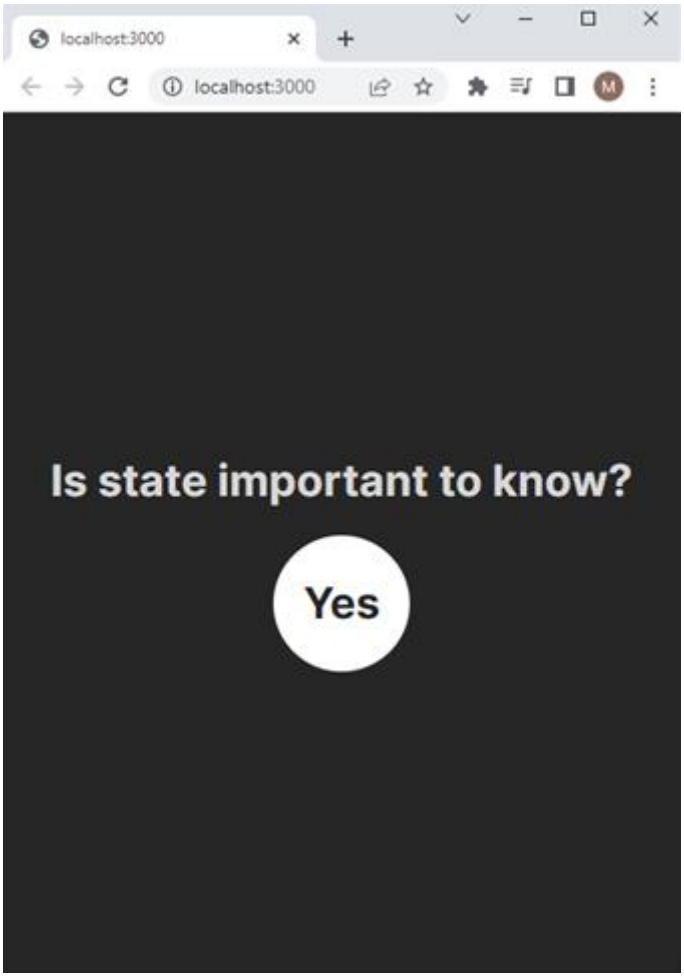
Hands-On - #7

Q7. Copy the given 'LAB-p3q7' folder and its files as follows:

- 'index.html' & 'style.css' files in 'public' folder
- 'index.js' and 'App.js' files in the 'src' folder
  NOTE: Replace the file(s) in the destination.

[a] Open the terminal and run the app. You browser should display something as shown below:



[b] Your Challenge:

Replace our hard-coded "Yes" on the page with some state initiated with React.useState() and render it.

Save your file(s) and observe the app.

[c] Now, we are aware that React.useState() returns an array.

With array de-structuring, perform the following:

- Give the name of the state as 'isImportant'
- Give the function name as 'setIsImportant'

NOTE: You are at liberty to give any names, but that is how the convention is followed in React.

[d] Create a function called `handleClick` that runs/invokes setIsImportant("No")

- Add a click event listener to the div.state--value that runs `handleClick` when the div is clicked.

Save your file(s) and observe the app.

Check to find if the user click, the state is changed.

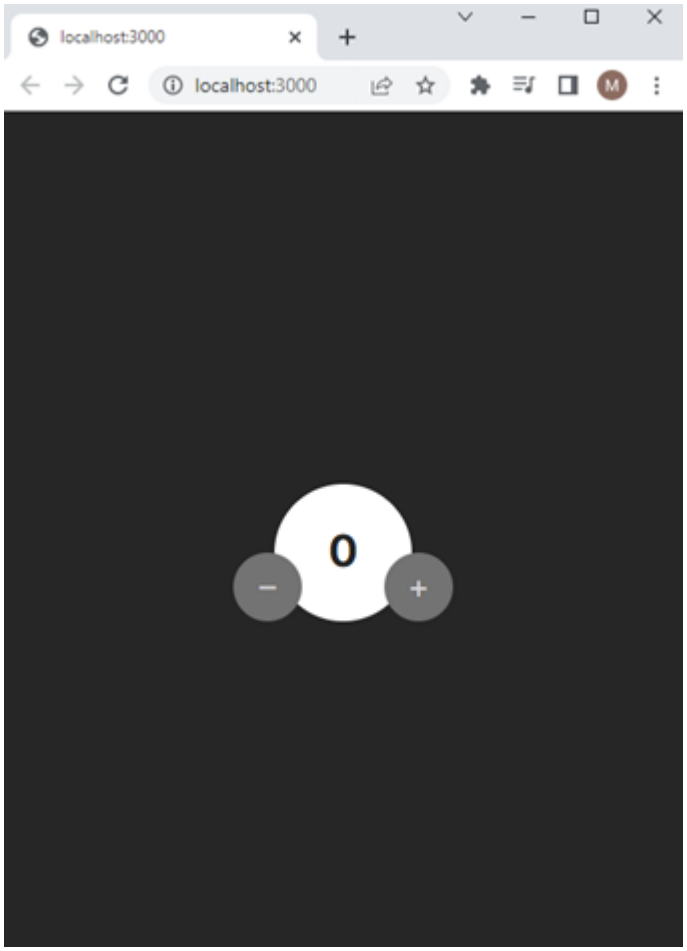
[e] Think is it possible to Toggle!

