https://shorturl.at/3ICHe 13 Jun 2024

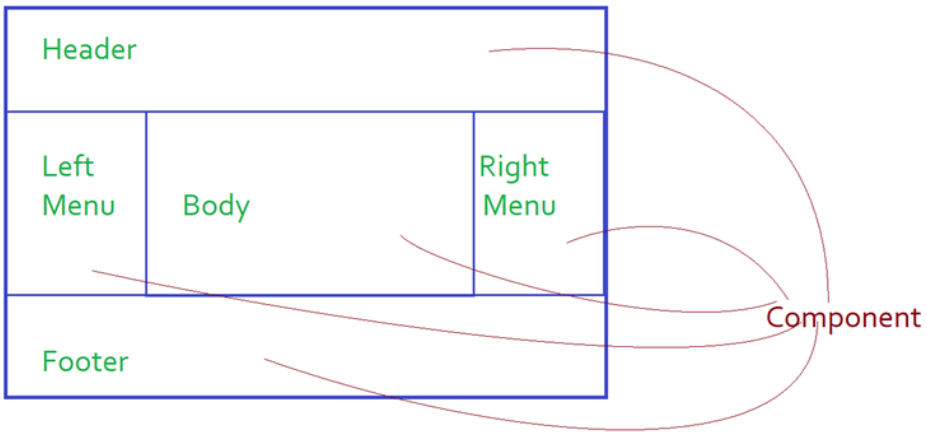
ReactJS Intro
Environmental Setup
Creating First React application
React Project Structure
First Component
Multiple Components in a Component
State
    Creating a state
    Setting and Changing state
Props
Forms and Validations
Single Page Application
    Routing and Navigation
    Nested Routing
    Lazy Loading
Asynchronous API Calls
    Api Calls using axios
    CRUD Operations using SPA

==========

ReactJS Intro

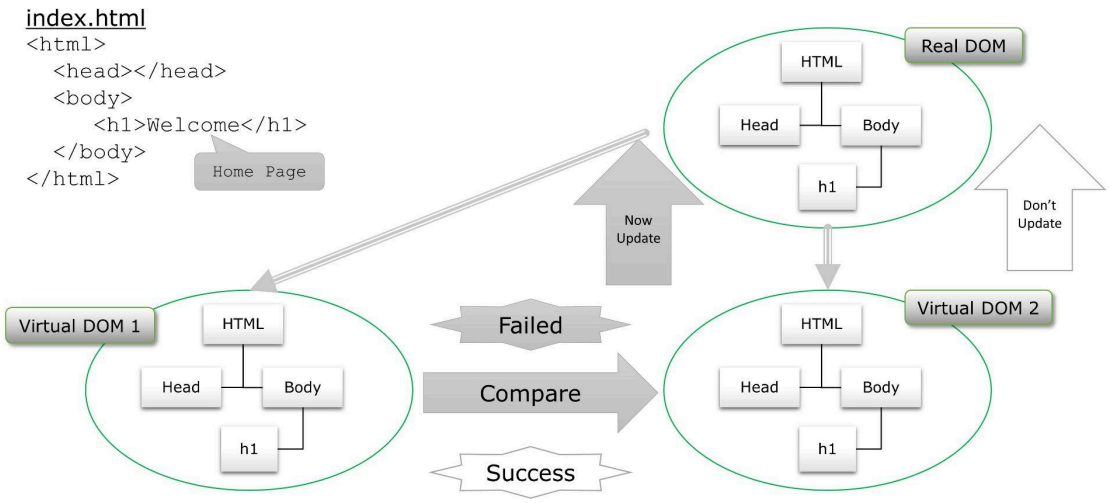
==========

- ReactJS is front end technology.
- React is introduced by facebook.
- There are two type of React
    i) React Native
    ii)ReactJS
- React Native is mostly used to develop 'Mobile Applications'.
- ReactJS is used to develop 'Web Applications'.
- Current version of ReactJS is 18.3.1
- It is released in 26th April 2024.
- ReactJS applications can be developed using 'JSX'.
- JSX stands for 'Javascript XML'.
- 'Babel' is a tool provided by facebook.
- this tool is used to convert ES2015+ code to equivalent backward versions Eg ES5. - ReactJS simplifies Complex UI with the help of components.
- Reusable part of a complex UI is called a component.



- As a React developer we can create more than one components.
- as a React developer we can reuse the components.
- as a React developer we can provide communication between components.
- React applications are faster applications as compared to other technologies because of the Virtual DOM concept.

What is virtual DOM in ReactJS?

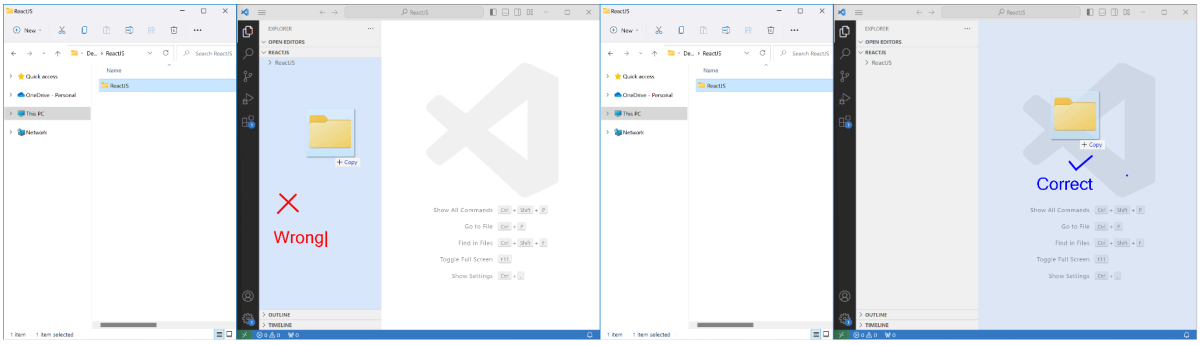


- React creates tree structures of html elements.
- this is called Real DOM.
- internally exactly two copies of Real DOM created.
- these are called Virtual DOM 1 and Virtual DOM 2 respectively.
- on change, VD1 is updated.
- this VD1 is compared with VD2.
- if Comparison is failed update RD.

- simultaneously update VD1 and VD2 also.

==========
Environmental Setup
==========
1. Download and install nodejs
	https://nodejs.org/en/download
2. Download and install git
	https://git-scm.com/downloads
3. Download and install VSCode
	https://code.visualstudio.com/
4. Install 'yarn' tool
	- yarn tool given by facebook
	- this tool is used to download libraries.
	>npm install -g yarn@latest
	$ sudo npm install -g yarn@latest
	% sudo npm install -g yarn@latest
		->npm -> node packaging manager
		->-g -> global installation
5. Install 'create-react-app' tool'
	- create-react-app tool given by facebook
	- this tool is used to create react applications.
		>npm install -g create-react-app@latest
		$ sudo npm install -g create-react-app@latest
		% sudo npm install -g create-react-app@latest
6. check react version
	>npm view react version

==========
Creating first react application
==========
1. Create a directory(folder)
	Demo	-> Drag n drop to VSCode



2. Create react application
	>create-react-app firstapp
		where	create-react-app tool to create react application
			firstapp is the name of react application
		-> enlist naming conventions for creating react application ?

3. Execution
	switch to application
		>cd firstapp
	execute
		>yarn start
		OR
		>npm start

Note:- by default react applications are running on port no 3000

Directory Structure of React Application

i) node_modules
 - this directory contains all libraries or modules
   required to execute react applications.

ii)public		/favicon.ico
			/logo192.png 192 x 192 px logo
			/logo512.png 512 x 512 px logo

iii)public	/index.html
		- react application starts execution from this file.

iv)public	/manifest.json
		/robots.txt
	- help to develop react native(mobile) applications

v) src
	- components are kept here

vi)src		/index.js
		/index.css
	- index.js is used to register first component
	- index.css in stylesheet for index.js (global styles here)

vii)src		/App.js
		/App.css
		/App.test.js
	- 'App' is the default component (i.e. .js)
	- 'App.css' is the stylesheet for the default component.
	- 'App.test.js' is the unit test case for the default component.

viii)package.json
		information about dependencies(downloaded libraries)

==========
Component:-

==========
- Reusable part of a complex UI is called a component.
- As a react developer we can create more than one component.
- React applications are component based applications.
- Components are kept in 'src' folder.

Type of Components
- Functional Components
- Class Components

Class Component
- Simple JSX class can behave like component.
- Class components are created by extending the Component class.
- Component class is available in React class.
- React class is available in the react package.
- 'render()' is the mandatory lifecycle method in the class component.

==========
First Component
==========

directory structure

```
<>
    src
        first
            - First.js
        - index.js
```

First.js

```
import React from 'react'
export default class First extends React.Component {
    render() {
        return (
            <div>
                <h1>Welcome to first Component </h1>
                <h3>React is Component Based</h3>
            </div>
        )
    }
}
```

index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import First from './first/First';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <First/>
);
```

```
// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

Multiple Components
Directory Structure
<>
src
components
- MEAN.js
- MERN.js
- MEVN.js
-
- index.js

MEAN.js

```
import React from "react";
export default class MEAN extends React.Component {
    render() {
        return (
            <div>
                <h1 style={{ color: 'red' }}>Welcome to MEAN stack </h1>
            </div>
        )
    }
}
```

Similarly create MERN and MEVN Components

FullStack.js

```
import React from "react";
import MEAN from "./MEAN";
import MERN from "./MERN";
import MEVN from "./MEVN";
export default class Fullstack extends React.Component {
    render() {
        return (
            <div>
                <MEAN />
                <MERN />
                <MEVN />
            </div>
        )
    }
}
```

index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
```

```
import First from './first/First';
import MEAN from './components/MEAN';
import MERN from './components/MERN';
import MEVN from './components/MEVN';
import Fullstack from './components/Fullstack';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
    /*
  <>
    <MEAN/>
    <MERN/>
    <MEVN/>
  </>
   */
  < Fullstack />
);

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

==========

State:-

==========

- State is used to store component data.
- {} used to display dynamic data.

Directory Structure

<>

src

Stateeg

- StateComponent.js

StateComponent.js

```
import React from 'react'
export default class StateComponent extends React.Component {
    constructor() {
        super()
        this.state = {
            data: `Data from db soon...!`,
            version: 18.3,
            flag: true,
            subs: ['ML', 'Maths', 'AI', 'IP', 'FSD'],
            obj: {
                fe: 'ReactJS',
                be: 'NodeJS',
                db: 'MongoDB'
            },
            products: [
                { "p_id": 111, "p_name": "P_one", "p_cost": 10000 },
                { "p_id": 222, "p_name": "P_two", "p_cost": 20000 },
                { "p_id": 333, "p_name": "P_three", "p_cost": 30000 },
```

```
                { "p_id": 444, "p_name": "P_four", "p_cost": 40000 },
                { "p_id": 555, "p_name": "P_five", "p_cost": 50000 }
            ]
        }
    }
    render() {
        return (
            <div>
                <p style={{ color: 'red' }}>{this.state.data} </p>
                <p style={{ color: 'blue' }}>{this.state.version}</p>
                <p style={{ color: 'green'
}}>{JSON.stringify(this.state.flag)}</p>
                <p style={{ color: 'maroon' }}>{this.state.subs.map((element,
index) => (
                    <>{element} </>
                ))} </p>
                <p style={{ color: 'magenta'
}}>{JSON.stringify(this.state.obj)} </p>
                <table border="1px"
                    cellSpacing="5px"
                    cellPadding="5px"
                    align="center">
                    <thead>
                        <tr>
                            <th>Srno</th>
                            <th>P_id</th>
                            <th>P_name</th>
                            <th>P_cost</th>
                        </tr>
                    </thead>
                    <tbody>
                        {
                            this.state.products.map((e, i) => (
                                <tr>
                                    <td>{i + 1} </td>
                                    <td>{e.p_id} </td>
                                    <td>{e.p_name} </td>
                                    <td>{e.p_cost} </td>
                                </tr>
                            ))
                        }
                    </tbody>
                </table>
            </div>
        )
    }
}
```

==========

Setting and Changing State

==========

```
import React from 'react'
export default class Statechange extends React.Component {
    constructor() {
        super()
```

```
        this.state = {
            count: 0
        }
    }
    render() {
        return (
            <div className='container mt-5'>
                <button className='btn btn-outline-success'
onClick={this.dec}> - </button>
                <button className='btn btn-secondary mx-3'>{this.state.count}
</button>
                <button className='btn btn-outline-success'
onClick={this.inc}> + </button>
            </div>
        )
    }
    inc = () => {
        this.setState({
            count: this.state.count
        })
        this.setState(prevState => {
            prevState.count += 1
        })
    }
    dec = () => {
        this.setState({
            count: this.state.count
        })
        this.setState(prevState => {
            prevState.count -= 1
        })
    }
}
```

==========

props:-

==========

- props are used to share data between components.

<>

src

propseg

- First.js
- Second.js

- index.js

First.js

```
import React from "react";
import Second from "./Second";
export default class First extends React.Component {
    constructor() {
        super()
        this.state = {
```

```
            tech: `Javascript`,
            number : 100,
            flag: true,
            obj: {
                fruit: 'Mango'
            },
            subs: [` Bootstrap `, ` Express `, ` React`],
            products: [
                { "p_id": 111, "p_name": "P_one", "p_cost": 10000 },
                { "p_id": 222, "p_name": "P_two", "p_cost": 20000 },
                { "p_id": 333, "p_name": "P_three", "p_cost": 30000 },
                { "p_id": 444, "p_name": "P_four", "p_cost": 40000 },
                { "p_id": 555, "p_name": "P_five", "p_cost": 50000 }
            ]
        }
    }
    render() {
        return (
            <div className="container mt-5">
                <h1>This is First Component</h1>
                <p>{this.state.tech} </p>
                <button className="btn btn-outline-primary"
onClick={this.chState}>Change State</button>
                <Second key1={this.state.tech}></Second>
                <Second data2 = {this.state.number}></Second>
                <Second data3={this.state.flag} />
                <Second data4={this.state.obj} />
                <Second data5={this.state.subs} />
                <Second data6={this.state.products} />
            </div>
        )
    }
    chState = () => {
        this.setState({
            tech: `ReactJS`
        })
    }
}
```

Second.js

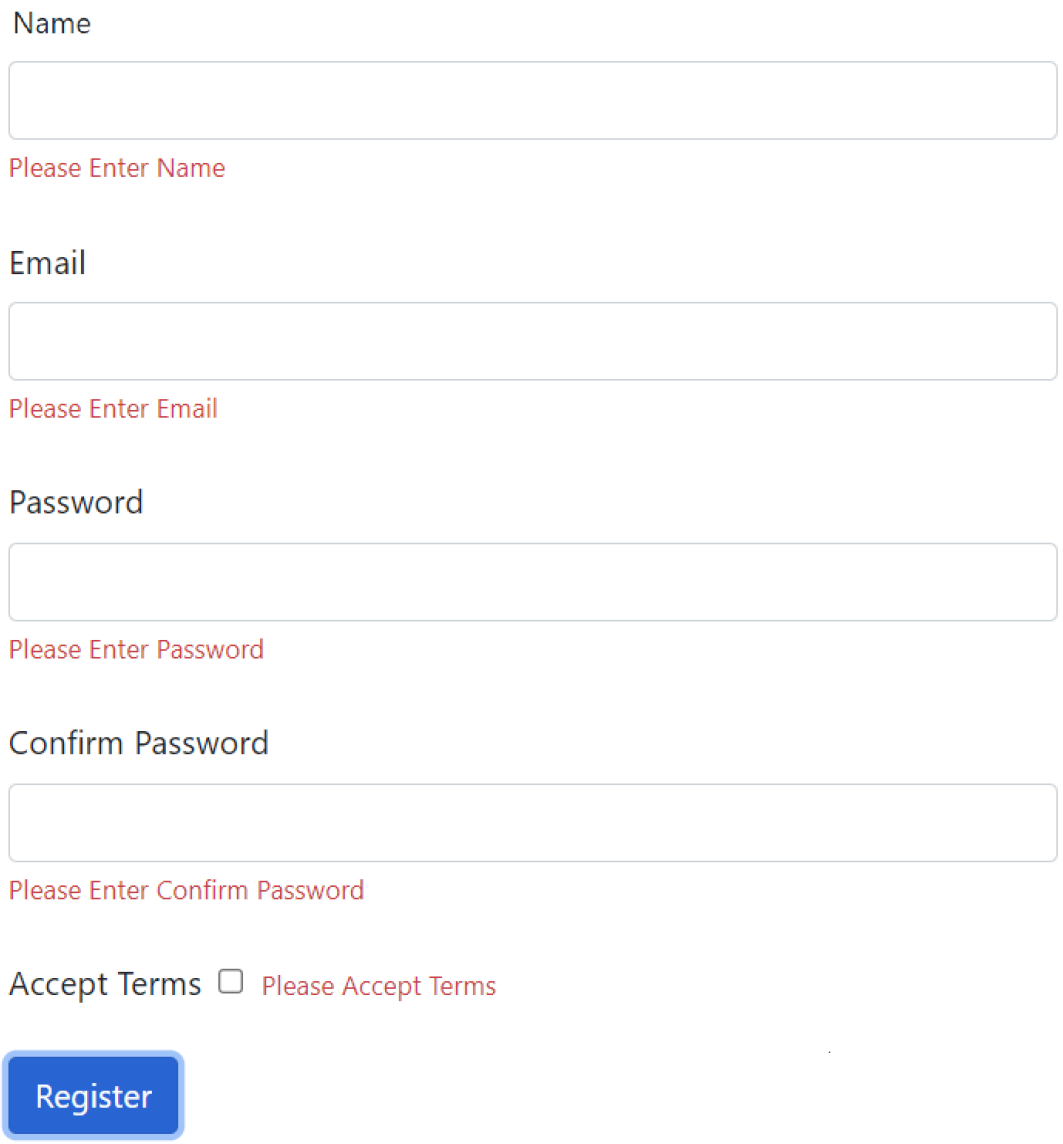
```
import React from "react";

export default class Second extends React.Component {
    render() {
        return (
            <div>
                <h1 style={{ color: 'red' }}>{this.props.key1} </h1>
                <h2 style={{ color: 'green' }}>{this.props.data2} </h2>
                <h2 style={{ color: 'blue'
}}>{JSON.stringify(this.props.data3)}</h2>
                <h2 style={{ color: 'maroon'
}}>{JSON.stringify(this.props.data4)}</h2>
                <h2 style={{ color: 'navy' }}>{this.props.data5}</h2>
                <h2 style={{ color: 'brown'
}}>{JSON.stringify(this.props.data6)} </h2>
```

```
            </div>
        )
    }
}
```

==========
Forms and Validations
==========
Formik
The most popular open-source form library for React and React Native.
A small library that helps us in:
 - Getting values in and out of form state
 - Validation and error messages
 - Handling form submission
There are four main components in Formik library: <Formik />, <Form />, <Field />, and <ErrorMessage />.

Form Validations
A process of checking the values in your form's input fields.
Conceptually, a validation rule needs the following things:
 - What field is being validated.
 - What function should be invoked to check if it's valid
 - What that function should return when the field is valid
 - What message should be displayed if the field isn't valid.
Yup is a library that is widely used to validate forms.

//>yarn add formik yup --save



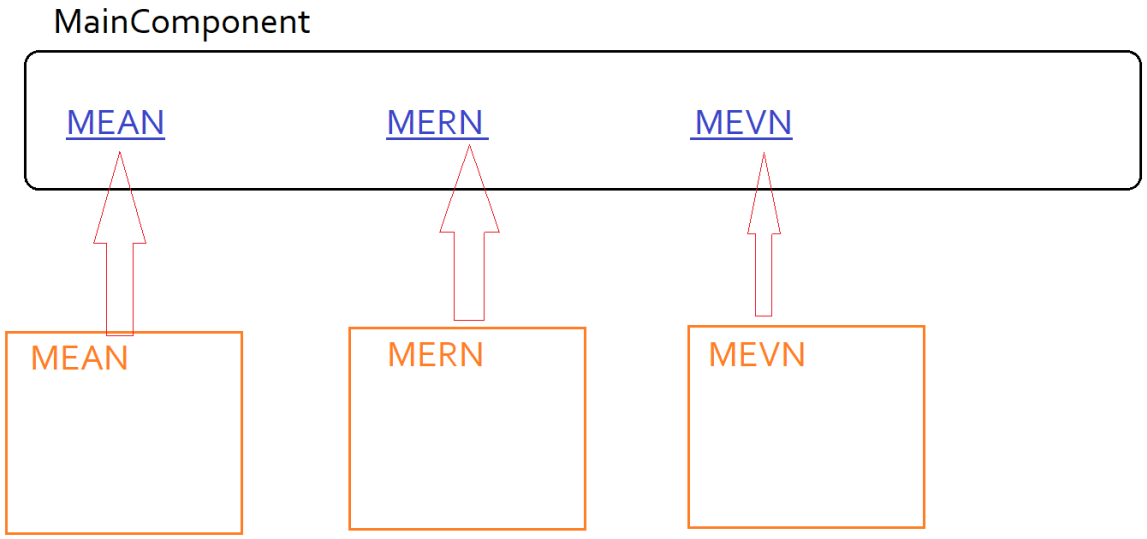

```
import React from 'react'
import * as Yup from 'yup'
import { Formik, Form, Field, ErrorMessage } from 'formik'
export default class MyFormik extends React.Component {
    constructor() {
        super()
        this.state = {
            user: {
                name: '',
                email: '',
                password: '',
                confirmPassword: '',
                terms: false
            },
            schema: Yup.object({
                name: Yup.string()
                    .required('Please Enter Name')
                    .min(3, 'Too Short')
```

```
                    .max(9, 'Too Large'),
                email: Yup.string().required('Please Enter email'),
                password: Yup.string()
                    .required('Please Enter Password')

.matches(/^(?=.*[a-z])(?=.*[A-Z])(?=.*[0-9])(?=.*[!@#%&])(?=.{8,})/,
                        "Must Contain 8 Characters, One Uppercase, One
Lowercase, One Number and One Special Case Character"),
                confirmPassword: Yup.string()
                    .required('Enter confirm password')
                    .oneOf([Yup.ref('password'), null], "Confirm password
dosent match"),
                terms: Yup.bool()
                    .required()
                    .oneOf([true], 'Please accept the terms')
            })
        }
    }
    render() {
        return (
            <div className='container mt-5'>
                <Formik initialValues={this.state.user}
                    validationSchema={this.state.schema}
                    onSubmit={(values) => {
                        console.log(values)
                    }}>
                    <Form>
                        <div className='my-3'>
                            <label>Name</label>
                            <Field name="name" type="text"
className="form-control"></Field>
                            <small><ErrorMessage name='name'
className='text-danger' component='label'></ErrorMessage></small>
                        </div>
                        <div className="my-3">
                            <label>Email</label>
                            <Field name="email" type="text"
className="form-control" />
                            <small>
                                <ErrorMessage component='label'
className="text-danger" name="email" />
                            </small>
                        </div>
                        <div className="my-3">
                            <label>Password</label>
                            <Field name="password" type="password"
className="form-control" />
                            <small>
                                <ErrorMessage component='label'
className="text-danger" name="password" />
                            </small>
                        </div>
                        <div className="my-3">
                            <label>Confirm Password</label>
                            <Field name="confirmPassword" type="password"
className="form-control" />
```

```
                            <small>
                                <ErrorMessage component='label'
className="text-danger" name="confirmPassword" />
                            </small>
                        </div>
                        <div className="my-3">
                            <label>Accept Terms</label>
                            <Field name="terms" type="checkbox"
className="m-3" />
                            <small>
                                <ErrorMessage component='label'
className="text-danger" name="terms" />
                            </small>
                        </div>
                        <div className="my-3">
                            <input type='submit' value='Register'
className='btn btn-primary' />
                        </div>
                    </Form>

                </Formik>
            </div>
        )
    }
}
```

==========
Single Page Application
==========

- Loading one component to another component without refreshing the whole webpage is called a single page application.
- Navigation of components in SPA is called routing.
- 'BrowserRouter' is used to perform navigation.
- 'Route' is used to render a particular component.
- All navigations are in 'Routes'.
- 'NavLink' defines a link to a particular component.
- BrowserRouter, Route, Routes and NavLink are available in 'react-router-dom'
- >yarn add react-router-dom --save



Dir Stru
<>
src
SPA
- Mean.js
- Mern.js
- Mevn.js
- MainComponent.js

Mean.js

```
import React from "react";
export default class MEAN extends React.Component {
    render() {
        return (
            <div>
                <h1 style={{ color: 'red' }}>Welcome to MEAN stack </h1>
            </div>
        )
    }
}
```

Similarly design Mern and Mevn Component

MainComponent.js

```
import React from "react";
import { NavLink, Route, BrowserRouter as Router, Routes } from
'react-router-dom'
import MEAN from "./MEAN";
import MERN from "./MERN";
import MEVN from "./MEVN";
export default class MainComponent extends React.Component {
    render() {
```

```
        return (
            <div>
                <Router>
                    <NavLink to="/mean" style={{ marginRight: '30px'
}}>MEAN</NavLink>
                    <NavLink to="/mern" style={{ marginRight: '30px'
}}>MERN</NavLink>
                    <NavLink to="/mevn" style={{ marginRight: '30px'
}}>MEVN</NavLink>
                    <Routes>
                        <Route path="/mean" element={<MEAN />}></Route>
                        <Route path="/mern" element={<MERN />}></Route>
                        <Route path="/mevn" element={<MEVN />}></Route>
                    </Routes>
                </Router>
            </div>
        )
    }
}
```

==========

Nested Routing

==========

Angular.js

```
import React from 'react'
export default class AngularComponent extends React.Component {
    render() {
        return (
            <div>
                <h2 style={{ color: 'maroon' }}>Angular </h2>
            </div>
        )
    }
}
```

similarly design ReactComponent and VueComponent

Mean.js

```
import React from "react";
import { NavLink } from "react-router-dom";
export default class MEAN extends React.Component {
    render() {
        return (
            <div>
                <h1 style={{ color: 'red' }}>Welcome to MEAN stack </h1>
                <NavLink to = "angular">Angular</NavLink>
            </div>
        )
    }
}
```

MainComponent.js

```
import React from "react";
import { NavLink, Route, BrowserRouter as Router, Routes } from
'react-router-dom'
```

```
import MEAN from "./MEAN";
import MERN from "./MERN";
import MEVN from "./MEVN";
import AngularComponent from "./Angular";
import ReactComponent from "./Reactc";
import VeuComponent from "./Veu";
export default class MainComponent extends React.Component {
    render() {
        return (
            <div>
                <Router>
                    <NavLink to="/mean" style={{ marginRight: '30px'
}}>MEAN</NavLink>
                    <NavLink to="/mern" style={{ marginRight: '30px'
}}>MERN</NavLink>
                    <NavLink to="/mevn" style={{ marginRight: '30px'
}}>MEVN</NavLink>
                    <Routes>
                        <Route path="/mean" element={<MEAN />}></Route>
                        <Route path="/mern" element={<MERN />}></Route>
                        <Route path="/mevn" element={<MEVN />}></Route>
                        <Route path="/mean/angular" element={<AngularComponent
/>}></Route>
                        <Route path="/mern/react" element={<ReactComponent
/>}></Route>
                        <Route path="/mevn/vue" element={<VeuComponent
/>}></Route>
                    </Routes>
                </Router>
            </div>
        )
    }
}
```

==========

Lazy Loading

==========

- Loading of Component on demand is called as Lazy Loading
- Loading time of application reduces.
- Performance of Application increases.
- Lazy Loading is used on slower internet.

1. Create LazyComponent
- 2. Import Suspense and lazy from react
- import React, { Suspense, lazy } from 'react';
- 3. In Main Component import Lazy Component
- //----------//
  const LazyComponent = lazy(() => { import('./LazyComponent')
  })
  //----------//
- 4. Provide Navigation Link
- {/*----------*/}
  <NavLink to="/myLazy" >Lazy</NavLink>
  {/*----------*/}

- 5. Provide Route
- {/*-------------------------*/}
<Route path='/myLazy' element={<Suspense fallback="Loading">
<LazyComponent />
</Suspense>} ></Route>
{/*-------------------------*/}

LazyComponent.js

```
import React from "react"
export default class LazyComponent extends React.Component {
    render() {
        return (
            <h1 style={{ background: 'red', color: 'white' }}>
                I am from Lazy Component
            </h1>
        )
    }
}
```

MainComponent.js

```
import React, { Suspense, lazy } from "react";
import { NavLink, Route, BrowserRouter as Router, Routes } from
'react-router-dom'
import MEAN from "./MEAN";
import MERN from "./MERN";
import MEVN from "./MEVN";
import AngularComponent from "./Angular";
import ReactComponent from "./Reactc";
import VeuComponent from "./Veu";
//-----------------------------------------------//
const LazyComponent = lazy(() => import('./LazyComponent'))
//-----------------------------------------------//

export default class MainComponent extends React.Component {
    render() {
        return (
            <div>
                <Router>
                    <NavLink to="/mean" style={{ marginRight: '30px'
}}>MEAN</NavLink>
                    <NavLink to="/mern" style={{ marginRight: '30px'
}}>MERN</NavLink>
                    <NavLink to="/mevn" style={{ marginRight: '30px'
}}>MEVN</NavLink>
                    {/*-------------------------*/}
                    <NavLink to="/myLazy" >Lazy</NavLink>
                    {/*-------------------------*/}
                    <Routes>
                        <Route path="/mean" element={<MEAN />}></Route>
                        <Route path="/mern" element={<MERN />}></Route>
                        <Route path="/mevn" element={<MEVN />}></Route>
                        {/*-------------------------*/}
```

```
                        <Route path='/myLazy' element={<Suspense
fallback="Loading">
                            <LazyComponent />
                        </Suspense>} ></Route>
                        {/*--------------------------*/}
                        <Route path="/mean/angular" element={<AngularComponent
/>}></Route>
                        <Route path="/mern/react" element={<ReactComponent
/>}></Route>
                        <Route path="/mevn/veu" element={<VeuComponent
/>}></Route>
                    </Routes>
                </Router>
            </div>
        )
    }
}
```

==========
Async Network calls APIs
==========
Download axios library
>yarn add axios --save

Assignment
1.
==========
CRUD Operations using SPA (Async Network calls APIs)
==========

src
CRUD
- IndexComponent.js Routing
- CreateComponent.js Insert record -> axios post
- ReadComponent.js Fetch records -> axios get
- UpdateComponent.js Update record -> axios post
- DeleteComponent.js Delete record -> axios post
- url.js api url

Note:- create all components skeleton and check for routing then go with api calls

ReadComponent.js

```
import React from 'react'
import axios from 'axios'
import url from './url'
export default class ReadComponent extends React.Component {
    constructor() {
        super()
        this.state = {
            products: [],
            status: ''
```

```
        }
    }
    componentDidMount() {
        this.setState({
            status: 'Loading'
        })
        axios.get(url + '/fetch')
            .then((posRes) => {
                this.setState({
                    products: posRes.data,
                    status: ''
                })
            }, (errRes) => {
                console.log(errRes)
            })
    }
    render() {
        return (
            <div className='container'>
                <div className='text-primary h1'>I am from Read
Component</div>
                <table className='table table-bordered table-warning
table-striped table-hover w-50 mx-auto'>
                    <thead>
                        <tr>
                            <th>Sr No</th>
                            <th>P_id</th>
                            <th>P_name</th>
                            <th>P_cost</th>
                        </tr>
                    </thead>
                    <tbody>
                        {
                            this.state.products.map((element, index) => (
                                <tr>
                                    <td>{index + 1} </td>
                                    <td>{element.p_id}</td>
                                    <td>{element.p_name}</td>
                                    <td>{element.p_cost}</td>
                                </tr>
                            ))
                        }
                    </tbody>
                </table>
                <h3 className='text-info'>{this.state.status} </h3>
            </div>
        )
    }
}
```

CreateComponent.js

```
import React from 'react'
import axios from 'axios'
import url from './url'
```

```
export default class CreateComponent extends React.Component {
    constructor() {
        super()
        this.state = {
            status: ""
        }
    }
    render() {
        return (
            <div className='container'>
                <div className='text-info h1'>I am from Create Component</div>
                <form onSubmit={this.insert} className='w-50 '>
                    <input type="number"
                        placeholder="P_id"
                        name="p_id"
                        className="form-control my-2"></input>
                    <input type="text"
                        placeholder="P_name"
                        name="p_name"
                        className="form-control my-2"></input>
                    <input type="number"
                        placeholder="P_cost"
                        name="p_cost"
                        className="form-control my-2"></input>
                    <input type='submit' value="Create" className="btn
btn-success"></input>
                </form>
                <h1 className='text-primary'>{this.state.status} </h1>
            </div>
        )
    }
    insert = (e) => {
        e.preventDefault()
        let obj = {
            "p_id": parseInt(e.target.p_id.value),
            "p_name": e.target.p_name.value,
            "p_cost": parseInt(e.target.p_cost.value)
        }
        axios.post(url + '/insert', obj)
            .then((posRes) => {
                this.setState({
                    status: posRes.data.insert
                })
            }, (errRes) => {
                console.log(errRes)
            })
    }
}
```

UpdateComponent.js

```
import axios from 'axios'
import React from 'react'
import url from './url'
export default class UpdateComponent extends React.Component {
    constructor() {
        super()
```

```
        this.state = {
            status: ""
        }
    }
    render() {
        return (
            <div className='container'>
                <div className='text-info h1'>I am from Update Component</div>
                <div>
                    <form onSubmit={this.update} className='w-50 '>
                        <input type="number"
                            placeholder="P_id"
                            name="p_id"
                            className="form-control my-2"></input>
                        <input type="text"
                            placeholder="P_name"
                            name="p_name"
                            className="form-control my-2"></input>
                        <input type="number"
                            placeholder="P_cost"
                            name="p_cost"
                            className="form-control my-2"></input>
                        <input type='submit' value="Update" className="btn
btn-success"></input>
                    </form>
                    <h1 className='text-primary'>{this.state.status} </h1>
                </div>
            </div>
        )
    }
    update = (e) => {
        e.preventDefault()
        this.setState({
            status: "Loading"
        })
        let obj = {
            "p_id": parseInt(e.target.p_id.value),
            "p_name": e.target.p_name.value,
            "p_cost": parseInt(e.target.p_cost.value)
        }
        axios.post(url + "/update", obj)
            .then((posRes) => {
                console.log(posRes)
                this.setState({
                    status: posRes.data.update
                })
            }, (errRes) => {
                console.log(errRes)
            })
    }
}
```

DeleteComponet.js

```
import axios from 'axios'
import React from 'react'
```

```
import url from './url'
export default class DeleteComponent extends React.Component {
    constructor() {
        super()
        this.state = {
            status: ""
        }
    }
    render() {
        return (
            <div className='container'>
                <div className='text-danger h1'>I am from Delete
Component</div>
                <div>
                    <form onSubmit={this.delete} className='w-50 '>
                        <input type="number"
                            placeholder="P_id"
                            name="p_id"
                            className="form-control my-2"></input>
                        <input type='submit' value="Delete" className="btn
btn-success"></input>
                    </form>
                    <h1 className='text-primary'>{this.state.status} </h1>
                </div>
            </div>
        )
    }
    delete = (e) => {
        e.preventDefault()
        this.setState({
            status: "Loading"
        })
        let obj = {
            "p_id": parseInt(e.target.p_id.value),
        }
        axios.post(url + "/delete", obj)
            .then((posRes) => {
                console.log(posRes)
                this.setState({
                    status: posRes.data.delete
                })
            }, (errRes) => {
                console.log(errRes)
            })
    }
}
```

. . .

2.
Display spinner in place of 'Loading message'

3.

From the capstone project backend fetch data populate in card layout MANDATORY