Contents

[SETTINGS UP LINKS 3](#_Toc76145115)

[Blog.js [FROME post react project] 3](#_Toc76145116)

[Blog.css 4](#_Toc76145117)

[SETTING UP ROUTER PACKAGES 4](#_Toc76145118)

[INSTALLATION: 4](#_Toc76145119)

[App.js 4](#_Toc76145120)

[Preparing Routing for project: 5](#_Toc76145121)

[Posts.js 5](#_Toc76145122)

[Blog.js 6](#_Toc76145123)

[Blog.css 7](#_Toc76145124)

[Posts.css 8](#_Toc76145125)

[SETTINGS UP AND RENDERING ROUTERS: 8](#_Toc76145126)

[RENDERING COMPONENTS FOR ROUTERS: 9](#_Toc76145127)

[SWITCHING BETWEEN PAGES: 9](#_Toc76145128)

[USING LINKS TO SWITCH PAGES: 9](#_Toc76145129)

[USING ROUTE RELATED PROPS: 10](#_Toc76145130)

[THE WITHROUTER HOC & ROUTE PROPS 10](#_Toc76145131)

[Post.js 10](#_Toc76145132)

[RELATIVE PATH VS ABSOLUTE PATH: 11](#_Toc76145133)

[Blog.js 12](#_Toc76145134)

[STYLING ACTIVE CLASS: 12](#_Toc76145135)

[Blog.css 12](#_Toc76145136)

[Blog.js 12](#_Toc76145137)

[Blog.js 13](#_Toc76145138)

[Posts.js 13](#_Toc76145139)

[EXTRACTING ROUTERS PARAMETERS: 14](#_Toc76145140)

[FullPosts.js: 14](#_Toc76145141)

[FullPost.js 15](#_Toc76145142)

[TO REMOVE THIS ISSUES: 16](#_Toc76145143)

[ONE METHOD: 16](#_Toc76145144)

[Blog.js 16](#_Toc76145145)

[Post.js 16](#_Toc76145146)

[ANOTHER METHOD: 16](#_Toc76145147)

[Blog.js: 16](#_Toc76145148)

[NAVIGATING PROGRAMMTICALLY: 17](#_Toc76145149)

[Posts.js 17](#_Toc76145150)

[RERENDERING ACTIVE LINKS: 18](#_Toc76145151)

[Issues: 18](#_Toc76145152)

[UNDERSTANDING NESTING ROUTES: 19](#_Toc76145153)

[Blog.js 19](#_Toc76145154)

[Posts.js 20](#_Toc76145155)

[App.js 22](#_Toc76145156)

[ANOTHER ONE: 22](#_Toc76145157)

[Blog.js 24](#_Toc76145158)

[CREATING DYNAMICALLY NESTED ROUTES: 25](#_Toc76145159)

[REDIRECT THE USER / THE REQUEST: 27](#_Toc76145160)

[Blog.js 27](#_Toc76145161)

[CONDITIONAL REDIRECT: 27](#_Toc76145162)

[USING THE HISTORY PROP TO REDIRECT(REPLACE); 29](#_Toc76145163)

[NewPost.js 29](#_Toc76145164)

[NAVIGATION GAURDS: 29](#_Toc76145165)

[HANDLING THE 404 CASE (UNKNOWN ROUTES) 31](#_Toc76145166)

[LOADING ROUTES LAZILLY - [IMPORTANT]: 31](#_Toc76145167)

[AsyncComponent.js 32](#_Toc76145168)

[Blog.js 32](#_Toc76145169)

[Lazy Loading with React Suspense(16.6) 34](#_Toc76145170)

[Dependencies: 34](#_Toc76145171)

[NEW PROJECT: 34](#_Toc76145172)

[App.js 34](#_Toc76145173)

[ADDING SUSPENSE AND LAZY LOADING: 35](#_Toc76145174)

[ANOTHER WAY IF CLIENT BASED APPLICATION: 36](#_Toc76145175)

[ROUTING AND THE SERVER (DEPLOYMENT) 36](#_Toc76145176)

[App.js 36](#_Toc76145177)

# SETTINGS UP LINKS

## Blog.js [FROME post react project]

import React, { Component } from 'react';

import NewPost from '../../components/NewPost/NewPost';

import Post from '../../components/Post/Post';

import FullPost from '../../components/FullPost/FullPost';

import './Blog.css';

import axios from 'axios';

class Blog extends Component {

        //map provides new array

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><a href='/'>Home</a></li>

                        <li><a href='/new-post'>New Post</a></li>

                        <li><a href='/'></a></li>

                        <li><a href='/'></a></li>

                    </ul>

                </nav>

            </header>

            <section className="Posts">

                {posts}

            </section>

            <section>

                <FullPost id={this.state.selectedPostId}/>

            </section>

            <section>

                <NewPost />

            </section>

        </div>

        );

    }

}

export default Blog;

## Blog.css

.Posts {

    display: flex;

    flex-flow: row wrap;

    justify-content: center;

    width: 80%;

    margin: auto;

}

.Blog ul {

    list-style: none;

    margin: 0;

    padding: 0;

    width: 100%;

    text-align: center;

}

.Blog li {

    display: inline-block;

    margin:10px;

}

.Blog a {

    text-decoration: none;

    color: black;

}

.Blog a:hover,

.Blog a:active{

    color:coral

}

# SETTING UP ROUTER PACKAGES

## INSTALLATION:

npm install --save react-router react-router-dom

## App.js

import React, { Component } from 'react';

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

import Blog from './container/Blog/Blog';

class App extends Component {

  render() {

    return (

      <BrowserRouter>

      <div className="App">

        <Blog />

      </div>

      </BrowserRouter>

    );

  }

}

export default App;

# Preparing Routing for project:

NOTE: Create file name Posts.js and Posts.css in Posts folder inside Blog folder of container. Tips: FullPost folder and NewPost folders are copied inside Blog folder of container folder.

