

ReactJS. Front to backend. BT 2022

<https://github.com/PacktPublishing/React-Front-to-Back-2022>

Table of Contents

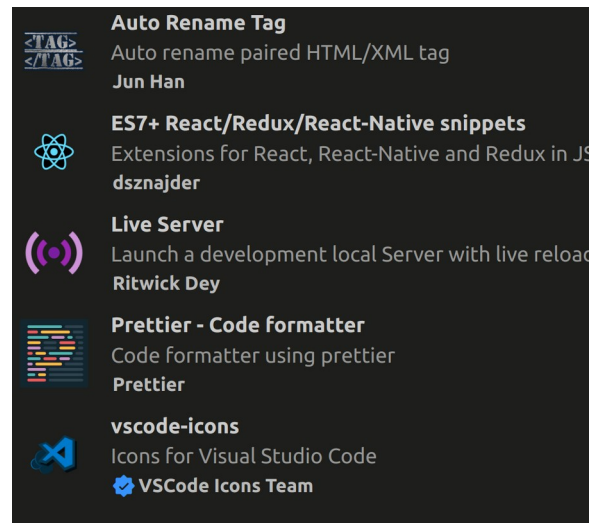
1 Hints.....	3
2 React basics and JSX.....	4
2.1 Feedback project intro.....	4
2.2 Create react app.....	4
2.3 Initializing React.....	4
2.3.1 How to create a new tag on your page with ReactDOM and vanillaJS?.....	4
2.3.2 How to create a main App.js file (main component)?.....	4
2.3.3 How to add a main css file?.....	5
2.4 Intro to JSX.....	5
2.5 Dynamic values and lists in JSX.....	5
2.6 Conditionals in JSX.....	5
3 Components, props and state.....	6
3.1 Creating a Component and Props.....	6
3.1.1 How to create a new Component and use in in Main Component?.....	6
3.1.2 How to use Props with a Component?.....	6
3.1.3 How to destructure and use props?.....	7
3.1.4 How to setup default props?.....	7
3.1.5 How to setup and use prop types?.....	7
3.2 Adding styles to the Component.....	8
3.2.1 How to add inline styles?.....	8
3.2.2 How to add styles using a variable?.....	8
3.2.3 How to add styles using props and a variable?.....	8
3.2.4 How to add styles using default props?.....	9
3.3 Component level State and useState hook.....	10
3.3.1 How to add useState hook to your component?.....	10
3.3.2 How to setup useState hook?.....	10
3.3.3 How to change useState hook variable to a hardcoded value?.....	10
3.3.4 How to change useState hook variable using a previous value?.....	11
3.4 Managing Global State.....	12
3.4.1 How to use additional files to store and use any additional data?.....	12
3.4.2 How to create a component that uses another component?.....	12
3.5 Card component and Conditional styles.....	13
3.5.1 How to implement conditional class?.....	13
3.5.2 How to implement conditional style?.....	14
3.5.2 How to setup propTypes for an array of objects?.....	15

1 Hints

[tagName].[className]	div.main-element	<div className="main-element"></div>
[tagName]#[idName]	div#root	<div id="root"></div>
ul>li*[quantity]	ul>li*25	unordered list with 25 list items will be created
#[idName].[container]	#root.container	<div id="root" className="container"></div>
[tagName]	button	<button></button>

VSCode settings:

```
{
  "editor.fontSize": 18,
  "editor.tabSize": 3,
  "editor.bracketPairColorization.independentColorPoolPerBracketType": true,
  "editor.guides.bracketPairs": true,
  "editor.guides.bracketPairsHorizontal": true,
  "editor.wordWrap": "on",
  "workbench.colorTheme": "Monokai",
  "window.zoomLevel": 2,
  "prettier.arrowParens": "avoid",
  "prettier.jsxSingleQuote": true,
  "prettier.printWidth": 50,
  "prettier.singleQuote": true,
  "editor.formatOnSave": true,
  "[javascript]": {
    "editor.defaultFormatter": "esbenp.prettier-vscode"
  },
  "[jsonc]": {
    "editor.defaultFormatter": "esbenp.prettier-vscode"
  },
  "[json]": {
    "editor.defaultFormatter": "esbenp.prettier-vscode"
  },
  "[javascriptreact]": {
    "editor.defaultFormatter": "esbenp.prettier-vscode"
  }
}
```



You can use snippet **rafce** (it comes from ES7+ React/Redux/React-Native snippets extension) to create a new **React Arrow Function Component with Export**

You can use snippet **impt** (it comes from ES7+ React/Redux/React-Native snippets extension) to **Import Prop Types** – just type **impt** and click **TAB** or **Enter** - It will add line **import PropTypes from 'prop-types'**

