Modal';
import {updateArticle} from "../../../features/articleSlice"
import {getScategories} from "../../../features/scategorieSlice";
import {useDispatch,useSelector} from "react-redux";
import {UploadFirebase} from '../../../utils/UploadFirebase';

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 EditArticle = ({art,show,handleClose}) => {


    const {scategories} = useSelector((state) =>state.storescategories);
    const [file, setFile] = 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 dispatch = useDispatch()



 useEffect(()=>{

 setReference(art.reference);
  setDesignation(art.designation);
  setPrix(art.prix);
  setScategorieID(art.scategorieID._id);
  console.log(art.scategorieID._id)
 setMarque(art.marque);
 setImageart(art.imageart)

 setFile([
  {
    source: art.imageart,
    options: { type: "local" }
  }
]);
 setQtestock(art.qtestock);


 },[art])
 useEffect(() => {
    dispatch(getScategories());
    },[dispatch]);



    const handleSubmit = (url) => {

        const article={
            ...art,
        reference: reference,
        designation: designation,
        prix: prix,
        marque: marque,
        qtestock: qtestock,
        imageart: url,
        scategorieID: scategorieID
        }
        dispatch(updateArticle(article))
        .then(res=>{
    })
    .catch(error=>{
      console.log(error)
      alert("Erreur ! Insertion non effectuée")
      })

    }

      const handleUpload = (event) => {
        event.preventDefault();
        const form = event.currentTarget;
       if (form.checkValidity() === true) {

              if (!file) {
                  const url = imageart;
                  handleClose()

                 handleSubmit(url)
                }
              else {

                resultHandleUpload(file[0].file,event);
            }


       setValidated(true);
      };
      }

      const resultHandleUpload = async(file) => {

        try {

        const url =  await UploadFirebase(file);
        handleClose()

        handleSubmit(url)
       } catch (error) {

       }

      }
return (
  <div>



<Modal show={show} onHide={handleClose}>
```

```
   <Form noValidate validated={validated} onSubmit={handleUpload}>
<Modal.Header closeButton>
    <h2>Modification Product</h2>
</Modal.Header>
<Modal.Body>
<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>

 <FilePond
        files={file}
        allowMultiple={false}
        onupdatefiles={setFile}
        labelIdle='<span className="filepond--label-action">Browse
One</span>'
        server={{
         load: (source, load, error, progress, abort, headers) => {
          var myRequest = new Request(source);
          fetch(myRequest).then(function(response) {
           response.blob().then(function(myBlob) {
            load(myBlob);
           });
          });
         }
        }}
```

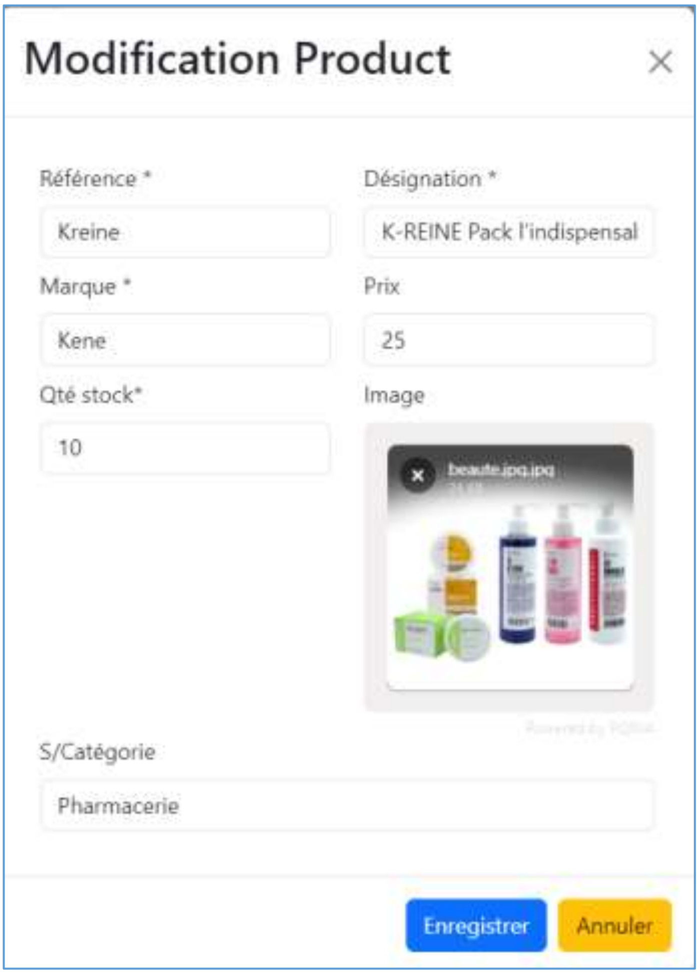
```
              />
  </Form.Group>

