React.js with PHP API REST

3. Composant ArticleCard, useEffect hook et conditional rendering

Nous reprenons la card déjà développée en php dans la vue home.index.php pour l'adapter.
Nous créons donc le composant ArticleCard.jsx suivant

src > components > articleCard > ArticleCard.jsx > ArticleCard

```jsx
/* eslint-disable react/prop-types */
function ArticleCard(props) {
    const {article} = props;
    return (
        <div className="col-12 col-md-6 col-lg-3 d-flex align-items-stretch justify-content-center">
        <div className="card mb-3" style={{width: "18rem"}}>
            <img src={article.img_src} className="card-img-top"
                alt={article.title} />
            <div className="card-body text-center">
                <h5 className="card-title">{article.title}</h5>
                <p className="card-text" style={{textAlign: "justify"}}>
                    {article.summary}
                </p>
                <a href={"articles/details/" + article.id_article} className="btn btn-primary">Lire</a>
            </div>
        </div>
    </div>
    );
}
export default ArticleCard;
```

Ce composant prend en props un article (ce qui permettra de le customiser en fonction de l'article à afficher)
Pensez à renommer les attributs class et à adapter les valeurs des attributs style.

Nous allons dans un premier temps afficher un composant ArticleCard dans le HomeScreen avec par exemple les valeurs de l'article ayant pour id 1 en DB

src > screens > home > HomeScreen.jsx > HomeScreen

```jsx
/* eslint-disable react/prop-types */
import ArticleCard from '../../components/articleCard/ArticleCard';
import './HomeScreen.css';

function HomeScreen() {
    const article = {
        id_article: 1,
        title: "nulla ac enim in tempor",
        summary: "Fusce posuere felis sed lacus. Morbi sem mauris, l...",
        img_src: "https://picsum.photos/id/1/400/300",
        published_at : "09/05/2022",
        updated_at : "26/08/2023",
        is_deleted : true,
        id_appuser: 8,
        id_serie: 9,
    }

    return (
        <main className="mt-5 pt-3 row">
            <ArticleCard article={article} />
        </main>
    );
}
export default HomeScreen;
```

Puis dans le composant App.jsx, nous ajoutons le HomeScreen sous la Navbar

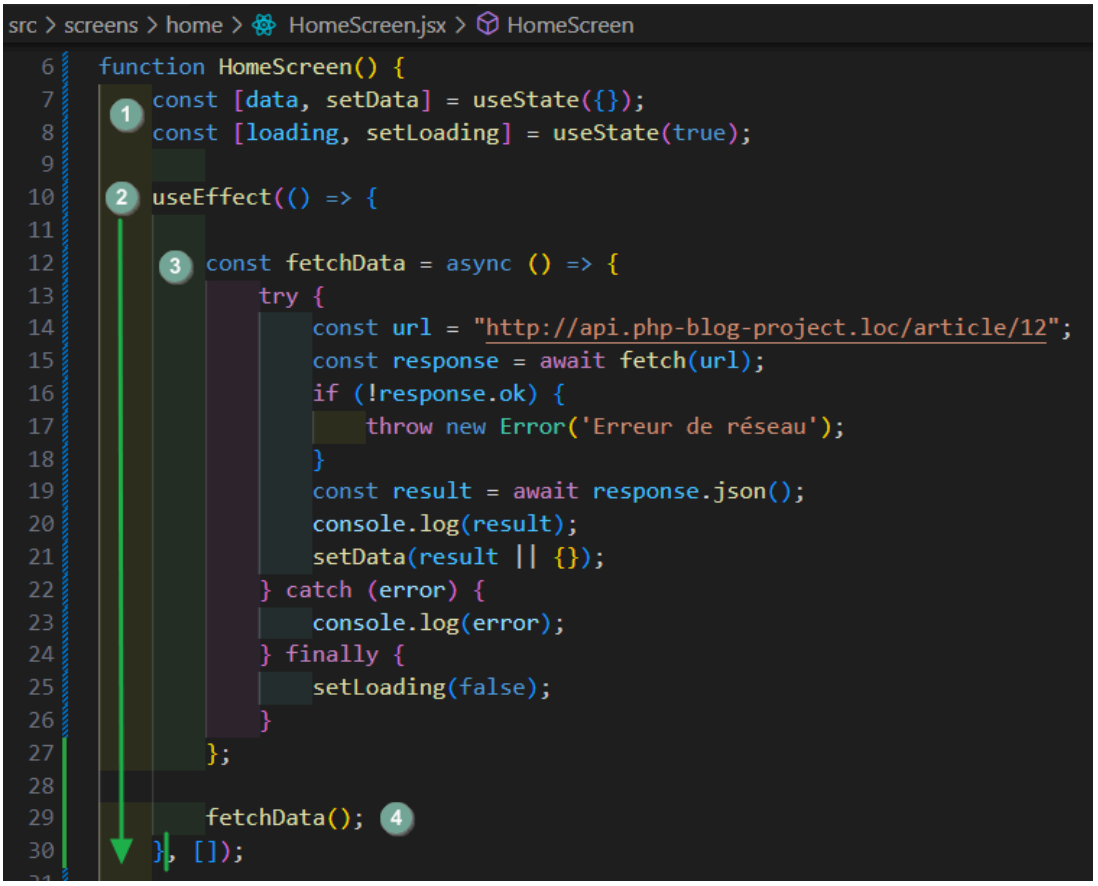
```jsx
src > App.jsx > App
import "./App.css";
import Navbar from "./components/navbar/Navbar";
import HomeScreen from "./screens/home/HomeScreen";

function App() {
    return (
        <>
            <Navbar/>
            <HomeScreen />
        </>
    );
}
export default App;
```