2 React basics and JSX

2.1 Feedback project intro

2.2 Create react app

You can create a new ReactJS app using next command:

CLI=> `npx create-react-app feedback-app --use-npm`

You can start your ReactJS app using next command (the terminal should be opened in root folder of your ReactJS app – in this case feedback-app):

CLI=> `npm run server`

Delete all the files from folder feedback-app/src

2.3 Initializing React

Create a main file feedback-app/src/index.js

2.3.1 How to create a new tag on your page with ReactDOM and vanillaJS?

```
import React from 'react';
import ReactDOM from 'react-dom';
ReactDOM.render(<h1>My App</h1>, document.getElementById('root'));
```

2.3.2 How to create a main App.js file (main component)?

Create a main component feedback-app/src/App.js and do some changes in index.js

```
JS index.js U X
feedback-app > src > JS index.js
1  import React from 'react';
2  import ReactDOM from 'react-dom';
3  import App from './App';
4
5  ReactDOM.render(
6    <React.StrictMode>
7      <App />
8    </React.StrictMode>,
9    document.getElementById('root')
10 );
11
```

```
JS App.js U X
feedback-app > src > JS App.js > ...
1  import React from 'react';
2
3  const App = () => {
4    |   return <h1>Hello from the App component</h1>;
5  };
6
7  export default App;
8
```

Feedback app x +

Not secure | 192.168.0.100:3000

Hello from the App component

2.3.3 How to add a main css file?

Create a main css file `feedback-app/src/index.css` and import it to `index.js`

2.4 Intro to JSX

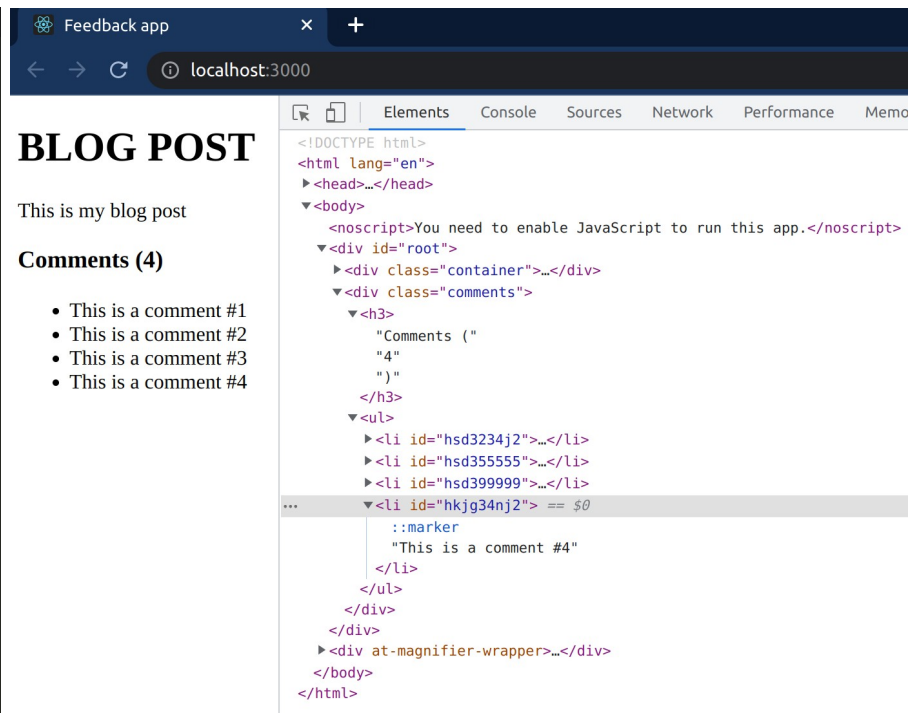
Each React component should return only 1 main/root/parent element. It's such a common case to use fragment element as a parent element `<>some JSX</>`

You can't use `class` attribute (as in HTML) instead of it you have to use `className`.

You can't use `for` attribute (as in HTML) instead of it you have to use `htmlFor`.

