**What I have found in programming experience with pure functions -----**

When you create a pure function in your react project:  
You have one main exportable function  
**You create functions “within” your main function**

One of the best articles I have read on React for explaining functions, classes concepts etc.  
https://www.freecodecamp.org/news/react-components-jsx-props-for-beginners/

Highlights:  
Graphical user interface, text, application

Description automatically generated  
  
Graphical user interface, text, application, Teams

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

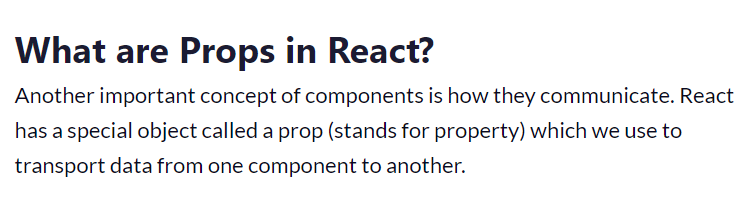
Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**Graphical user interface, text, application

Description automatically generated**

****

**Graphical user interface, text, application

Description automatically generated**

**Functions can take a props argument (an object)**

[**https://reactjs.org/docs/components-and-props.html**](https://reactjs.org/docs/components-and-props.html)

Props are read-only, they are passed down from parent child

Text

Description automatically generated  
  
Text

Description automatically generated

The keyword props does not have to be used, you can pass a single value as child value as shown below

Text

Description automatically generated

Generally when you use the keyword props (this means a props object that can contain multiple values)

**My Findings on Hooks**

--You use **useState** when you want to use state within your funcional component

const [tblFoodsToBeOmmited, settblFoodsToBeOmmitedData] = useState([])

--you set your state as shown below

async function fetchblFoodsToBeOmmitedData() {

let \_FTBOM = [];

var myAPI = new studentInfoApi;

\_FTBOM = await myAPI.fetchblFoodsToBeOmmitedData()

//console.log(\_FTBOM)

settblFoodsToBeOmmitedData(\_FTBOM)

}

--then to use your state variable:

<BootstrapTable data={tblFoodsToBeOmmited} striped hover options={options}

--**useSelector**: Used to tap into your REDUX Store

const counter = useSelector((state) => state.counter);

--**useDispatch** allows you to "dispatch" actions against your REDUX store

const dispatch = useDispatch();

//below is how to use **useCallBack** - keeps a cached copy of a memozized component (export default React.memo(CustomButton))

//and only renders the component that needs to be rendered (not all components ) when a prop or state changes

const incrementAge = useCallback( () => {

setAge(age +1)

},[age])

--this is good for when the DOM changes and you want to perform some other action

**useEffect**(() => {

...... do something else when the DOM changes (like componentDid or DidNot Mount

});

**Working with Parent Child and Props with functional components**

**Graphical user interface, text

Description automatically generated**

**Text

Description automatically generated**

**Text

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

**------------------------------------------------------------------------------------------------------------------------------------------**

**GIT**

**Project Locations on GIT**

git clone <https://github.com/lionel5116/SBHISDCustomDevAppsWebAPI.git>

git clone <https://github.com/lionel5116/SandBoxReactWithWebPack.git>

**PUSHING CODE**

git init

**git remote add origin** [**https://github.com/lionel5116/SBHISDCustomDevAppsWebAPI.git**](https://github.com/lionel5116/SBHISDCustomDevAppsWebAPI.git)

git add -A

git add .

git commit -m "Adding PRop example code"

git push origin master

**PROJECT LOCATIONS**

Graphical user interface

Description automatically generated

**VIRTUAL DIRECTORY NAME  
HISDCustAppsWebAPI**

**Graphical user interface, text, application

Description automatically generated**

**REACT HOOKS & REDUX**

**Functional Programming**

https://medium.com/@shaistha24/functional-programming-vs-object-oriented-programming-oop-which-is-better-82172e53a526

From a maintenance, logical and structural standpoint, functional programming excels when there are no histories to deal with. It works particularly well when there are no boundaries required, or those boundaries are already predefined. It thrives in situations where the state is not a factor and there is very little to no involvement with mutable data.

Functional programming provides the advantages like efficiency, lazy evaluation, nested functions, bug-free code, parallel programming. In simple language, functional programming is to write the function having statements to execute a particular task for the application. Each small function does its part and only its part. The function can be easily invoked and reused at any point. It also helps the code to be managed and the same thing or statements does not need to be written again and again. It allows for very modular and clean code that all works together in harmony.