<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>
</Modal.Body>
<Modal.Footer>
<Button type="submit">Enregistrer</Button>
<Button type="button" className="btn btn-warning" onClick={handleClose}
>Annuler</Button>
</Modal.Footer>
</Form>

</Modal>
    </div>

  )
}

export default EditArticle
```

Composant AffichearticlesAdmin.js

```
import React,{useState} from 'react'
import ReactLoading from 'react-loading';
import {useSelector} from "react-redux"
import Button from 'react-bootstrap/Button';
import EditArticle from './EditArticle';
```

```
const AffichearticlesAdmin = () => {

  const [showModal, setShowModal] = useState(false);

    const {articles,isLoading,error} = useSelector((state)=>state.storearticles);
    const [selectedItem, setSelectedItem] = useState(null);
   const handleClose= () => {
      setShowModal(false);
      setSelectedItem(null);
    }

const handleEdit = (item) => {
      setSelectedItem(item);
      setShowModal(true);
    };

    if (isLoading) return  <center><ReactLoading type='spokes' color="red"
height={'8%'} width={'8%'}  /></center>
    if (error) return <p>Impossible d'afficher la liste des articles...</p>


    return (
        <>

        <div style={{overflowX: "auto"}}>

    <div className="table_responsive">
    <table>
      <thead>
        <tr>
          <th>Image</th>
          <th>Référence</th>
          <th>Désignation</th>
          <th>Quantité</th>
          <th>Prix</th>
          <th>Action</th>
        </tr>
      </thead>

      <tbody>
      {articles.map((art,ind)=>
        <tr key={ind}>

          <td><img src={art.imageart} alt=""/></td>
          <td>{art.reference}</td>
          <td>{art.designation}</td>
          <td>{art.qtestock}</td>
          <td>{art.prix}</td>
          <td>
            <span className="action_btn">
            <Button

                     onClick={() => handleEdit(art)}
                     size="md"
                     className="text-warning btn-link edit"
                   >
                    <i class="fa-solid fa-pen-to-square"></i>
                   </Button>
            <Button

                    onClick={(e) => {

                    }}

                    size="md"
                    className="text-danger btn-link delete"
                  >
                  <i className="fa fa-trash" />
                  </Button>

            </span>
          </td>
        </tr>
)}


      </tbody>
    </table>

  </div>


</div>
{showModal && (
        <EditArticle
          show={showModal}
          handleClose={handleClose}
          art={selectedItem}
        />
      )}
</>
  )
}
```

```
export default AfficheArticlesAdmin
```



Modification Product

Référence * Kreine

Désignation * K-REINE Pack l'indispensal

Marque * Kene

Prix 25

Qté stock* 10

Image beaute.jpg.jpg

S/Catégorie Pharmacerie

• Suppression d’un article

Ouvrir le composant `AfficheArticlesAdmin`.js

```
import React,{useState} from 'react'
import ReactLoading from 'react-loading';
import {useSelector} from "react-redux"
import Button from 'react-bootstrap/Button';
import EditArticle from './EditArticle';
import {delArticle} from "../../../features/articleSlice"
import {useDispatch} from "react-redux";

const AfficheArticlesAdmin = () => {
const dispatch=useDispatch()
```

```
  const [showModal, setShowModal] = useState(false);

    const {articles,isLoading,error} = useSelector((state)=>state.storearticles);
    const [selectedItem, setSelectedItem] = useState(null);
   const handleClose= () => {
      setShowModal(false);
      setSelectedItem(null);
    }

    const handleDelete=(id)=>{
      if(window.confirm("supprimer Article O/N"))  {
         dispatch(delArticle(id));

       }
  }

    if (isLoading) return  <center><ReactLoading type='spokes' color="red"
height={'8%'} width={'8%'}  /></center>
    if (error) return <p>Impossible d'afficher la liste des articles...</p>

    const handleEdit = (item) => {
      setSelectedItem(item);
      setShowModal(true);
    };

    return (
        <>

        <div style={{overflowX: "auto"}}>

    <div className="table_responsive">
    <table>
      <thead>
        <tr>
          <th>Image</th>
          <th>Référence</th>
          <th>Désignation</th>
          <th>Quantité</th>
          <th>Prix</th>
          <th>Action</th>
        </tr>
      </thead>

      <tbody>
      {articles.map((art,ind)=>
        <tr key={ind}>

          <td><img src={art.imageart} alt=""/></td>
          <td>{art.reference}</td>
          <td>{art.designation}</td>
          <td>{art.qtestock}</td>
          <td>{art.prix}</td>
          <td>
            <span className="action_btn">
            <Button

                    onClick={() => handleEdit(art)}
                    size="md"
                    className="text-warning btn-link edit"
                  >
                   <i class="fa-solid fa-pen-to-square"></i>
                  </Button>
            <Button

                    onClick={(e) => {

                      handleDelete(art._id);
                      }}


                    size="md"
                    className="text-danger btn-link delete"
                  >
                  <i className="fa fa-trash" />
                  </Button>

            </span>
          </td>
        </tr>
)}


      </tbody>
    </table>

  </div>


</div>
{showModal && (
        <EditArticle
          show={showModal}
          handleClose={handleClose}
          art={selectedItem}
        />
      )}
</>
  )
```

```
}

export default AffichearticlesAdmin
```

Dans le composant AfficheArticleAdmin.js, on ajoute toast lors de la suppression d’un article

```
 <Button

                    onClick={(e) => {

                      handleDelete(art._id,art.reference);
                      }}


                    size="md"
                    className="text-danger btn-link delete"
                  >
                  <i className="fa fa-trash" />
                  </Button>
```

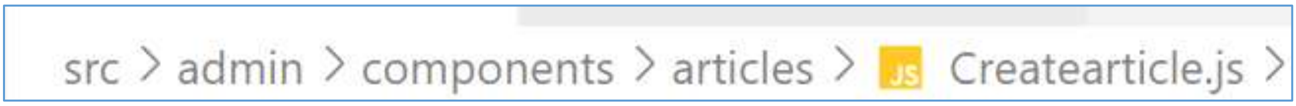
- Méthode handleDelete

```
const handleDelete=(id,ref)=>{
      if(window.confirm("supprimer Article O/N"))  {
         dispatch(delArticle(id));
         toast(`Article ${ref} Supprimé`, {
          position: "top-right",
          autoClose: 5000,
          hideProgressBar: false,
          closeOnClick: true,
          pauseOnHover: true,
          draggable: true,
          progress: undefined,
          })
        }
  }
```

Progress Bar

https://www.npmjs.com/package/rc-progress

```
npm i rc-progress
```



src > admin > components > articles > JS Createarticle.js >

```
import React, { useState, useEffect } from 'react';
import Button from 'react-bootstrap/Button';
import Form from 'react-bootstrap/Form';
import Modal from 'react-bootstrap/Modal';
import Col from 'react-bootstrap/Col';
import InputGroup from 'react-bootstrap/InputGroup';
import Row from 'react-bootstrap/Row';
import {useDispatch,useSelector} from "react-redux";
import {createArticle} from "../../../features/articleSlice"
import {  toast } from 'react-toastify';
import {getScategories} from "../../../features/scategorieSlice";

import {UploadFirebase} from '../../../utils/UploadFirebase';
import { Line } from 'rc-progress';

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 Createarticle = () => {

  const [uploadProgress, setUploadProgress] = useState(0);

    const [file, setFile] = useState("");

const [show, setShow] = useState(false);
const handleClose = () => setShow(false);
const handleShow = () => setShow(true);
const [validated, setValidated] = useState(false);
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 dispatch = useDispatch();
const {scategories,isLoading} = useSelector((state) =>state.storescategories);

useEffect(() => {
dispatch(getScategories());
},[dispatch]);
```

```
const handleSubmit = (url) => {

  const article={
  reference: reference,
  designation: designation,
  prix: prix,
  marque: marque,
  qtestock: qtestock,
  imageart: url,
  scategorieID: scategorieID
  }
  dispatch(createArticle(article))

  .then(res=>{


  setReference("");
  setDesignation("");
  setPrix("");
  setMarque("");
  setQtestock("");
  setImageart("");
  setScategorieID("");
  setValidated(false);

  setFile("")

  setUploadProgress(0)

  handleClose()
  toast("Article ajouté", {
    position: "top-right",
    autoClose: 5000,
    hideProgressBar: false,
    closeOnClick: true,
    pauseOnHover: true,
    draggable: true,
    progress: undefined,
    })

})
.catch(error=>{
  console.log(error)
  alert("Erreur ! Insertion non effectuée")
  })

}

  const handleUpload = (event) => {
```

```
  event.preventDefault();
  const form = event.currentTarget;
 if (form.checkValidity() === true) {

     if(file){
        if (!file[0].file) {
            alert("Please upload an image first!");
        }
        else {
          console.log(file[0].file)
          resultHandleUpload(file[0].file,event);
      }
     } else {
        alert("Please upload an image first!");
    }

 setValidated(true);
};
}

const resultHandleUpload = async(file) => {

  try {

  const url =  await UploadFirebase(file, setUploadProgress);

