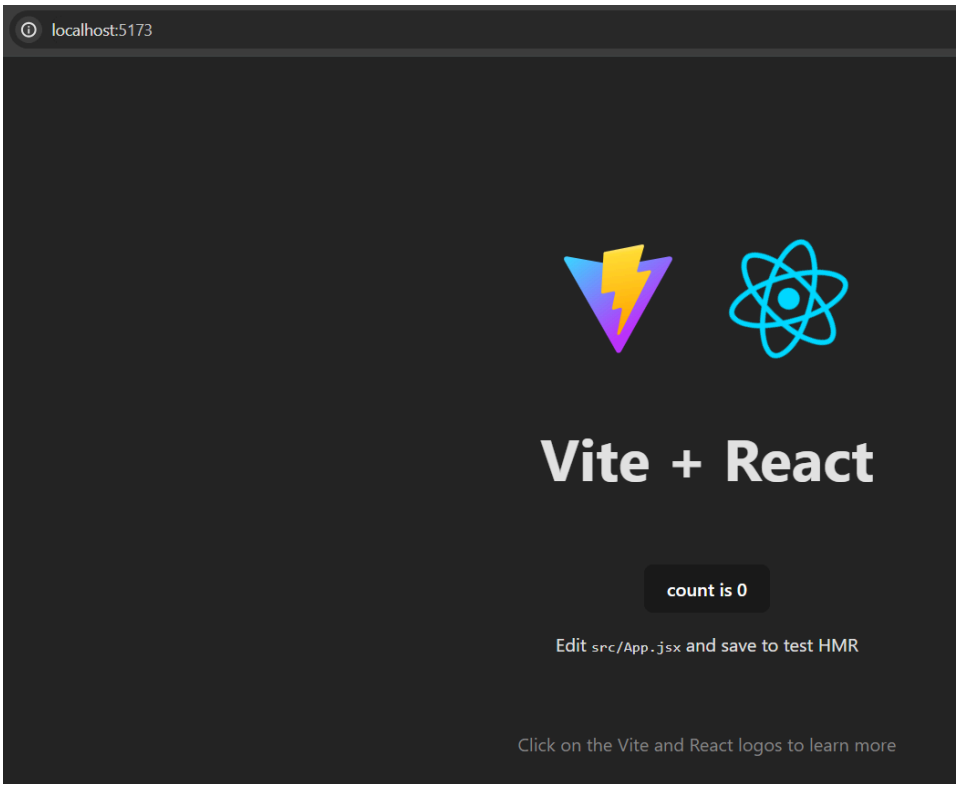
React.js with PHP API REST

1. Premiers pas en React.js

Nous commençons par créer une application React.js avec l'outil Vite.js
https://vitejs.dev/guide/#getting-started

N'oubliez pas d'installer les dépendances avant de démarrer le serveur.

Résultat



Avant d'écrire notre premier composant fonctionnel, nous allons nettoyer le boilerplate proposé par Vite.
Nous vidons les fichiers CSS index et App et nous réécrivons le fichier App.jsx (qui est un composant)

```
src > App.jsx > App
1   import './App.css'
2
3   function App() {
4
5     return (
6       <p>
7         App.jsx est le composant qui va contenir toute notre application
8       </p>
9     )
10  }
11
12  export default App
```

Notre composant jsx est défini entre les lignes 3 et 10. En ligne 1, nous importons les CSS contenu dans App.css et la ligne 12 permet d'exporter le composant afin qu'il soit utilisable.

Dans le fichier main.jsx, le composant App.jsx est importé en même temps que React et ReactDOM. C'est également dans ce fichier qu'est importé index.css (qui pourra nous servir de reset CSS)

src > main.jsx

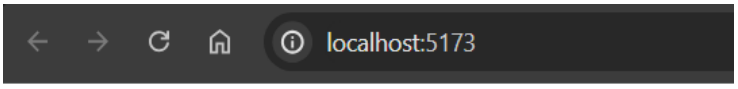
```jsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
import './index.css'

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)
```

Les lignes 6 à 10 permettent d'afficher le contenu de notre application (App.jsx) dans l'élément ayant pour id #root. Cet élément est dans le fichier index.html qui importe également le fichier main.jsx afin qu'il soit exécuté.

