React Components

A FRONTEND LIBRARY

Starter Code

https://1drv.ms/f/s!AtGKdbMmNBGd1GL9UY0KVY0Szrgz

There are two projects Vidly and Counter

Go to specific folder and run

>npm install

Then

>npm start

Using bootstrap in react app

>npm i bootstrap

>npm install font-awesome

Then you can use

Use below code in either index.js or in any component

```
import "bootstrap/dist/css/bootstrap.css";

import "font-awesome/css/font-awesome.css";
```

Function and Class Components

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

Using a Component

<Welcome name="Sara" />

Attributes passed to a component are props

Props are read only

Don’t be afraid to split components into smaller components.

HTML ? FancyBorder Tag?

```
<FancyBorder color="blue">
    <h1 className="Dialog-title">
      Welcome
    </h1>
    <p className="Dialog-message">
      Thank you for visiting our spacecraft!
    </p>
  </FancyBorder>
```

FancyBorder

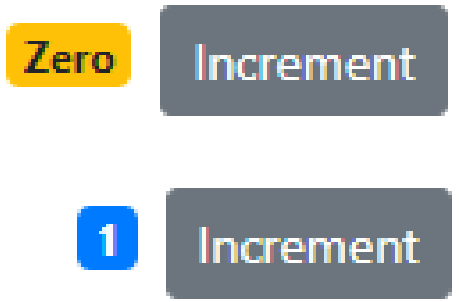
```
function FancyBorder(props) {
  return (
    <div className={'FancyBorder FancyBorder-' + props.color}>
      {props.children}
    </div>
  );
}// Child elements are passed in props.children
```

Composing Components

```
function App() {
  return (
    <div>
      <Welcome name="Sara" />
      <Welcome name="Cahal" />
      <Welcome name="Edite" />
    </div>
  );
}
```

A Simple Counter components/counter.jsx

```
import React, { Component } from 'react';
class Counter extends Component {
 state = { }
 render() {
  return ( <div></div> );
 }
}
export default Counter;
```



Using a Component

```
import Counter from "./components/counter";

ReactDOM.render(<Counter />,
document.getElementById("root"));
```

//instead of using App we are using Counter

Using a Component inside another

```
import React, { Component } from 'react';
import Counter from './components/counter';
class App extends Component {
 render() {
  return (
   <div className="App"><Counter /></div>
  );
 }
}
export default App;
```

What render Method can return a Single Element

```
render() {
return (
  <div className="App">
    <h1>Hello Class</h1>
    <p>Lorem ipsum dolor</p>
  </div>
 );
}
```

Embedding Expressions

```
render() {
    const count=0;
    return (
      <div className="App">
        {count}
      </div>
    );
  }
```

If Short Hand

```
{count === 0 ? "Zero" : count}

{count !== 0 && count} //print count if not zero
```

State

```
state = {
  count: 0
};
//A special variable to control Component State
```

Counter With State

```
state = {
      count: 0
   };
   render() {
      return (
         <div>
            <span>{this.state.count === 0 ? "Zero" : this.state.count}</span>
         </div>
      );
   }
```

Object Destructuring

```
var o = {p: 42, q: true};
var {p, q} = o;


console.log(p); // 42
console.log(q); // true
```

A Clean Approach

```
render() {
      return (
         <div>
            <span>{this.formatCount()}</span>
         </div>
      );
   }
   formatCount() {
      const { count } = this.state;
      return count === 0 ? "Zero" : count;
    }
```

formatCount can also return jsx expression

```
formatCount() {
  const { count } = this.state;
  return count === 0 ? "Zero" :
    <span> count</span>;
}
```

Setting Attributes

```
import React, { Component } from "react";
class Counter extends Component {
    state = {
            imageURL: "https://picsum.photos/200"
    };
    render() {
        return <img src={this.state.imageURL} alt="" />;
    }
}
export default Counter;
```