Résultat

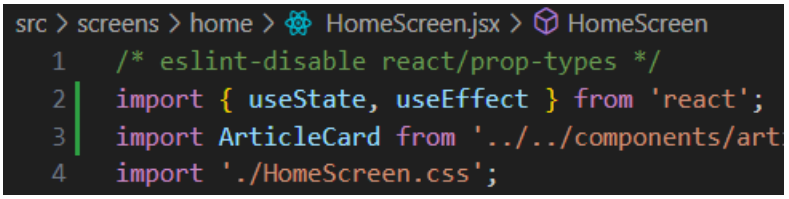
useEffect hook et conditional rendering

Pour récupérer réellement l'article en DB nous allons avoir besoin du hook useEffect et de la méthode fetch
Nous modifions donc le composant HomeScreen.jsx



1) - Nous déclarons 2 states, un premier (data) pour stocker les données qui seront récupérés après une requête vers l'API Rest, un autre (loading) pour stocker l'état de la requête
2) - Nous utilisons le hook useEffect (entre la ligne 10 et 30) qui prend en premier paramètre une fonction fléchée anonyme ()=>{ … } et en deuxième paramètre un tableau vide.
3) - A l'intérieur du hook useEffect, nous créons une méthode asynchrone fetchData() qui va exécuter la requête vers l'API Rest avec la fonction fetch() et attendre la réponse (ligne 15) puis convertir la réponse en objet json (ligne 19) pour enfin mettre à jour le state data c'est à dire (ici) l'article ayant pour id 12 en DB (ligne 21)
Un fois tout le travail effectué, le state loading est passé à false dans la clause finally du try catch (ligne 25)
4) - Nous appelons la méthode fetchData précédemment créée pour modifier les states data et loading

N'oubliez pas les imports :



Ressources :
https://react.dev/reference/react/useEffect
https://fr.javascript.info/fetch

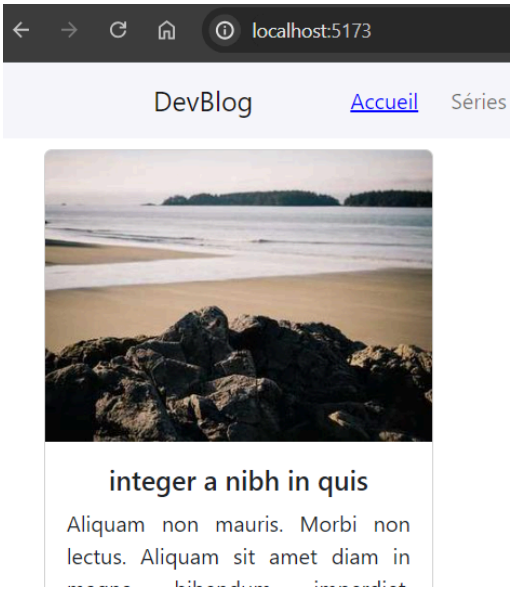
Pour le render (return) du composant :

```
29          fetchData();
30      }, []);
31
32      return (
33          <main className="mt-5 pt-3 row">
34          1 {loading &&
35              <p className='col-12 text-center'>
36                  Chargement des données ...
37              </p>}
38          2 {('id_article' in data)
39              ? <ArticleCard article={data} />
40              : !loading && <p className='col-12 text-center'>
41                  Aucun article trouvé ...
42              </p>}
43          </main>
44      );
45  }
46  export default HomeScreen;
```

1) - Si le state loading a pour valeur true nous affichons un message “chargement des données …” (ligne 34)
2) - Si le state data possède un attribut id_article (ligne 38), les données ont été récupérées avec succès suite à la requête vers l’API Rest. Nous affichons alors le composant ArticleCard auquel nous passons les données par la props article (ligne 39)
Sinon (ligne 40) et si le chargement des données est fini, c’est qu’aucun article n’a été récupéré en DB, nous affichons alors le message “Aucun article trouvé …”

Testez avec 2 id différents, l’un existant en DB, l’autre non.

Résultat pour id = 12



Résultat pour id = 120

```
const fetchData = async () => {
    try {
        const url = "http://api.php-blog-project.loc/article/120";
        const response = await fetch(url);
        if (!response.ok) {
```

Ressources :
https://react.dev/learn/conditional-rendering

Exercices sur le useEffect :
https://www.clientside.dev/blog/react-use-effect-practice-exercises

A vous de réaliser la page détail d'un article en React.js à partir de la vue articles.details.php
Commencez par créer un composant **ArticleScreen** puis ajoutez le dans App.jsx à la place de HomeScreen pour pouvoir tester et debugger.
Codez ensuite le useEffect pour récupérer les données et finissez par le render (return) à partir du code adapté de la vue php.

git : https://github.com/DWWM-23526/REACT_BLOG_PROJECT/tree/Step03