  handleSubmit(url)
 } catch (error) {
    console.log(error);
 }

}

return (
  <>
  <Button


      onClick={handleShow}
      variant="success"
      size="sm"
       style={{'margin':10,'left':10,fontFamily:'Arial'}}
    >
    <i className="fa-solid fa-circle-plus"></i>
    &nbsp;
    Nouveau
    </Button>
  <Modal show={show} onHide={handleClose}>
  <Form noValidate validated={validated} onSubmit={handleUpload}>
```

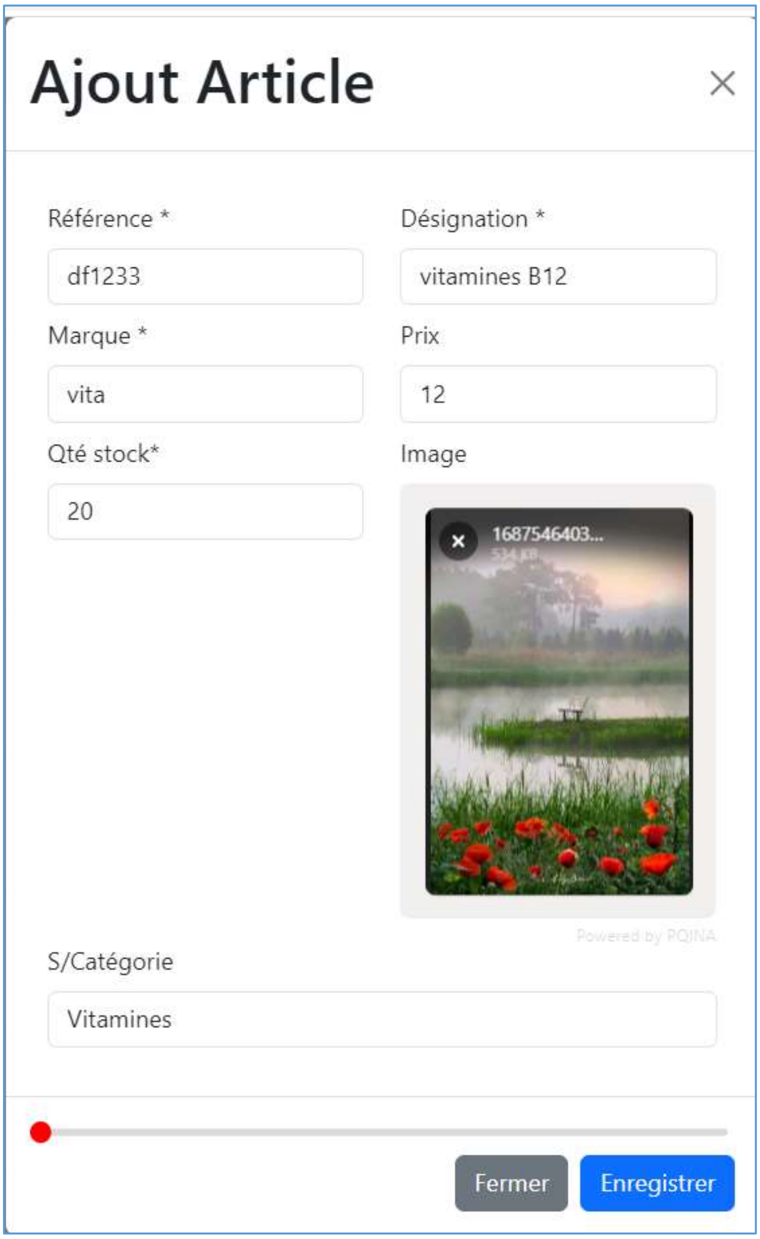
```
<Modal.Header closeButton>
<Modal.Title> <h1 align="center">Ajout Article</h1></Modal.Title>
</Modal.Header>
<Modal.Body>
<div className="container w-100 d-flex justify-content-center">
<div>

<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>
 <FilePond
        files={file}
        allowMultiple={false}
        onupdatefiles={setFile}
        labelIdle='<span className="filepond--label-action">Browse
One</span>'

       />

 </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>
  {!isLoading? scategories.map((scat)=><option key={scat._id}
  value={scat._id}>{scat.nomscategorie}</option>):null}
  </Form.Control>
  </Form.Group>
  </Row>
  </div>
  </div>
  </div>
  </Modal.Body>
  <Modal.Footer>
  <Line percent={uploadProgress} strokeWidth={3} strokeColor="#ff0000" />
  <Button variant="secondary" onClick={handleClose}>
  Fermer
  </Button>
  <Button type="submit">Enregistrer</Button>
  </Modal.Footer>
  </Form>
  </Modal>
  </>

  )
}

export default Createarticle
```

src > utils > JS UploadFirebase.js >

```
import  storage from "../firebaseConfig";
import { ref, uploadBytesResumable, getDownloadURL } from "firebase/storage";

export const UploadFirebase=async(file, setUploadProgress)=>{
    let imageurl=""

    const storageRef = ref(storage, `/files/${file.name}`);

    const uploadTask = uploadBytesResumable(storageRef, file);

    await new Promise(resolve => {
    uploadTask.on(
        "state_changed",
        (snapshot) => {
            const percent = Math.round(
                (snapshot.bytesTransferred / snapshot.totalBytes) * 100
            );
           setUploadProgress(percent);
                   },
        (err) => console.log(err),
        async () => {
                await getDownloadURL(uploadTask.snapshot.ref).then((url) => {

                imageurl=url;
                console.log(imageurl);
                resolve();
            });
        }
    );
}
)
    return imageurl;

