React

**Catchup**

Array Iterator Methods:

**Map** – can apply a function to each value in the array and creates a new array

**Filter** - creates a new array with all elements that pass the test implemented by the provided function

**Reduce** - executes a reducer function (that you provide) on each element of the array, resulting in a single output value

Classes:

A class is a type of function, but instead of using the keyword function to initiate it, we use the keyword class, and the properties are assigned inside a constructor() method.

Use the keyword class to create a class, and always add the constructor() method. The constructor method is called each time the class object is initialized.

What is React?

React is a declarative, efficient, and flexible JavaScript library for building user interfaces. React deals with all the DOM stuff for us: we just write basic components, that look almost like HTML, and it wires everything up.

React is built on the philosophy that large complex apps should be built by com- bining small simple components.

Scripts:

* Start
* Build
* Test
* Eject

Packs:

**Babel** - Babel is a toolchain that is mainly used to convert ECMAScript 2015+ code into a backwards compatible version of JavaScript in current and older browsers or environments.

**Webpack** - It’s a tool that lets you bundle your JavaScript applications (supporting both ESM and CommonJS), and it can be extended to support many different assets such as images, fonts, and stylesheets.

**Es Lint** - statically analyzes your code to quickly find problems.

Process for initialising React App:

1. npm init react-app “App Name”
2. cd “App Name”
3. npm start

Process for declaring a class:

1. Render HTML on Page
2. Track data in state and use state to update display
3. Source data in state from props or elseware
4. Add interactivity so that state is updated when the user does something
5. Add event listeners to elements and test they run
6. Update the state

React Router

React Router is a collection of **navigational components** that compose declaratively with your application.

<Router>

<Header title=”Title text” Links={links}/>

<Switch>

<Link to=”/path-name>Path Name</link>

<Link to=”/path-name>Path Name</link>

<Route path=”/path-name” component={component-name}>

<Component-path1 />

</Route>

<Route path=”/path-name”>

<Component-path1 />

</Route>

<Route path=”/path-name/:id” render={({ match })} => {

<Component article={match.params.id} />

};

</Switch>

</Router>