# Day 1 Exercise - Introducing Components

# Making payments-ui frontend part 1

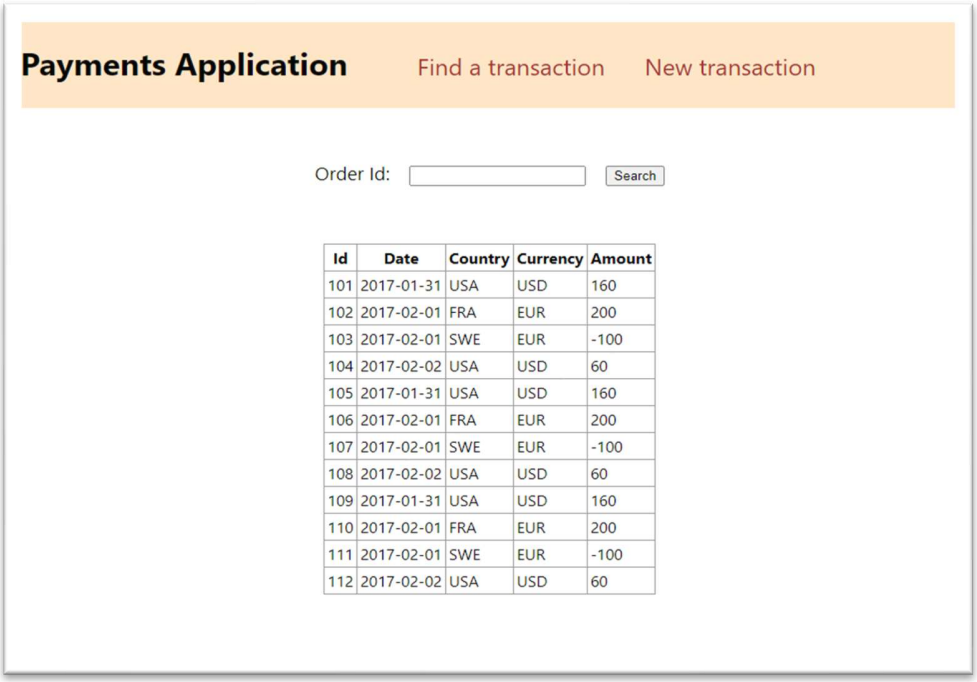
## Task

- Create a new project called payments-ui. This will be the front-end that will connect to the payments database rest application.

- Create a set of components that will come together to form the user interface for the application. This should contain:

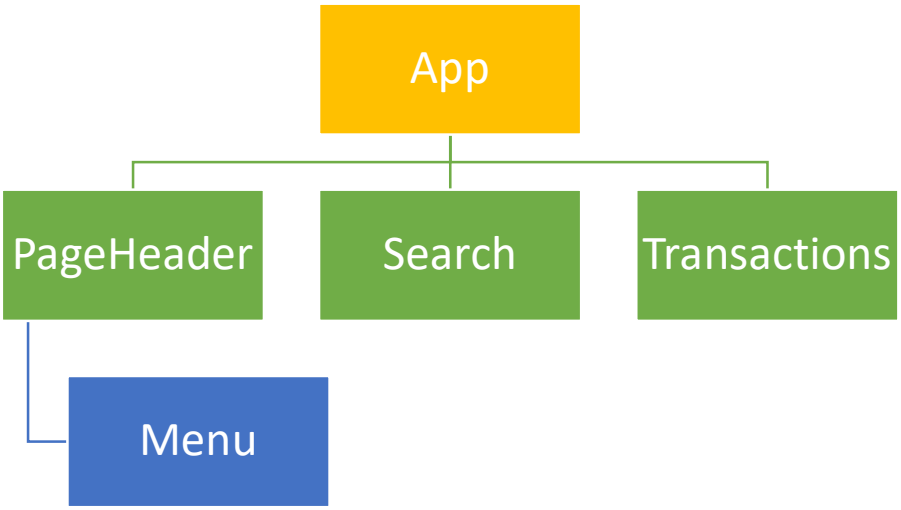
    - The application name at the top of the page
    - A menu with the options: “find a transaction” and “new transaction”
    - A search box which would ask for an order_id
    - A List of all the transactions which are found from the search

- Here is an example of what you might build.