[f] Backup the ‘src’ and ‘public’ directory and save it as solution ‘sol-p3q7’

Q8. Copy the given 'LAB-p3q8' folder and its files as follows:

- 'index.html' & 'style.css' files in 'public' folder
- 'index.js' and 'App.js' files in the 'src' folder
  NOTE: Replace the file(s) in the destination.

[a] Open the terminal and run the app. (If it has STOPPED!). You should now see something like this in your browser.



[b] Your Challenge:

Set up state to track our count (initial value is 0)

- See if you can think of a way to add 1 to the count every time the **(+)** button is clicked

- See if you can think of a way to subtract 1 to the count every time the **(-)** button is clicked

[c] Save your file(s) and observe the app for its functionality.

[d] Change required as per React Best Practice
OPTIONAL

[e] Backup the 'src' and 'public' directory and save it as solution 'sol-p3q8'

Q9. Now, once again open your 'memegen' folder using VS Code.

[a] Your Next Challenge!

- Save the random meme URL in state

- Create new state called `memeImage` with an 'nature.jpg' image as default.

- When the getMemeImage function is called, update the `memeImage` state to be the random chosen image URL

- Below the div.form, add an <img /> and set the 'src' to the new `memeImage` state you created

- Don't forget to set the className for the <img /> Element

```
max-width: 100%
```

- You should initially see the 'nature.jpg' image.

However, on clicking the 'Get a new meme image' button, the image should change in the bottom portion of the web page

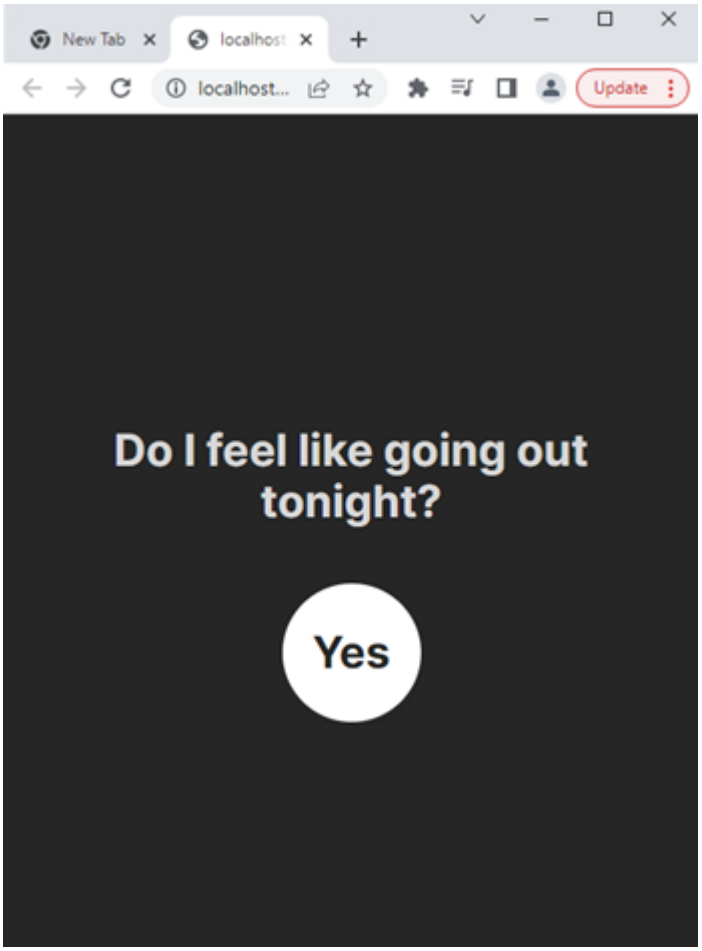
[b] Save your files and close VS Code.

Q10. Open your 'demo3' folder using VS Code.

Copy the 'LAB-p3q10' folder and its as files as follows:

- 'index.html' & 'style.css' files in 'public' folder
- 'index.js' and 'App.js' files in the 'src' folder
  NOTE: Replace the file(s) in the destination.

