Hands-On - #4

Q6. The following challenges are a review of
    Array.map() method.

Take out your scratch pad and try them out. You can also use Node.js REPL to try them.

Challenge [1]:
Given an array of numbers, return an array of each number, squared

```
    const nums = [1, 2, 3, 4, 5]
    // -->       [1, 4, 9, 16, 25]
```

Challenge [2]:
Given an array of strings, return an array where the first letter of each string is capitalized

```
const names = ["alice", "bob", "charlie", "danielle"]
// -->        ["Alice", "Bob", "Charlie", "Danielle"]
```

Challenge [3]:
Given an array of strings, return an array of strings that wraps each of the original strings in an HTML-like tag.

E.g. given: ["Bulbasaur", "Charmander", "Squirtle"]
    return: [" Bulbasaur ", " Charmander ", " Squirtle "]

```
const pokemon = ["Bulbasaur", "Charmander",
                 "Squirtle"]
// -->          ["<p>Bulbasaur</p>",
                 "<p>Charmander</p>",
                 "<p>Squirtle</p>"]
```

Make a note of the answers in the Notepad document as it is required during your assignment submission.

Q7. Let's now applying the map() method to our
    'airbnb' project.

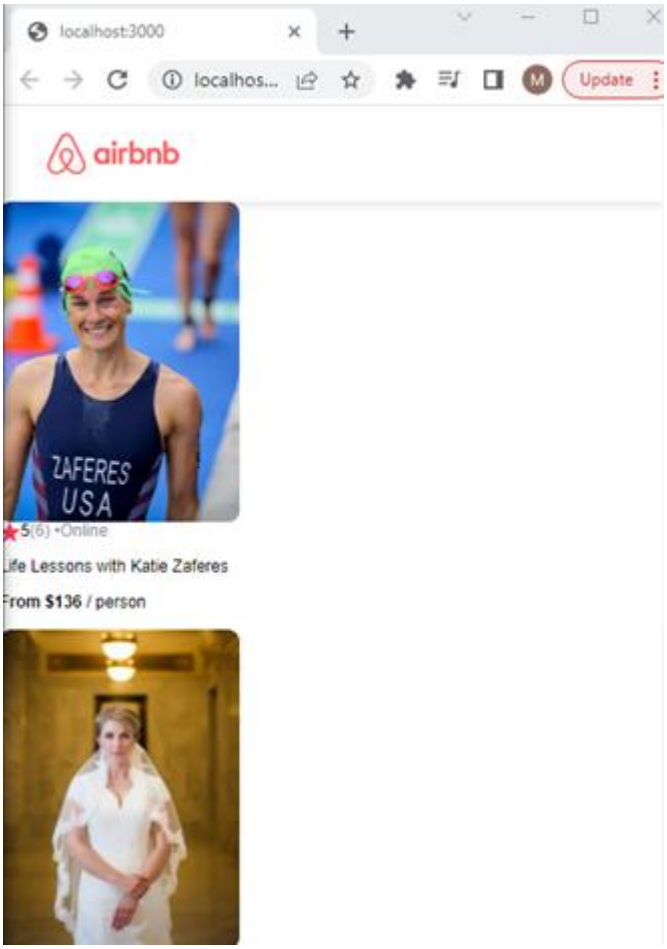
[a] Your Next Challenge!

- import the array of data from data.js
- map over the array to create <Card /> components
- display the array of card components under the navbar (in place of the current <Card /> component)

NOTE: We haven't styled the group of components yet, so they'll still be block elements, stacked vertically. We'll add styling later.

[b] Save your files and observe the app.

It should now be displaying an array of cards.



[c] We have already done the styling for the card in our CSS file.

However, we want the cards to be one besides the other, to achieve this set the display property to 'flex' and flex direction to 'column'

You can add the following in the CSS file for the 'card' class selector.

```
    flex: 0 0 auto;          /* ADD */
    display: flex;           /* ADD */
    flex-direction: column; /* ADD */
```

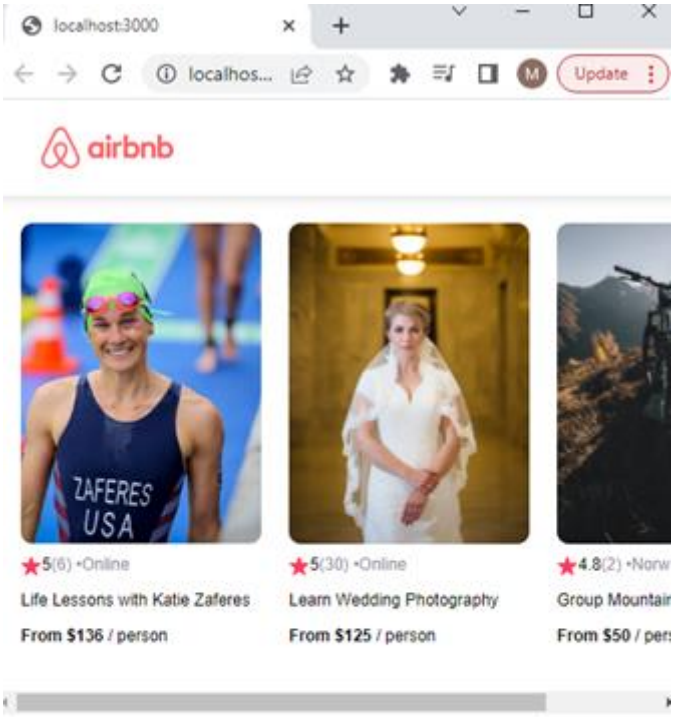
[d] The cards which were rendered below the 'Navbar' in the app can be styled.

- To do this, puts the cards inside a section element
- Give the className as "card--list" and set the Following properties for the card list.

```
    .cards-list {
        display: flex;
        flex-wrap: nowrap;
        gap: 20px;
        overflow-x: auto;
    }
```

- Save your CSS file and other files
- Observe the app once again.

Awesome!



Q8. Conditional Rendering.
If you observe carefully, there is 'Sold Out' badge for those cards where the 'openSpots' props is ZERO.

You challenge is to display the 'SOLD OUT' badge only for those cards.

[a] Firstly, we need to pass the 'openSpots' data to the Card component.

Console log and check if the Card component is now getting the 'openSpots' prop

[b] Next let's display the 'SOLD OUT' badge with a element

Moreover, set the className to 'card--badge'

Since, we want the 'SOLD OUT' badge to be inside the card, set the 'postion' to relative in the CSS style

[c] Set the '.card--badge' properties as follows in the CSS file

```
.card--badge {
position: absolute;
top: 6px;
left: 6px;
background-color: white;
padding: 5px 7px;
border-radius: 2px;
font-weight: bold;
}
```

Save your files and observe the app.

[d] We don't want the card badge to be displayed for all the cards

Perform conditional rendering using the '&&' operator.

Syntax: <condition> && <do-some-thing>

Do some thing is executed only if the condition is TRUE. Feature of JS

- Thus check to see if the openSpots are equal to ZERO and if TRUE display the card badge

- Save your Cards component and observe the app once

[e] Your next challenge:
- Display the correct text in the badge based on the following conditions

(1) If openSpots is set to ZERO, set text as 'SOLD OUT'
(2) If locatiion is 'Online', set text as 'ONLINE'

- Only display the badge if badgeText has a value

Save your Card component and refresh your app.

The application should look as shown below: