Redux Toolkit Panier

Référence : https://react-redux.js.org/

Dépendances *:*

```
npm install axios

npm install react-router-dom

npm install react-redux @reduxjs/toolkit

npm i bootstrap react-bootstrap

npm i react-toastify

npm install @mui/material @emotion/react @emotion/styled
```

Remarque : Ajouter l'option **--force** en cas ou l'installation échoue

Ajouter l'appel à bootstrap dans le fichier index.js

```
import 'bootstrap/dist/css/bootstrap.min.css';
```

Fichier src/Axios/Api.js

```
import axios from 'axios'

export default axios.create({
baseURL:"https://ecommercebackend2023j.vercel.app/api"
})
```

Création de services Frontend

Fichier src/services/ArticleService.js

```
import Api from "../Axios/Api";
```

```
const ARTICLE_API="/articles"
export const fetchArticles=async()=> {
return await Api.get(ARTICLE_API);
}
export const fetchArticleById=async(articleId)=> {
return await Api.get(ARTICLE_API + '/' + articleId);
}
export const deleteArticle=async(articleId) =>{
return await Api.delete(ARTICLE_API + '/' + articleId);
}
export const addArticle=async(article)=> {
return await Api.post(ARTICLE_API, article);
}
export const editArticle=(article) =>{
return Api.put(ARTICLE_API + '/' + article._id, article);
}
```

CategorieService.js

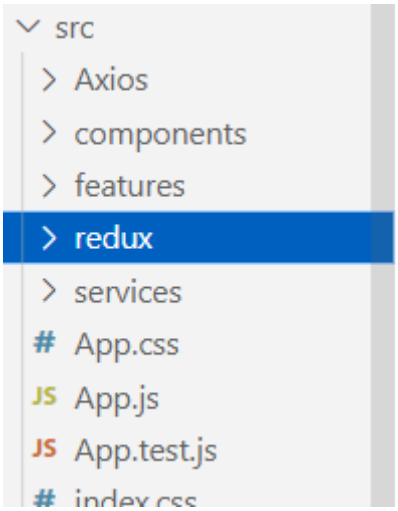
```
import Api from "../Axios/Api";
const CATEGORIE_API="/categories"
export const fetchCategories=async()=> {
return await Api.get(CATEGORIE_API);
}
export const fetchCategorieById=async(categorieId)=> {
return await Api.get(CATEGORIE_API + '/' + categorieId);
}
export const deleteCategorie=async(categorieId) =>{
return await Api.delete(CATEGORIE_API + '/' + categorieId);
}
export const addCategorie=async(categorie)=> {
return await Api.post(CATEGORIE_API,categorie);
}
export const editCategorie=(categorie) =>{
return Api.put(CATEGORIE_API + '/' + categorie._id, categorie);
}
```

ScategorieService.js

```
import Api from "../Axios/Api";
const SCATEGORIE_API="/scategories"
export const fetchSCategories=async()=> {
return await Api.get(SCATEGORIE_API);
}
export const fetchSCategorieById=async(scategorieId)=> {
return await Api.get(SCATEGORIE_API + '/' + scategorieId);
}
```

```
const fetchSCategorieByCat=(idcat) =>{return Api.get(SCATEGORIE_API + '/cat/' +
idcat);
}
export const deleteSCategorie=async(scategorieId) =>{
return await Api.delete(SCATEGORIE_API + '/' + scategorieId);
}
export const addSCategorie=async(scategorie)=> {
return await Api.post(SCATEGORIE_API,scategorie);
}
export const editSCategorie=(scategorie) =>{
return Api.put(SCATEGORIE_API + '/' + scategorie._id, scategorie);
}
```

Ajouter les dossiers features et redux comme indiqué ci-dessous



Créer les slices (Redux toolkit)

