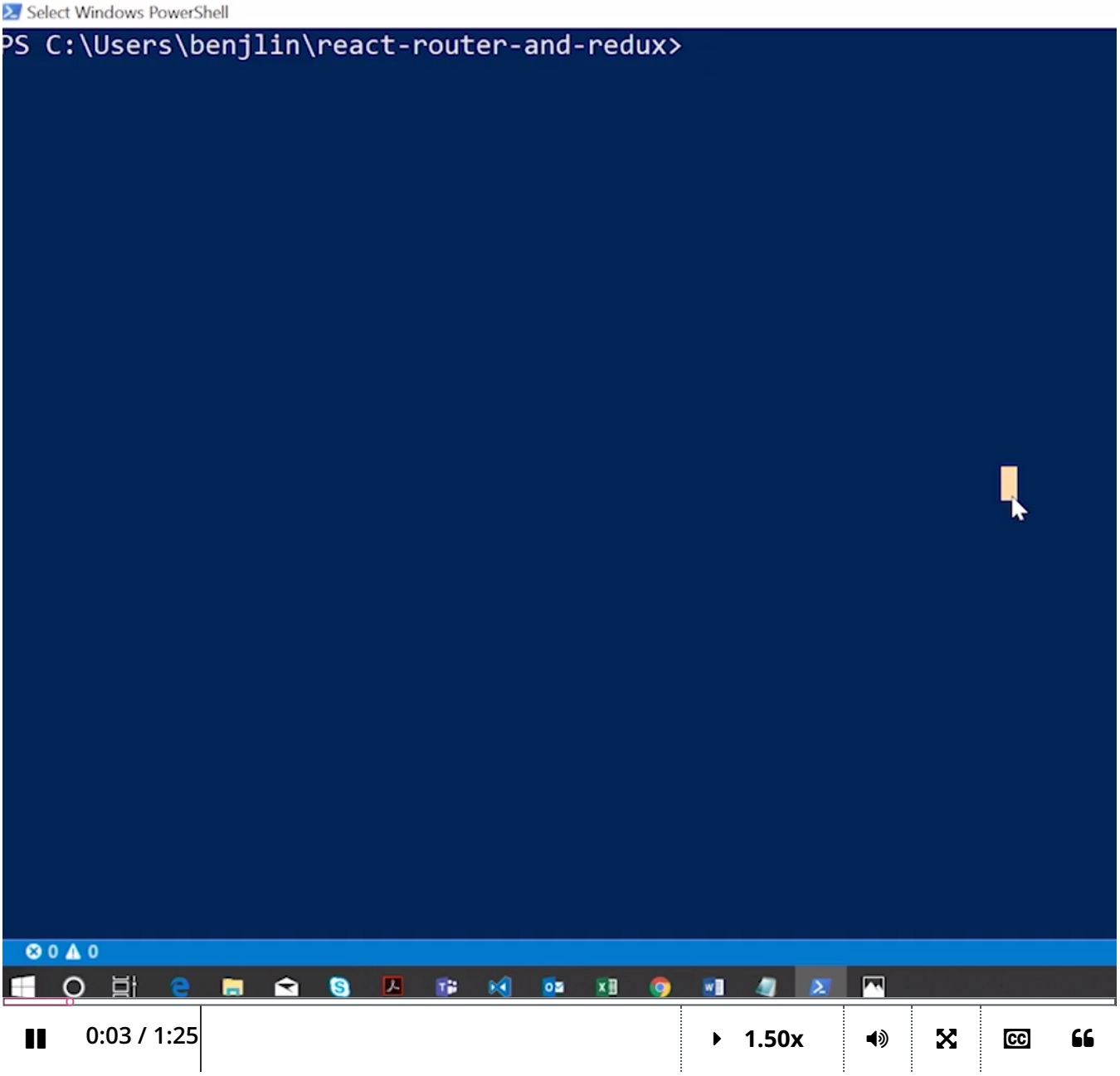
Audit Access Expires Jun 20, 2020
You lose all access to this course, including your progress, on Jun 20, 2020.

Setting up React Redux locally



Video

Download video file

Transcripts

Download SubRip (.srt) file

Download Text (.txt) file

Setting up React Redux locally

Setting up React Redux locally

If you haven't already done so, install `create-react-app` and then create a new React project:

```
npm install -g create-react-app
create-react-app project-name
cd project-name
```

React Redux does not come installed with React, so we must install the `react-redux` library along with the core `redux` library with the following command:

```
npm install 'redux' 'react-redux'
```

After that, you can import the React Redux methods into your `app.js` file:

```
import React from 'react'
import { render } from 'react-dom'
import { createStore } from 'redux'
import { Provider, connect } from 'react-redux'
//...
```

Try copying and pasting this basic example into `app.js` to get started:

```
import React from "react";
import ReactDOM from "react-dom";
import { createStore, combineReducers } from "redux";
import { Provider, connect } from "react-redux";

//action creator
const addItem = (name, price) => {
  return {
    type: "ADD_ITEM",
    item: {
      name: name,
      price: price
    }
  };
};

const deleteItem = index => {
  return {
    type: "DELETE_ITEM",
    index: index
  };
};

//reducer
const reducer = (state = [], action) => {

  switch (action.type) {
   case "ADD_ITEM":
      return [...state, action.item];
    case "DELETE_ITEM":
      return [
        ...state.slice(0, action.index),
        ...state.slice(action.index + 1)
      ];
    default:
      return state;
  }
};

//store
var store = createStore(reducer);
```

```
//presentational components
const Item = props => {
  return (
    <div>
      <div>
        Item : {props.name} | Price: {props.price}
      </div>
      <button onClick={() => props.onDelete(props.index)}>Delete</bu
    </div>
  );
};

class Input extends React.Component {
  constructor(props) {
    super(props);
    this.state = { name: "", price: "" };
  }
  handleChangeName(event) {
    this.setState({ name: event.target.value });
  }
  handleChangePrice(event) {
    this.setState({ price: event.target.value });
  }
  addItem(){
    this.props.onAdd(this.state.name, this.state.price)
    this.setState({ name: "", price: "" })
  }
  render() {
    return (
      <div>
        <input
          onChange={this.handleChangeName.bind(this)}
          value={this.state.name}
        />
        <input
          onChange={this.handleChangePrice.bind(this)}
          value={this.state.price}
        />
        <button
          onClick={() => this.addItem()}>
```

```
          Add
        </button>
      </div>
    );
  }
}

const ItemsList = props => {
  return (
    <div>
      <Input onAdd={props.onAdd} />
      {props.items.map((item, index) => {
        return (
          <Item
            onDelete={props.onDelete}
            index={index}
            name={item.name}
            price={item.price}
          />
        );
      })}
    </div>
  );
};

const mapStateToProps = state => {
  return {
    items: state
  };
};

const mapDispatchToProps = dispatch => {
  return {
    onAdd: (name, price) => {
      console.log(dispatch(addItem(name, price)));
      console.log(store.getState());
    },
    onDelete: id => {
      console.log(dispatch(deleteItem(id)));
    }
  };
};
```

```
//container components
const ItemsListContainer = connect(mapStateToProps, mapDispatchToPro
  ItemsList
);

const App = () => {
  return (
    <Provider store = {store}>
      <div>
        <ItemsListContainer />
      </div>
    </Provider>
  );
}



export default App;
```