[a] Open the terminal and run the app (if it is STOPPED). Your browser should display something similar as shown below:



[b] Now your challenge is as follows:

- Initialize state for `isGoingOut` as a boolean
- Make it so clicking the div.state--value flips that boolean value (true -> false, false -> true)
- Display "Yes" if `isGoingOut` is `true`, "No" otherwise

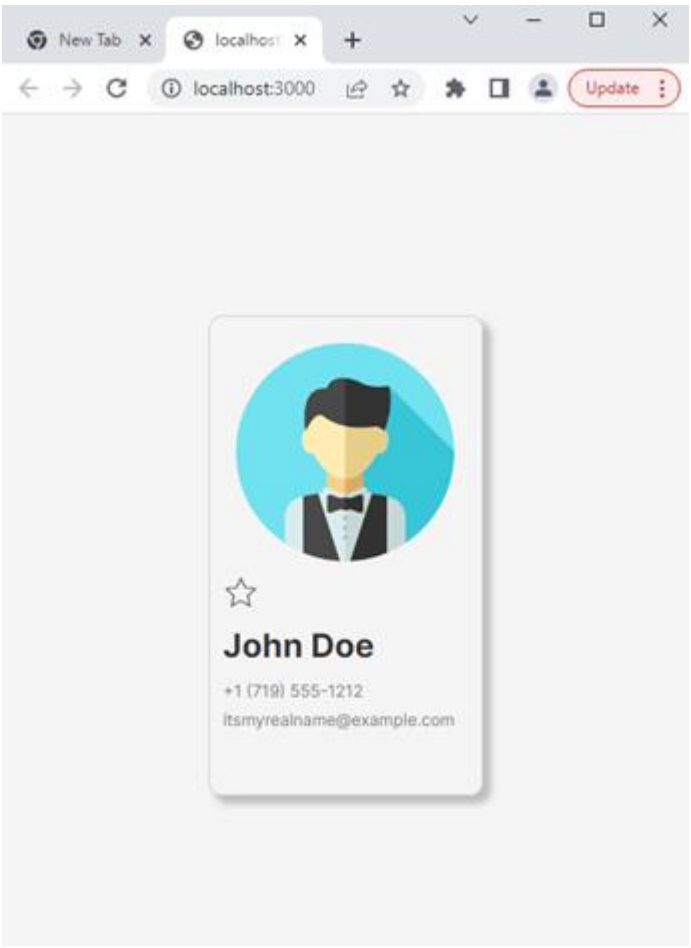
[c] Save your files and check if the app is functional as per our requirement.

[d] Backup the 'src' and 'public' directory and save it as solution 'sol-p3q10'

Q11. Copy the folder 'LAB-p3q11' and its files as follows:

- 'index.html' & 'style.css' files in 'public' folder
- 'index.js' and 'App.js' files in the 'src' folder
- The 'images' folder in 'public' folder
  NOTE: Replace the file(s) in the destination.

[a] Open the terminal and run the app (if it is STOPPED). Your browser should look as shown below:



[b] Fill in the values in the markup using the properties of our state object (given above)
NOTE: Ignore `isFavorite` for now

[c] Save the 'App.js' file and check if the app is working.

[d] Change the firstName and lastName in the contact object to your choice and check if the card is changing.

[e] Your Next Challenge:

Use a ternary to determine which star image filename should be used based on the `contact.isFavorite` property

* `true` => "star-filled.png"
* `false` => "star-empty.png"

Then use the starIcon value to display the correct image

[f] Change the object 'isFavorite' property value from 'true' to 'false' and vice-versa and observe the UI

The STAR ICON should change accordingly.

[g] Modify the `toggleFavorite' function which will enable us to toggle the ICON depending on the state 'isFavorite'

We already have the `setContact` setter function which is defined by the useState hook.

NOTE-1: We need to use the `setContact` function inside the the `toggleFavorite` function

NOTE-2: 'isFavorite' is one of the property of the contact obj.

[h] Save the file(s). Check the functionality of the app

[i] Backup the ‘src’ and ‘public’ directory and save it as solution ‘sol-p3q11’

Q12. Once again open the ‘memegen’ project in VS Code

Your next challenge!

[a] Update our state to save the meme-related data as an object called `meme`. It should have the following 3 properties:

* topText, bottomText, randomImage.

- The 2 text states can default to empty strings for now, and randomImage should default to "http://i.imgflip.com/1bij.jpg"

- Next, create a new state variable called `allMemeImages` which will default to `memesData`, which we imported above

- Lastly, update the `getMemeImage` function and the markup to reflect our newly reformed state object and array in the correct way.