## Posts.js

import React,{ Component } from 'react';

import axios from 'axios';

import Post from '../../../components/Post/Post';

import './Posts.css';

class Posts extends Component {

    state = {

        posts: [],

        // selectedPostId: null,

        // error: false,

    }

    componentDidMount() {

        axios.get('/posts')

        .then(response => {

            //transform data [from 1 to 4]

            const posts = response.data.slice(0,4);

            const updatedPosts = posts.map(

                post => {

                    return{

                        ...post,

                        author: 'Max'

                }

                }

            )

            this.setState({posts: updatedPosts})

            // console.log(response);

        })

        .catch(error => {

            console.log(error)

            // this.setState({error: true})

        });

    }

    postSelectedHandler = (id) => {

        this.setState({selectedPostId: id});

    }

    render(){

        let posts = <p>Something wrong</p>

        if(!this.state.error){

            posts = this.state.posts.map(

                post => {

                    return <Post key={post.id} clicked={() => this.postSelectedHandler(post.id)} title={post.title} author={post.author}/>

                }

            );

        }

        return(

            <section className="Posts">

                {posts}

            </section>

        )

    }

}

export default Posts;

## Blog.js

import React, { Component } from 'react';

import './Blog.css';

import Posts from '../../container/Blog/Posts/Posts';

class Blog extends Component {

    render(){

            //map provides new array

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><a href='/'>Home</a></li>

                        <li><a href='/new-post'>New Post</a></li>

                        <li><a href='/'></a></li>

                        <li><a href='/'></a></li>

                    </ul>

                </nav>

            </header>

            <Posts />

            {/\* <section>

                <FullPost id={this.state.selectedPostId}/>

            </section>

            <section>

                <NewPost />

            </section> \*/}

        </div>

        );

    }

}

export default Blog;

## Blog.css

.Blog ul {

    list-style: none;

    margin: 0;

    padding: 0;

    width: 100%;

    text-align: center;

}

.Blog li {

    display: inline-block;

    margin:10px;

}

.Blog a {

    text-decoration: none;

    color: black;

}

.Blog a:hover,

.Blog a:active{

    color:coral

}

## Posts.css

.Posts {

    display: flex;

    flex-flow: row wrap;

    justify-content: center;

    width: 80%;

    margin: auto;

}

# SETTINGS UP AND RENDERING ROUTERS:

import React, { Component } from 'react';

import { Route } from 'react-router-dom';

import './Blog.css';

import Posts from '../../container/Blog/Posts/Posts';

class Blog extends Component {

    render(){

            //map provides new array

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><a href='/'>Home</a></li>

                        <li><a href='/new-post'>New Post</a></li>

                        <li><a href='/'></a></li>

                        <li><a href='/'></a></li>

                    </ul>

                </nav>

            </header>

            {/\* localhost:3000/ => Post component  \*/}

            <Route path="/" exact render={() => <Posts/> }/>

        </div>

        );

    }

}

export default Blog;

# RENDERING COMPONENTS FOR ROUTERS:

            <Route path="/" exact component={Posts} />

## SWITCHING BETWEEN PAGES:

  return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><a href='/'>Home</a></li>

                        <li><a href='/new-post'>New Post</a></li>

                        <li><a href='/'></a></li>

                        <li><a href='/'></a></li>

                    </ul>

                </nav>

            </header>

            {/\* localhost:3000/ => Post component  \*/}

            <Route path="/" exact component={Posts} />

            <Route path="/new-post" exact component={NewPost} />

        </div>

        );

## USING LINKS TO SWITCH PAGES:

NOTE: while switching pages using Link, it does not reload or send request. It will just update the components in same page.

import React, { Component } from 'react';

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

import './Blog.css';

import Posts from '../../container/Blog/Posts/Posts';

import NewPost from '../../container/Blog/NewPost/NewPost';

class Blog extends Component {

    render(){

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><Link to="/">Home</Link></li>

                        <li><Link to={{

                            pathname: '/new-post',

                            hash: '#submit',

                            search: '?quick-submit=true'

                        }}>New Post</Link></li>

                    </ul>

                </nav>

            </header>

            <Route path="/" exact component={Posts} />

            <Route path="/new-post" exact component={NewPost} />

        </div>

        );

    }

}

export default Blog;

# USING ROUTE RELATED PROPS:

componentDidMount() {

        console.log(this.props);

    }

# THE WITHROUTER HOC & ROUTE PROPS

## Post.js

import React from 'react';

import { withRouter } from 'react-router-dom';

import './Post.css';

const post = (props) => {

    console.log(props);

    return(

        <article className="Post"  onClick={props.clicked}>

        <h1>{props.title}</h1>

        <div className="Info">

            <div className="Author">{props.author}</div>

        </div>

     </article>

    )

    };

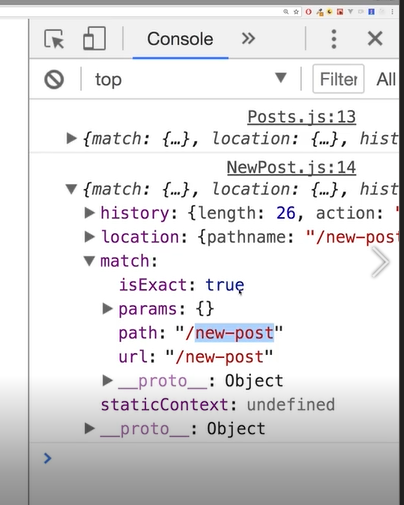
export default withRouter(post);

# RELATIVE PATH VS ABSOLUTE PATH:

NOTE: Absolute path means: [www.shakti/posts/post](http://www.shakti/posts/post).

