How to do simple routing

Finally our site is going to behave like a real website! No more hacky solutions like changing the startup component in App.js. Routing, here we come!

Let's say we wanted to design our routes like this:

path	component
/	LandingPage
/account	Account
/register	Account
/login	Login
/logout	Logout
/checkout	Checkout
/pickseats/:showingId	PickSeats
/film/:filmId	FilmDetails

1. We clearly can't do any routing until we've installed React Router.

```
npm install react-router-dom
```

2. Look at package.json to make sure react-router-dom is in there now.
3. Open App.js in your IDE. Wrap all of its contents in a BrowserRouter. Add a <BrowserRouter> open tag at the very top of the JSX and its matching close </BrowserRouter> tag at the very bottom.
4. Run and test. You should see no difference so far.
5. Find the <main> in App.js. If you have any components in there like <Account /> or <PickSeats /> or whatever, remove them.
6. Inside the <main>, add a <Routes> and </Routes>
7. And finally inside the Routes add a <Route>:

```
<Route path="/" element={<LandingPage {...state} />} />
```

Remember that { …state } trick is our way of passing each part of state into the component. This will allow our components who need props to get them. It is not strictly necessary, and some developers would frown on this practice. It won't affect performance but may be considered less clean by some. In the next lab we'll show you a reason why it can be so handy.

8. Run and test by typing in the URL to the root route. Did it work?
9. Try typing in the URL for /account. That didn't work, did it? Your next assignment is to get it working by adding another <Route>. Go for it.

Is it good now? Good!

10. Go through your table of routes above and implement all of the site's routes.
11. Test some of them by using the top menu navigation. Those links should be working now.

What about 404?

12. Try typing in a nonsense route, one that you know just doesn't exist. What happens? _____ Does that seem like the right thing to you?

By default, the user gets no feedback that their navigation failed. That is just confusing. What if they have a small typo in the URL? Wouldn't it be better to tell them that their navigation attempt failed?

13. Add one more route to the table at the end. The component should be NotFound.js. (You thought we had forgotten about NotFound, didn't you?) Make the path attribute be a "*" so that it matches every request.
14. Run and test again with a nonsense route. You should see your NotFound component.