}
```

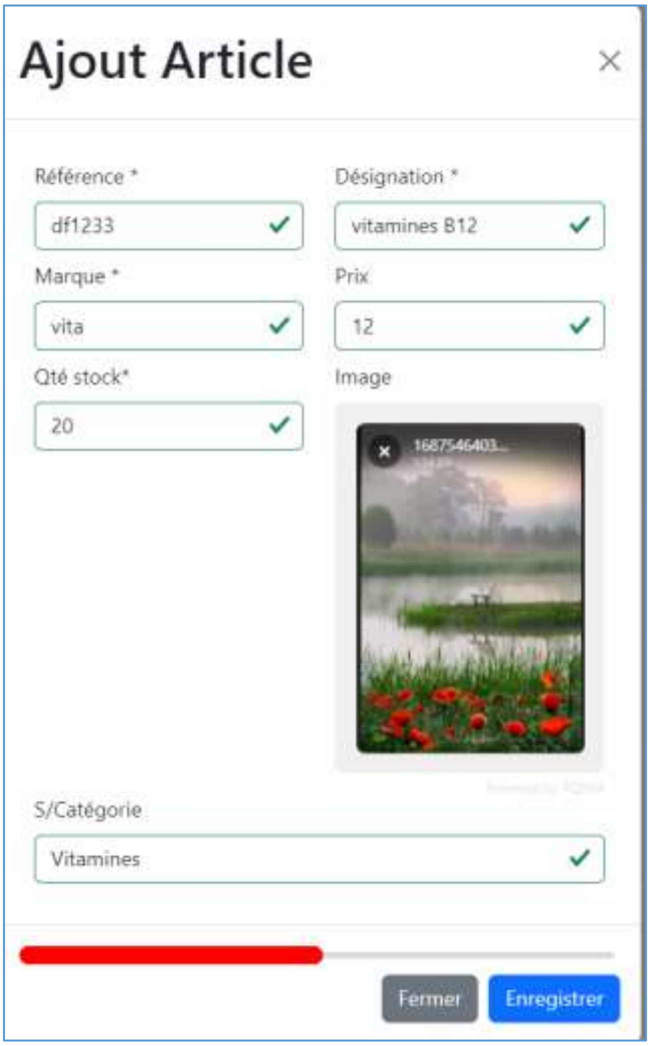


Ajout Article

Référence *	Désignation *
df1233	vitamines B12
Marque *	Prix
vita	12
Qté stock*	Image
20	

S/Catégorie

Vitamines



Ajout Article

Référence *
df1233

Désignation *
vitamines B12

Marque *
vita

Prix
12

Qté stock*
20

Image
1687546403...

S/Catégorie
Vitamines

src > admin > components > articles > JS EditArticle.js >

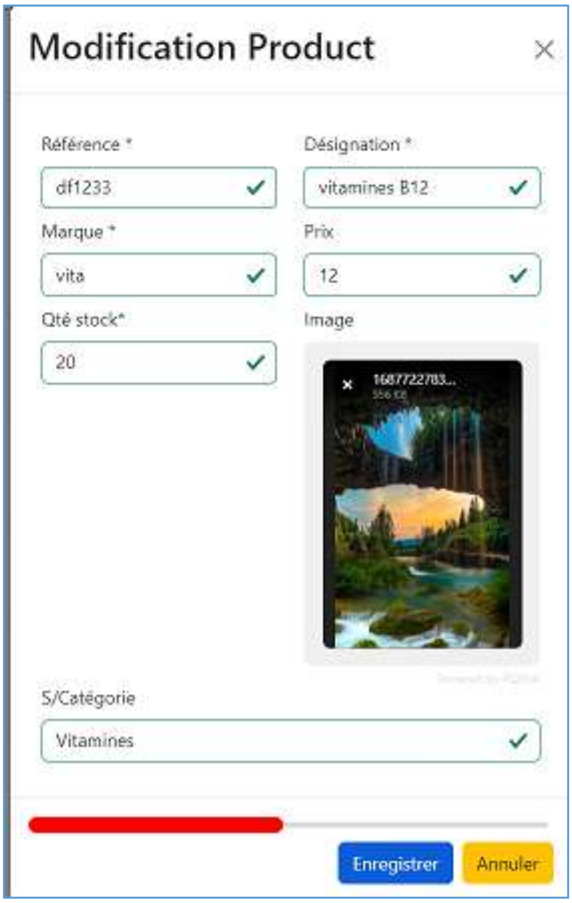
```
import React, { useState,useEffect } from 'react';
import Button from 'react-bootstrap/Button';
import Form from 'react-bootstrap/Form';
import Col from 'react-bootstrap/Col';
import InputGroup from 'react-bootstrap/InputGroup';
import Row from 'react-bootstrap/Row';
import Modal from 'react-bootstrap/Modal';
import {updateArticle} from "../../../features/articleSlice"
import {getScategories} from "../../../features/scategorieSlice";
import {useDispatch,useSelector} from "react-redux";