Relative path means: [www.shakti/post](http://www.shakti/post)





## Blog.js

return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><Link to="/">Home</Link></li>

                        <li><Link to={{

                            pathname: this.props.match.url + '/new-post',

                            hash: '#submit',

                            search: '?quick-submit=true'

                        }}>New Post</Link></li>

                    </ul>

                </nav>

            </header>

            <Route path="/" exact component={Posts} />

            <Route path="/new-post" exact component={NewPost} />

        </div>

        );

# STYLING ACTIVE CLASS:

## Blog.css

.Blog a:hover,

.Blog a:active,

.Blog a.active{

    color:coral

}

## Blog.js

import React, { Component } from 'react';

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

import './Blog.css';

import Posts from '../../container/Blog/Posts/Posts';

import NewPost from '../../container/Blog/NewPost/NewPost';

class Blog extends Component {

    render(){

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><NavLink

                            to="/"

                            exact

                            activeClassName='MyActive-class'

                            activeStyle={{

                                color: 'fa923f',

                                textDecoration: 'underline'

                            }}

                            // activeClassName='active'

                            >Home</NavLink></li>

                        <li><NavLink to={{

                            pathname:'/new-post',

                            hash: '#submit',

                            search: '?quick-submit=true'

                        }}>New Post</NavLink></li>

                    </ul>

                </nav>

            </header>

            <Route path="/" exact component={Posts} />

            <Route path="/new-post" exact component={NewPost} />

        </div>

        );

    }

}

export default Blog;

PASSING ROUTERS PARAMETER:

## Blog.js

import FullPost from '../../container/Blog/FullPost/FullPost';

NOTE: id is dynamic value. Example: /1 or /2 or /3.

<Route path="/" exact component={Posts} />

            <Route path="/new-post" exact component={NewPost} />

            <Route path="/:id" exact component={FullPost} />

## Posts.js

NOTE: post.id selects post which we have click. Key should be always on top of the loop otherswise it throws error.

render(){

        let posts = <p>Something wrong</p>

        if(!this.state.error){

            posts = this.state.posts.map(

                post => {

                    return (

                        <Link to={'/'+post.id}  key={post.id} >

                            <Post

                            clicked={() => this.postSelectedHandler(post.id)}

                            title={post.title}

                            author={post.author}/>

                        </Link>

                    );

                }

            );

        }

# EXTRACTING ROUTERS PARAMETERS:

## FullPosts.js:

componentDidUpdate(){

        if(this.props.id){

            if( !this.state.loadedPost ||  this.state.loadedPost && this.state.loadedPost.id !== this.props.id){

                axios.get('/posts/'+this.props.id)

                .then(response => {

                    // console.log(resposne)

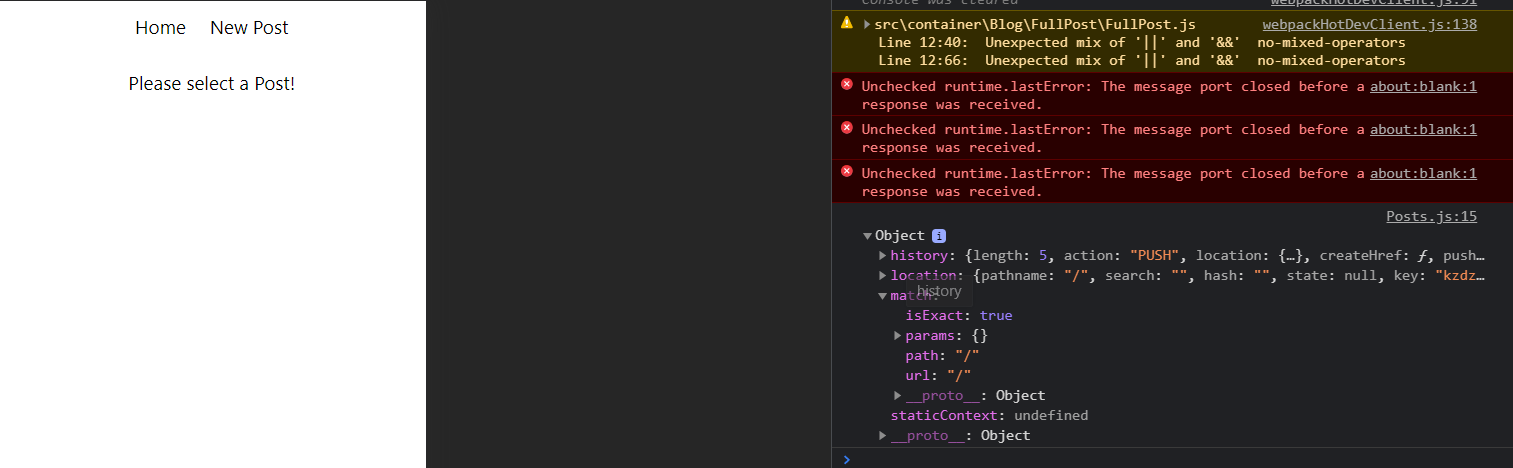
                    this.setState({loadedPost: response.data});