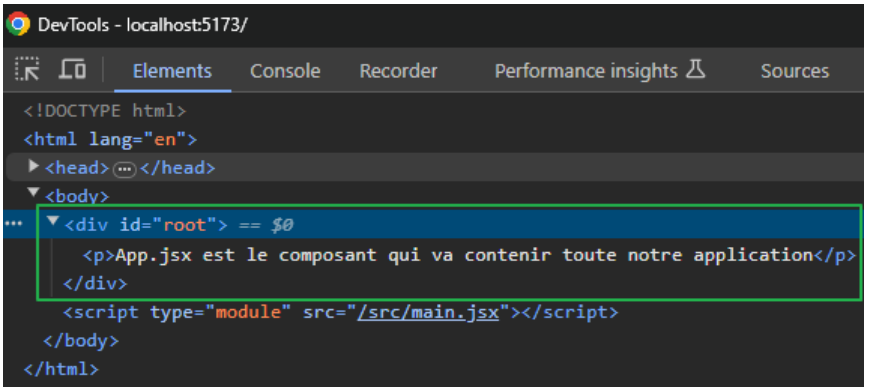
index.html > html

```html
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Vite + React</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

Résultat

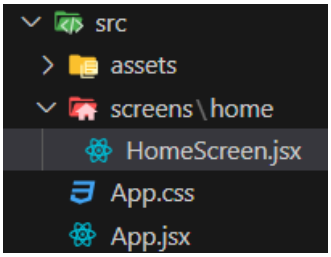


App.jsx est le composant qui va contenir toute notre application



Un premier composant

Nous créons l'arborescence (src>screens>home) et y ajoutons le fichier Home.jsx ci-dessous

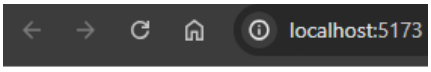



```jsx
function HomeScreen() {

    return ( <div>
        <h1>Le titre de la page</h1>
        <p>Un paragraphe</p>
    </div> );
}

export default HomeScreen;
```

Pour afficher ce composant, il faut l'ajouter (et l'importer) dans App.jsx

```jsx
import './App.css'
import HomeScreen from './screens/home/HomeScreen'

function App() {

  return (
    <HomeScreen />
  )
}

export default App
```

Résultat

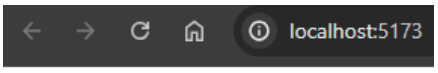


Le titre de la page

Un paragraphe

Si nous ajoutons du style dans App.css, il sera appliqué à tous les composant importés dans App.jsx

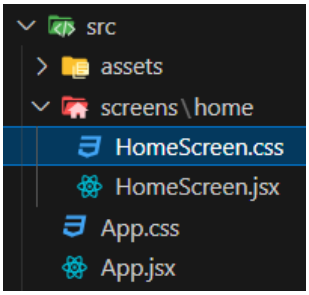
```css
h1{
    color: blue;
}
```



Le titre de la page

Un paragraphe

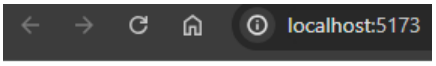
Il est préférable de définir le style spécifique à chaque composant dans un fichier css dédié au composant. Nous créons donc un fichier HomeScreen.css (toujours dans src>screens>home)




```
src > screens > home > HomeScreen.css > h1
1   h1{
2       color: red;
3   }
```

```
src > screens > home > HomeScreen.jsx > HomeScreen
1   import './HomeScreen.css';
2
3   function HomeScreen() {
4
```

Résultat



Le titre de la page

Un paragraphe

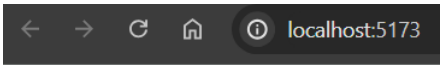
Les styles spécifiques au composant définis dans HomeScreen.css prennent le dessus sur les styles généraux de l'application définis dans App.css

Si nous voulons utiliser des classes pour définir le style, le nom de l'attribut class en HTML est remplacé par className en JSX (class étant un mot réservé au JS)