-

- If you are not confident with css, do not worry about the design aspect.
- For now – just use dummy data to build up the interface (3 or 4 lines of data is sufficient).
- You will probably adjust what we build as we progress through the course, so don't worry about accuracy – the key purpose of this exercise is to create some different components, and then place them on the page.




## Pre-requisites

None

## 1. Create a new project called payments-ui

**Create a new react applicaiton** by running the command

```
npx create-react-app payments-ui --template typescript
```

This will create a new React application called "payments-ui".

## 2. Start the react development server

When we create a new React applicaiton, we get a full application with some dummy content ready to edit.

To be able to see the website in the browser we need a webserver - this is because the code files need to be **transpiled** before we can see the generated content. The transpiling process converts the code files into plain javascript that the browser can execute.

React comes with a built in webserver designed for use in development, so we'll use that to help us view the site.

1. **Navigate into the folder** that was created when you generated the app with

   ```
   cd payments-ui
   ```

2. **Start the web server** with

   ```
   npm start
   ```

3. When the server tells you that the site is ready to view, **open your browser** and visit: http://localhost:3000

## 3.Create the Menu component file

Components sit in typescript files, however with a typical application containing many components we need to think about how to structure our files. It is usual to create a folder called components, and then place components either directly in that folder or in subfolders. We use subfolders to create logical groups of components.

1. **Create a new folder** under the src folder called "components".

2. **Create a new folder** under the components folder called "pageHeader".

3. **Create a new file** in the Greeting folder called "Menu.tsx". Components always start with an uppercase letter and it is convention to name the file with the same name as the component it contains.

4. Create the outline of a **regular typescript function** in the Menu.tsx file. The function should be called Menu. It will return a ReactElement (which you will need to import). You should also make it the default export function for the file.

```
import { ReactElement } from "react";

const Menu = () : ReactElement => {};

export default Menu;
```

5. Make the function **return valid JSX** to display the two menu items in an unordered list. Add styling with className of nav. import { ReactElement } from "react";

```
import ‘./PageHeader.css’;

const Menu = ():ReactElement => {
    return (
        <ul className="nav">
            <li>Find a transaction</li>
            <li>New transaction</li>
        </ul>
    );
}
export default Menu;
```

*Note: At this point we have created the menu component, but we won't yet see it on the web page.*

6. **Create a new file** in the pageHeader folder called "PageHeader.tsx". Components always start with an uppercase letter and it is convention to name the file with the same name as the component it contains.

7. Create the outline of a **regular typescript function** in the PageHeader.tsx file. The function should be called PageHeader. It will return a ReactElement (which you will need to import). You should also make it the default export function for the file.

```
import { ReactElement } from "react";

const PageHeader = () : ReactElement => {};

export default PageHeader;
```

8. Make the function **return valid JSX** to display the title “Payments Application: and the Menu component.

```
import Menu from "./Menu";
import ‘./PageHeader.css’;
```

```
const PageHeader = () => {
    return (
        <div className="pageHeader">
            <h1>Payments Application</h1>
            <Menu/>
        </div>
    );
}

export default PageHeader
```

9. Create the CSS file PageHeader.css

```
.pageHeader {
    background-color: bisque;
    min-height: 80px;
    width: 100%;
    padding: 10px;
}

.pageHeader h1 {
    float: left;
    margin-right: 50px;
}

.pageHeader ul.nav {
    margin-top: 30px;
}

.pageHeader ul.nav li {
    display:inline;
    padding: 20px;
    font-size: 1.5em;
    color: brown;
}
```

*Note: At this point we have created the PageHeader component, but we won't yet see it on the web page.*

10. Make the PageHeader component appear on the screen by adding it to App.tsx

```
import React from 'react';
import './App.css';
import PageHeader from './components/pageHeader/PageHeader';
```

```
function App() {
  return (
    <>
    <PageHeader />
    </>
  );
}

export default App;
```

## 4. Create the Search component file

Components sit in typescript files, however with a typical application containing many components we need to think about how to structure our files. It is usual to create a folder called components, and then place components either directly in that folder or in subfolders. We use subfolders to create logical groups of components.