                });

            }

        }

    }



NOTE: params: {}

## FullPost.js

NOTE: componentDidMount() is used for add and remove

componentDidMount(){

        console.log(this.props)

        if(this.props.match.params.id){

            if( !this.state.loadedPost ||  this.state.loadedPost && this.state.loadedPost.id !== this.props.id){

                axios.get('/posts/'+this.props.match.params.id)

                .then(response => {

                    // console.log(resposne)

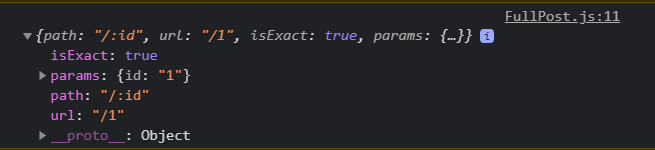
                    this.setState({loadedPost: response.data});

                });

            }

        }

    }



BUT we are having some issues in new post in web



# TO REMOVE THIS ISSUES:

## ONE METHOD:

## Blog.js

            <Route path="/post:id" exact component={FullPost} />

## Post.js

                        <Link to={'/post'+post.id}  key={post.id} >

## ANOTHER METHOD:

NOTE: WRAPPING ROUTE CONFIG WITH SWITCH. IN SWTICH WRAPPER, ROUTE MUST BE IN ORDER.

## Blog.js:

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

<Switch>

                <Route path="/" exact component={Posts} />

                <Route path="/new-post" exact component={NewPost} />

                <Route path="/:id" exact component={FullPost} />

            </Switch>

OR

 <Route path="/" exact component={Posts} />

            <Switch>

                <Route path="/new-post" exact component={NewPost} />

                <Route path="/:id" exact component={FullPost} />

            </Switch>

# NAVIGATING PROGRAMMTICALLY:

## Posts.js

    postSelectedHandler = (id) => {

        // this.setState({selectedPostId: id});

        //HISTORY OBJECT IN CONSOLE, like goBack and push method. push will navigate to new page

        this.props.history.push({pathname: '/' +id});

        // OR

        // this.props.history.push('/' + id);

    }

    render(){

        let posts = <p>Something wrong</p>

        if(!this.state.error){

            posts = this.state.posts.map(

                post => {

                    return (

                        // <Link to={'/'+post.id}  key={post.id} >

                            <Post

                            key={post.id}

                            clicked={() => this.postSelectedHandler(post.id)}

                            title={post.title}

                            author={post.author}/>

                        // </Link>

                    );

                }

            );

        }

        return(

            <section className="Posts">

                {posts}

            </section>

        )

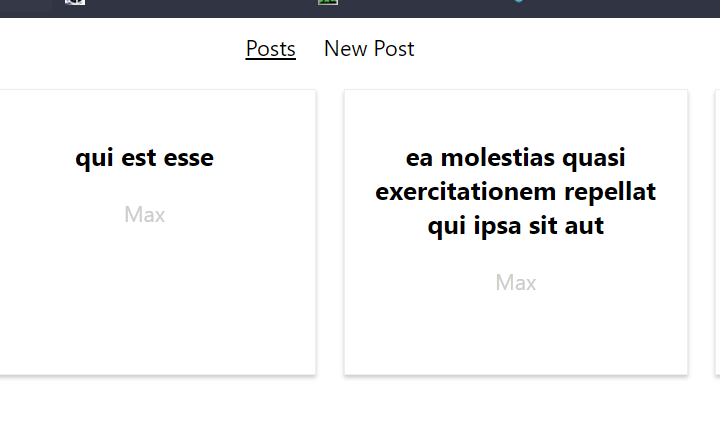
    }

}

export default Posts;

# RERENDERING ACTIVE LINKS:

## Issues:



BUT IN SINGLE POST, IT DOESNOT APPLY EVENTHOUGH IT IS PART OF POSTS.



WE CAN add post after ‘/’:

  <nav>

                    <ul>

                        <li><NavLink

                            to="/post"

                            exact

                            activeClassName='MyActive-class'

                            activeStyle={{

                                color: 'fa923f',

                                textDecoration: 'underline'

                            }}

                            // activeClassName='active'

                            >Posts</NavLink></li>

                    </ul>

                </nav>

# UNDERSTANDING NESTING ROUTES:

## Blog.js

import React, { Component } from 'react';

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

import './Blog.css';

import Posts from '../../container/Blog/Posts/Posts';

import NewPost from '../../container/Blog/NewPost/NewPost';

import FullPost from '../../container/Blog/FullPost/FullPost';

class Blog extends Component {

    render(){

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><NavLink

                            to="/ "

                            exact

                            activeClassName='MyActive-class'

                            activeStyle={{

                                color: 'fa923f',

                                textDecoration: 'underline'

                            }}

                            // activeClassName='active'

                            >Posts</NavLink></li>

                        <li><NavLink to={{

                            pathname:'/new-post',

                            hash: '#submit',

                            search: '?quick-submit=true'