2.5 Dynamic values and lists in JSX

```
1 import React from 'react';
2
3 const App = () => {
4   const title = 'Blog Post';
5   const body = 'This is my blog post';
6   const comments = [
7     {
8       id: 'hsd3234j2',
9       text: 'This is a comment #1',
10    },
11    {
12      id: 'hsd355555',
13      text: 'This is a comment #2',
14    },
15    {
16      id: 'hsd399999',
17      text: 'This is a comment #3',
18    },
19    {
20      id: 'hkjg34nj2',
21      text: 'This is a comment #4',
22    },
23  ];
24  return (
25    <>
26      <div className='container'>
27        <h1>{title.toUpperCase()}</h1>
28        <p>{body}</p>
29      </div>
30      <div className='comments'>
31        <h3>Comments ({comments.length})</h3>
32        <ul>
33          {comments.map(comment => (
34            <li id={comment.id}>
35              {comment.text}
36            </li>
37          ))}
38        </ul>
39      </div>
40    </>
41  );
42 };
43
44 export default App;
```



2.6 Conditionals in JSX

3 Components, props and state

3.1 Creating a Component and Props

3.1.1 How to create a new Component and use in in Main Component?

Create a new folder to store all the components feedback-app/src/**components**. Create your first Component – Header.jsx (or Header.js)

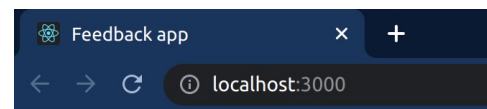
You can use snippet **rafce** (it comes from ES7+ React/Redux/React-Native snippets extension) to create a new **React Arrow Function Component** with **Export** – just type **rafce** and click **TAB** or **Enter**

```
Header.jsx U X
src > components > Header.jsx > ...
1  import React from 'react';
2
3  const Header = () => {
4    return <div>Header</div>;
5  };
6
7  export default Header;
8
```

```
Header.jsx U X
src > components > Header.jsx > ...
1  const Header = () => {
2    return (
3      <header>
4        <div className='container'>
5          <h2>Feedback UI</h2>
6        </div>
7      </header>
8    );
9  };
10 export default Header;
11
```

import it in App.js

```
App.jsx U X
src > App.jsx > ...
1  import Header from './components/Header';
2  const App = () => (
3    <>
4      <Header />
5      <div className='container'>
6        <h1>My app</h1>
7      </div>
8    </>
9  );
10
11 export default App;
12
```



Feedback UI

My app

3.1.2 How to use Props with a Component?

```
App.jsx U X Header.jsx U
src > App.jsx > ...
1  import Header from './components/Header';
2  const App = () => (
3    <>
4      <Header anyAttributeName='Hello world!' />
5      <div className='container'>
6        <h1>My app</h1>
7      </div>
8    </>
9  );
10
11 export default App;
12
```

Hello world!

My app

```
App.jsx U Header.jsx U X
src > components > Header.jsx > ...
1  const Header = props => {
2    return (
3      <header>
4        <div className='container'>
5          <h2>{props.anyAttributeName}</h2>
6        </div>
7      </header>
8    );
9  };
10 export default Header;
11
```

3.1.3 How to destructure and use props?

```
App.jsx | Header.jsx |
src > components > Header.jsx > ...
1  const Header = ({ anyAttributeName }) => {
2
3    return (
4      <header>
5        <div className='container'>
6          <h2>{anyAttributeName}</h2>
7        </div>
8      </header>
9    );
10 }
11 export default Header;
```

3.1.4 How to setup default props?

```
App.jsx | Header.jsx |
src > App.jsx > ...
1  import Header from './components/Header';
2  const App = () => {
3
4    <Header />
5    <div className='container'>
6      <h1>My app</h1>
7    </div>
8  </>
9  };
10
11 export default App;
```

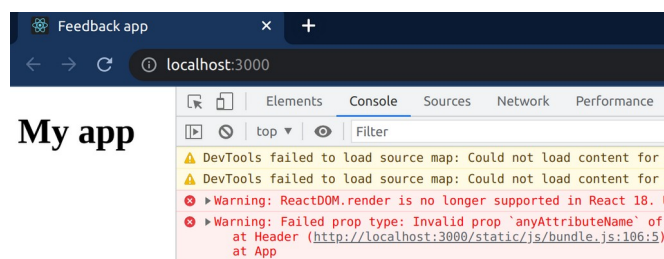
```
App.jsx | Header.jsx |
src > components > Header.jsx > ...
1  const Header = ({ anyAttributeName }) => {
2
3    return (
4      <header>
5        <div className='container'>
6          <h2>{anyAttributeName}</h2>
7        </div>
8      </header>
9    );
10 };
11 Header.defaultProps = {
12   anyAttributeName: 'Feedback UI!',
13 };
14 export default Header;
```

3.1.5 How to setup and use prop types?

You can use snippet **impt** (it comes from ES7+ React/Redux/React-Native snippets extension) to **Import Prop Types** – just type **impt** and click **TAB** or **Enter**. It will add line **import PropTypes from 'prop-types'**

```
App.jsx | Header.jsx |
src > App.jsx > ...
1  import Header from './components/Header';
2  const App = () => {
3
4    <Header anyAttributeName={true} />
5    <div className='container'>
6      <h1>My app</h1>
7    </div>
8  </>
9  };
10
11 export default App;
```