Pour chaque classe on crée les slices des Cruds

Fichier src/features/articleSlice.js

```
import { createSlice,createAsyncThunk } from '@reduxjs/toolkit'
import {fetchArticles,addArticle,deleteArticle,editArticle,fetchArticleById} from
"../services/ArticleService"

export const getArticles = createAsyncThunk(
    "article/getArticles",
    async (_, thunkAPI) => {
      const { rejectWithValue } = thunkAPI;
      try {
      const res = await fetchArticles();
```

```
    return res.data;
    }
    catch (error) {
      return rejectWithValue(error.message);
    }
  }
);

export const createArticle = createAsyncThunk(
  "article/createArticle",
  async (article, thunkAPI) => {
    const { rejectWithValue } = thunkAPI;
    try{
    const res= await addArticle(article);
    return res.data
  }
  catch (error) {
    return rejectWithValue(error.message);
  }
  }
);

export const delArticle = createAsyncThunk(
  "article/delArticle",
  async (id,thunkAPI) => {
    const { rejectWithValue } = thunkAPI;
    try{
    await deleteArticle(id);
    return  id ;
  }
  catch (error) {
    return rejectWithValue(error.message);
  }
  });

export const updateArticle = createAsyncThunk(
    "article/updateArticle",
    async (article, thunkAPI) => {
    const { rejectWithValue } = thunkAPI;
      try{
     const res= await editArticle(article);
      return res.data

    }
    catch (error) {
      return rejectWithValue(error.message);
    }

    }
```

```
    );

  export const findArticleByID = createAsyncThunk(
    "article/findArticleByID",
    async (id,thunkAPI) => {
      const { rejectWithValue } = thunkAPI;
      try{
      const res = await fetchArticleById(id);
      return res.data;
    }
    catch (error) {
      return rejectWithValue(error.message);
    }
    });
export const articleSlice = createSlice({
  name: 'article',
  initialState:{
    articles:[],
    article:{},
    isLoading: false,
    success:null,
    error:null,
  },


  extraReducers: (builder) => {
    //get articles
    builder
    .addCase(getArticles.pending, (state, action) => {

      state.isLoading=true;
      state.error=null;
    })
    .addCase(getArticles.fulfilled, (state, action) => {
      state.isLoading=false;
      state.error = null;
      state.articles=action.payload;
    })
    .addCase(getArticles.rejected, (state, action) => {
      state.isLoading=false;
      state.error=action.payload;
      console.log("impossible de se connecter au serveur")
    })

    //insertion article
    .addCase(createArticle.pending, (state, action) => {
      state.isLoading=true;
      state.error=null;
      state.success=null;
```

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

    state.articles.push(action.payload);
    state.isLoading=false;
    state.error=null;
    state.success=action.payload;
  })
  .addCase(createArticle.rejected, (state, action) => {
    state.isLoading=false;
    state.error=action.payload;
    state.success=null;
  })
  //Modification article
  .addCase(updateArticle.pending, (state, action) => {
    state.isLoading=true;
    state.error=null;
    state.success=null;
  })
  .addCase(updateArticle.fulfilled, (state, action) => {
    state.articles = state.articles.map((item) =>
    item._id === action.payload._id ? action.payload : item
  );
state.isLoading=false;
state.error=null;
state.success=action.payload;
   })
  //Delete article
  .addCase(delArticle.pending, (state, action) => {
    state.isLoading=true;
    state.error=null;
  })
  .addCase(delArticle.fulfilled, (state, action) => {
    state.isLoading=false;
    state.error=null;
    state.articles=state.articles.filter((item)=> item._id!==action.payload)

  })
  .addCase(delArticle.rejected, (state, action) => {
    state.isLoading=false;
    state.error=action.payload;
  })
//Fectch article
  .addCase(findArticleByID.pending, (state, action) => {
    state.isLoading = true
    state.error=null;

    })
  .addCase(
```

```
     findArticleByID.fulfilled,(state, action) => {
      state.isLoading = false
      state.error = null
      state.article=action.payload;
   })

  }

  }

)

export default articleSlice.reducer;
```

Fichier src/features/scategorieSlice.js

```
import { createSlice,createAsyncThunk } from '@reduxjs/toolkit'
import
{fetchSCategories,addSCategorie,deleteSCategorie,editSCategorie,fetchSCategorieById
} from "../services/ScategorieService"

export const getScategories = createAsyncThunk(
    "scategorie/getScategories",
    async (_, thunkAPI) => {
      const { rejectWithValue } = thunkAPI;
      try {
      const res = await fetchSCategories();
      return res.data;
      }
      catch (error) {
        return rejectWithValue(error.message);
      }
    }
  );

  export const createScategorie = createAsyncThunk(
    "scategorie/createScategorie",
    async (scategorie, thunkAPI) => {
      const { rejectWithValue } = thunkAPI;
      try{
      const res= await addSCategorie(scategorie);
      return res.data
    }
    catch (error) {
      return rejectWithValue(error.message);
```

```
    }
    }
  );

  export const deleteScategorie = createAsyncThunk(
    "scategorie/deleteScategorie",
    async (id,thunkAPI) => {
      const { rejectWithValue } = thunkAPI;
      try{
      await deleteSCategorie(id);
      return  id ;
    }
    catch (error) {
      return rejectWithValue(error.message);
    }
    });

  export const updateScategorie = createAsyncThunk(
      "scategorie/updateScategorie",
      async (scategorie, thunkAPI) => {
      const { rejectWithValue } = thunkAPI;
        try{
       const res= await editSCategorie(scategorie);
        return res.data

