**4 – React Router and Routing**

Routing in a single page application 🡪 is going back and forth /using urls . ***HOC -- higher order component***

**APP.js**

import React from 'react';

import './App.css';

import HomePage from './pages/homepage.component';

**import {Route, Switch} from 'react-router-dom';**

const HatsPage = () =>{

return(

<div>

<h1> hats page</h1>

</div>

);

}

function App() {

return (

<div>

**<Switch>**

**<Route exact path='/' component={HomePage}/>**

**<Route exact path='/hats' component={HatsPage}/>**

**</Switch>**

</div> ); }

**react-router-dom (library)**

Here routing defines , routing to different page components using urls .

**npm install react-router-dom**

**or**

**yarn add react-router-dom**

**Component :** component to be loaded when path is

**path =”/”** at the base page . property that the router is looking to match

**exact :** true or false property (default true) without exact , / and /hats are technically true

**switch :** when a match is found, it doesn’t not render anything else

**Index.js**

import React from 'react';

import ReactDOM from 'react-dom';

import './index.css';

import App from './App';

**import {BrowserRouter} from 'react-router-dom';**

ReactDOM.render(

**<BrowserRouter>**

**<App />**

**</BrowserRouter>**

, document.getElementById('root'));

**<BrowserRouter>** is a component wrapped around our Application .It provides our application all of the functionalities of routing

<Link to={`${props.match.url}/21`}>TO TOPIC 21</Link>

</div>

); };

const TopicDetail = props => {

return (

<div>

<h1>TOPIC DETAIL PAGE**: {props.match.params.topicId**}</h1>

</div> // match property is useful for nested routing

);};

function App() {

return (

<div>

<Route exact path='/' component={HomePage} />

<Route exact path='/blog/asdqw/topics' component={TopicsList} />

<Route path='/blog/asdqw/topics/:topicId' component={TopicDetail} />

<Route exact path='/blog/topics' component={TopicsList} />

<Route path='/blog/topics/**:topicId'** component={TopicDetail} />

</div>

);} **:topicId // url parameter or router parameter (dynamically changing)**

// values passed as parameter

**Example** program

import React from 'react';

import { Route, **Link** } from 'react-router-dom';

import './App.css';

**// Link : dynamically pass in property**

const HomePage = props => {

console.log(props);

return (

<div>

<button onClick={() => props.history.push('/topics')}>Topics </button>

<h1>HOME PAGE</h1>

</div>

);};

const TopicsList = props => {

return (

<div>

<h1>TOPIC LIST PAGE</h1>

**<Link to**={`${props.match.url}/13`}>TO TOPIC 13**</Link>**

<Link to={`${props.match.url}/17`}>TO TOPIC 17</Link>

**withRouter**

***higher order component*** that takes in component as arguments and returns modified component

import {withRouter } from ‘react-router-dom’;

**export default withRouter(MenuItem);**

**example prg**

**<Route exact path='/' component={HomePage}/>**

this.state={

sections:[

{ title: 'hats',

imageUrl: 'https://i.ibb.co/cvpntL1/hats.png',

id: 1,

**linkUrl: 'hats'** },

**<div className="directory-menu">**

**{ this.state.sections.map(({id,title,imageUrl,size,linkUrl}) =>**

**<MenuItem id={id} title={title} imageurl={imageUrl} size={size} linkUrl={linkUrl}/> ) }</div>**

**(or) (using spread operator)**

**<div className="directory-menu">**

**{ this.state.sections.map(({id,...otherSectionProps}) =>**

**<MenuItem id={id} {...otherSectionProps}/> )**

**} </div>**

const MenuItem = ({title,imageUrl,size,linkUrl,**match,history**}) =>(

<div className={`menu-item ${size}`} >

<div className="background-image" tyle={{background:`url(${imageUrl})`}}

**onClick={()=>history.push(`${match.url}${linkUrl}`)}>**

**export default withRouter(MenuItem);**

Two ways to navigate:

**Link compone**nt: special component .does not re-render like <a> , it just mounts the component that is necessary

**History. Push:** call the link. Dynamically functionally control when we want this to work

Props -> history -> push pushes url link

location **Location props** - tells us where we are currently, full path name

match -> isExact

params

path

url