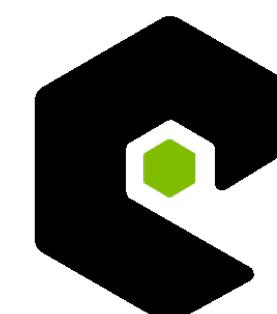
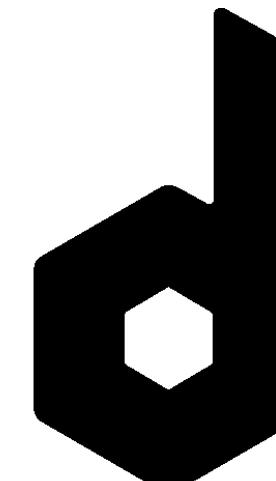
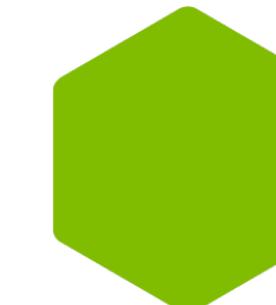
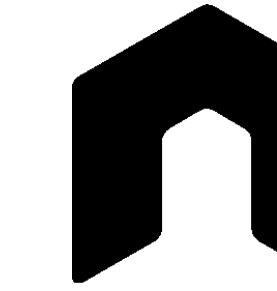
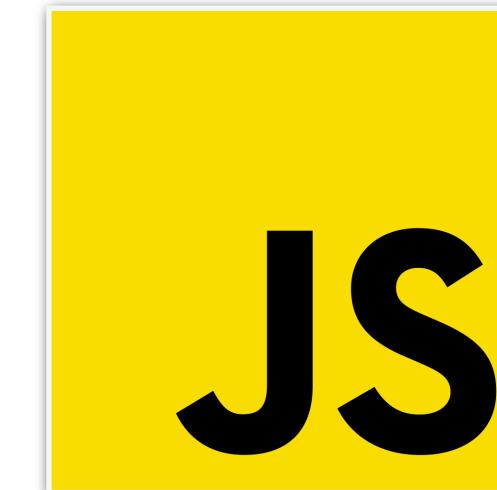
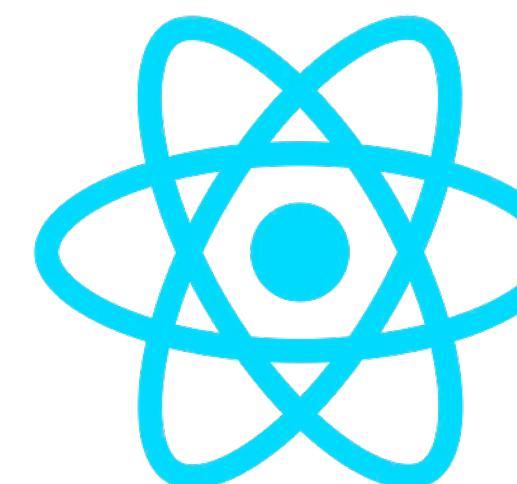

JAVASCRIPT

REACT FRONT-END PART 2

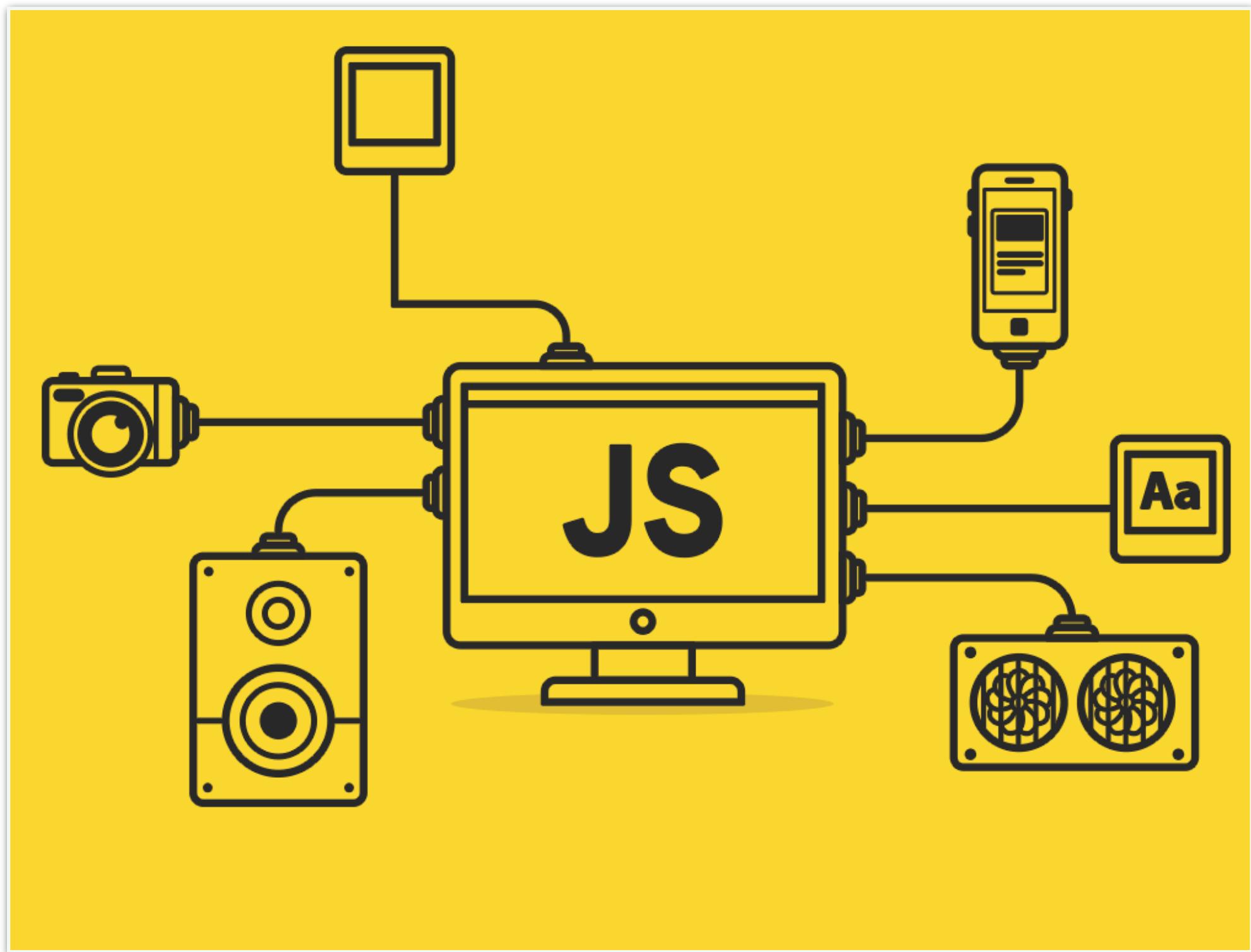
Anirach Mingkhwan

Anirach.m@fitm.kmutnb.ac.th



OUTLINE

- Thinking React II
- Form and OnChange
- React CRUD



THINKING REACT II

The mockup look like this:

```
[  
  { category: "Fruits", price: "$1", stocked: true, name: "Apple" },  
  { category: "Fruits", price: "$1", stocked: true, name: "Dragonfruit" },  
  { category: "Fruits", price: "$2", stocked: false, name: "Passionfruit" },  
  { category: "Vegetables", price: "$2", stocked: true, name: "Spinach" },  
  { category: "Vegetables", price: "$4", stocked: false, name: "Pumpkin" },  
  { category: "Vegetables", price: "$1", stocked: true, name: "Peas" }  
]
```

