Passing Information

In the first part of the app, we created a new react app and built all of the static components for the project. Additionally we simulated what the project would look like after returning search results.

In this second part we'll be passing information to our components instead of hard-coding it in. This will call for a little code cleanup and some additional functionality adds. This **WILL** take some time, plan accordingly!

If you don't understand how to implement a certain part of the project, do a little research and refer back to your CC exercises first. Then use Slack for additional help.

Creating a business Array

When the Yelp API is implemented, a list of businesses will be returned to App.js. Therefore, we should create a hard-coded list of businesses there and then pass them to the appropriate components.

1. Open Business.js. **Cut** the entire business object and paste it into `App.js`.

Make sure to paste it right after the imports and above the App class declaration.

2. Take a look at `BusinessList.js` - what is different about the BusinessList div?

There's a lot of repetition here. Specifically, the `<Business />` component is repeated *six times*. Recall that we manually did this in the BusinessList component so that we can simulate a list of returned business. We'll refactor this to remove some of the component repetition.

Go back to `App.js`. Under the `business` object, create a `businesses` array. The array should hold six references to the `business` object.

3. In `app.js`, add a `businesses property` to the `<BusinessList />` component inside of the `render()` method. Set the property equal to the `businesses` array. (Don't forget your curly braces!)

   `<ComponentName propName={value} />`

Adding Functionality to the BusinessList Prop

When the `businesses` prop is set in the `BusinessList` component, it should iterate through the businesses array. Recall that you created an `array of businesses` in `App.js`.

4. Open `BusinessList.js`. **Remove** everything inside of the `BusinessList div` (the repeated components).

Inside of the `BusinessList div`, access the `businesses prop` and call the `.map()` method on it.

example:

```
<div className="BusinessList">
  {
    this.props.businesses.map();
  }
```

```
</div>
```

5. Inside of the `map()` method, pass a callback function with one parameter called `business`.
6. The callback function should return a `<Business />` component. The returned `<Business />` should have a property called `business`. Set the property equal to the parameter of the callback function. (Don't forget your curly braces and semicolon!)

Modifying Business.js using Props

7. Open `Business.js`. What do you notice about how information is accessed inside of the return statement?

Because the `business` object was removed, statements like the following no longer have any meaning:

```
{business.imageSrc}
```

Business information now has to be accessed via the business prop you set in `BusinessList.js`. Modify all statements (like the one in the example above) by prepending them with `this.props`.

This is typical of the flow of information in React apps. You'll continue to build on this structure as you move on to future parts of this project. Think about these lingering questions as we move to the next part of the project.

```
1. The "Let's Go" button doesn't do anything at the moment. How might you simulate
a query to the Yelp API?
2. A user may decide to search with a different sorting option (for example,
"Highest Rated", or "Most Reviewed"). How can you handle this change in state
using React?
```

Happy Coding, Happy Thanksgiving!! See You Next Class.