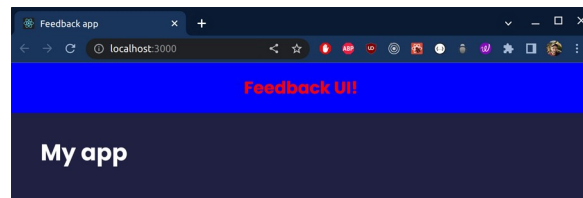
```
App.jsx | Header.jsx |
src > components > Header.jsx > ...
1  import PropTypes from 'prop-types';
2  const Header = ({ anyAttributeName }) => {
3
4    return (
5      <header>
6        <div className='container'>
7          <h2>{anyAttributeName}</h2>
8        </div>
9      </header>
10    );
11 };
12 Header.defaultProps = {
13   anyAttributeName: 'Feedback UI!',
14 };
15 Header.propTypes = {
16   anyAttributeName: PropTypes.string,
17 };
18 export default Header;
```



3.2 Adding styles to the Component

3.2.1 How to add inline styles?

```
src > components > Header.jsx > ...
1 import PropTypes from 'prop-types';
2 const Header = ({ anyAttributeName }) => {
3   return (
4     <header
5       style={{
6         backgroundColor: 'blue',
7         color: 'red',
8       }}
9     >
10    <div className='container'>
11      <h2>{anyAttributeName}</h2>
12    </div>
13    </header>
14  );
15 };
16 Header.defaultProps = {
17   anyAttributeName: 'Feedback UI!',
18 };
19 Header.propTypes = {
20   anyAttributeName: PropTypes.string,
21 };
22 export default Header;
```

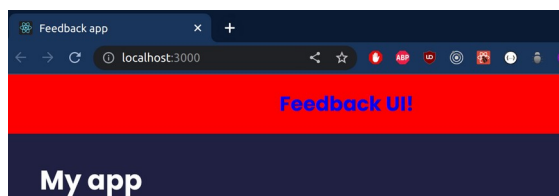


3.2.2 How to add styles using a variable?

```
src > components > Header.jsx > Header
1 import PropTypes from 'prop-types';
2 const Header = ({ anyAttributeName }) => {
3   const headerStyles = {
4     backgroundColor: 'blue',
5     color: 'red',
6   };
7   return (
8     <header style={headerStyles}>
9       <div className='container'>
10        <h2>{anyAttributeName}</h2>
11      </div>
12    </header>
13  );
14 };
15 Header.defaultProps = {
16   anyAttributeName: 'Feedback UI!',
17 };
18 Header.propTypes = {
19   anyAttributeName: PropTypes.string,
20 };
21 export default Header;
```

3.2.3 How to add styles using props and a variable?

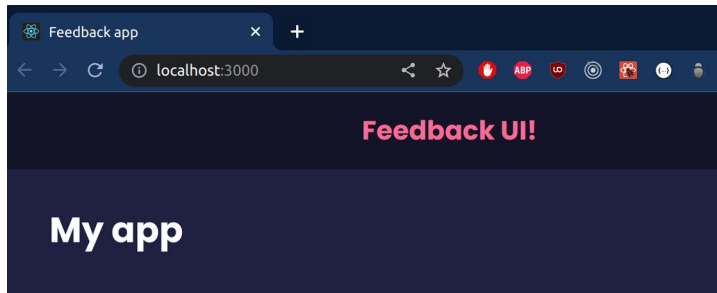
```
src > App.jsx > App
1 import Header from './components/Header';
2 const App = () => {
3   <>
4     <Header bgColor='red' textColor='blue' />
5     <div className='container'>
6       <h1>My app</h1>
7     </div>
8   </>
9 };
10
11 export default App;
```



```
src > components > Header.jsx > Header
1 import PropTypes from 'prop-types';
2 const Header = ({
3   anyAttributeName,
4   bgColor,
5   textColor,
6 }) => {
7   const headerStyles = {
8     backgroundColor: bgColor,
9     color: textColor,
10  };
11   return (
12     <header style={headerStyles}>
13       <div className='container'>
14        <h2>{anyAttributeName}</h2>
15      </div>
16    </header>
17  );
18 };
19 Header.defaultProps = {
20   anyAttributeName: 'Feedback UI!',
21 };
22 Header.propTypes = {
23   anyAttributeName: PropTypes.string,
24 };
25 export default Header;
```


3.2.4 How to add styles using default props?

```
App.jsx  U X
src > App.jsx > ...
1  import Header from './components/Header';
2  const App = () => (
3
4    <Header />
5    <div className='container'>
6      <h1>My app</h1>
7    </div>
8  </>
9  );
10
11 export default App;
12
```

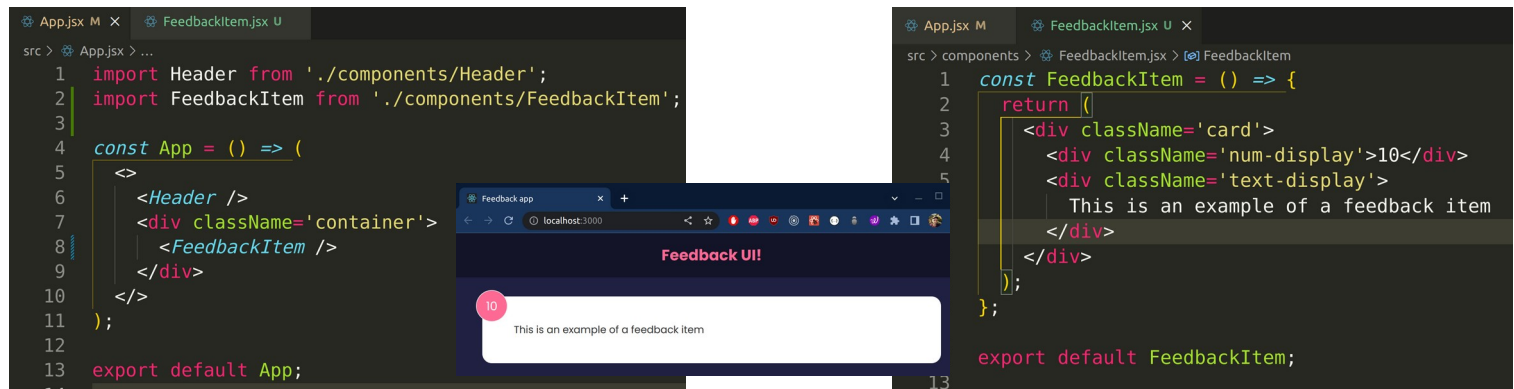


```
Header.jsx  U X
src > components > Header.jsx > ...
1  import PropTypes from 'prop-types';
2  const Header = ({
3    anyAttributeName,
4    bgColor,
5    textColor,
6  }) => {
7    const headerStyles = {
8      backgroundColor: bgColor,
9      color: textColor,
10   };
11   return (
12     <header style={headerStyles}>
13       <div className='container'>
14         <h2>{anyAttributeName}</h2>
15       </div>
16     </header>
17   );
18 };
19 Header.defaultProps = {
20   anyAttributeName: 'Feedback UI!',
21   bgColor: 'rgba(0,0,0,0.4)',
22   textColor: '#ff6a95',
23 };
24 Header.propTypes = {
25   anyAttributeName: PropTypes.string,
26   bgColor: PropTypes.string,
27   textColor: PropTypes.string,
28 };
29 export default Header;
30
```

3.3 Component level State and useState hook

State is just a data that you store and use in your app. There are 2 main types of states: app level (global) states and component level states.

Let's create a new component to display each Feedback and implement it using hardcoded values:



```
src > App.jsx > ...
1 import Header from './components/Header';
2 import FeedbackItem from './components/FeedbackItem';
3
4 const App = () => (
5   <>
6     <Header />
7     <div className='container'>
8       <FeedbackItem />
9     </div>
10   </>
11 );
12
13 export default App;
```

```
src > components > FeedbackItem.jsx > FeedbackItem
1 const FeedbackItem = () => {
2   return (
3     <div className='card'>
4       <div className='num-display'>10</div>
5       <div className='text-display'>
6         This is an example of a feedback item
7       </div>
8     </div>
9   );
10 };
11
12 export default FeedbackItem;
```

3.3.1 How to add useState hook to your component?

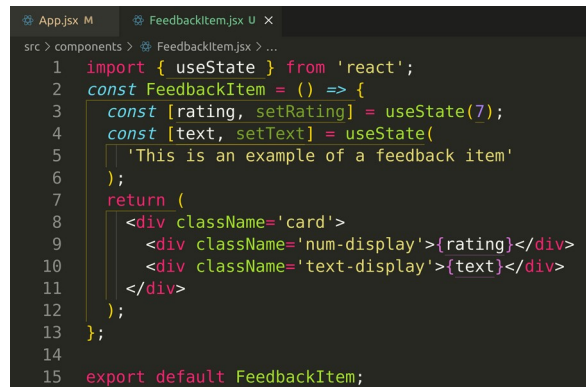
First of all you need to import useState to your component

```
import { useState } from 'react';
```

3.3.2 How to setup useState hook?

Now you need to setup useState (define state variable (OR piece of state) and function that will change/update the state variable) and initial state variable value.