                        }}>New Post</NavLink></li>

                    </ul>

                </nav>

            </header>

            <Route path="/" exact component={Posts} />

            <Switch>

                <Route path="/new-post" exact component={NewPost} />

            </Switch>

        </div>

        );

    }

}

export default Blog;

## Posts.js

import React,{ Component } from 'react';

import axios from 'axios';

import Post from '../../../components/Post/Post';

import './Posts.css';

// import {Link} from 'react-router-dom';

import {Route} from 'react-router-dom';

import FullPost from '../FullPost/FullPost';

class Posts extends Component {

    state = {

        posts: [],

        // selectedPostId: null,

        // error: false,

    }

    componentDidMount() {

        console.log(this.props)

        axios.get('/posts')

        .then(response => {

            //transform data [from 1 to 4]

            const posts = response.data.slice(0,4);

            const updatedPosts = posts.map(

                post => {

                    return{

                        ...post,

                        author: 'Max'

                }

                }

            )

            this.setState({posts: updatedPosts})

            // console.log(response);

        })

        .catch(error => {

            console.log(error)

            // this.setState({error: true})

        });

    }

    postSelectedHandler = (id) => {

        // this.setState({selectedPostId: id});

        //HISTORY OBJECT IN CONSOLE, like goBack and push method. push will navigate to new page

        this.props.history.push({pathname: '/' +id});

        // OR

        // this.props.history.push('/' + id);

    }

    render(){

        let posts = <p>Something wrong</p>

        if(!this.state.error){

            posts = this.state.posts.map(

                post => {

                    return (

                        // <Link to={'/'+post.id}  key={post.id} >

                            <Post

                            key={post.id}

                            clicked={() => this.postSelectedHandler(post.id)}

                            title={post.title}

                            author={post.author}/>

                        // </Link>

                    );

                }

            );

        }

        return(

            <div>

            <section className="Posts">

                {posts}

            </section>

            <Route path="/:id" exact component={FullPost} />

            </div>

        )

    }

}

export default Posts;

NOTE: you can use Route wherever you like as long as it is inside <BroswerROuter>

## App.js

<BrowserRouter>

      <div className="App">

        <Blog />

      </div>

      </BrowserRouter>

AFTER THIS CHANGED CODE, YOU WILL SEE CHANGE IN URL OF SINGLE POST BUT SINGLE POST WILL NOT BEEN SHOWN. REASON: WE HAVE CREATED NESTED ROUTES.

YOU CAN REMOVE ‘exact’

<Switch>

                <Route path="/new-post" component={NewPost} />

                <Route path="/" component={Posts} />

            </Switch>

## ANOTHER ONE:

import React,{ Component } from 'react';

import axios from 'axios';

import Post from '../../../components/Post/Post';

import './Posts.css';

// import {Link} from 'react-router-dom';

import {Route} from 'react-router-dom';

import FullPost from '../../Blog/FullPost/FullPost';

class Posts extends Component {

    state = {

        posts: [],

        // selectedPostId: null,

        // error: false,

    }

    componentDidMount() {

        console.log(this.props)

        axios.get('/posts')

        .then(response => {

            //transform data [from 1 to 4]

            const posts = response.data.slice(0,4);

            const updatedPosts = posts.map(

                post => {

                    return{

                        ...post,

                        author: 'Max'

                }

                }

            )

            this.setState({posts: updatedPosts})

            // console.log(response);

        })

        .catch(error => {

            console.log(error)

            // this.setState({error: true})

        });

    }

    postSelectedHandler = (id) => {

        // this.setState({selectedPostId: id});

        //HISTORY OBJECT IN CONSOLE, like goBack and push method. push will navigate to new page

        this.props.history.push({pathname: '/posts/' +id});

        // OR

        // this.props.history.push('/posts/' + id);

    }

    render(){

        let posts = <p>Something wrong</p>

        if(!this.state.error){

            posts = this.state.posts.map(

                post => {

                    return (

                        // <Link to={'/posts/'+post.id}  key={post.id} >

                            <Post

                            key={post.id}

                            clicked={() => this.postSelectedHandler(post.id)}

                            title={post.title}

                            author={post.author}/>

                        // </Link>

                    );

                }

            );

        }

        return(

            <div>

            <section className="Posts">

                {posts}

            </section>

            {/\* <Route path="/posts/:id" exact component={FullPost} /> \*/}

            {/\* GETTING DYnamic path  \*/}

            <Route path={this.props.match.url + '/:id'} exact component={FullPost} />

            </div>

        )

    }

}

export default Posts;

## Blog.js

import React, { Component } from 'react';

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

import './Blog.css';

import Posts from '../../container/Blog/Posts/Posts';

import NewPost from '../../container/Blog/NewPost/NewPost';

class Blog extends Component {

    render(){

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><NavLink

                            to="/posts/"

                            exact

                            activeClassName='MyActive-class'

                            activeStyle={{

                                color: 'fa923f',

                                textDecoration: 'underline'

                            }}

                            // activeClassName='active'

                            >Posts</NavLink></li>

                        <li><NavLink to={{

                            pathname:'/new-post',

                            hash: '#submit',

                            search: '?quick-submit=true'

                        }}>New Post</NavLink></li>

                    </ul>

                </nav>

            </header>

            <Switch>

                <Route path="/new-post" component={NewPost} />

                <Route path="/posts" component={Posts} />

            </Switch>

        </div>

        );

    }

}

export default Blog;