**HOC**

**A HOC is a pure function. That means it has no side effects. It only returns a new component.**

**Graphical user interface, application

Description automatically generated**

**Below is a good video on how to use react hooks with nodejs API**

<https://www.youtube.com/watch?v=PEGUFi9Sx-U>

**Good Video Series on React**

**React Hooks:**

Allow you to use React features without having to write a class, Ex; State of a component

<https://www.youtube.com/watch?v=cF2lQ_gZeA8&list=PLC3y8-rFHvwgg3vaYJgHGnModB54rxOk3&index=44>  
  
**React Effects**  
Mimics lifecycle hooks for a functional component. i:e componentDidMount, it essentially adds "effects" to a functional component

<https://www.youtube.com/watch?v=BH4xvzHa7H8&list=PLC3y8-rFHvwgg3vaYJgHGnModB54rxOk3&index=52>

**Great Article on React Hooks with REDUX**

<https://thoughtbot.com/blog/using-redux-with-react-hooks>  
**How to use Redux with Hooks**

React Redux now includes its own useSelector and useDispatch Hooks that can be used instead of **connect**.

**useSelector** is analogous to connect’s **mapStateToProps**. You pass it a function that takes the Redux store state and returns the pieces of state you’re interested in.

**useDispatch** replaces connect’s **mapDispatchToProps** but is lighter weight. All it does is return your store’s dispatch method so you can manually dispatch actions. I like this change, as binding action creators can be a little confusing to newcomers to React Redux.

Alright, so now let’s convert a React component that formerly used connect into one that uses Hooks.

Text

Description automatically generated

**Here a working example – This is in the sandbox project**

<https://www.freecodecamp.org/news/learn-redux-by-making-a-counter-application/>

A screenshot of a computer

Description automatically generated with medium confidence  
  
**And it works:**

Graphical user interface, text, application

Description automatically generated

**­­A recap of REDUX  
Diagram

Description automatically generated  
Actions:  
Text

Description automatically generated  
  
Text

Description automatically generated**

**Definition of a pure function**

A function application has no side effects (no mutation of local static variables,

non-local variables, mutable reference arguments or input/output streams).

**Graphical user interface, text, application, email

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

**Use Effect**[**https://reactjs.org/docs/hooks-effect.html**](https://reactjs.org/docs/hooks-effect.html)Since we are using pure functions, (that don’t allow for side effects), the only way to add side effects is to use .. well (useEffect)

**Understanding Side Effects**

React is centered around functional programming. A side effect is any execution that affects something

outside the scope of the function being executed. It is not a React specific term,

it’s a general concept about the behavior of a function. For example, if a function modifies a

global variable, then that function is introducing a side effect —

the global variable doesn’t belong to the scope of the current function.

Data fetching, setting up a subscription, and manually changing the DOM in React components are all

examples of side effects. Whether or not you’re used to calling these operations “side effects”

(or just “effects”), you’ve likely performed them in your components before.  
  
Graphical user interface, text, application, email

Description automatically generated

**A screenshot of a computer

Description automatically generated**

**Diagram

Description automatically generated**

**Text, letter

Description automatically generated**

**To keep useEffect from an infinite loop, use and empty array as the dependent argument as explained below**

**A screenshot of a computer

Description automatically generated with medium confidence  
When I did not include the empty array in the useEfffect, it ran over 1,000 times**

**Graphical user interface, text, application

Description automatically generated**

**MORE REDUX HERE**

<https://reactjs.org/docs/hooks-state.html>  
What is a hook in functional programming  
A Hook is a special function that lets you “hook into” React features  
**When would I use a Hook?**

If you write a function component and realize you need to add some state to it,

previously you had to convert it to a class. Now you can use a Hook inside the existing function component  
  
**Example:**

const [count, setCount] = useState(0);

What do we pass to useState as an argument? The only argument to the useState() Hook is the initial state.

Above we pass in 0 as the initial value

What does useState return?

It returns a pair of values: the current state and a **function** that updates it.

This is why we write const [count, **setCount**] = useState()

This is similar to this.state.count and this.**setState** in a class

Graphical user interface, text

Description automatically generated

Graphical user interface

Description automatically generated

Real world example I used in my file upload component example:  
Text

Description automatically generated

**REACT PAGE LAYOUT**

For Components used by bootstrap react

<https://react-bootstrap.github.io/components/alerts/>

**React Bootstrap5 Sample Page Demo**

<https://www.youtube.com/watch?v=l2131Rok8XU>  
We implemented a react bootstrap web page  
**NOTE**: WE ALSO DID LIVE MARKUP CODING, WE STARTED THE WEBSITE AND CODED AND SAVED THE FILE WHILE IT DID LIVE RELOADING!!!  
Docs:  
<https://react-bootstrap.github.io/getting-started/introduction>  
Page: “SampleReactBootStrapWebPage.js”  
Key Notes on the Page  
Text