For this example you need to create 2 state variables to work with feedback rating and feedback text



```
src > components > FeedbackItem.jsx > ...
1 import { useState } from 'react';
2 const FeedbackItem = () => {
3   const [rating, setRating] = useState(7);
4   const [text, setText] = useState(
5     'This is an example of a feedback item'
6   );
7   return (
8     <div className='card'>
9       <div className='num-display'>{rating}</div>
10      <div className='text-display'>{text}</div>
11    </div>
12  );
13 };
14
15 export default FeedbackItem;
```

3.3.3 How to change useState hook variable to a hardcoded value?

Let's add a button to your FeedbackItem component and setup a setter to be able to change state variable

```

App.jsx M FeedbackItem.jsx U X
src > components > FeedbackItem.jsx > ...
1 import { useState } from 'react';
2 const FeedbackItem = () => {
3   const [rating, setRating] = useState(7);
4   const [text, setText] = useState(
5     'This is an example of a feedback item'
6   );
7   const handleClick = () => {
8     setRating(100);
9   };
10  return (
11    <div className='card'>
12      <div className='num-display'>{rating}</div>
13      <div className='text-display'>{text}</div>
14      <button onClick={handleClick}>click</button>
15    </div>
16  );
17 };
18
19 export default FeedbackItem;

```

In this example default feedback rating is 7 but click on the button invokes handleClick() function that invokes setter function for rating state variable with parameter 100. It means that after click on the button rating of the feedback will be changed from 7 to 100

3.3.4 How to change useState hook variable using a previous value?

You need to pass into setter another anonymous function that takes in prev variable (it's a default name for working with a previous value) and returns changed value (in this example increased by 1)

```

App.jsx M FeedbackItem.jsx U X
src > components > FeedbackItem.jsx > ...
1 import { useState } from 'react';
2 const FeedbackItem = () => {
3   const [rating, setRating] = useState(7);
4   const [text, setText] = useState(
5     'This is an example of a feedback item'
6   );
7   const handleClick = () => {
8     setRating(prev => prev + 1);
9   };
10  return (
11    <div className='card'>
12      <div className='num-display'>{rating}</div>
13      <div className='text-display'>{text}</div>
14      <button onClick={handleClick}>click</button>
15    </div>
16  );
17 };
18
19 export default FeedbackItem;
20

```

In this example after each click on the button rating of the feedback will be increased by 1

3.4 Managing Global State

To be able to work with App level state (Global) first of all you need to import it to a main component – App.js

3.4.1 How to use additional files to store and use any additional data?

Create a new folder feedback-app/src/**data** to store additional data. Create there a new file – FeedbackData.js – to store data about feedbacks (array of objects). Don't forget to export it.

```
1 const FeedbackData = [
2   {
3     id: 3245,
4     rating: 4,
5     text: 'This is a feedback with ID 3245',
6   },
7   {
8     id: 3577,
9     rating: 6,
10    text: 'This is a feedback with ID 3577',
11  },
12  {
13    id: 1111,
14    rating: 2,
15    text: 'This is a feedback with ID 1111',
16  },
17  {
18    id: 7777,
19    rating: 1,
20    text: 'This is a feedback with ID 7777',
21  },
22  {
23    id: 9999,
24    rating: 9,
25    text: 'This is a feedback with ID 9999',
26  },
27 ];
28 export default FeedbackData;
```

Now you need to import that files to a the main App component using

```
import FeedbackData from './data/FeedbackData';
```

3.4.2 How to create a component that uses another component?

The main idea: you need to create FeedbackList.jsx component that returns list of all the feedbacks. Each item inside of that list is another component – FeedbackItem.jsx

```
1 import { useState } from 'react';
2 import Header from './components/Header';
3 import FeedbackList from './components/FeedbackList';
4 import FeedbackData from './data/FeedbackData';
5
6 const App = () => {
7   const [feedback, setFeedback] =
8     useState(FeedbackData);
9   return (
10     <Header />
11     <div className='container'>
12       <FeedbackList feedback={feedback} />
13     </div>
14   );
15 };
16
17 export default App;
```

```
1 import FeedbackItem from './FeedbackItem';
2
3 const FeedbackList = ({ feedback }) => {
4   if (!feedback || feedback.length === 0) {
5     return <p>No any Feedbacks yet</p>;
6   }
7   return (
8     <div className='feedback-list'>
9       {feedback.map(eachFeedback => {
10         return (
11           <FeedbackItem
12             key={eachFeedback.id}
13             item={eachFeedback}
14           />
15         );
16       })}
17     </div>
18   );
19 };
20 export default FeedbackList;
```

```
1 const FeedbackItem = ({ item }) => {
2   return (
3     <div className='card'>
4       <div className='num-display'>
5         {item.rating}
6       </div>
7       <div className='text-display'>
8         {item.text}
9       </div>
10     </div>
11   );
12 };
13 export default FeedbackItem;
```

workflow: Main component App.js renders FeedbackList component (that takes in an array of all the feedbacks).

