Prop\_Types

Ví dụ 3:

Code:

import React, { useState, useReducer } from "react";

import { Button, Form, Container, Alert } from "react-bootstrap";

import PropTypes from "prop-types";

const initialState = {

  name: "",

  email: "",

  password: "",

  isSubmitted: false,

};

const formReducer = (state, action) => {

  switch (action.type) {

    case "SET\_FIELD":

      return { ...state, [action.field]: action.value };

    case "SUBMIT":

      return { ...state, isSubmitted: true };

    default:

      return state;

  }

};

const MyForm = ({ title, onSubmit }) => {

  const [state, dispatch] = useReducer(formReducer, initialState);

  const [errors, setErrors] = useState({});

  const [showAlert, setShowAlert] = useState(false);

  const handleChange = (e) => {

    const { name, value } = e.target;

    dispatch({ type: "SET\_FIELD", field: name, value });

  };

  const handleValidation = () => {

    const newErrors = {};

    if (!state.name) newErrors.name = "Tên không được để trống!";

    if (!state.email) newErrors.email = "Email không được để trống!";

    if (!state.password) newErrors.password = "Mật khẩu không được để trống!";

    setErrors(newErrors);

    setShowAlert(Object.keys(newErrors).length > 0);

    return Object.keys(newErrors).length === 0;

  };

  const handleSubmit = (e) => {

    e.preventDefault();

    if (handleValidation()) {

      dispatch({ type: "SUBMIT" });

      onSubmit(state);

    }

  };

  return (

    <Container>

      <h3>{title}</h3>

      {showAlert && <Alert variant="danger">Vui lòng điền đầy đủ thông tin.</Alert>}

      <Form onSubmit={handleSubmit}>

        <Form.Group>

          <Form.Label>Tên</Form.Label>

          <Form.Control

            name="name"

            value={state.name}

            onChange={handleChange}

            isInvalid={!!errors.name}

          />

          <Form.Control.Feedback type="invalid">{errors.name}</Form.Control.Feedback>

        </Form.Group>

        <Form.Group>

          <Form.Label>Email</Form.Label>

          <Form.Control

            name="email"

            value={state.email}

            onChange={handleChange}

            isInvalid={!!errors.email}

          />

          <Form.Control.Feedback type="invalid">{errors.email}</Form.Control.Feedback>

        </Form.Group>

        <Form.Group>

          <Form.Label>Mật khẩu</Form.Label>

          <Form.Control

            type="password"

            name="password"

            value={state.password}

            onChange={handleChange}

            isInvalid={!!errors.password}

          />

          <Form.Control.Feedback type="invalid">{errors.password}</Form.Control.Feedback>

        </Form.Group>

        <Button type="submit">Submit</Button>

      </Form>

    </Container>

  );

};

MyForm.propTypes = {

  title: PropTypes.string.isRequired,

  onSubmit: PropTypes.func.isRequired,

};

export default MyForm;

import logo from './logo.svg';

import './App.css';

import 'bootstrap/dist/css/bootstrap.min.css';

import MyForm from './vd3';

function App() {

  const handleFormSubmit = (formData) => {

    console.log("Dữ liệu đăng ký:", formData);

  };

  return (

    <div className="App">

      <h1>Ví dụ 3: Đăng ký sử dụng useReducer + validation</h1>

      <MyForm title="Đăng Ký Người Dùng" onSubmit={handleFormSubmit} />

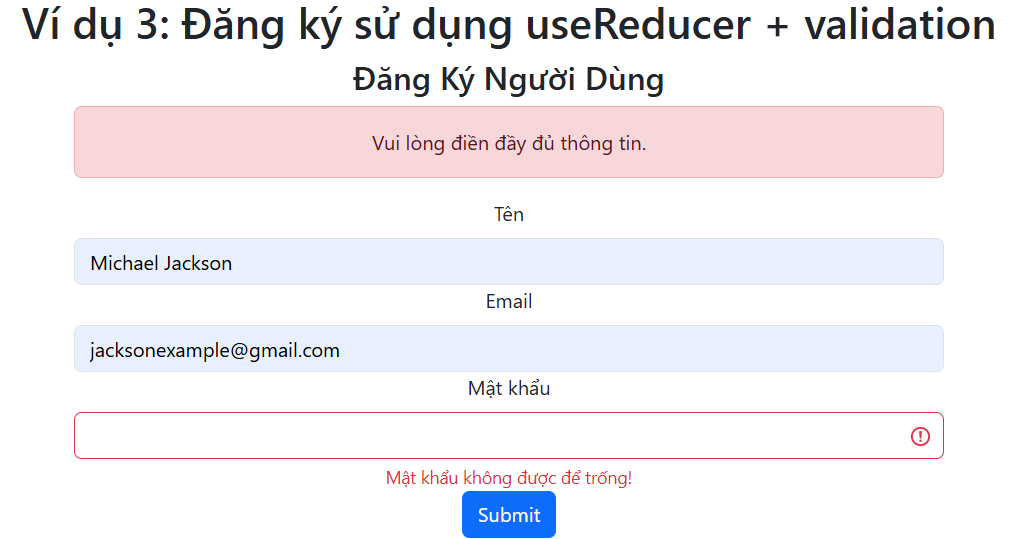
    </div>

  );

