* Step 1 – Create React App
* Step 2 – Install validator and Bootstrap
* Step 3 – Create Form Validation Class
* Step 4 – Create Registration Form in App.js

npx create-react-app my-react-app

npm start

npm install bootstrap --save

npm install --save validator

//Add **bootstrap.min.css** file in src/App.js file:

import React, { Component } from 'react'

import '../node\_modules/bootstrap/dist/css/bootstrap.min.css';

function App() {

  return (

    <div>

      <h2>How to Add Custom Validatin with Forms in React</h2>

    </div>

  );

}

export default App;

import validator from 'validator';

class FormValidator {

    constructor(validations) {

        this.validations = validations;

    }

    validate(state) {

        let validation = this.valid();

        // for each validation rule

        this.validations.forEach(rule => {

            if(!validation[rule.field].isInvalid) {

                const field\_value = state[rule.field].toString();

                const args = rule.args || [];

                const validation\_method = typeof rule.method === 'string' ? validator[rule.method] : rule.method

                if(validation\_method(field\_value, ...args, state) !== rule.validWhen) {

                    validation[rule.field] = {

                        isInvalid: true,

                        message: rule.message

                    }

                    validation.isValid = false;

                }

            }

        });

        return validation;

    }

    valid() {

        const validation = {}

        this.validations.map(rule => (validation[rule.field] = {

            isInvalid: false,

            message: ''

        }));

        return {

            isValid: true,

            ...validation

        };

    }

}

export default FormValidator;

import React, {Component} from 'react';

import FormValidator from './FormValidator';

import './App.css';

class App extends Component{

constructor(){

super();

this.validator = new FormValidator([{

field: 'full\_name',

method: 'isEmpty',

validWhen: false,

message: 'Enter full name.'

}, {

field: 'email',

method: 'isEmpty',

validWhen: false,

message: 'Enter your email address.'

}, {

field: 'email',

method: 'isEmail',

validWhen: true,

message: 'Enter valid email address.'

}, {

field: 'phone',

method: 'isEmpty',

validWhen: false,

message: 'Enter a phone number.'

}, {

field: 'phone',

method: 'matches',

args: [/^\(?\d\d\d\)? ?\d\d\d-?\d\d\d\d$/],

validWhen: true,

message: 'Enter valid phone number.'

}, {

field: 'password',

method: 'isEmpty',

validWhen: false,

message: 'Enter password.'

}, {

field: 'password\_confirmation',

method: 'isEmpty',

validWhen: false,

message: 'Enter Password confirmation.'

}, {

field: 'password\_confirmation',

method: this.passwordMatch, // notice that we are passing a custom function here

validWhen: true,

message: 'Password and password confirmation do not match.'

}]);

this.state = {

full\_name: '',

email: '',

phone: '',

password: '',

password\_confirmation: '',

validation: this.validator.valid(),

}

this.submitted = false;

}

passwordMatch = (confirmation, state) => (state.password === confirmation)

handleInputChange = event => {

event.preventDefault();

this.setState({

[event.target.name]: event.target.value,

});

}

handleFormSubmit = event => {

event.preventDefault();

const validation = this.validator.validate(this.state);

this.setState({

validation

});

this.submitted = true;

if(validation.isValid) {

//reaches here if form validates successfully...

}

}

render() {

let validation = this.submitted ?this.validator.validate(this.state) : this.state.validation

return (

<div className="container">

<div className="row">

<div className="col-md-4 col-md-offset-4">

<form className="registrationForm">

<h2>Registration form validation react </h2>

<div className={validation.email.isInvalid && 'has-error'}>

<label htmlFor="full\_name">Full Name</label>

<input type="string" className="form-control" name="full\_name" placeholder="Full Name" onChange={this.handleInputChange} /> <span className="help-block">{validation.full\_name.message}</span> </div>

<div className={validation.email.isInvalid && 'has-error'}>

<label htmlFor="email">Email address</label>

<input type="email" className="form-control" name="email" placeholder="Email address" onChange={this.handleInputChange} /> <span className="help-block">{validation.email.message}</span> </div>

<div className={validation.phone.isInvalid && 'has-error'}>

<label htmlFor="phone">Phone(enter only 10 digit number)</label>

<input type="phone" className="form-control" name="phone" placeholder="Phone Number" onChange={this.handleInputChange} /> <span className="help-block">{validation.phone.message}</span> </div>

<div className={validation.password.isInvalid && 'has-error'}>

<label htmlFor="password">Password</label>

<input type="password" className="form-control" placeholder="Password" name="password" onChange={this.handleInputChange} /> <span className="help-block">{validation.password.message}</span> </div>

<div className={validation.password\_confirmation.isInvalid && 'has-error'}>

<label htmlFor="password\_confirmation">Confirm Password</label>

<input type="password" className="form-control" placeholder="Confirm Password" name="password\_confirmation" onChange={this.handleInputChange} /> <span className="help-block">{validation.password\_confirmation.message}</span> </div>

<button onClick={this.handleFormSubmit} className="btn btn-primary"> Register </button>

</form>

</div>

</div>

</div>

)

}

}

export default App;