FeedbackList component iterates over that array and for each element renders FeedbackItem component (that takes in props). FeedbackItem component renders each feedback element.

3.5 Card component and Conditional styles

Create a new folder `feedback-app/src/components/shared` to store all the shared components. Create there a new file – `Card.js` – to render information about each element in form of Card.

```
src > components > FeedbackItem.jsx > ...
1 import Card from './shared/Card';
2 const FeedbackItem = ({ item }) => {
3   return (
4     <Card>
5       <div className='num-display'>
6         {item.rating}
7       </div>
8       <div className='text-display'>
9         {item.text}
10      </div>
11    </Card>
12  );
13 };
14 export default FeedbackItem;
```

```
src > components > shared > Card.jsx > ...
1 const Card = ({ children }) => (
2   <div className='card'>{children}</div>
3 );
4 export default Card;
```

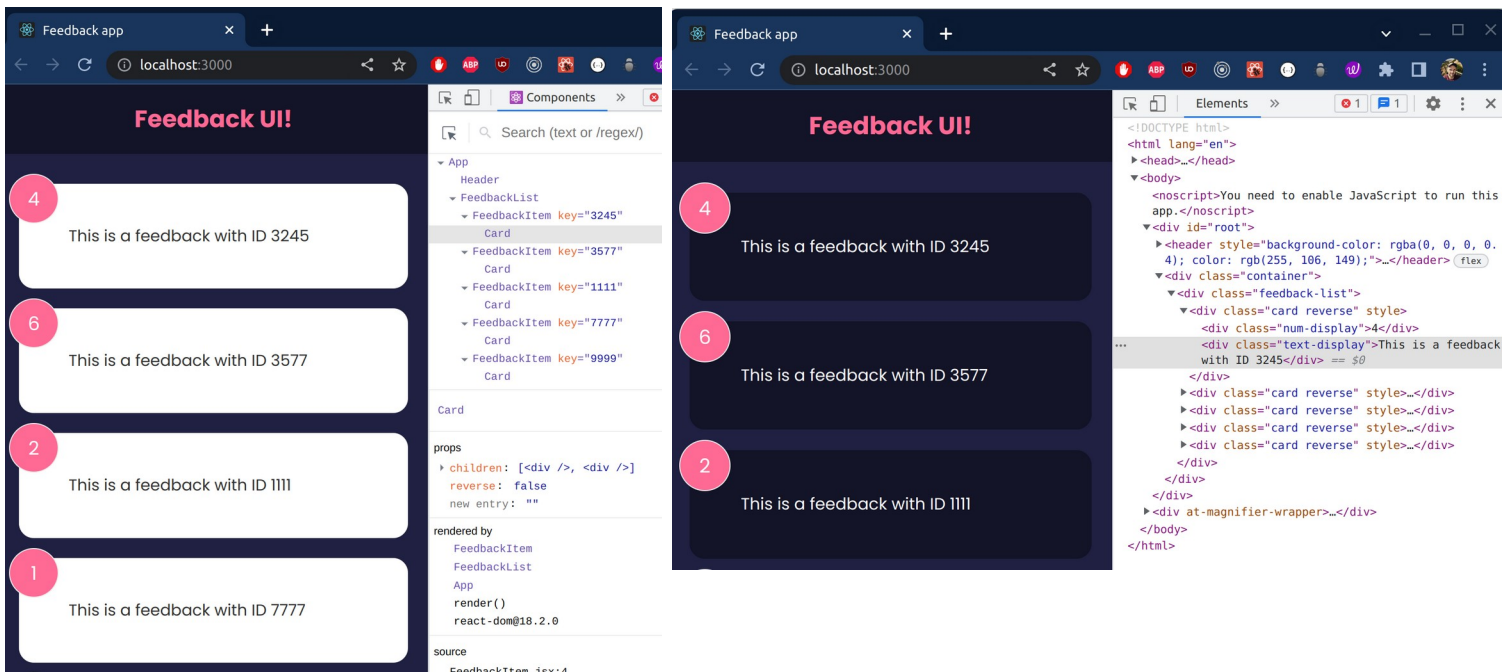
```
.card {
  background-color: #fff;
  color: #333;
  border-radius: 15px;
  padding: 40px 50px;
  margin: 20px 0;
  position: relative;
}
```

