

React.js with PHP API REST

6. Composants Génériques et composants as props

Le composant ListScreen.jsx

Ce composant va reprendre le code déjà écrit dans les 3 screens Home, Series et Techs

Pour la partie logique, seul l'url de la requête vers l'api pour récupérer les données va changer

```
src > components > screens > ListScreen.jsx > ListScreen
1  import { useState, useEffect } from 'react';
2  import PropTypes from 'prop-types';
3  function ListScreen({apiUrl, title, Card}) {
4      const [data, setData] = useState([]);
5      const [loading, setLoading] = useState(true);
6
7      useEffect(() => {
8          const fetchData = async () => {
9              try {
10                 const response = await fetch(apiUrl);
11                 if (!response.ok) {
12                     throw new Error('Erreur de réseau');
13                 }
14                 const result = await response.json();
15                 console.log(result);
16                 setData(result || []);
17             } catch (error) {
18                 console.log(error);
19             } finally {
20                 setLoading(false);
21             }
22         };
23         fetchData();
24     }, []);
25
26     return (
```

Pour la partie render, le titre va changer d'un screen à l'autre, ainsi que la Card utilisée pour afficher les données.

```
src > components > screens > ListScreen.jsx > ListScreen
3  function ListScreen({apiUrl, title, Card}) {
26     return (
27         <main className="mt-5 pt-3 row">
28             <h2>{title}</h2>
29             {loading &&
30                 <p className='col-12 text-center'>
31                     Chargement des données ...
32                 </p>
33             {(data.length > 0 )
34                 ? (data.map((item, i) => <Card key={i} data={item} />))
35                 : (!loading && (<p className='col-12 text-center'>
36                     Aucune donnée trouvée ...
37                 </p>))
38             )}
39         </main>
40     );
41 }
```

N'oubliez pas les propTypes

```
41 }
42 ListScreen.propTypes = {
43   apiUrl: PropTypes.string.isRequired,
44   title: PropTypes.string.isRequired,
45   Card: PropTypes.func.isRequired,
46 }
47 export default ListScreen;
```

Nous devons adapter les composants de type Card pour que la props data se retrouvent dans tous les composants.

Pour ArticleCard nous renommons la props article en data

```
src > components > cards > ArticleCard.jsx > ArticleCard
1  import PropTypes from "prop-types";
2  import Card from "../Card";
3  function ArticleCard(props) {
4    const { data } = props;
5    return (
6      <Card
7        title={data.title}
8        imgSrc={data.img_src}
9        link={"articles/details/" + data.id_article}
10       btnText="Lire"
11     >
12       <p className="card-text" style={{ textAlign: "justify" }}>
13         {data.summary}
14       </p>
15     </Card>
16   );
17 }
18 ArticleCard.propTypes = {
19   data: PropTypes.shape({
20     id_article: PropTypes.number.isRequired,
```

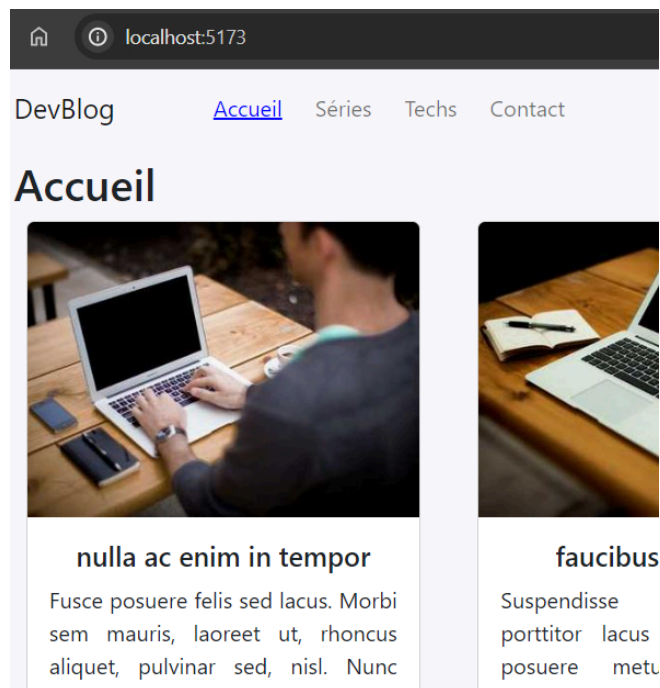
Même chose pour SerieCard (serie -> data) et pour TechCard (tech -> data)

Nous allons maintenant modifier HomeScreen pour utiliser ListScreen et lui passer ArticleCard en props (ligne 10)

```
src > screens > home > HomeScreen.jsx > HomeScreen
1  import "../HomeScreen.css";
2  import ListScreen from "../../components/screens/ListScreen";
3  import ArticleCard from "../../components/cards/ArticleCard";
4
5  function HomeScreen() {
6    return (
7      <ListScreen
8        apiUrl="http://api.php-blog-project.loc/article?orderby=published_at&sort=DESC&limit=12"
9        title="Accueil"
10       Card={ArticleCard}
11     />
12   );
13 }
14 export default HomeScreen;
15
```

Nous modifierons l'API Rest ensuite pour gérer les params que nous avons ajouté dans l'url d'appel à l'API

Résultat



Même principe pour SerieScreen

```
src > screens > series > SeriesScreen.jsx > SeriesScreen
1  import SerieCard from '../components/cards/SerieCard';
2  import ListScreen from '../components/screens/ListScreen';
3
4  function SeriesScreen() {
5    return (
6      <ListScreen
7        apiUrl="http://api.php-blog-project.loc/serie?orderby=title"
8        title="Les séries"
9        Card={SerieCard}
10      />
11    );
12  }
13  export default SeriesScreen;
```