Setting CSS Classes

```
render() {
    const count = 0;
    return (
      <div className="App">
        {count === 0 ? "Zero" : count}
        {count !== 0 && count}
      </div>
    );
  }
```

Setting Styles from jsx

```
styles = {
   fontWeight: 'bold'
 }
 render() {
   const count = 0;
   return (
     <div className="App" style={this.styles}>
       Hello
     </div>
   );
 }
```

Dynamic Classes

```
getBadgeClasses() {
    let classes = "badge m-2 badge-";
    classes += this.state.count === 0 ? "warning" : "primary";
    return classes;
  }
//Then in render
//className={this.getBadgeClasses()}
```

Handling Lists

```
state = { tags: ['Funny', 'SciFi', 'Horror'] }
render() {
 return (
 <ul>
 {this.state.tags.map(tag => <li>{tag}</li>)}
 </ul>
);
}
//This will generate Warning. Each li should have a key
```

Specifying Key

```
const numbers = [1, 2, 3, 4, 5];
const listItems = numbers.map((number) =>
  <li key={number.toString()}>
    {number}
  </li>
);
```

Specifying Key

```
const todoItems = todos.map((todo, index) =>
  // Only do this if items have no stable IDs
  <li key={index}>
    {todo.text}
  </li>
);
```

Conditional Rendering

```
renderTags() {
    if(this.state.tags.length===0) return <p>No Tags</p>;
    return this.state.tags.map(tag => <li>{tag}</li>);
  }
//Then in Render
//{this.state.tags.length === 0 && "Please Create Tags"}
//{this.renderTags()}
```

Handling Events
Raising an Event

```
<button
onClick={this.handleIncrement}
className="btn btn-secondary btn-sm"
>
Increment
</button>
```

Handling Events
Capturing an Event

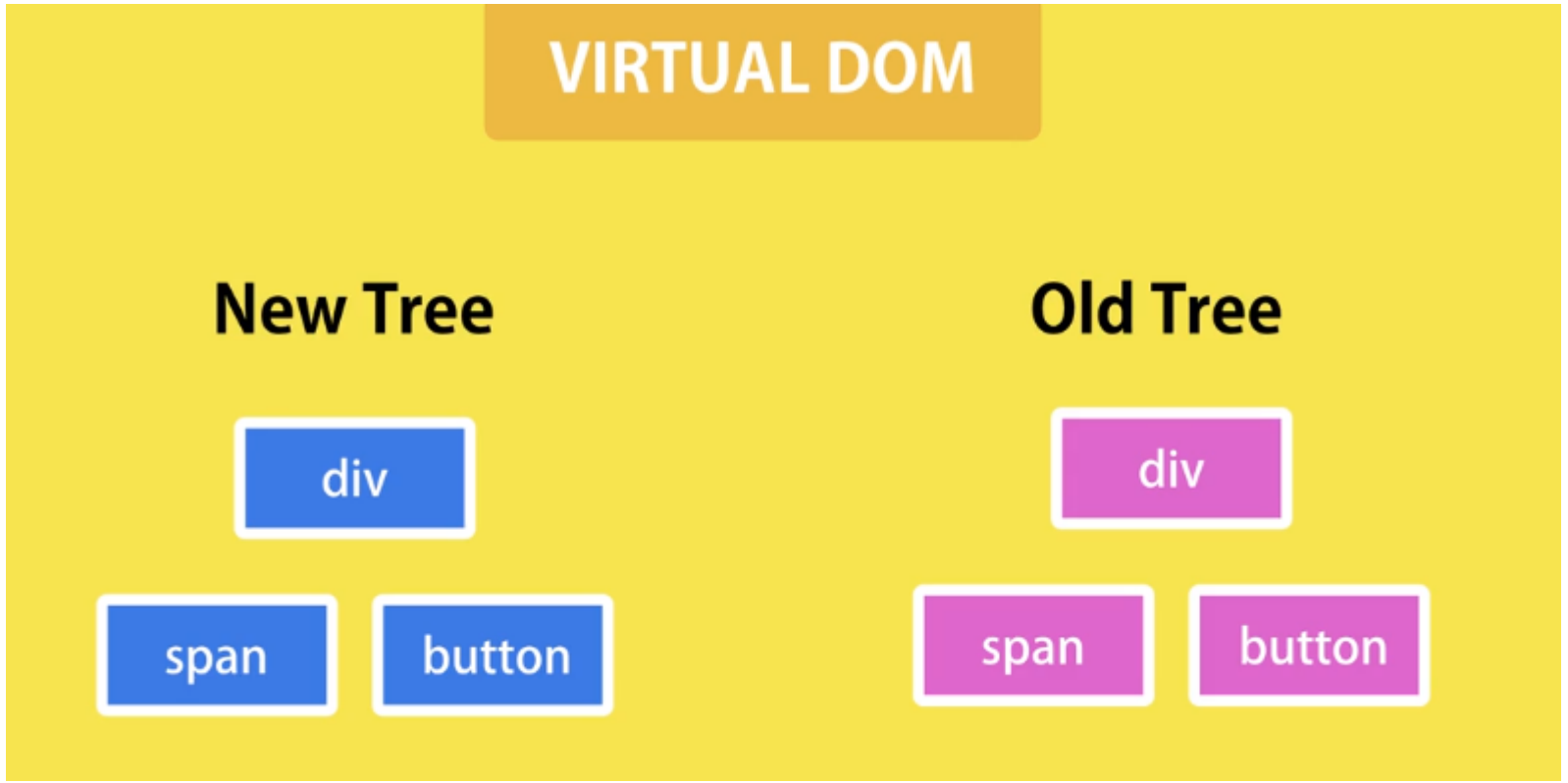
```
handleIncrement = () => {
 // Do Something
 //Always use arrow functions for events
};
```