import {UploadFirebase} from '../../../utils/UploadFirebase';
import { Line } from 'rc-progress';

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 EditArticle = ({art,show,handleClose}) => {

  const [uploadProgress, setUploadProgress] = useState(0);

  console.log(show)
    const {scategories} = useSelector((state) =>state.storescategories);
    const [file, setFile] = 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 dispatch = useDispatch()



 useEffect(()=>{

 setReference(art.reference);
  setDesignation(art.designation);
  setPrix(art.prix);
  setScategorieID(art.scategorieID._id);
  console.log(art.scategorieID._id)
 setMarque(art.marque);
 setImageart(art.imageart)

 setFile([
  {
    source: art.imageart,
    options: { type: "local" }
  }
]);
 setQtestock(art.qtestock);


 },[art])
```

```
useEffect(() => {
   dispatch(getScategories());
   },[dispatch]);



   const handleSubmit = (url) => {

       const article={
           ...art,
       reference: reference,
       designation: designation,
       prix: prix,
       marque: marque,
       qtestock: qtestock,
       imageart: url,
       scategorieID: scategorieID
       }
       dispatch(updateArticle(article))
       .then(res=>{

         setUploadProgress(0)



     })
     .catch(error=>{
       console.log(error)
       alert("Erreur ! Insertion non effectuée")
       })

     }

       const handleUpload = (event) => {
         event.preventDefault();
         const form = event.currentTarget;
        if (form.checkValidity() === true) {

               if (!file) {
                   const url = imageart;
                   handleClose()

                  handleSubmit(url)
                 }
               else {

                 resultHandleUpload(file[0].file,event);
             }
        setValidated(true);
       };
       }

       const resultHandleUpload = async(file) => {

         try {

           const url =  await UploadFirebase(file, setUploadProgress);

           handleClose()

         handleSubmit(url)
        } catch (error) {

        }

       }
 return (
   <div>



 <Modal show={show} onHide={handleClose}>
    <Form noValidate validated={validated} onSubmit={handleUpload}>
 <Modal.Header closeButton>
     <h2>Modification Product</h2>
 </Modal.Header>
 <Modal.Body>
 <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>

  <FilePond
                files={file}
                allowMultiple={false}
                onupdatefiles={setFile}
                labelIdle='<span className="filepond--label-action">Browse
One</span>'
                server={{
                  load: (source, load, error, progress, abort, headers) => {
                    var myRequest = new Request(source);
                    fetch(myRequest).then(function(response) {
                      response.blob().then(function(myBlob) {
                        load(myBlob);
                      });
                    });
                  }
                }}
              />
  </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>
</Modal.Body>
```

```
<Modal.Footer>
<Line percent={uploadProgress} strokeWidth={3} strokeColor="#ff0000" />

<Button type="submit">Enregistrer</Button>
<Button type="button" className="btn btn-warning" onClick={handleClose}
>Annuler</Button>
</Modal.Footer>
</Form>

</Modal>
    </div>

  )
}

export default EditArticle
```



Modification Product

Référence * df1233

Désignation * vitamines B12

Marque * vita

Prix 12

Qté stock* 20

Image

1687722783...
556 KB

Powered by PQINA

S/Catégorie Vitamines