# CREATING DYNAMICALLY NESTED ROUTES:

NOTE: WHEN RENDER THE POST THEN YOU TRY TO RENDER ANOTHER POST, IT WILL NOT UPDATE THE COMPONENT. YOU CAN ONLY SEE URL UPDATED: ‘/posts/1’ or ‘/posts/2’

import React, { Component } from 'react';

import axios from 'axios';

import './FullPost.css';

class FullPost extends Component {

    state = {

        loadedPost: null,

    }

    componentDidMount(){

        console.log(this.props)

        this.loadData();

    }

    componentDidUpdate() {

        this.loadData();

    }

    loadData(){

        if(this.props.match.params.id){

            if( !this.state.loadedPost ||  this.state.loadedPost && this.state.loadedPost.id !== + this.props.match.params.id){

                axios.get('/posts/'+this.props.match.params.id)

                .then(response => {

                    // console.log(resposne)

                    this.setState({loadedPost: response.data});

                });

            }

        }

    }

    // IN console website, look network where we have request infinte time i.e. infinite loops because of

    // this.setState({loadedPost: response.data});

    // we are continously updating state

    deletePostHandler = () => {

        axios.delete('/posts'+this.props.match.params.id)

        .then(response => {

            console.log(response);

        });

    }

    render () {

        let post = <p style={{textAlign:'center'}}>Please select a Post!</p>;

        //ERROR REMOVING

        if(this.props.match.params.id){

            post = <p style={{textAlign:'center'}}>Loading...</p>;

        }

        if(this.state.loadedPost){

            post = (

                <div className="FullPost">

                    <h1>{this.state.loadedPost.title}</h1>

                    <p>l{this.state.loadedPost.body}</p>

                    <div className="Edit">

                        <button className="Delete" onClick={this.deletePostHandler}>Delete</button>

                    </div>

                </div>

            );

        }

        //NULL TITLE ERROR KINA AYO VANE... componentDidUpdate() vanda aagi nai

        //yo tala ko if wala statement run vayo jun ma {this.state.loadedPost.title}, loadedPost null nai hunxa

        // if(this.props.id){

        //     post = (

        //         <div className="FullPost">

        //             <h1>{this.state.loadedPost.title}</h1>

        //             <p>l{this.state.loadedPost.content}</p>

        //             <div className="Edit">

        //                 <button className="Delete">Delete</button>

        //             </div>

        //         </div>

        //     );

        // }

        return post;

    }

}

export default FullPost;

# REDIRECT THE USER / THE REQUEST:

NOTE: when you enter url: localhost:300/ then it will redirect you to localhost:300/posts

## Blog.js

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

<Switch>

                <Route path="/new-post" component={NewPost} />

                <Route path="/posts" component={Posts} />

                <Redirect from='/' to='/posts' />

            </Switch>

# CONDITIONAL REDIRECT:

IN NewPost.js, When we submit the form then it will redirect to posts page.

import React, { Component } from 'react';

import axios from 'axios';

import { Redirect } from 'react-router-dom';

import './NewPost.css';

class NewPost extends Component {

    state = {

        title: '',

        content: '',

        author: 'Max',

        submitted: false,

    }

    componentDidMount() {

        console.log(this.props);

    }

    postDataHandler = () => {

        // JS Object

        const post = {

            title: this.state.title,

            body: this.state.content,

            author: this.state.author,

        };

        axios.post('/posts', post)

        .then( response => {

            console.log(response);

            this.setState({submitted: true})

        });

    }

    render () {

        let redirected = null;

        if(this.state.submitted){

            redirected = <Redirect to="/posts" />

        }

        return (

            <div className="NewPost">

            {redirected}

                <h1>Add a Post</h1>

                <label>Title</label>

                <input type="text" value={this.state.title} onChange={(event) => this.setState({title: event.target.value})} />

                <label>Content</label>

                <textarea rows="4" value={this.state.content} onChange={(event) => this.setState({content: event.target.value})} />

                <label>Author</label>

                <select value={this.state.author} onChange={(event) => this.setState({author: event.target.value})}>

                    <option value="Max">Max</option>

                    <option value="Manu">Manu</option>

                </select>

                <button onClick={this.postDataHandler}>Add Post</button>

            </div>

        );

    }

}

export default NewPost;

# USING THE HISTORY PROP TO REDIRECT(REPLACE);

## NewPost.js

 postDataHandler = () => {

        // JS Object

        const post = {

            title: this.state.title,

            body: this.state.content,

            author: this.state.author,

        };

        axios.post('/posts', post)

        .then( response => {

            console.log(response);

            // this.setState({submitted: true})

            this.props.history.push('/posts');

        });

    }

NOTE: history objects will be generated when we use <BroswerRouter>.

In push, we can go back to page when we last visited.

In redirect, it will replace the current page.

We can use replace in here to replicate redirect property. ( it can be alternative)

            this.props.history.replace('/posts');

# NAVIGATION GAURDS:

NOTE: YOU ONLY ALLOW USERS TO VISIT SOME PAGES IF THEY ARE AUTHENTICATED.

import React, { Component } from 'react';

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

import './Blog.css';

import Posts from '../../container/Blog/Posts/Posts';

import NewPost from '../../container/Blog/NewPost/NewPost';

class Blog extends Component {

    state  ={

        auth: false,

    }

    render(){

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><NavLink

                            to="/posts/"

                            exact

                            activeClassName='MyActive-class'

                            activeStyle={{

                                color: 'fa923f',

                                textDecoration: 'underline'

                            }}

                            // activeClassName='active'

                            >Posts</NavLink></li>

                        <li><NavLink to={{

                            pathname:'/new-post',

                            hash: '#submit',

                            search: '?quick-submit=true'

                        }}>New Post</NavLink></li>

                    </ul>

                </nav>

            </header>

            <Switch>

                {this.state.auth ?

<Route path="/new-post" component={NewPost} />

: null }

                <Route path="/posts" component={Posts} />

                <Redirect from='/' to='/posts' />

            </Switch>

        </div>

        );

    }

}

export default Blog;

# HANDLING THE 404 CASE (UNKNOWN ROUTES)

<Switch>

                {this.state.auth ? <Route path="/new-post" component={NewPost} /> : null}

                <Route path="/posts" component={Posts} />

                <Route render={() => <h1>NOT FOUND</h1>} />

                // {/\* <Route path="/posts" component={Posts} />

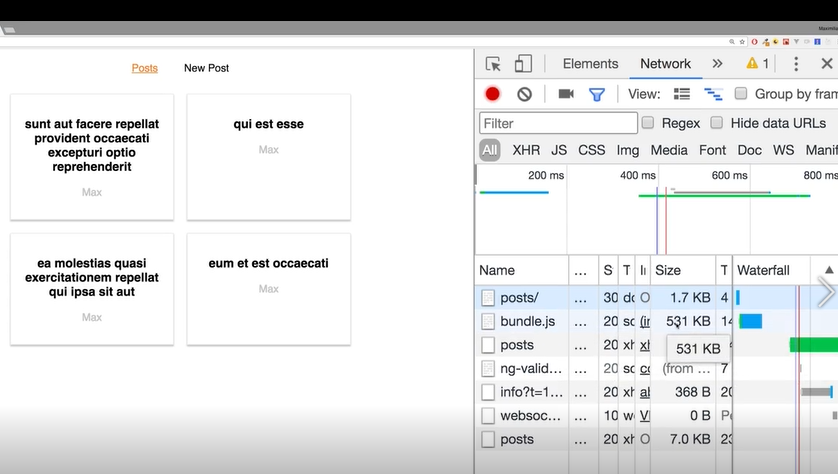
                // <Redirect from='/' to='/posts' /> \*/}

            </Switch>



# LOADING ROUTES LAZILLY - [IMPORTANT]:

NOTE:



Here we are loading the bundle.js, it contains our whole source codes. It can be bad to load all codes if we have big featured app where user might not visit certain page. We should only load the codes of pages or components which user will be navigated. It can be solved by lazy loading or implementing code splitting.

1. Create Folder name: hoc in src folder
2. Create file named: AsyncComponent.js

## AsyncComponent.js

import React, { Component } from 'react';

const asyncComponet = (importComponent) => {

    return class extends Component {

        state ={

            component: null

        }

//this state will set dynamically by loaded importComponent

        componentWillMount () {

//importComponent should be function reference

            importComponent()

            //function that’s return Promise

            .then(cmp => {

//cmp = argument which will have one property ‘default’ which will that component will load dynamically.

                this.setState({component : cmp.default});

            });

        }

//our loaded component i.e newpost will be set in component state

        render () {

            const C =  this.state.component;

//return JSX codes, C is returned as normal components

            return C ? < C {...this.props} /> : null;

        }

    }

}

export default asyncComponet;

## Blog.js

import React, { Component } from 'react';

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

import './Blog.css';

import Posts from '../../container/Blog/Posts/Posts';

// import NewPost from '../../container/Blog/NewPost/NewPost';

import asyncComponet from './hoc/asyncComponent';

//asyncComponet() requires some arguments which must be function so we have to pass some function i.e anaymous function

//import() is function where parenthesis will be only what we need to render.

const AsyncNewPost =  asyncComponet(() => {

    return import('../../container/Blog/NewPost/NewPost');

});

class Blog extends Component {

    state  ={

        auth: true,

    }

    render(){

        return(

            <div className='Blog'>

            <header>

                <nav>

                    <ul>

                        <li><NavLink

                            to="/posts/"

                            exact

                            activeClassName='MyActive-class'

                            activeStyle={{

                                color: 'fa923f',

                                textDecoration: 'underline'

                            }}

                            // activeClassName='active'

                            >Posts</NavLink></li>

                        <li><NavLink to={{

                            pathname:'/new-post',

                            hash: '#submit',

                            search: '?quick-submit=true'

                        }}>New Post</NavLink></li>

                    </ul>

                </nav>

            </header>

            <Switch>

                {this.state.auth ?

<Route path="/new-post"

component={AsyncNewPost} /> : null}

                <Route path="/posts" component={Posts} />

                <Route render={() => <h1>NOT FOUND</h1>} />

                // {/\* <Route path="/posts" component={Posts} />

                // <Redirect from='/' to='/posts' /> \*/}

            </Switch>

        </div>

        );

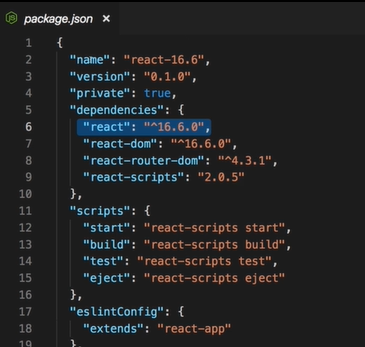
    }

}

export default Blog;