      }
      catch (error) {
        return rejectWithValue(error.message);
      }

      }
    );

  export const findScategorieByID = createAsyncThunk(
    "scategorie/findScategorieByID",
    async (id,thunkAPI) => {
      const { rejectWithValue } = thunkAPI;
      try{
      const res = await fetchSCategorieById(id);
      return res.data;
    }
    catch (error) {
      return rejectWithValue(error.message);
    }
    });
export const scategorieSlice = createSlice({
  name: 'scategorie',
  initialState:{
    scategories:[],
```

```
  scategorie:{},
  isLoading: false,
  success:null,
  error:null,
},


extraReducers: (builder) => {
  //get scategories
  builder
  .addCase(getScategories.pending, (state, action) => {

    state.isLoading=true;
    state.error=null;
  })
  .addCase(getScategories.fulfilled, (state, action) => {
    state.isLoading=false;
    state.error = null;
    state.scategories=action.payload;
  })
  .addCase(getScategories.rejected, (state, action) => {
    state.isLoading=false;
    state.error=action.payload;
    console.log("impossible de se connecter au serveur")
  })

  //insertion scategorie
  .addCase(createScategorie.pending, (state, action) => {
    state.isLoading=true;
    state.error=null;
    state.success=null;
  })
  .addCase(createScategorie.fulfilled, (state, action) => {

    state.scategories.push(action.payload);
    state.isLoading=false;
    state.error=null;
    state.success=action.payload;
  })
  .addCase(createScategorie.rejected, (state, action) => {
    state.isLoading=false;
    state.error=action.payload;
    state.success=null;
  })
  //Modification scategorie
  .addCase(updateScategorie.pending, (state, action) => {
    state.isLoading=true;
    state.error=null;
    state.success=null;
```

```
    })
    .addCase(updateScategorie.fulfilled, (state, action) => {
      state.scategories = state.scategories.map((item) =>
      item._id === action.payload._id ? action.payload : item
    );
  state.isLoading=false;
  state.error=null;
  state.success=action.payload;
     })
    //Delete scategorie
    .addCase(deleteScategorie.pending, (state, action) => {
      state.isLoading=true;
      state.error=null;
    })
    .addCase(deleteScategorie.fulfilled, (state, action) => {
      state.isLoading=false;
      state.error=null;
      state.scategories=state.scategories.filter((item)=>
item._id!==action.payload)

    })
    .addCase(deleteScategorie.rejected, (state, action) => {
      state.isLoading=false;
      state.error=action.payload;
    })
  //Fectch scategorie
    .addCase(findScategorieByID.pending, (state, action) => {
      state.isLoading = true
      state.error=null;

      })
    .addCase(
     findScategorieByID.fulfilled,(state, action) => {
      state.isLoading = false
      state.error = null
      state.scategorie=action.payload;
   })

  }

  }

)


export default scategorieSlice.reducer;
```

Créer le dossier redux sous src

Créer le Fichier src/redux/store.js


```
import { configureStore } from '@reduxjs/toolkit'
import articlesReducer from "../features/articleSlice"
import scategoriesReducer from "../features/scategorieSlice"

const store = configureStore({
  reducer: {
    storearticles:articlesReducer,
    storescategories: scategoriesReducer,
  }
})
export default store
```

Importer le store dans le fichier index.js

Fichier src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import 'bootstrap/dist/css/bootstrap.min.css';
import reportWebVitals from './reportWebVitals';
import store from './redux/store';
import { Provider } from 'react-redux' ;

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <Provider store={store}>

    <App />

    </Provider>
  </React.StrictMode>
);
```

```
// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

Gestion des composants Articles

Fichier src/components/articlesRedux/Listarticles.js

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

const Listarticles = () => {


    const dispatch = useDispatch();

    const initFetch = useCallback(() => {
        dispatch(getArticles());
      }, [dispatch])

      useEffect(() => {
        initFetch()
      }, [initFetch])


  return (
    <div>
              <AfficheArticles/>
    </div>
  )
}

export default Listarticles
```

Installer react-loading

Site : https://www.npmjs.com/package/react-loading

```
npm i react-loading --force
```