3.5.1 How to implement conditional class?

```
src > components > FeedbackItem.jsx > ...
1 import Card from './shared/Card';
2 const FeedbackItem = ({ item }) => {
3   return (
4     <Card reverse={false}>
5       <div className='num-display'>
6         {item.rating}
7       </div>
8       <div className='text-display'>
9         {item.text}
10      </div>
11    </Card>
12  );
13 };
14 export default FeedbackItem;
```

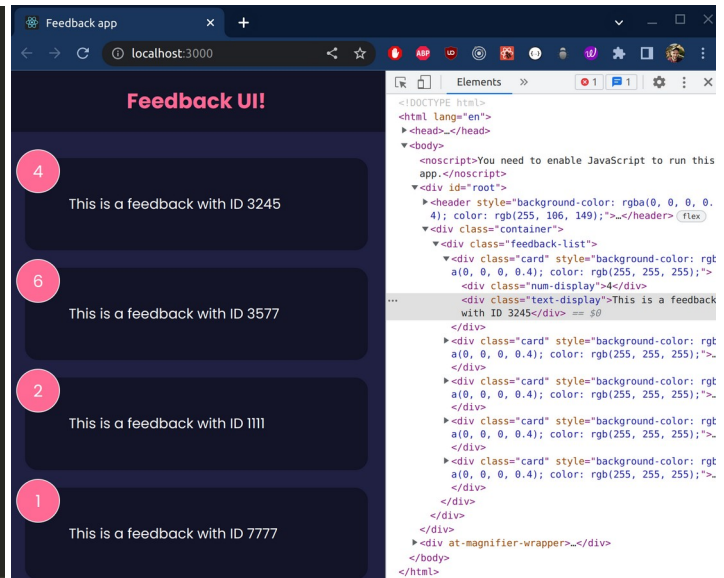
```
src > components > shared > Card.jsx > ...
1 const Card = ({ children, reverse }) => (
2   <div className={`card ${reverse && 'reverse'}`}>
3     {children}
4   </div>
5 );
6 export default Card;
```

```
.card.reverse {
  background-color: rgba(0, 0, 0, 0.4);
  color: #fff;
}
```



3.5.2 How to implement conditional style?

```
src > components > shared > Card.jsx > ...
1  const Card = ({ children, reverse }) => (
2    //Conditional class
3    /* <div className={`card ${reverse && 'reverse'}}`>
4      {children}
5    </div> */
6    ///////////////
7    //Conditional style
8    <div
9      className='card'
10     style={{
11       backgroundColor: reverse
12         ? 'rgba(0,0,0,0.4)'
13         : '#fff',
14       color: reverse ? '#fff' : '#000',
15     }}
16   >
17     {children}
18   </div>
19 );
20 export default Card;
```



Now you can add default props values and propTypes:

```
src > components > shared > Card.jsx > Card
1  import PropTypes from 'prop-types';
2  const Card = ({ children, reverse }) => (
3    //Conditional class
4    <div className={`card ${reverse && 'reverse'}}`>
5      {children}
6    </div>
7    ///////////////
8    //Conditional style
9    /* <div
10     className='card'
11     style={{
12       backgroundColor: reverse
13         ? 'rgba(0,0,0,0.4)'
14         : '#fff',
15       color: reverse ? '#fff' : '#000',
16     }}
17   >
18     {children}
19   </div> */
20 );
21 Card.defaultProps = {
22   reverse: false,
23 };
24 Card.propTypes = {
25   children: PropTypes.node.isRequired,
26   reverse: PropTypes.bool,
27 };
28 export default Card;
```

3.5.2 How to setup propTypes for an array of objects?

```
src > components > FeedbackList.jsx > ...
1  import PropTypes from 'prop-types';
2  import FeedbackItem from './FeedbackItem';
3  const FeedbackList = ({ feedback }) => {
4    if (!feedback || feedback.length === 0) {
5      return <p>No any Feedbacks yet</p>;
6    }
7    return (
8      <div className='feedback-list'>
9        {feedback.map(eachFeedback => {
10          return (
11            <FeedbackItem
12              key={eachFeedback.id}
13              item={eachFeedback}
14            />
15          );
16        })}
17      </div>
18    );
19  };
20  FeedbackList.defaultProps = {
21    feedback: false,
22  };
23  FeedbackList.propTypes = {
24    feedback: PropTypes.arrayOf(
25      PropTypes.shape({
26        id: PropTypes.number.isRequired,
27        text: PropTypes.string.isRequired,
28        rating: PropTypes.number.isRequired,
29      })
30    ),
31  };
32  export default FeedbackList;
33
```