Et pour TechsScreen

```
src > screens > techs > TechsScreen.jsx > TechsScreen
1  import TechCard from '../components/cards/TechCard';
2  import ListScreen from '../components/screens/ListScreen';
3
4  function TechsScreen() {
5    return (
6      <ListScreen
7        apiUrl="http://api.php-blog-project.loc/tech?orderby=label"
8        title="Les Techs"
9        Card={TechCard}
10      />
11    );
12  }
13  export default TechsScreen;
```

Ressources :

<https://www.dhiwise.com/post/react-pass-component-as-prop-guide-to-component-composition#role-of-parent-and-child-components>

Modification de l'API Rest

Pour prendre en compte les paramètres GET de l'url que nous avons ajoutés dans nos 3 screens (orderby, sort, limit) nous devons modifier légèrement notre API Rest en php.

Dans BaseController, méthode get(), nous récupérons les paramètres grâce à la classe HttpRequest et les passons à la méthode getAll du repository.

```
api > Controller > BaseController.php > BaseController > get()
6  class BaseController
    0 references | 0 overrides
34  protected function get() : array | BaseEntity | null
35  {
36      $repositoryClassName = $this->getRepositoryClassName();
37      $repository = new $repositoryClassName();
38      if($this->id <= 0){
39          $params = HttpRequest::get(HttpReqAttr::PARAMS);
40          $entities = $repository->getAll($params);
41          return $entities;
42      }
43      $entity = $repository->getOneById($this->id);
44      return $entity;
45  }
46
```

Puis nous modifions la requête sql dans la méthode getAll() de BaseRepository pour tenir compte des éventuels paramètres GET de l'url.

```
api > Repository > BaseRepository.php > BaseRepository > getAll()
8  class BaseRepository
    0 references | 0 overrides
58  public function getAll(array $params) : array
59  {
60      $sql = "SELECT * FROM ".$this->getTableName();
61      if(isset($params['orderby'])){
62          $sql .= " ORDER BY ".$params['orderby'];
63          if(isset($params['sort'])){
64              $sql .= " " . $params['sort'];
65          }
66      }
67      if(isset($params['limit'])){
68          $sql .= " LIMIT " . $params['limit'];
69      }
70      $queryResponse = $this->preparedQuery($sql);
71      $entities = $queryResponse->statement->fetchAll(PDO::F
72      return $entities;
73  }
```

Résultat pour HomeScreen (Les 12 derniers articles publiés)

localhost:5173

DevBlog

Accueil

Séries

Techs

Contact

Accueil



sapien in sapien iaculis
congue vivamus

In hac habitasse platea dictumst.
Etiam faucibus cursus urna. Ut
tellus. Nulla ut erat id mauris
velutata, elementum. Nullam



nec ni

Duis aliquam
at turpis a
Integer
neque, ut

Pour SeriesScreen (par ordre alphabétique sur le titre)

localhost:5173

DevBlog

Accueil

Séries

Techs

Contact

Les séries



augue luctus tincidunt

Duis ac nibh. Fusce lacus purus,
aliquet at, feugiat non, pretium
quis, lectus. Suspendisse potenti.

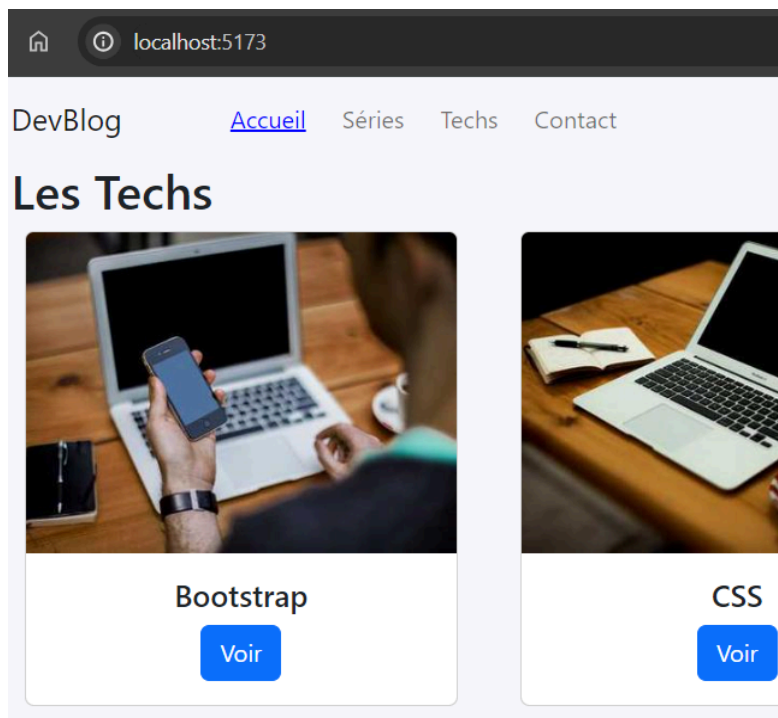
Voir



dapibu

Curabitur con
dui nec nis
Donec ut doli
quam fringilla
leo, rhoncus
amet, cursus i

Pour TechsScreen (par ordre alphabétique sur le label)



git : https://github.com/DWWM-23526/REACT_BLOG_PROJECT/tree/Step06
https://github.com/DWWM-23526/PHP_BLOG_PROJECT/tree/Step16-6