**NOTE 3:**

Q: how to rename folder in terminal?

A: mv old\_name new\_name

Node server.js : to start the backend server

**PROPS:**

Look at this custom attribute

<User username=’Bryan’ />

* This is the same as giving a function an argument to call it: Function username(Bryan)

Now we can access that attribute by using this.props

Ex:

class User extends React.Component {

render(){

return(

<p> Username: {this.props.username} </p>

<p> Is Friend?: {this.props.friend}</p>

)

}

}

<User **username=’Bryan’ friend={ true}**/>

NOTE: {} denotes Javascript expression

Highlighted in red is the props of that **class**

**CREATE-REACT-APP:**

1)

\_Sudo npm –g install Create-React-App

\_Then create-react-app project\_name

This will give you an src file with App.cs and App.js

2) But if you clone it from the Facebook repo here <https://github.com/facebookincubator/create-react-app>

Those won’t have that, but it’s unnecessary to clone since it’s just Facebook backend files.

create-react-app my-app

**TERMINAL:**

\_mkdir creates a folder/ directory

\_To remove empty folder, use rm –r name

\_To create an empty file, use touch. Ex: touch filename

\_Install Shell Command on Atom: Use cmd + shift + P

\_To open a file, just use atom filename

\_You can create an empty file with atom using atom newfilename. After putting content into that file, you can save it, then it’ll show up

**REACT:**

\_We can store state in a component

Q: Why do I need to do **import React, { Component} from 'react'**?

A: First, we import the React module from the external react module

We use {Component} so that we will have access to use Component as opposed to writing React.Component

Ex: Without the {Component} imported:

class ListContacts extends React.Component{}

But with it:

class ListContacts extends Component{}

Q: Why do we need to extend React.Component class?

A: (not totally confirmed)

But we do it so that we can initialize the constructor and change by using this.state

\_Within the render(){

return(

<UI>

)

}

}

Within the return{}, give a UI

\_ import ListContacts from './ListContacts'

* This means I import the ListContacts from that URL to use in the UI view
* In the ListContacts file, I have to export default ListContacts

\_ <ListContacts contacts={contacts}/>

* Pass data from 1 component/ file down to a component, I have to give it an attribute, and then you can access that via this.props
* The {contacts} refers to the variable contacts above

\_ So the console.log(‘Props’, this.props) from the ListContacts page means that you access your attribute by using this.props

(Like we can access arguments passed into a regular JS function, we can access components’ props with this.props)

Q: How to pass a prop?

A: <Component prop\_name={prop\_attribute} />

Ex:

<Clock currentTime={newDate().getTime()} />

Q: How to access the value of a prop from inside the component?

Ex: <Clock currentTime='3:41pm' />

A: **this.props.**currentTime

Q: how to map?

A: this.props.contacts.map(contact) => (

<li key={contact.id}>

{contact.name}

</li>

))}

</ol>

}

}

}

1. Need to give a key
2. Use {} to indicate JS
3. Give className so that I can highlight the CSS class

Q: How to give an avatar image (background image)?

A: First, give a className to initialize the CSS class, then put in an inline style

For Background Image of that (for example, if it’s in a list, then the images will appear horizontally), use the **backgroundImage: `url(${backgroundURL})`**

Remember to use the ` symbol, not ‘. Otherwise, will have error. The ` symbol is below ~

Ex:

<li key ={contact.id} className='contact-list-item'>

<**div className= 'contact-avatar' style={{**

**backgroundImage: `url(${contact.avatarURL})`**

}}/>

This <li> is wrapped inside the <ol>. So unordered list, then “list” inside

Q: how to create a remove button> (appears as an X)?

A: <button> Remove </button>

Ex:

<button className= 'contact-remove'>

Remove

</button>

Q: how to pass multiple props?

A: Pass them in individually

Ex: <button className= 'contact-remove'>

Remove

</button>

**SUMMARY ON PROP**

Q: How to pass and access a prop?

A: First, pass a prop:

<LogoutButton text=’Wanna log out?’ />

Text is the prop. The string is the value.

To access the text prop from inside the component, we use this.props.text

Ex:

render() {

return

<div>{this.props.text}</div>

}

* Here we don’t use () at return because we only need to return 1 thing

Q: Components and Props?

A: Props is an attribute of the component