Only show products in stock

Name	Price
------	-------

Fruits

Apple	\$1
Dragonfruit	\$1
Passionfruit	\$2

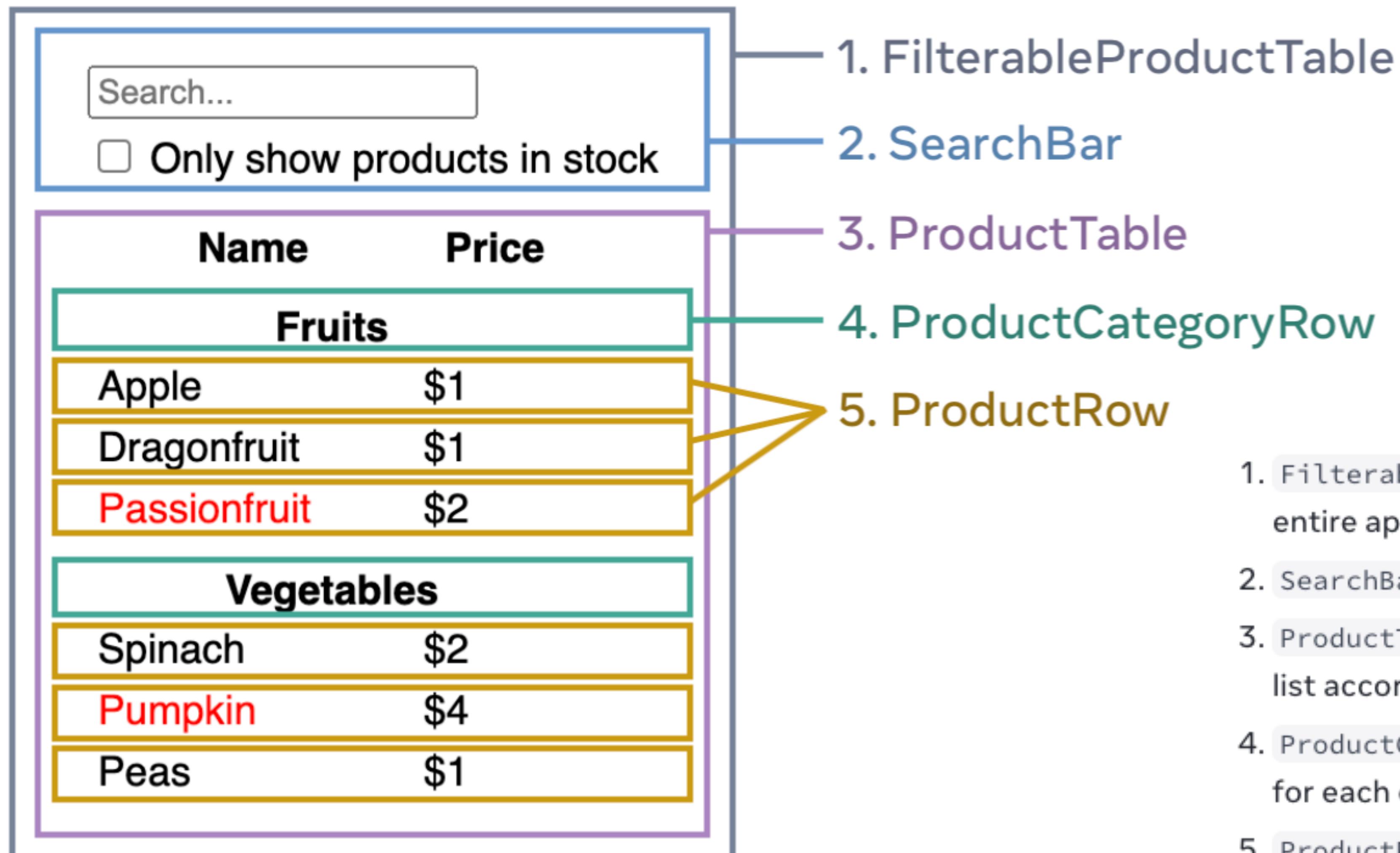
Vegetables

Spinach	\$2
Pumpkin	\$4
Peas	\$1

Step 1: Break the UI into a component hierarchy

Slide # 5

There are five components on this screen:



1. `FilterableProductTable` (grey) contains the entire app.
2. `SearchBar` (blue) receives the user input.
3. `ProductTable` (lavender) displays and filters the list according to the user input.
4. `ProductCategoryRow` (green) displays a heading for each category.
5. `ProductRow` (yellow) displays a row for each product.

The screenshot shows a code editor interface with a dark theme. On the left, the file tree (EXPLORER) shows a project structure with files like App.js, index.js, styles.css, and package.json. The main editor area displays the code for App.js:

```
src > App.js > FilterableProductTable
1 import { useState } from 'react';
2
3 function FilterableProductTable({ products }) {
4   const [filterText, setFilterText] = useState('');
5   const [inStockOnly, setInStockOnly] = useState(false);
6
7   return (
8     <div>
9       <SearchBar
10         filterText={filterText}
11         inStockOnly={inStockOnly}
12         onFilterTextChange={setFilterText}
13         onInStockOnlyChange={setInStockOnly} />
14       <ProductTable
15         products={products}
16         filterText={filterText}
17         inStockOnly={inStockOnly} />

```

To the right, the rendered output is displayed in a preview window. It includes a search bar, a checkbox for filtering by stock status, and two sections: "Fruits" and "Vegetables", each listing products with their names and prices.

Name	Price
Apple	\$1
Dragonfruit	\$1
Passionfruit	\$2
Spinach	\$2
Pumpkin	\$4
Peas	\$1

https://codesandbox.io/p/sandbox/react-dev-w6ykwl?file=%2Fsrc%2FApp.js&utm_medium=sandpack

The screenshot shows the VS Code interface with the following details:

- EXPLORER View:** Shows the project structure under "REACT-S03".
 - node_modules
 - src
 - components
 - FilterableProductTable.jsx
 - ProductCategoryRow.jsx
 - ProductRow.jsx
 - ProductTable.jsx
 - SearchBar.jsx
 - App.jsx (selected)
 - main.jsx
 - .editorconfig
 - .eslintrc.cjs
 - .gitignore
 - index.html
 - package-lock.json
 - package.json
 - README.md
 - vite.config.js
- Code Editor View:** Displays the content of "App.jsx".

```
1 import { useState } from "react";
2 import FilterableProductTable from "./components/FilterableProductTable";
3
4 export default function App() {
5   const products = [
6     { category: "Fruits", price: "$1", stocked: true, name: "Apple" },
7     { category: "Fruits", price: "$1", stocked: true, name: "Dragonfruit" },
8     { category: "Fruits", price: "$2", stocked: false, name: "Passionfruit" },
9     { category: "Vegetables", price: "$2", stocked: true, name: "Spinach" },
10    { category: "Vegetables", price: "$4", stocked: false, name: "Pumpkin" },
11    { category: "Vegetables", price: "$1", stocked: true, name: "Peas" },
12  ];
13
14  const [filterText, setFilterText] = useState("");
15  const [inStockOnly, setInStockOnly] = useState(false);
16
17  return (
18    <FilterableProductTable
19      products={products}
20      filterData={filterText}
21      filterAction={setFilterText}
22      inStockOnlyData={inStockOnly}
23      inStockOnlyAction={setInStockOnly}
24    />
25  );
26}
27
```

```
FilterableProductTable.jsx M X  
src > components > FilterableProductTable.jsx > ...  
1 import PropTypes from "prop-types";  
2  
3 import SearchBar from "./SearchBar";  
4 import ProductTable from "./ProductTable";  
5  
6 export default function FilterableProductTable(  
7   { products, filterData, filterAction, inStockOnlyData, inStockOnlyAction } ) {  
8   return (  
9     <div>  
10       <SearchBar  
11         filterText={filterData}  
12         inStockOnly={inStockOnlyData}  
13         onFilterTextChange={filterAction}  
14         onInStockOnlyChange={inStockOnlyAction}  
15       />  
16       <ProductTable  
17         products={products}  
18         filterText={filterData}  
19         inStockOnly={inStockOnlyData}  
20       />  
21     </div>  
22   );  
23 }  
24  
25 FilterableProductTable.propTypes = {  
26   products: PropTypes.object.isRequired,  
27   filterData: PropTypes.string.isRequired,  
28   filterAction: PropTypes.func.isRequired,  
29   inStockOnlyData: PropTypes.bool.isRequired,  
30   inStockOnlyAction: PropTypes.func.isRequired  
31 };  
32
```

 SearchBar.jsx X

```
src > components >  SearchBar.jsx > ...
1 import PropTypes from "prop-types";
2
3 export default function SearchBar({
4   filterText,
5   inStockOnly,
6   onFilterTextChange,
7   onInStockOnlyChange
8 }) {
9   return (
10     <form>
11       <input
12         type="text"
13         value={filterText} placeholder="Search..." 
14         onChange={(e) => onFilterTextChange(e.target.value)} />
15       <label>
16         <input
17           type="checkbox"
18           checked={inStockOnly}
19           onChange={(e) => onInStockOnlyChange(e.target.checked)} />
20         {' '}
21         Only show products in stock
22       </label>
23     </form>
24   );
25 }
26
27 SearchBar.propTypes = {
28   filterText: PropTypes.string.isRequired,
29   inStockOnly: PropTypes.bool.isRequired,
30   onFilterTextChange: PropTypes.func.isRequired,
31   onInStockOnlyChange: PropTypes.func.isRequired,
32 };
33
```

```
ProductTable.jsx X  
src > components > ProductTable.jsx > ProductTable > constructor  
1 import PropTypes from "prop-types";  
2  
3 import ProductRow from "./ProductRow";  
4 import ProductCategoryRow from "./ProductCategoryRow";  
5  
6 export default function ProductTable({ products, filterText, inStockOnly }) {  
7   const rows = [];  
8   let lastCategory = null;  
9  
10  products.forEach((product) => {  
11    if (  
12      product.name.toLowerCase().indexOf(  
13        filterText.toLowerCase()  
14      ) === -1  
15    ) {  
16      return;  
17    }  
18    if (inStockOnly && !product.stocked) {  
19      return;  
20    }  
21    if (product.category !== lastCategory) {  
22      rows.push(  
23        <ProductCategoryRow  
24          category={product.category}  
25          key={product.category} />  
26      );  
27    }  
28    rows.push(  
29      <ProductRow  
30        product={product}  
31        key={product.name} />  
32    );  
33    lastCategory = product.category;  
34  });  
35}
```

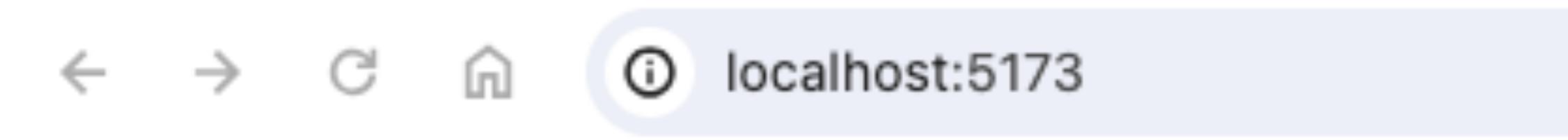
```
35
36     return (
37         <table>
38             <thead>
39                 <tr>
40                     <th>Name</th>
41                     <th>Price</th>
42                 </tr>
43             </thead>
44             <tbody>{rows}</tbody>
45         </table>
46     );
47
48
49 ProductTable.propTypes = {
50     products: PropTypes.object.isRequired,
51     filterText: PropTypes.string.isRequired,
52     inStockOnly: PropTypes.bool.isRequired
53 };
54
```

 ProductCategoryRow.jsx X

```
src > components >  ProductCategoryRow.jsx > ...
1  import PropTypes from "prop-types";
2
3  export default function ProductCategoryRow({ category }) {
4      return (
5          <tr>
6              <th colSpan="2">
7                  {category}
8              </th>
9          </tr>
10     );
11 }
12
13 ProductCategoryRow.propTypes = {
14     category: PropTypes.string.isRequired,
15 };
16
```

```
ProductRow.jsx X  
src > components > ProductRow.jsx > ...  
1 import PropTypes from "prop-types";  
2  
3 export default function ProductRow({ product }) {  
4     const name = product.stocked ? product.name :  
5           
6             {product.name}  
7           
8       
9     return (  
10        |  
11             {name} |  
12             {product.price} |  
13        
  
14    );  
15 }  
16  
17 ProductRow.propTypes = {  
18     product: PropTypes.object.isRequired,  
19 };  
20
```

FORM AND ON CHANGE



App Form

Form to Input Data

Name: Submit

The Input Data is[]

The screenshot shows a code editor interface with the following details:

- EXPLORER View:** Shows the project structure:
 - REACT-S02**: Root folder
 - node_modules**: Subfolder
 - src**: Subfolder
 - assets**: Subfolder
 - Form.jsx**: File (status: 2, U)
 - ShowName.jsx**: File (status: U)
 - App.jsx**: File (status: M) - This file is selected.
 - main.jsx**: File
 - .eslintrc.cjs**: Configuration file
 - .gitignore**: Ignore file
 - index.html**: HTML file
 - package-lock.json**: Lock file
 - package.json**: Project configuration file
 - README.md**: Readme file
 - vite.config.js**: Vite configuration file
- Code Editor View:** Displays the content of **App.jsx**:

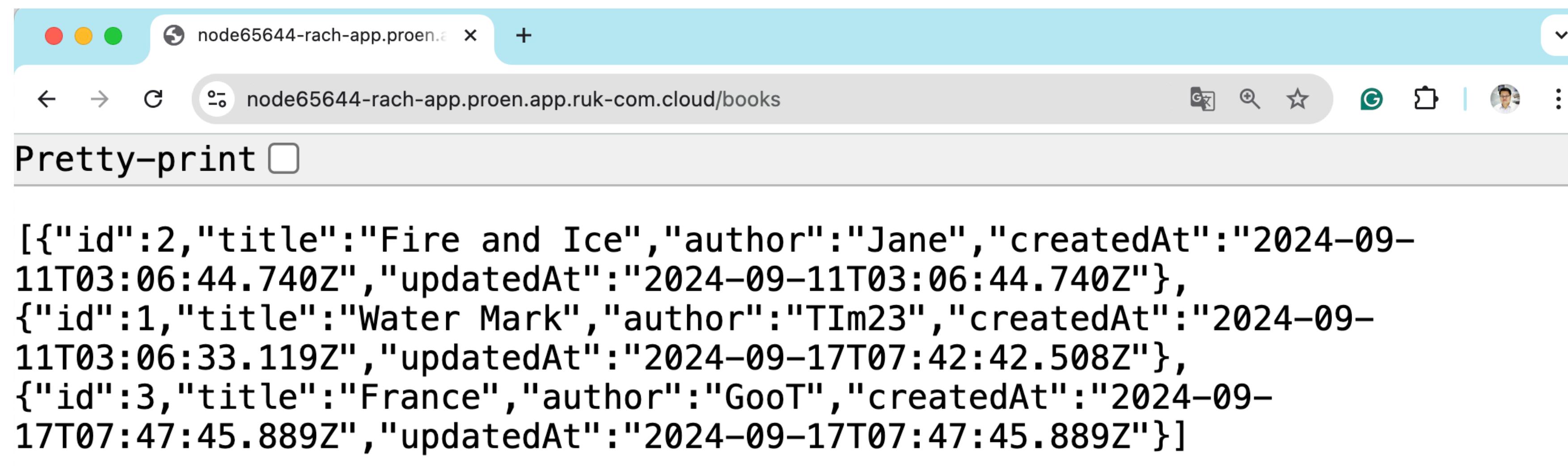
```
src > App.jsx > ...
1 import Form from "./assets/Form";
2
3 function App() {
4     return (
5         <div>
6             <h1>App Form</h1>
7             <Form />
8         </div>
9     );
10
11
12 export default App;
13 }
```

```
Form.jsx  X
src > assets > Form.jsx > ...
1 import { useState } from "react";
2 import ShowName from "./ShowName";
3
4 export default function Form() {
5   const [name, setName] = useState("");
6   const [displayName, setDisplayName] = useState(false);
7
8   const handleChange = (event) => {
9     console.log(event);
10    setName(event.target.value);
11  };
12
13   const handleSubmit = (event) => {
14     event.preventDefault();
15     alert("A name was submitted: " + name);
16     console.log("A name was submitted: " + name);
17
18     name.length >= 5 ? setDisplayName(true) : setDisplayName(false);
19   };
20
21   return (
22     <div>
23       <h1>Form to Input Data</h1>
24       <form>
25         <label>
26           Name:
27           <input type="text" value={name} onChange={handleChange} />
28         </label>
29         <input type="submit" value="Submit" onClick={handleSubmit} />
30       </form>
31       <ShowName name={name} show={displayName} />
32     </div>
33   );
34 }
35
```

```
>ShowName.jsx M X  
src > assets > ShowName.jsx > ...  
1 import PropTypes from "prop-types";  
2  
3 export default function ShowName({ name, show }) {  
4   console.log("name:", name);  
5   console.log("show:", show);  
6  
7   return <div>{show && <h2>Your name is: {name}</h2>}</div>;  
8 }  
9  
10 ShowName.propTypes = {  
11   name: PropTypes.string.isRequired,  
12   show: PropTypes.bool.isRequired,  
13 };  
14
```

```
▼ SyntheticBaseEvent {_reactName: 'onChange', _targetInst: null, type: 'change',
  bubbles: true
  cancelable: false
  currentTarget: null
  defaultPrevented: false
  eventPhase: 3
  ▶ isDefaultPrevented: f functionThatReturnsFalse()
  ▶ isPropagationStopped: f functionThatReturnsFalse()
  isTrusted: true
  ▶ nativeEvent: InputEvent {isTrusted: true, data: 'h', isComposing: false, inp
    ▼ target: input
      value: "Anirach"
      ▶ InputEvent
    ▶ __reactEvents$m869wjrdwn: Set(1) {'invalid__bubble'}
    ▶ __reactFiber$m869wjrdwn: FiberNode {tag: 5, key: null, elementType: 'input'
    ▶ __reactProps$m869wjrdwn: {type: 'text', value: 'Anirach', onChange: f}
    ▶ _valueTracker: {getValue: f, setValue: f, stopTracking: f}
```

REACT WITH API



A screenshot of a web browser window titled "node65644-rach-app.proen.a...". The address bar shows the URL "node65644-rach-app.proen.app.ruk-com.cloud/books". The main content area is titled "Pretty-print" and contains the following JSON data:

```
[{"id":2,"title":"Fire and Ice","author":"Jane","createdAt":"2024-09-11T03:06:44.740Z","updatedAt":"2024-09-11T03:06:44.740Z"}, {"id":1,"title":"Water Mark","author":"TIm23","createdAt":"2024-09-11T03:06:33.119Z","updatedAt":"2024-09-17T07:42:42.508Z"}, {"id":3,"title":"France","author":"GooT","createdAt":"2024-09-17T07:47:45.889Z","updatedAt":"2024-09-17T07:47:45.889Z"}]
```

The screenshot shows the VS Code interface. On the left is the Explorer sidebar, which lists the project structure. The 'src' folder is selected. Inside 'src', there are files: App.jsx, main.jsx, .eslintrc.cjs, .gitignore, index.html, package-lock.json, package.json, README.md, and vite.config.js. The status bar at the bottom indicates there are 2 changes pending. The main editor area shows the 'App.jsx' file with the following code:

```
1 function App() {  
2   return (  
3     <div>  
4       <h1>Hello</h1>  
5       <h2> My First React App.</h2>  
6     </div>  
7   )  
8 }  
9  
10 export default App  
11  
12
```



Hello

My First React App.

