11.7.1 Key Terms

action
Angular
build tool
class components
component
conditional rendering
context
controlled form components
create-react-app
CSS-in-JS
currying
dependencies
functional components
functional composition
higher-order functions
jQuery
props
prop-drilling
pure functions
React
React Router
reducers
Single-Page Application (SPA)
software framework
state
store
tagged template literal
uncontrolled form components
vanilla JavaScript
Vue.js
webpack

11.7.2 Review Questions

1. What are the key advantages and disadvantages of using a software framework when constructing web applications?
2. What is a single-page application? Why are frameworks helpful in their creation?
3. What were the original use cases for jQuery? Why is jQuery less important today than it was a decade ago?
4. How are functional components different from class components?
5. What are the advantages of React's component-based approach to constructing user interfaces?
6. Describe the difference between props and state in React.
7. What are the differences between controlled versus uncontrolled form components? Why would you use one over the other?
8. What does the term props-drilling refer to? Why is it necessary in React?
9. Why are build tools necessary for production React sites?
10. Provide a brief discussion of the React lifecycle. When should data be fetched from an external API?
11. Why is routing needed in React?
12. Why might one make use of a state mechanism such as Redux or Context Provider?

11.7.3 Hands-On Practice

PROJECT 1: Editor

DIFFICULTY LEVEL: Beginner

Overview

This project requires the creation of multiple interconnected components. Figure 11.20 illustrates what the finished version should look like along with the required

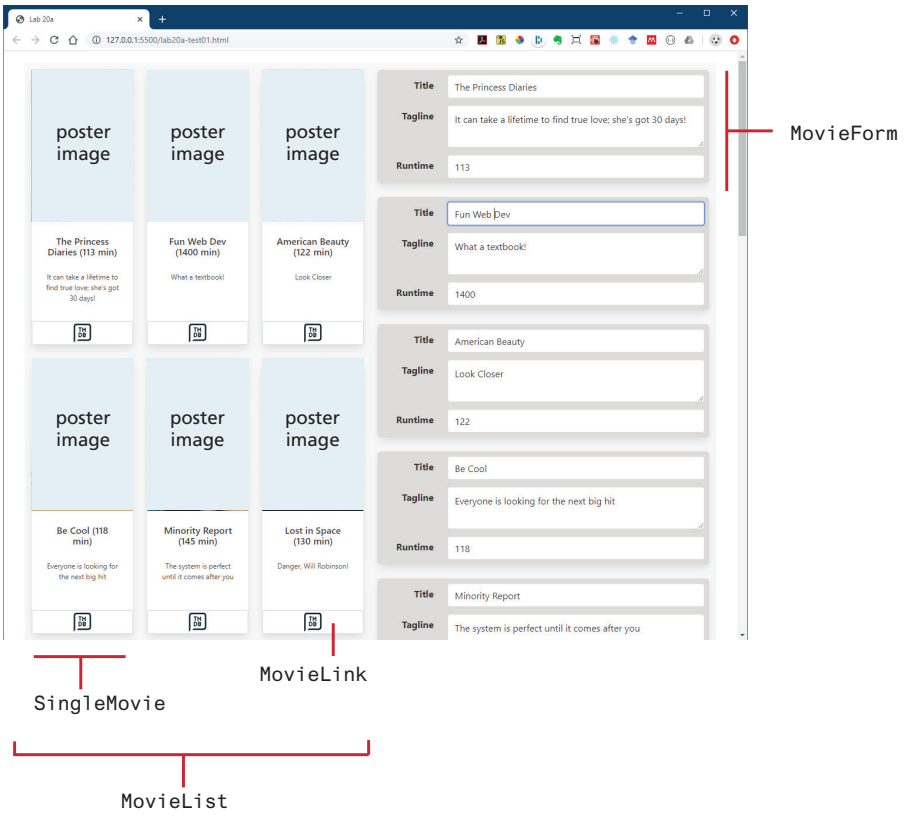

FIGURE 11.20 Completed Project 1

component hierarchy. Changing any of the form fields on the right will update the movie display on the left.

Instructions

1. The starting of the components and their `render()` methods have been provided. The data is contained within the file **movie-data.js**. You can complete this project using the techniques covered in sections 11.2 and 11.3. You could instead use create-react-app as your starting point.
2. Implement the rest of the `MovieList`, `SingleMovie`, and `MovieLink` components. For `MovieList`, you will need to render a `<SingleMovie>` for each movie in the passed list of movies using `map()`. You will need to pass a movie object to `SingleMovie`. For `SingleMovie`, you will need to replace the sample data with data from the passed-in movie object. In the footer area, you will render a `<MovieLink>` and pass it the `tmdbID` property from the movie object.

`MovieLink` must be a functional component. It will return markup similar to the following (though you will replace 1366 with the passed `tmdbID` value):

```
<a className="button card-footer-item"
   href="https://www.themoviedb.org/movie/1366" >
        <img src="images/tmdb.svg" width="30" />
</a>
```

3. In the `App` component, you will add the `<MovieList>` component to the render. Be sure to pass it the list of movies in state. Test.
4. In the `App` component, use `map()` to output a `<MovieForm>` for each movie. Be sure to pass both index and key values to each `MovieForm`. Also pass the `saveChanges` method to each `MovieForm`. Test.
5. Make `MovieForm` a Controlled Form Component. This will require creating some type of handler method within `MovieForm` that will call the `saveChanges` method that has been passed in (see also next step).
6. Implement `saveChanges` in the `App` component. Notice that it expects a movie object that contains within it the new data. Your method will use the index to replace the movie object from the movies data with the new data, and then update the state. Test.

Guidance and Testing

1. Break this problem down into smaller steps. Verify each component works as you create them.

PROJECT 2: Favorites List

DIFFICULTY LEVEL: Intermediate

Overview

This project builds on the completed Test Your Knowledge #2 by adding a favorites list.

Instructions

1. Use your completed Test Your Knowledge #2 as the starting point for this project.
2. The `PhotoThumb` component already has an Add Favorite button (the heart icon). You are going to implement its functionality. When the user clicks on this button, it will add a new favorite to an array stored in state, which should update the display in the Favorites component (it will be blank row at first between the header and the photo browser).
3. You will need to create two new components: one called `FavoriteBar`, the other called `FavoriteItem` (there is already a CSS class defined in the provided CSS named favorites). `FavoriteBar` should display a list of `FavoriteItem` components. Since the `favorites` CSS class uses grid layout, your `FavoriteItem` component will need to wrap the image in a block-level item, such as a `<div>`.