}

export default App;

Kết quả:



Ví dụ 4:

Code:

// components/UserValidationForm.js

import React, { useState } from "react";

import { Form, Button, Container, Alert } from "react-bootstrap";

import PropTypes from "prop-types";

const UserValidationForm = ({ onSubmit }) => {

  const [formData, setFormData] = useState({

    name: "",

    age: "",

    email: "",

    phone: "",

    agreeToTerms: false,

  });

  const [errors, setErrors] = useState({});

  const [showAlert, setShowAlert] = useState(false);

  const handleChange = (e) => {

    const { name, value, type, checked } = e.target;

    setFormData({ ...formData, [name]: type === "checkbox" ? checked : value });

  };

  const validate = () => {

    const newErrors = {};

    const age = parseInt(formData.age, 10);

    const phoneRegex = /^\d{10,15}$/;

    const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

    if (!formData.name || formData.name.length < 3 || formData.name.length > 50)

      newErrors.name = "Tên phải từ 3 đến 50 ký tự.";

    if (!formData.age || isNaN(age) || age < 18 || age > 100)

      newErrors.age = "Tuổi phải từ 18 đến 100.";

    if (!formData.email || !emailRegex.test(formData.email))

      newErrors.email = "Email không hợp lệ.";

    if (!phoneRegex.test(formData.phone))

      newErrors.phone = "Số điện thoại phải từ 10-15 chữ số.";

    if (!formData.agreeToTerms)

      newErrors.agreeToTerms = "Bạn phải đồng ý với điều khoản.";

    setErrors(newErrors);

    setShowAlert(Object.keys(newErrors).length > 0);

    return Object.keys(newErrors).length === 0;

  };

  const handleSubmit = (e) => {

    e.preventDefault();

    if (validate()) {

      setShowAlert(false);

      onSubmit(formData);

    }

  };

  return (

    <Container>

      <h3>Form Xác Thực Người Dùng</h3>

      {showAlert && <Alert variant="danger">Vui lòng kiểm tra lại thông tin!</Alert>}

      <Form onSubmit={handleSubmit}>

        <Form.Group>

          <Form.Label>Tên</Form.Label>

          <Form.Control

            name="name"

            value={formData.name}

            onChange={handleChange}

            isInvalid={!!errors.name}

          />

          <Form.Control.Feedback type="invalid">{errors.name}</Form.Control.Feedback>

        </Form.Group>

        <Form.Group>

          <Form.Label>Tuổi</Form.Label>

          <Form.Control

            name="age"

            value={formData.age}

            onChange={handleChange}

            isInvalid={!!errors.age}

          />

          <Form.Control.Feedback type="invalid">{errors.age}</Form.Control.Feedback>

        </Form.Group>

        <Form.Group>

          <Form.Label>Email</Form.Label>

          <Form.Control

            name="email"

            value={formData.email}

            onChange={handleChange}

            isInvalid={!!errors.email}

          />

          <Form.Control.Feedback type="invalid">{errors.email}</Form.Control.Feedback>

        </Form.Group>

        <Form.Group>

          <Form.Label>Số điện thoại</Form.Label>

          <Form.Control

            name="phone"

            value={formData.phone}

            onChange={handleChange}

            isInvalid={!!errors.phone}

          />

          <Form.Control.Feedback type="invalid">{errors.phone}</Form.Control.Feedback>

        </Form.Group>

        <Form.Group>

          <Form.Check

            type="checkbox"

            name="agreeToTerms"

            checked={formData.agreeToTerms}

            onChange={handleChange}

            label="Tôi đồng ý với điều khoản"

            isInvalid={!!errors.agreeToTerms}

          />

          <Form.Control.Feedback type="invalid">

            {errors.agreeToTerms}

          </Form.Control.Feedback>

        </Form.Group>

        <Button type="submit">Gửi</Button>

      </Form>

    </Container>

  );

};

UserValidationForm.propTypes = {

  onSubmit: PropTypes.func.isRequired,

};

export default UserValidationForm;

import logo from './logo.svg';

import './App.css';

import 'bootstrap/dist/css/bootstrap.min.css';

import MyForm from './vd3';

import UserValidationForm from './vd4';

function App() {

  //Ví dụ 4

  const handleFormSubmit = (formData) => {

    console.log("Dữ liệu gửi:", formData);

  };

  return (

    <div className="App">

      <h1>Ví dụ 4: Form validate nâng cao</h1>

      <UserValidationForm onSubmit={handleFormSubmit} />

    </div>

  );

}

export default App;

Kết quả:

