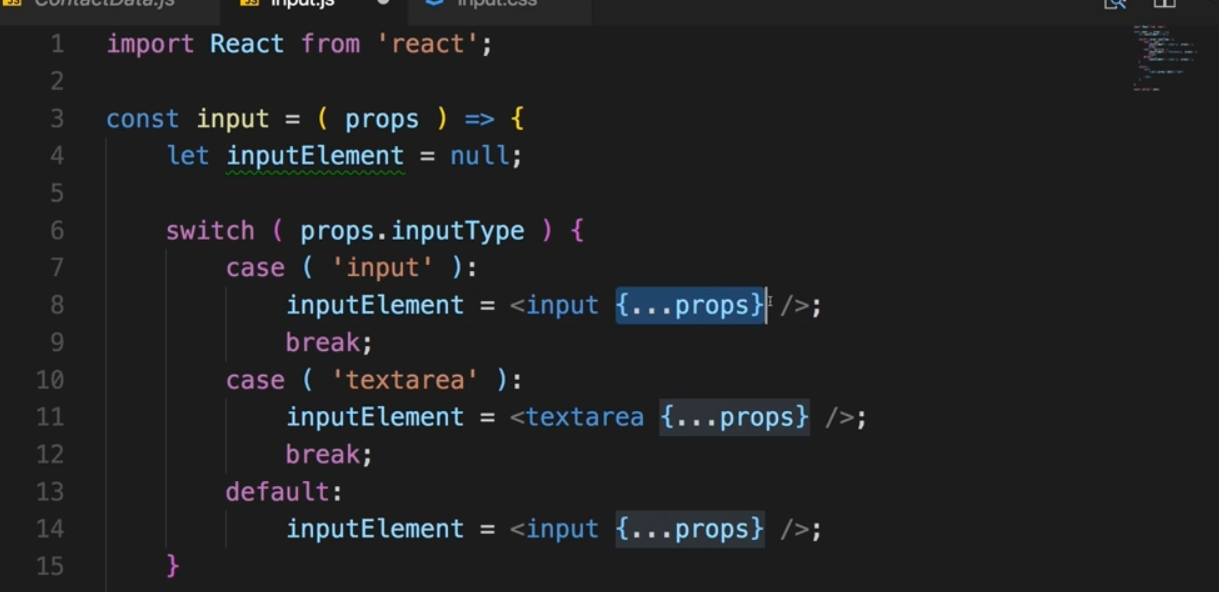
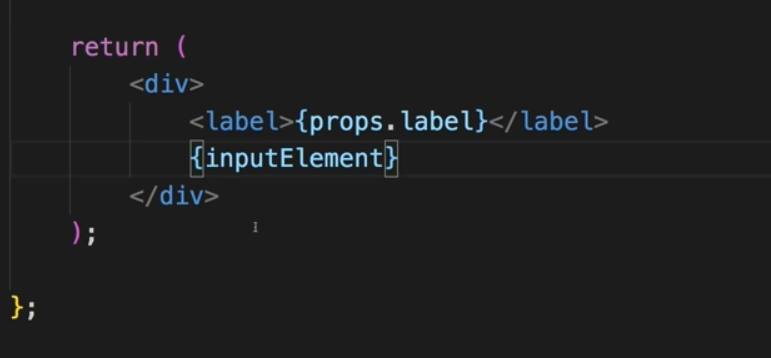
Tạo component input dùng switch, {…props} nhận các thông số cấu hình của input đó, như value,name,id…



Input thường đi kèm với label



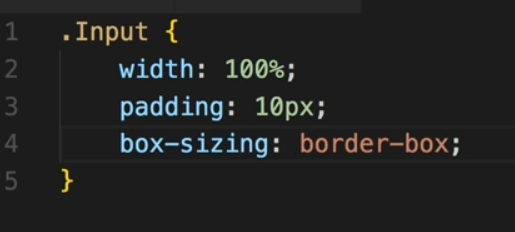
Style div chưa label và input: thường vật chứa sẽ được set

width: 100% chiếm hết ô màn hình

box-sizing: border-box: tính toán margin và padding chuẩn hơn

padding: khoảng cách giữa nội dung và border

margin: khoảng cách giữa border với các thành phần khác



Label là chữ sẽ được style như chữ

Font-size:

Font-weight:

Font-height:

Display:block; hiển thị hết một hàng

Xét margin với các đối tượng khác



Luôn nghĩ đến display đầu tiên khi hiển thị đối tượng đó trên hàng

Outline là lớp viền ngoài đối tượng

Border, background-color là các đặc tính tùy chọn luôn xuât hiện

Thêm box-sizing: border box vào để padding và margin đúng, khi dùng margin, padding thì phải có

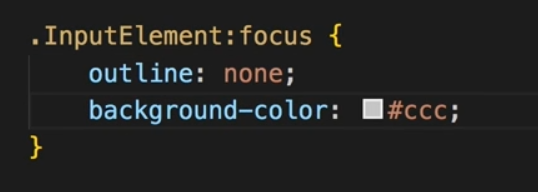
Font: inherit; kế thừa font của lớp cha

Padding margin tùy chọn



Đối tượng forcus được sử dụng khi nhấn con trỏ vào để nhập

Thay đổ background color để làm đối tượng khác đi



Thêm các class trên theo dạng import classes form ‘../’

Sử dụng state để lưu cho thuộc tính cho từng input

  state = {

        orderForm: {

            name: {

                elementType: 'input',

                elementConfig: {

                    type: 'text',

                    placeholder: 'Your Name'

                },

                value: ''

            },

            street: {

                elementType: 'input',

                elementConfig: {

                    type: 'text',

                    placeholder: 'Street'

                },

                value: ''

            },

            zipCode: {

                elementType: 'input',

                elementConfig: {

                    type: 'text',

                    placeholder: 'ZIP Code'

                },

                value: ''

            },

            country: {

                elementType: 'input',

                elementConfig: {

                    type: 'text',

                    placeholder: 'Country'

                },

                value: ''

            },

            email: {

                elementType: 'input',

                elementConfig: {

                    type: 'email',

                    placeholder: 'Your E-Mail'

                },

                value: ''

            },

            deliveryMethod: {

                elementType: 'select',

                elementConfig: {

                    options: [

                        {value: 'fastest', displayValue: 'Fastest'},

                        {value: 'cheapest', displayValue: 'Cheapest'}

                    ]

                },

                value: ''

            }

        },

        loading: false

    }

Các thuộc tính được đưa vào trong form component dùng prop

  <Input

                        key={formElement.id}

                        elementType={formElement.config.elementType}

                        elementConfig={formElement.config.elementConfig}

                        value={formElement.config.value}

                        changed={(event) => this.inputChangedHandler(event, formElement.id)} />

                ))}

   switch ( props.elementType ) {

        case ( 'input' ):

            inputElement = <input

                className={classes.InputElement}

                {...props.elementConfig}

                value={props.value}

                onChange={props.changed} />;

            break;

        case ( 'textarea' ):

            inputElement = <textarea

                className={classes.InputElement}

                {...props.elementConfig}

                value={props.value}

                onChange={props.changed} />;

            break;

        case ( 'select' ):

            inputElement = (

                <select

                    className={classes.InputElement}

                    value={props.value}

                    onChange={props.changed}>

                    {props.elementConfig.options.map(option => (

                        <option key={option.value} value={option.value}>

                            {option.displayValue}

                        </option>

                    ))}

                </select>

            );

            break;

        default:

            inputElement = <input

                className={classes.InputElement}

                {...props.elementConfig}

                value={props.value}

                onChange={props.changed} />;

    }

Với state chưa input như trên, tạo ra mãng chứa các input trên, mỗi phần tử mãng chưa id là tên của input và config chứa object chưa các thông số config

 const formElementsArray = [];

        for (let key in this.state.orderForm) {

            formElementsArray.push({

                id: key,

                config: this.state.orderForm[key]

            });

        }

Dùng map sinh ra input từ mãng này

{formElementsArray.map(formElement => (

                    <Input

                        key={formElement.id}

                        elementType={formElement.config.elementType}

                        elementConfig={formElement.config.elementConfig}

                        value={formElement.config.value}

                        changed={(event) => this.inputChangedHandler(event, formElement.id)} />

                ))}

Hàm inputChangedHandler được sinh dùng

inputChangedHandler = (event, inputIdentifier) => {

        const updatedOrderForm = {

            ...this.state.orderForm

        };

        const updatedFormElement = {

            ...updatedOrderForm[inputIdentifier]

        };

        updatedFormElement.value = event.target.value;

        updatedOrderForm[inputIdentifier] = updatedFormElement;

        this.setState({orderForm: updatedOrderForm});

    }

Đối với select element

Config sẽ chưa mãng chứa các select

Mỗi slect là một object chứa value và display value

deliveryMethod: {

                elementType: 'select',

                elementConfig: {

                    options: [

                        {value: 'fastest', displayValue: 'Fastest'},

                        {value: 'cheapest', displayValue: 'Cheapest'}

                    ]

                },

                value: ''

            }

Select chưa classname và onChange như những đối tượng khác

Phần option dùng map loop qua mãng được tạo ở trên

   case ( 'select' ):

            inputElement = (

                <select

                    className={classes.InputElement}

                    value={props.value}

                    onChange={props.changed}>

                    {props.elementConfig.options.map(option => (

                        <option key={option.value} value={option.value}>

                            {option.displayValue}

                        </option>

                    ))}

                </select>

            );

            break;

Validation:

Thêm ở mỗi input validation chứa các yêu cầu valid

Và valid return xem trường đó valid hay không

String.trym() : remove whitespace from both sides of a string:

state = {

        orderForm: {

            name: {

                elementType: 'input',

                elementConfig: {

                    type: 'text',

                    placeholder: 'Your Name'

                },

                value: '',

                validation: {

                    required: true

                },

                valid: false,

                touched: false

            },

            street: {

                elementType: 'input',

                elementConfig: {

                    type: 'text',

                    placeholder: 'Street'

                },

                value: '',

                validation: {

                    required: true

                },

                valid: false,

                touched: false

            },

            zipCode: {

                elementType: 'input',

                elementConfig: {

                    type: 'text',

                    placeholder: 'ZIP Code'

                },

                value: '',

                validation: {

                    required: true,

                    minLength: 5,

                    maxLength: 5

                },

                valid: false,

                touched: false

            },

            country: {

                elementType: 'input',

                elementConfig: {

                    type: 'text',

                    placeholder: 'Country'

                },

                value: '',

                validation: {

                    required: true

                },

                valid: false,

                touched: false

            },

            email: {

                elementType: 'input',

                elementConfig: {

                    type: 'email',

                    placeholder: 'Your E-Mail'

                },

                value: '',

                validation: {

                    required: true

                },

                valid: false,

                touched: false

            },

            deliveryMethod: {

                elementType: 'select',

                elementConfig: {

                    options: [

                        {value: 'fastest', displayValue: 'Fastest'},

                        {value: 'cheapest', displayValue: 'Cheapest'}

                    ]

                },

                value: '',

                valid: true

            }

        },

        formIsValid: false,

        loading: false

    }

Hàm check valid return về giá trị valid

Để giá trị ban đầu là true, các điều kiện sau check có điều kiện không, sau đó isValid sẽ check thõa mãn điều kiện, và && với isValid => nếu 1 cái false thì dẫn đến false

 checkValidity(value, rules) {

        let isValid = true;

        if (rules.required) {

            isValid = value.trim() !== '' && isValid;

        }

        if (rules.minLength) {

            isValid = value.length >= rules.minLength && isValid

        }

        if (rules.maxLength) {

            isValid = value.length <= rules.maxLength && isValid

        }

        return isValid;

    }

Dùng hàm này trong input, vì khi nhập sẽ check luôn

inputChangedHandler = (event, inputIdentifier) => {

        const updatedOrderForm = {

            ...this.state.orderForm

        };

        const updatedFormElement = {

            ...updatedOrderForm[inputIdentifier]

        };

        updatedFormElement.value = event.target.value;

        updatedFormElement.valid = this.checkValidity(updatedFormElement.value, updatedFormElement.validation);

        updatedFormElement.touched = true;

        updatedOrderForm[inputIdentifier] = updatedFormElement;

        let formIsValid = true;

        for (let inputIdentifier in updatedOrderForm) {

            formIsValid = updatedOrderForm[inputIdentifier].valid && formIsValid;

        }

        this.setState({orderForm: updatedOrderForm, formIsValid: formIsValid});

    }

Feedback to user

Trên input componet

Tạo biến chứa class của các input dưới dạng array

Invalid là biến thể hiện field đó valid hay ko

shouldValidate là biến trỏ về validation của component cha, nghĩa là khi filed đó ko cần validate thì sholdvalidate sẽ false

touched là một biến trong state của mỗi input(xem trên state), touched chuyển sang true mỗi khi điền vào filed, thêm touched vào điều kiện, thì khi người dùng chưa thay đổi field thì ko bị add class invalid

const input = ( props ) => {

    let inputElement = null;

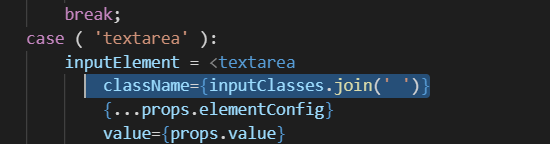
    const inputClasses = [classes.InputElement];

    if (props.invalid && props.shouldValidate && props.touched) {

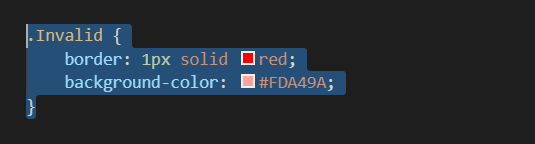
        inputClasses.push(classes.Invalid);

    }

Trong các case, như vậy input sẽ được có thêm class invalid nếu điều kiện invalid xảy ra



File css của input component



Trên contact form là class cha, state ở trên

Invalid là biến điều kiện để thêm class invalid vào class của component input

for (let key in this.state.orderForm) {

            formElementsArray.push({

                id: key,

                config: this.state.orderForm[key]

            });

        }

        let form = (

            <form onSubmit={this.orderHandler}>

                {formElementsArray.map(formElement => (

                    <Input

                        key={formElement.id}

                        elementType={formElement.config.elementType}

                        elementConfig={formElement.config.elementConfig}

                        value={formElement.config.value}

                        invalid={!formElement.config.valid}

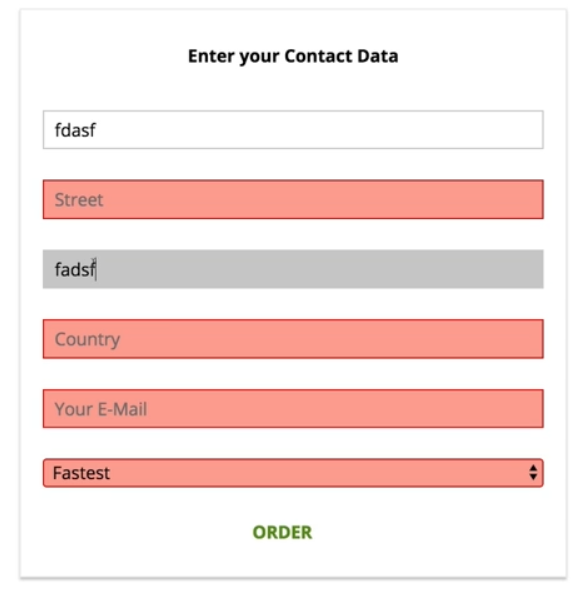
                        shouldValidate={formElement.config.validation}

                        touched={formElement.config.touched}

                        changed={(event) => this.inputChangedHandler(event, formElement.id)} />

                ))}

Nhứng field invalid sẽ hiện màu đỏ



Thông báo khi submid form

Biến formIsvalid: loop hết state nếu valid của thằng nào false thì biến này false

 inputChangedHandler = (event, inputIdentifier) => {

        const updatedOrderForm = {

            ...this.state.orderForm

        };

        const updatedFormElement = {

            ...updatedOrderForm[inputIdentifier]

        };

        updatedFormElement.value = event.target.value;

        updatedFormElement.valid = this.checkValidity(updatedFormElement.value, updatedFormElement.validation);

        updatedFormElement.touched = true;

        updatedOrderForm[inputIdentifier] = updatedFormElement;

        let formIsValid = true;

        for (let inputIdentifier in updatedOrderForm) {

            formIsValid = updatedOrderForm[inputIdentifier].valid && formIsValid;

        }

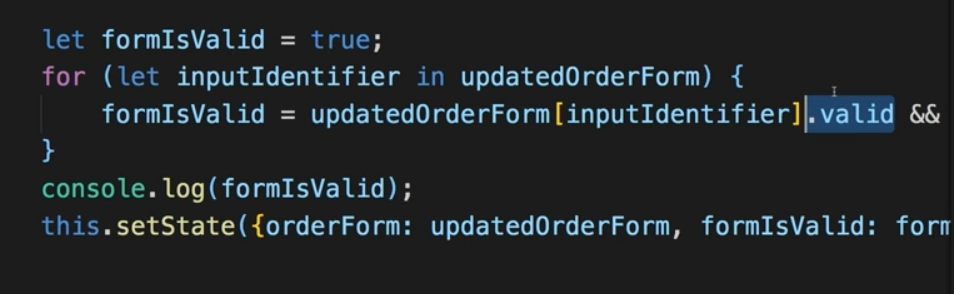
        this.setState({orderForm: updatedOrderForm, formIsValid: formIsValid});

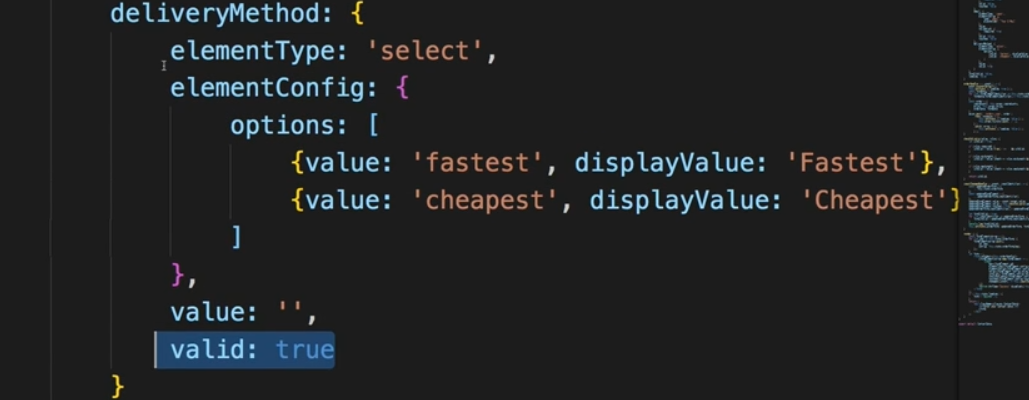
    }

Nút order sẽ bị disable khi formIsValid false

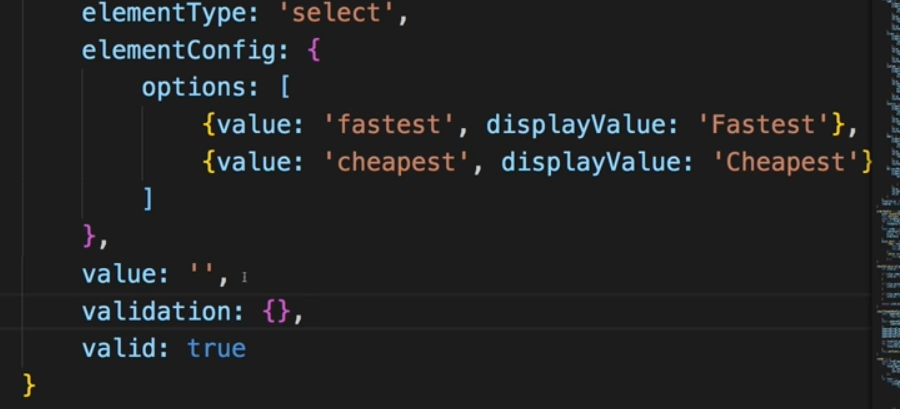


Khi loop qua những biến ko có valid vì ko cần sẽ có giá trị valid underfined là sai kết quả, nên thêm valid cho những giá trị này

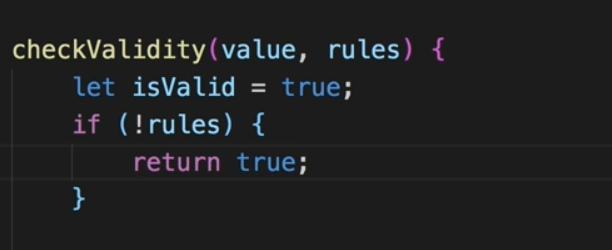




Với select ko có validation nhưng cũng phải thêm vào, nếu ko thêm hàm valid sẽ lỗi vì ko có validation khi loop đến select



Hoặc có thể kẹp điều kiện



button

import React from 'react';

import classes from './Button.css';

const button = (props) => (

    <button

        disabled={props.disabled}

        className={[classes.Button, classes[props.btnType]].join(' ')}

        onClick={props.clicked}>{props.children}</button>

);

export default button;

.Button {

    background-color: transparent;

    border: none;

    color: white;

    outline: none;

    cursor: pointer;

    font: inherit;

    padding: 10px;

    margin: 10px;

    font-weight: bold;

}

.Button:first-of-type {

    margin-left: 0;

    padding-left: 0;

}

.Button:disabled {

    color: #ccc;

    cursor: not-allowed;

}

.Success {

    color: #5C9210;

}

.Danger {

    color: #944317;

}