Fichier src/components/articlesRedux/AfficheArticles.js

```
import React from 'react'
import Card from '@mui/material/Card';
import CardActions from '@mui/material/CardActions';
import CardContent from '@mui/material/CardContent';
import CardMedia from '@mui/material/CardMedia';
import Button from '@mui/material/Button';
import Typography from '@mui/material/Typography';
import ReactLoading from 'react-loading';
import {useSelector} from "react-redux"


const AfficheArticles = () => {

    const {articles,isLoading,error} = useSelector((state)=>state.storearticles);


const renderArticles = () => {
    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 <React.Fragment>

    {articles  &&
          <div
style={{"display":"flex","flexWrap":"wrap","justifyContent":"left"}}>
          {articles.map((art,ind)=>{
              return  <Card sx={{ maxWidth: 'auto',margin: 1 }} key={ind}>
              <CardMedia
                component="img"
                alt="image"
                height="160"
                image={art.imageart}
              />
              <CardContent>s
                <Typography gutterBottom variant="h6" component="div">
                {art.reference}
                </Typography>
                <Typography variant="body2" color="text.secondary">
                Prix : {art.prix} DT
                </Typography>
              </CardContent>
              <CardActions>
              <Button disabled={art.qtestock<=1}
                  variant="contained" color="secondary" size="large"
```

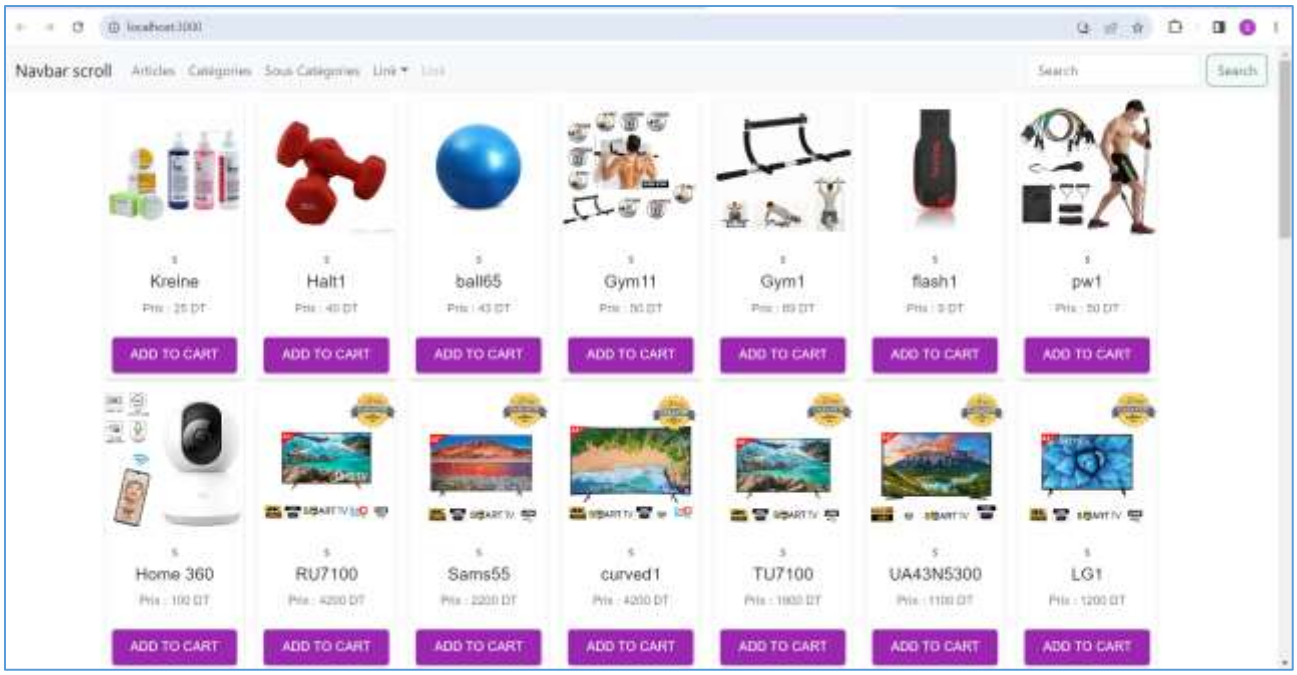
```
                >
                 {art.qtestock<=1? "OUT OF SOLD": "Add to cart"}
            </Button>

                </CardActions>
              </Card>

          })}</div>
        }
  </React.Fragment>
  }


  return (
    <div className="container">
        {renderArticles()}
   </div>

  )
}

export default AfficheArticles
```

Fichier src/App.js

```
import Listarticles from './components/articlesRedux/Listarticles';
```

…

```
<Route path='/' element={<Listarticles/>}/>
```


Navbar scroll
Articles
Categories
Sous Categories
Link
Search
Search
Kreine
Halt1
ball65
Gym11
Gym1
flash1
pw1
ADD TO CART
Home 360
RU7100
Sams55
curved1
TU7100
UA43N5300
LG1
ADD TO CART