Description automatically generated  
  
The **’px-4 my-5’** is padding an margins in bootstrap react  
  
**Responsive Grids:**  
<https://react-bootstrap.netlify.app/layout/grid/#grid-system>  
  
**Images**  
Text

Description automatically generated **ClassName Keyword**Text

Description automatically generated

**Notice how when you using regular markup (not components):**class (for regular elements)  
className (for components)  
  
https://react-bootstrap.github.io/components/cards/  
A screenshot of a computer

Description automatically generated  
He also just copied and pasted the sample card code from the website (sweet), just like regular bootrap example coding  
  
After coding:

A screenshot of a computer

Description automatically generated with medium confidence  
SWEET!!!!!  
He also has some information on free web page hosting of your react application, kinda like Heroku  
<https://www.youtube.com/watch?v=VLGLlIhn3Ng>  
  
For spacing in bootstrap react

https://getbootstrap.com/docs/5.1/utilities/spacing/  
Text

Description automatically generated  
**You first specify what spacing you want either:**  
m (margin)  
p( padding)  
Then you specify the sides  
t (top  
b(bottom)  
…..  
Then a # between 0 and 5 (also auto) (You cannot specify any other numbers)

So for this example below, he used “mt-4” (margin-top) with a 4 spacer (\* 1.5)

Text

Description automatically generated

And for Text  
https://getbootstrap.com/docs/5.1/utilities/text/  
Its in the docs

Graphical user interface, text, application, email

Description automatically generated  
  
For example:

Text

Description automatically generated

**Adding a Custom Style css attribute**

If you want to add your own custom style for a component, you have to use in the format below:  
{ { <style: ‘ ‘}}  
Text

Description automatically generated

**DEPLOYMEMT**

**Testing the App out at HISD**  
Nutritional Services  
Server - SQLINT01  
NutritionalServices  
User Name : u\_NutSvs  
Password: 8~QXCob/Zp  
  
Building the project  
**run npm build**

Changed the web.config file’s connection string out in the WEBAPI Project  
Published the files out the Dev Server  
  
On the Server  
WAPPDEVWEB01  
  
Graphical user interface, text, application

Description automatically generated  
  
Named the Virtual Directory  
SBHISDCustomDevAppsWebAPI  
  
Graphical user interface, text, application, email

Description automatically generated  
  
Graphical user interface, text, application, chat or text message

Description automatically generated  
  
<http://appsdev.houstonisd.org/SBHISDCustomDevAppsWebAPI>

**The Final URLS:**  
<http://appsdev.houstonisd.org/SBHISDCustomDevAppsWebAPI>  
<http://appsdev.houstonisd.org/SandBoxReactWithWebPack>

Graphical user interface, text, application

Description automatically generated  
  
Now for installing the React App  
I changed the config.js file:  
A screenshot of a computer

Description automatically generated with medium confidence  
  
Did a test to see if the files would upload from my local build  
  
Looked at SQL  
Graphical user interface, text

Description automatically generated  
  
Ran the page  
Graphical user interface, application

Description automatically generated  
  
And it works

A screenshot of a computer

Description automatically generated with medium confidence

Now I am going to do a build and deploy:  
npm run build  
  
Graphical user interface, application

Description automatically generated  
  
  
I named web application the same as the project on the dev server  
Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated  
  
The URL

<http://appsdev.houstonisd.org/SandBoxReactWithWebPack>  
Graphical user interface, text, application

Description automatically generated  
  
Starting the test:

Graphical user interface, application

Description automatically generated  
And it works:

Graphical user interface, text, application

Description automatically generated  
  
Graphical user interface, table

Description automatically generated  
  
**HELPER – TROUBLESHOOTING TOPICS**

**How to kill server when seeing “EADDRINUSE: address already in use”**

<https://levelup.gitconnected.com/how-to-kill-server-when-seeing-eaddrinuse-address-already-in-use-16c4c4d7fe5d>

C:\Users\admin>netstat -ano|findstr "PID :3000"

Proto Local Address Foreign Address State PID

TCP 0.0.0.0:3000 0.0.0.0:0 LISTENING 18264

To kill this process (the /f is force):

taskkill /pid 18264 /f

**Issue running npm install and running your project**

For HISD  
**First check your node version**  
Graphical user interface, text, application

Description automatically generated

If you are having an issue getting the node\_modules to install running   
npm install  
Delete the **node\_modu**les directory  
Make sure you don’t have any files open or in use in your project  
Open a command prompt as “Administrator”  
run  
npm install  
If no errors, run  
npm run start:dev  
Text

Description automatically generated

**Handling React Errors with the DOM -Rendering  
When you see something like this:**

**Text

Description automatically generated with low confidence**

You can click on the link to see the true error:

**Graphical user interface, text, application, email

Description automatically generated**  
  
In this case it was the way that I had one of my functions coded inside of one of my components

This was the fix

Graphical user interface, text, application, chat or text message

Description automatically generated  
  
I had my function inside of one of my components like below:

Text

Description automatically generated  
  
The fix:

Text

Description automatically generated  
  
So when you are calling methods inside of one of your components, you have to make sure that the method is already bound, or use the syntax above.