The screenshot shows a web browser window with the URL `localhost:5173`. The page title is "Books". The main content is a table listing three books:

TITLE	AUTHOR	ACTIONS
Fire and Ice	Jane	View Edit Delete
Water Mark	TIm23	View Edit Delete
France	GooT	View Edit Delete

At the bottom right of the table, there is a button labeled "Create New Book". The browser interface includes standard navigation icons like back, forward, and search.

TITLE	AUTHOR	ACTIONS
Fire and Ice	Jane	View Edit Delete
Water Mark	TIm23	View Edit Delete
France	GooT	View Edit Delete



View Book

TITLE	AUTHOR
Fire and Ice	Jane

[Edit](#) [Delete](#) [Back to List](#)

A screenshot of a web browser window. The address bar shows 'localhost:5173'. The page title is 'Create Book'. The main content is a form with two fields: 'Title:' and 'Author:', each with a grey input field. Below the fields is a 'Save' button. At the bottom is a 'Back to List' link.

localhost:5173

Create Book

Title:

Author:

Save

Back to List

localhost:5173

Edit Book

Title:
Water Mark

Author:
TIm23

Save

Back to List

The screenshot shows the VS Code interface with the following details:

- EXPLORER View:** Shows the project structure under "REACT-BOOKS".
 - node_modules
 - src
 - components
 - BookForm.jsx
 - BookList.jsx
 - ViewBook.jsx
 - App.jsx (highlighted)
 - main.jsx
 - style
 - App.css
 - .eslintrc.cjs
 - .gitignore
 - index.html
 - package-lock.json
 - package.json
 - README copy.md
 - README.md
 - vite.config.js
- Code Editor View:** Displays the content of `App.jsx`.

```
src > App.jsx > [o] App
1 import { useState, useEffect } from 'react';
2 import axios from 'axios';
3 import BookList from './components/BookList';
4 import ViewBook from './components/ViewBook';
5 import BookForm from './components/BookForm';
6
7 const API_URL = 'https://node65644-rach-app.proen.app.ruk-com.cloud/books';
8
9 const App = () => {
10   const [books, setBooks] = useState([]);
11   const [selectedBook, setSelectedBook] = useState(null);
12   const [viewMode, setViewMode] = useState('list'); // 'list', 'view', 'edit'
13   const [loading, setLoading] = useState(true);
14   const [error, setError] = useState(null);
15
16   // Function to handle API errors
17   const handleError = (err) => {
18     if (err.response) {
19       // Server responded with a status code outside 2xx range
20       setError(`Error: ${err.response.status} - ${err.response.data.message}`);
21     } else if (err.request) {
22       // Request was made but no response received
23       setError('Network error: No response received from server.');
24     }
25   }
26
27   useEffect(() => {
28     axios.get(API_URL)
29       .then(response => {
30         setBooks(response.data);
31         setLoading(false);
32       })
33       .catch(handleError);
34   }, []);
35
36   return (
37     <div>
38       <BookList books={books} selectedBook={selectedBook} setViewMode={setViewMode} />
39       <ViewBook selectedBook={selectedBook} viewMode={viewMode} setViewMode={setViewMode} />
40       <BookForm selectedBook={selectedBook} setSelectedBook={setSelectedBook} setViewMode={setViewMode} />
41     </div>
42   );
43 }
```

```
src > ⚛ App.jsx > 🏃 App > 🕸️ useEffect() callback
  1 import { useState, useEffect } from 'react';
  2 import axios from 'axios';
  3 import BookList from './components/BookList';
  4 import ViewBook from './components/ViewBook';
  5 import BookForm from './components/BookForm';
  6
  7 const API_URL = 'https://node65644-rach-app.proen.app.ruk-com.cloud/books';
  8
  9 const App = () => {
10   const [books, setBooks] = useState([]);
11   const [selectedBook, setSelectedBook] = useState(null);
12   const [viewMode, setViewMode] = useState('list'); // 'list', 'view', 'edit'
13   const [loading, setLoading] = useState(true);
14   const [error, setError] = useState(null);
15
16   // Function to handle API errors
17   const handleError = (err) => {
18     if (err.response) {
19       // Server responded with a status code outside 2xx range
20       setError(`Error: ${err.response.status} - ${err.response.data.message}`);
21     } else if (err.request) {
22       // Request was made but no response received
23       setError('Network error: No response received from server.');
24     } else {
25       // Something else caused the error
26       setError(`Error: ${err.message}`);
27     }
28   };
29
```

```
29
30 // Fetch books when component mounts
31 useEffect(() => {
32   // Fetch books from API
33   const fetchBooks = async () => {
34     try {
35       setLoading(true);
36       const response = await axios.get(API_URL);
37       setBooks(response.data);
38       setError(null); // Clear any previous errors
39       setLoading(false);
40     } catch (err) {
41       handleError(err);
42       setLoading(false);
43     }
44   };
45
46   fetchBooks();
47 }, []);
48
49 const handleView = (id) => {
50   setSelectedBook(books.find(book => book.id === id));
51   setViewMode('view');
52 };
53
54 const handleEdit = (id) => {
55   setSelectedBook(books.find(book => book.id === id) || null);
56   setViewMode('edit');
57 };
58
59 const handleDelete = async (id) => {
```

```
59  const handleDelete = async (id) => {
60    try {
61      await axios.delete(`/${API_URL}/${id}`);
62      setBooks(books.filter((book) => book.id !== id));
63      setViewMode('list');
64      setError(null); // Clear any previous errors
65    } catch (err) {
66      handleError(err);
67    }
68  };
69
70  const handleSave = async (book) => {
71    try {
72      if (book.id) {
73        // Update existing book
74        await axios.put(`/${API_URL}/${book.id}`, book);
75        setBooks(books.map((b) => (b.id === book.id ? book : b)));
76      } else {
77        // Create new book
78        const response = await axios.post(API_URL, book);
79        setBooks([...books, response.data]);
80      }
81      setViewMode('list');
82      setError(null); // Clear any previous errors
83    } catch (err) {
84      handleError(err);
85    }
86  };
87
88  const handleBack = () => {
89    setViewMode('list');
```

```
89 |     setViewMode('list');
90 | };
91 |
92 | // Function to handle creating a new book
93 | const handleCreateNewBook = () => {
94 |     setSelectedBook(null); // Reset selectedBook to null for new book creation
95 |     setViewMode('edit');    // Switch to 'edit' mode to show the form
96 | };
97 |
98 | // Rendering loading and error states
99 | if (loading) {
100 |     return <div>Loading...</div>;
101 | }
102 |
103 | if (error) {
104 |     return <div style={{ color: 'red' }}>{error}</div>;
105 | }
106 |
107 | return (
108 |     <div>
109 |         {viewMode === 'list' && (
110 |             <div>
111 |                 <BookList
112 |                     books={books}
113 |                     onView={handleView}
114 |                     onEdit={handleEdit}
115 |                     onDelete={handleDelete}
116 |                     onCreate={handleCreateNewBook} // Pass onCreate prop to BookList
117 |                 />
118 |             </div>