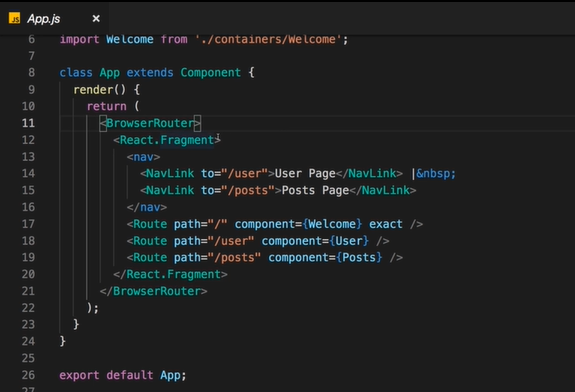
# Lazy Loading with React Suspense(16.6)

## Dependencies:

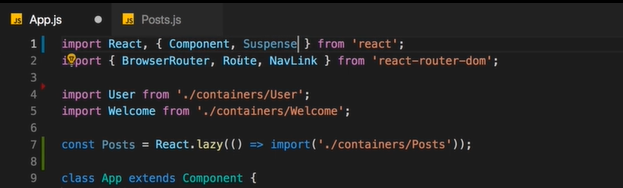


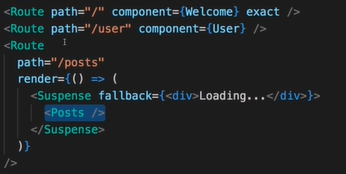
## NEW PROJECT:

### App.js

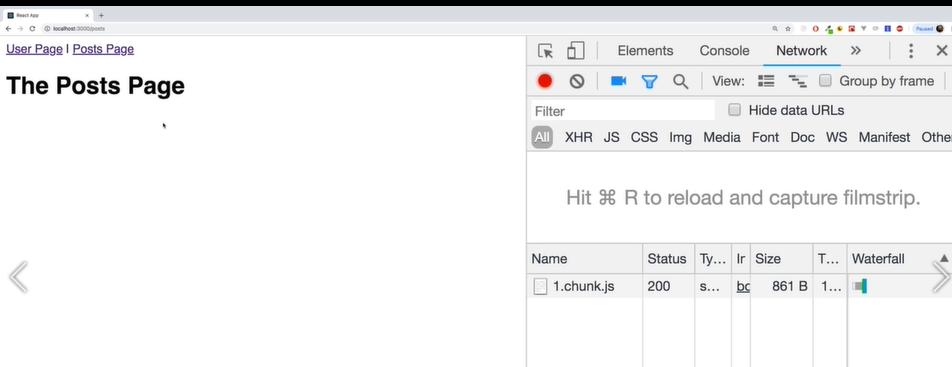


## ADDING SUSPENSE AND LAZY LOADING:

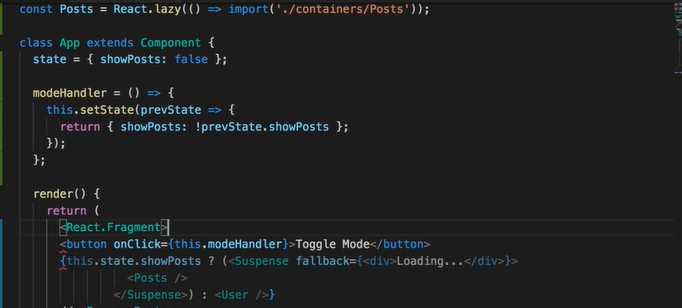




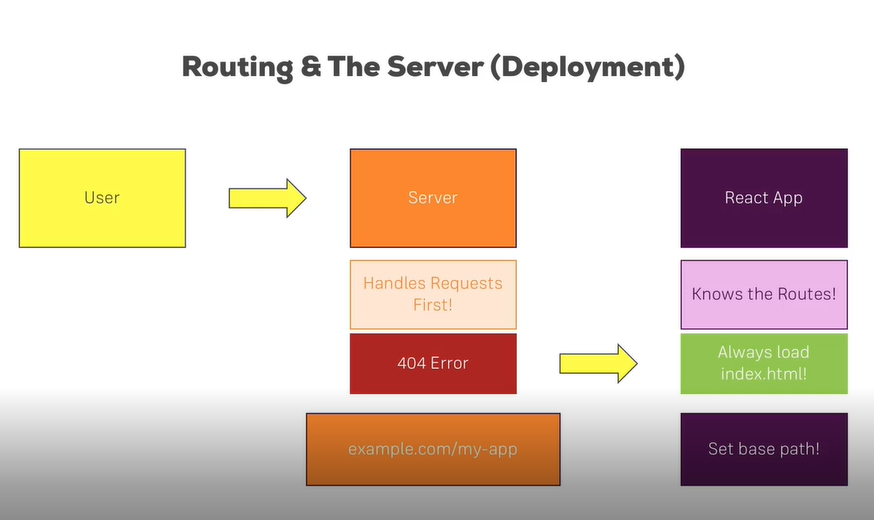
WEB BROSWER:



## ANOTHER WAY IF CLIENT BASED APPLICATION:



# ROUTING AND THE SERVER (DEPLOYMENT)



## App.js

import React, { Component } from 'react';

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

import Blog from './container/Blog/Blog';

class App extends Component {

  render() {

    return (

      <BrowserRouter basename="/my-app">

      <div className="App">

        <Blog />

      </div>

      </BrowserRouter>

    );

  }

}

export default App;