<Button onClick={() =>setShow(false)}>Close</Button>

The reason for this (I saw this before), is because it call the click function without being clicked if you write it like:  
<Button onClick={setShow(false)}>Close</Button>

**It causes an unexpected “re-render” of the DOM**

Here is another example that was broken before I re-wrote the functions as “arrow” functions  
When I was trying to pass props to my child component, and then used a bootstrap button, it kept breaking when I was using a regular function call.

**A screenshot of a computer

Description automatically generated with medium confidence**

**REACT FINDINGS – CONCEPTS**

**How React let’s you do an import as an “an”**

A screenshot of a computer

Description automatically generated with medium confidence  
  
Notice above how you can reference a reducer as an alias without spelling out the actual function name

**FINDINGS – VETTED OUT SNIPPETS**  
  
**Getting the attachment code to work from the Grid**  
Graphical user interface, application, Word

Description automatically generated  
  
I used a cell formatter

Text

Description automatically generated  
  
A picture containing graphical user interface

Description automatically generated  
  
I updated the service with the same code in HISDNutritional Services  
Text

Description automatically generated  
  
  
**Dialog Boxes:**

Graphical user interface, application

Description automatically generated

**The traditional Fix (The last resort)**

Graphical user interface, text, application, email

Description automatically generated

**It is populated by**

**Text

Description automatically generated**

**From useEffect**

**Text

Description automatically generated**

**And getting values is just pure JavaScript**

**Text

Description automatically generated**

**But with the difficulty of managing adding and removing items, we can use good old traditional JavaScript**

**Works like a charmGraphical user interface, application

Description automatically generated**

**Multi-Select – The one below is still in investigation (cannot get it work with dynamic)**

Graphical user interface, application, Word

Description automatically generated<https://www.npmjs.com/package/react-multiselect-dropdown-bootstrap>  
  
Graphical user interface, application, Word

Description automatically generated

Very good docs on the control

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**HELPER SECTION**

**React Snippets for VSCODE**

**Text

Description automatically generated**

**Stop UseEffect from firing indefinitely**useEffect uses a dependent array, if it not supplied, it will continuously run. Just place an [] array in the function and it will run only once.

**Text

Description automatically generated**

**Boilerplate code for new component**import {Button,

Card,

Container,

Row,

Col,

Image} from 'react-bootstrap';

<div>

<main>

<Container>

<Row>

<Col sm={12}>

</Col>

</Row>

</Container>

</main>

</div>

**REFERNCE INFO**

**React NPM’s for file-upload**  
<https://www.npmjs.com/package/use-file-upload>  
**React-dropzone-npm  
  
Bootstrap 5 Selects**  
<https://getbootstrap.com/docs/5.0/forms/select/>

<https://react-bootstrap.github.io/forms/select/>  
<https://www.npmjs.com/package/react-multiselect-dropdown-bootstrap>  
<https://stackoverflow.com/questions/54573926/how-to-use-multi-select-dropdown-in-react-bootstrap>

**Passing data between Parent and Child Components**

https://www.geeksforgeeks.org/how-to-pass-data-from-child-component-to-its-parent-in-reactjs/<https://www.freecodecamp.org/news/pass-data-between-components-in-react/>

**Callbacks**:  
The way that callbacks work is that you are essentially creating a pointer to a method back to your function in your parent component. You are making a “**CALL BACK**” to the parent. In the examples below you will see how you define your actual function/method in your parent component. In your child component (in the case of the custom button example below), the function that we are firing exists in the parent, we then pass a pointer to the props of the child component, the onClick handler uses that pointer to call the function in the parent. This is typical behavior of callbacks in any language. You are making a call back to the parent to satisfy the click event. If you see the example below of SWAV, it is doing the same thing. He is using custom components in the waivers page, the custom components have markup and buttons that have actions that need to be fired, we use callbacks to make the function calls on the parent.  
**A screenshot of a computer

Description automatically generated with medium confidence**  
  
**A screenshot of a computer

Description automatically generated with medium confidence**