Handling Events
Updating State

```
handleIncrement = () => {
 this.setState({ count: this.state.count + 1 });
};

//Don’t do this.state.counter=1;
```

What Happens when state change



Passing Parameters to event handler

```
handleIncrement = (id) => {
//Handle ID
};
```

Call it like this

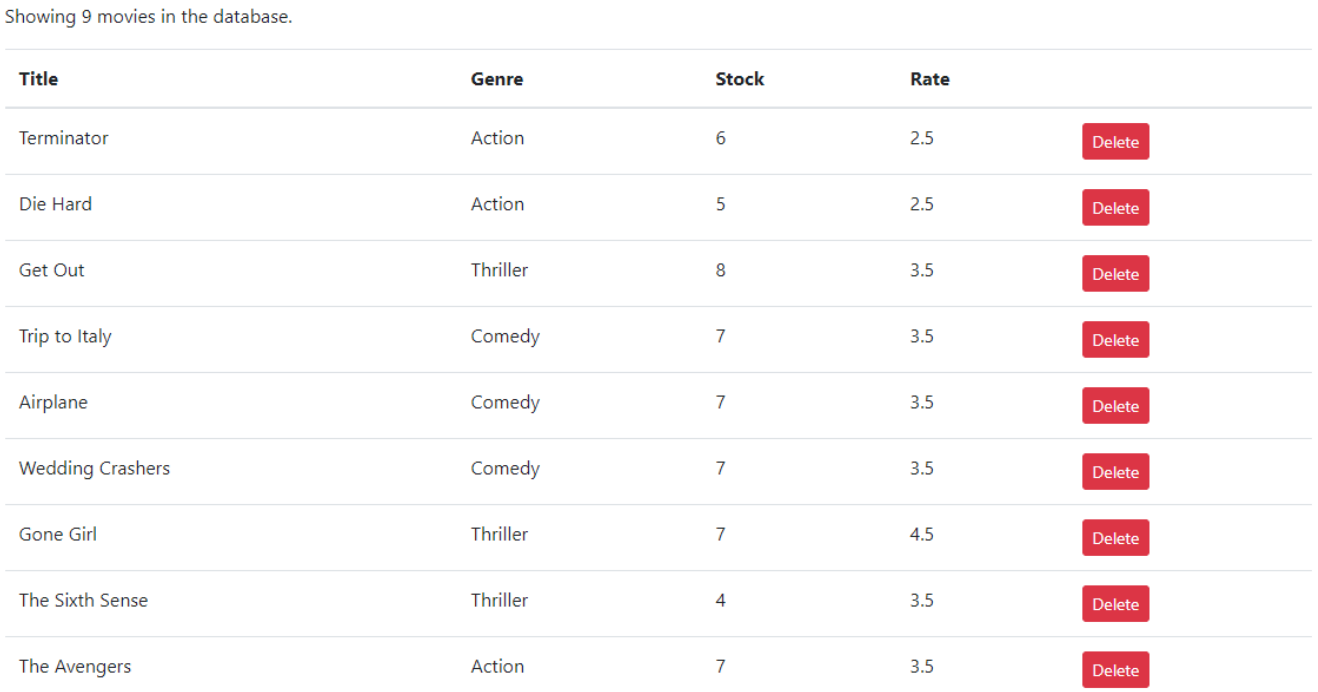
```
onClick={()=>this.handleIncrement(3)}
```

Vidly Project

https://1drv.ms/f/s!AtGKdbMmNBGd1GXjgiJoFiImccTx

>npm install

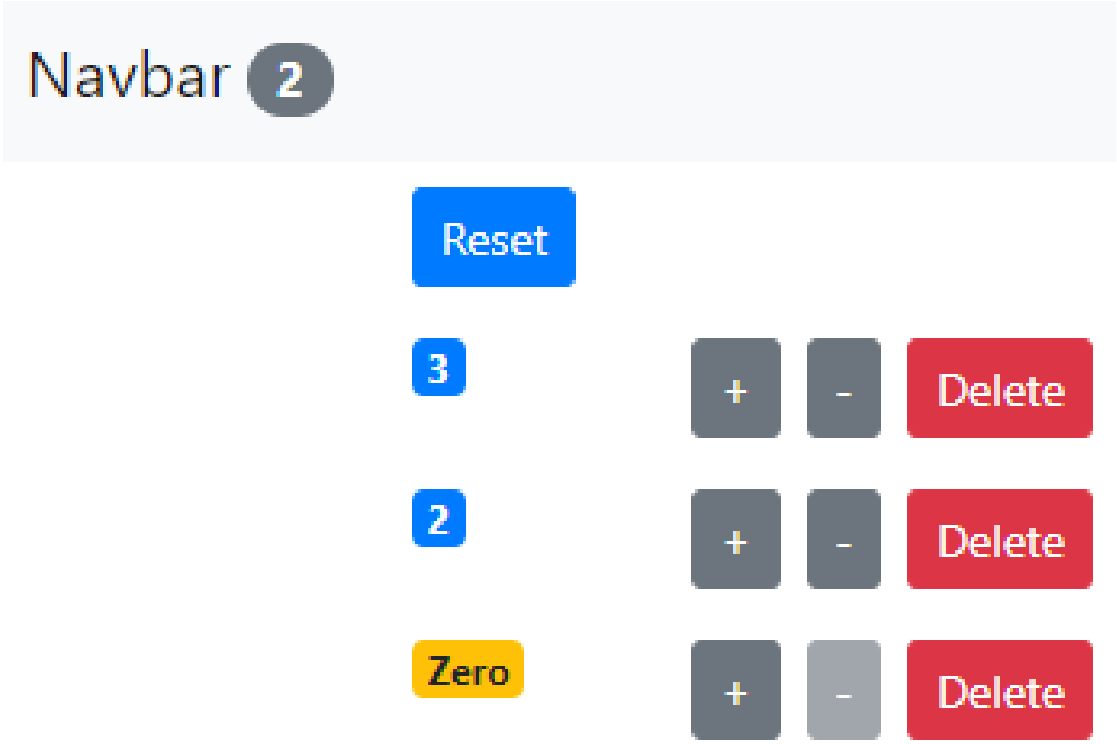
>npm start



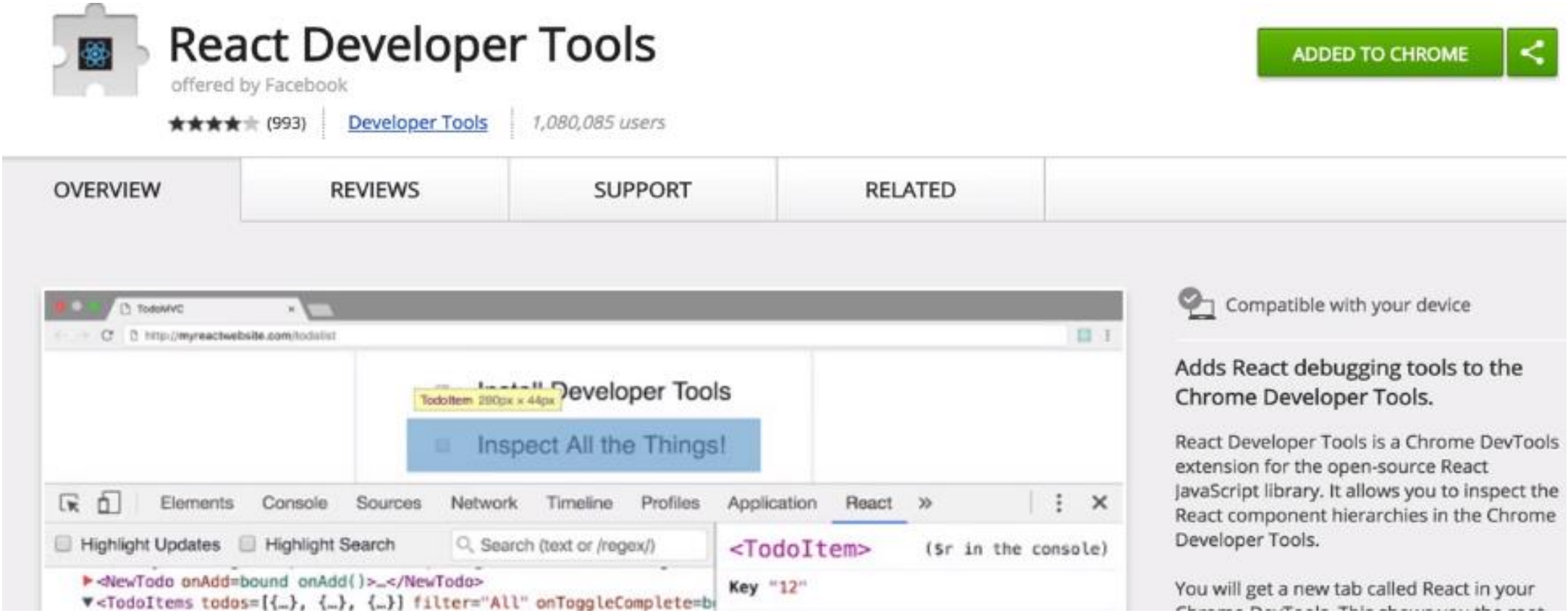
Composing Components

https://1drv.ms/f/s!AtGKdbMmNBGd1zi0xRJmY1kA7tqt

Start to Finish



React Debugging



Single Source of Truth

Try to make Stateless components as much as possible.

Data should be kept at only one location

E.g. Movies Components has many SingleMovie Components then movies data should only be placed in parent .

Passing Functions as props

```
<Like
        liked={movie.liked}
        onClick={() => this.handleLike(movie)}
      />
```

Calling Passed Function (Like.jsx)

```
<i
    onClick={props.onClick}
    style={{ cursor: "pointer" }}
    className={classes}
    aria-hidden="true"
  />
```