```

```
118     </div>
119   )
120   {viewMode === 'view' && (
121     <ViewBook
122       book={selectedBook}
123       onEdit={handleEdit}
124       onDelete={handleDelete}
125       onBack={handleBack}
126     />
127   )
128   {viewMode === 'edit' && (
129     <BookForm book={selectedBook} onSave={handleSave} onBack={handleBack} />
130   )
131   </div>
132 );
133 );
134
135 export default App;
136
```

```
src > components > 📚 BookList.jsx > [🔗] BookList
```

```
1 import PropTypes from 'prop-types'; // Import PropTypes
2
3 const BookList = ({ books, onView, onEdit, onDelete, onCreate }) => {
4   return (
5     <div className="book-list-container">
6       <h1 className="center">Books</h1> /* Center the title */
7       <table className="book-table"> /* Use global table styles */
8         <thead>
9           <tr>
10             <th>TITLE</th>
11             <th>AUTHOR</th>
12             <th>ACTIONS</th>
13           </tr>
14         </thead>
15         <tbody>
16           {books.map((book) => (
17             <tr key={book.id}>
18               <td>{book.title}</td>
19               <td>{book.author}</td>
20               <td className="action-buttons">
21                 <button onClick={() => onView(book.id)}>View</button>
22                 <button onClick={() => onEdit(book.id)}>Edit</button>
23                 <button onClick={() => onDelete(book.id)}>Delete</button>
24               </td>
25             </tr>
26           ))}
27         </tbody>
28       </table>
29
30      /* Center the Create New Book button */
31      <div className="button-container">
32        <button className="create-button" onClick={onCreate}>Create New Book</button>
```

```
32     |     <button className="create-button" onClick={onCreate}>Create New Book</button>
33     |   </div>
34   </div>
35 );
36 };
37
38 BookList.propTypes = {
39   books: PropTypes.array.isRequired,
40   onView: PropTypes.func.isRequired,
41   onEdit: PropTypes.func.isRequired,
42   onDelete: PropTypes.func.isRequired,
43   onCreate: PropTypes.func.isRequired, // Add onCreate as a required prop
44 };
45
46 export default BookList;
```

src > components >  ViewBook.jsx >  ViewBook

```
1 import PropTypes from 'prop-types';
2
3 const ViewBook = ({ book, onEdit, onDelete, onBack }) => {
4   const handleDelete = () => {
5     onDelete(book.id);
6   };
7
8   return (
9     <div className="view-book-container">
10      <h1 className="center">View Book</h1>
11      <table className="book-table">
12        <thead>
13          <tr>
14            <th>TITLE</th>
15            <th>AUTHOR</th>
16          </tr>
17        </thead>
18        <tbody>
19          <tr>
20            <td>{book.title}</td>
21            <td>{book.author}</td>
22          </tr>
23        </tbody>
24      </table>
```

```
23     </tbody>
24   </table>
25   <div className="button-container">
26     <button className="edit-button" onClick={() => onEdit(book.id)}>Edit</button>
27     <button className="delete-button" onClick={handleDelete}>Delete</button>
28     <button className="back-button" onClick={onBack}>Back to List</button>
29   </div>
30 </div>
31 );
32 };
33
34 ViewBook.propTypes = {
35   book: PropTypes.object.isRequired,
36   onEdit: PropTypes.func.isRequired,
37   onDelete: PropTypes.func.isRequired,
38   onBack: PropTypes.func.isRequired,
39 };
40
41 export default ViewBook;
```

```
src > components > 📚 BookForm.jsx > 🏷 BookForm
  1 import { useState, useEffect } from 'react';
  2 import PropTypes from 'prop-types';
  3
  4 const BookForm = ({ book, onSave, onBack }) => {
  5   const [title, setTitle] = useState('');
  6   const [author, setAuthor] = useState('');
  7
  8   useEffect(() => {
  9     if (book) {
10       setTitle(book.title);
11       setAuthor(book.author);
12     }
13   }, [book]);
14
15   const handleSubmit = (e) => {
16     e.preventDefault();
17     onSave({ id: book ? book.id : null, title, author });
18   };
19
20   return (
21     <div>
22       <h1>{book ? 'Edit Book' : 'Create Book'}</h1>
23       <form onSubmit={handleSubmit}>
24         <div>
25           <label>Title:</label>
26           <input
27             type="text"
28             value={title}
29             onChange={(e) => setTitle(e.target.value)}
30           />
31         </div>
32         <div>
```

```
30      >
31    </div>
32    <div>
33      <label>Author:</label>
34      <input
35        type="text"
36        value={author}
37        onChange={(e) => setAuthor(e.target.value)}
38      />
39    </div>
40    <button type="submit">Save</button>
41  </form>
42  <div className="button-container">
43    <button onClick={onBack}>Back to List</button>
44  </div>
45
46</div>
47);
48};
49
50 BookForm.propTypes = {
51   book: PropTypes.object,
52   onSave: PropTypes.func.isRequired,
53   onBack: PropTypes.func.isRequired,
54 };
55
56 export default BookForm;
57
```

```
style >  App.css > ...
1  /* Set font family and size for the whole page */
2  body {
3      font-family: 'Open Sans', sans-serif;
4      font-size: 16px;
5  }
6
7  /* Center the main heading */
8  h1 {
9      text-align: center;
10     margin-top: 30px;
11     margin-bottom: 50px;
12     font-size: 36px;
13     color: #2C3E50;
14 }
15
16 /* Style the table */
17 table {
18     border-collapse: collapse;
19     width: 100%;
20 }
21
22 th, td {
23     text-align: left;
24     padding: 16px;
25 }
26
27 th {
28     background-color: #2C3E50;
29     color: white;
30     font-weight: 600;
31     text-transform: uppercase;
32 }
```

```
32 }
33
34 tr:nth-child(even) {
35   background-color: #f5f5f5;
36 }
37
38 /* Style the form */
39 form {
40   width: 50%;
41   margin: auto;
42   border: 2px solid #2C3E50;
43   padding: 20px;
44   border-radius: 5px;
45 }
46
47 label {
48   display: block;
49   font-weight: bold;
50   margin-bottom: 10px;
51 }
52
53 input[type="text"] {
54   width: 100%;
55   padding: 8px;
56   box-sizing: border-box;
57   border-radius: 5px;
58   border: none;
59   background-color: #f2f2f2;
60 }
61
62 input[type="submit"] {
```

```
61
62  input[type="submit"] {
63    background-color: #2C3E50;
64    color: white;
65    border: none;
66    padding: 10px 16px;
67    font-size: 16px;
68    cursor: pointer;
69    border-radius: 5px;
70    margin-top: 10px;
71  }
72
73 /* Style the links */
74 a {
75   display: inline-block;
76   margin: 10px;
77   color: #2C3E50;
78   text-decoration: none;
79   font-size: 18px;
80   font-weight: 600;
81   border-bottom: 2px solid transparent;
82   transition: border-bottom-color 0.2s;
83 }
84
85 a:hover {
86   border-bottom-color: #2C3E50;
87 }
88
89 .link-center {
90   text-align: center;
91   margin-top: 20px;
92 }
```

```
92 }
93
94 .link-center a {
95   display: inline-block;
96   margin: 10px;
97   color: #2C3E50;
98   text-decoration: none;
99   font-size: 18px;
100  font-weight: 600;
101  border-bottom: 2px solid transparent;
102  transition: border-bottom-color 0.2s;
103 }
104
105 /* Center the button */
106 .button-container {
107   text-align: center; /* Center the Create New Book button */
108   margin-top: 20px;
109 }
110
111 .link-center a:hover {
112   border-bottom-color: #2C3E50;
113 }
114
115 /* Style the table */
116 .book-table {
117   border-collapse: collapse;
118   width: 75%;
119   margin: 0 auto; /* center the table */
120   margin-bottom: 20px;
121   border-bottom: 2px solid #2C3E50;
122 }
```

```
122    }
123
124    th, td {
125        text-align: left;
126        padding: 16px;
127    }
128
```

The screenshot shows a web application titled "Books" running on "localhost:5173". The interface includes a header with navigation icons and a main content area displaying a list of books with columns for Title, Author, and Actions.

TITLE	AUTHOR	ACTIONS
Fire and Ice	Jane	View Edit Delete
Water Mark	Tlm23	View Edit Delete
France	GooT	View Edit Delete

[Create New Book](#)

THANK YOU

- React Introduction
- Fundamentals of React
- Create React Project
- React props/state
- Thinking React