1. **Create a new folder** under the components folder called "Search".
2. **Create a new file** in the Search folder called "Search.tsx". Components always start with an uppercase letter and it is convention to name the file with the same name as the component it contains.
3. Create the outline of a **regular typescript function** in the Search.tsx file. The function should be called Search. It will return a ReactElement (which you will need to import). You should also make it the default export function for the file.

```
import { ReactElement } from "react";

const Search = () : ReactElement => {};

export default Search;
```

11. Make the function **return valid JSX** to display an input box with label Order Id and a button to start the search

```
Search.tsx file

import './Search.css'


const Search = (): JSX.Element => {
    return (
        <div className='searchBox'>
```

```
            <label htmlFor='orderId'>Order Id:</label>
            <input id='orderId' type='text' />
            <button> Search order</button>

            </div>
    )

};

export default Search;


Search.css file
.searchBox {
    text-align: center;
    margin-top: 50px;
    font-size: 1.2em;
}

.searchBox label, input, button {
    margin: 10px;
}
```

*Note: At this point we have created the Search component, but we won't yet see it on the web page.*

12. Make the Search component appear on the screen by adding it to App.tsx

```
import React from 'react';
import './App.css';
import PageHeader from './components/pageHeader/PageHeader';
import Search from './components/Search/Search';

function App() {
  return (
    <>
    <PageHeader />
     <Search />
    </>
  );
}

export default App;
```

# 5. Create the Transactions component file

Components sit in typescript files, however with a typical application containing many components we need to think about how to structure our files. It is usual to create a folder called components, and then place components either directly in that folder or in subfolders. We use subfolders to create logical groups of components.

4. **Create a new folder** under the components folder called "Transactions".

5. **Create a new file** in the Transaction folder called " Transactions.tsx". Components always start with an uppercase letter and it is convention to name the file with the same name as the component it contains.

6. Create the outline of a **regular typescript function** in the Transactions.tsx file. The function should be called Transactions. It will return a ReactElement (which you will need to import). You should also make it the default export function for the file.

```
import { ReactElement } from "react";

const Transactions = () : ReactElement => {};

export default Transactions;
```

13. Make the function **return valid JSX** to display an input box with label Order Id and a button to start the search

```
import './Transactions.css';

const Transactions = () => {
    return (
      <table className="transactionsTable">
      <thead>
<tr>

<th>Id</th><th>Date</th><th>Country</th><th>Currency</th><th>Amount</th>
</tr>
       </thead>
       <tbody>
      <tr>
<td>1</td><td>2020-05-22</td><td>USA</td><td>USD</td><td>17.55</td>
       </tr>
      <tr>
       <td>2</td><td>2020-05-
23</td><td>UK</td><td>GBP</td><td>36.50</td>
      </tr>
      <tr>
```

```
      <td>3</td><td>2020-05-
24</td><td>SWE</td><td>EUR</td><td>42.00</td>
      </tr>
       </tbody>
      </table>
    );
}

export default Transactions
```

Transactions.css file

```
.transactionsTable {
    margin-top: 50px;
    margin-left: auto;
    margin-right: auto;
    border-collapse: collapse;
    border: 1px solid #999;
}
.transactionsTable td, .transactionsTable th {
    padding: 3px;
    border: 1px solid #999;
}
```

*Note: At this point we have created the Transactions component, but we won't yet see it on the web page.*

14. Make the Transactions component appear on the screen by adding it to App.tsx

```
import React from 'react';
import './App.css';
import PageHeader from './components/pageHeader/PageHeader';
import Search from './components/Search/Search';
import Transactions from './components/Transactions/Transactions';

function App() {
  return (
    <>
    <PageHeader />
    <Search />
    <Transactions />
    </>
  );
}

export default App;
```

## Summary

In this lab we have:

- Created a component Menu and included in component PageHeader
- Created a component Search
- Created a component Transactions
- Modified the App component to include PageHeader, Search, Transactions