```
src > screens > home > HomeScreen.css > .mainTitle
5   .mainTitle {
6       text-decoration: underline;
7   }
```

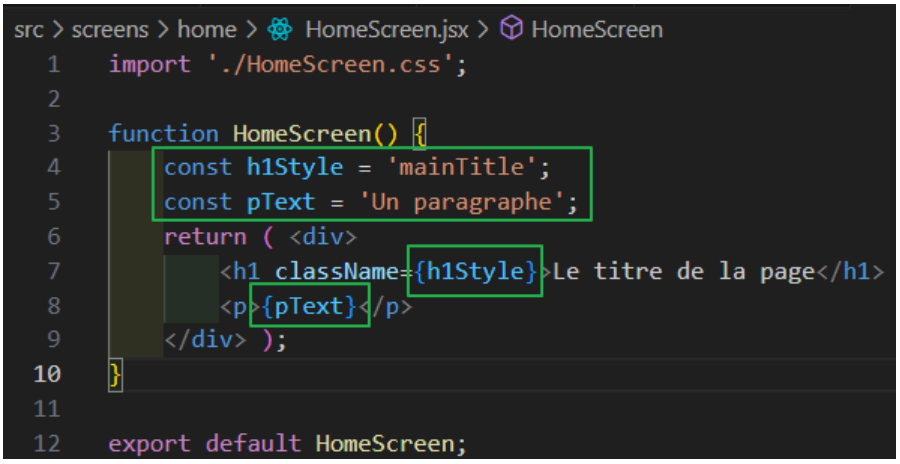
```
src > screens > home > HomeScreen.jsx > HomeScreen
1   import './HomeScreen.css';
2
3   function HomeScreen() {
4
5       return ( <div>
6           <h1 className='mainTitle'>Le titre de la page</h1>
7           <p>Un paragraphe</p>
8       </div> );
9   }
10
11  export default HomeScreen;
```

Résultat

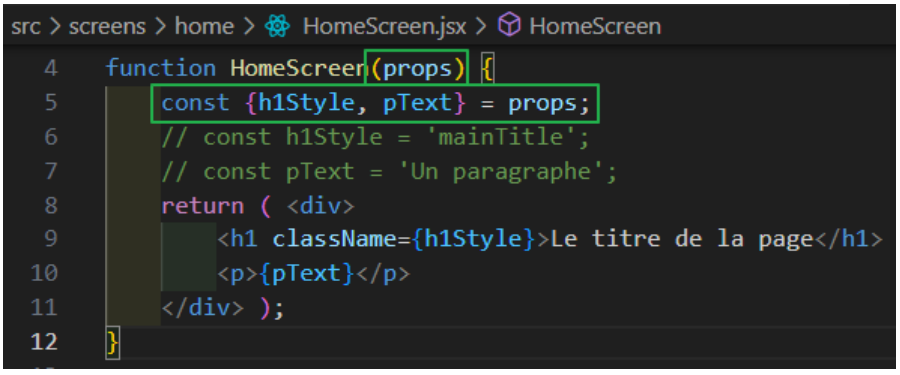


Le titre de la page

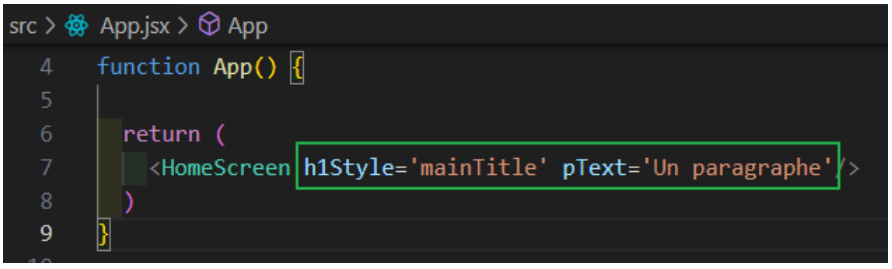
Il est possible de créer des variables pour stocker certaines informations avant le return du composant fonctionnel. Dans ce cas, l'inclusion de ces variables dans le jsx se fait entre accolades.



Plutôt que de figer ces valeurs dans le composant, nous pouvons le rendre dynamique à l'aide des props



Lorsque nous utilisons le composant (ici dans App.jsx), nous lui passons les valeurs désirées



Nous pouvons maintenant utiliser notre composant autant de fois que nous le souhaitons en le customisant en fonction des props définies dans celui-ci

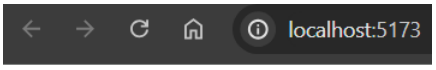
Lignes 7 et 10, nous utilisons un **fragment jsx**. Un composant ne peut retourner qu'une seul élément, il faut donc qu'il y ait un élément racine qui englobe tous les composants (ou balises html)

Nous ajoutons le css pour la classe .blue

src > screens > home > HomeScreen.css > ...

```
4   .mainTitle {
5       text-decoration: underline;
6   }
7   .blue {
8       color: blue;
9   }
```

Résultat



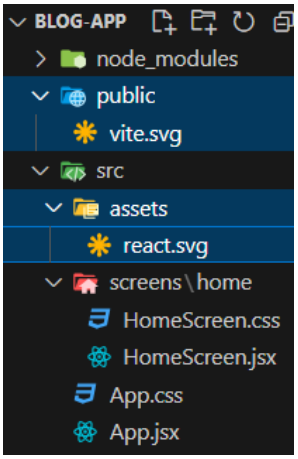
Le titre de la page

Un paragraphe

Le titre de la page

Un autre paragraphe

Nous allons utiliser les images du boilerplate pour voir leur insertion dans les composants

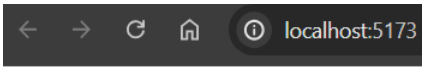


src > screens > home > HomeScreen.jsx > ...

```
2   import './HomeScreen.css';
3   import ViteLogo from '/vite.svg';
4   import ReactLogo from '/src/assets/react.svg';
5
6   function HomeScreen(props) {
7       const {h1Style, pText} = props;
8       return ( <div>
9           <h1 className={h1Style}>Le titre de la page</h1>
10          <p>{pText}</p>
11          <img src={ViteLogo} alt="image depuis public"/><br/>
12          <img src={ReactLogo} alt="image depuis src/assets"/><br/>
13      </div> );
14  }
```

L'image est importée et nommée en début de fichier puis ajoutée dans l'attribut src de la balise img (lignes 11 et 12)

Résultat



Le titre de la page

Un paragraphe

Il est bien sûr possible de passer des images dans les props

src > screens > home > HomeScreen.jsx > HomeScreen

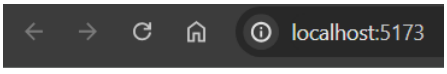
```jsx
import './HomeScreen.css';

function HomeScreen(props) {
    const {h1Style, pText, imgSrc, imgAlt} = props;
    return ( <div>
        <h1 className={h1Style}>Le titre de la page</h1>
        <p>{pText}</p>
        <img src={imgSrc} alt={imgAlt}/><br/>
    </div> );
}
```

Il faut dans ce cas les importer dans le composant appelant pour les passer en props des composants appelés

src > App.jsx > App

```jsx
import "./App.css";
import HomeScreen from "./screens/home/HomeScreen";
import ViteLogo from "/vite.svg";
import ReactLogo from "/src/assets/react.svg";

function App() {
    return (
        <>
            <HomeScreen
                h1Style="mainTitle"
                pText="Un paragraphe"
                imgSrc={ViteLogo}
                imgAlt="Logo Vite"
            />
            <HomeScreen
                h1Style="blue"
                pText="Un autre paragraphe"
                imgSrc={ReactLogo}
                imgAlt="Logo React"
            />
        </>
    );
}
```

Résultat



Le titre de la page

Un paragraphe

Le titre de la page

Un autre paragraphe