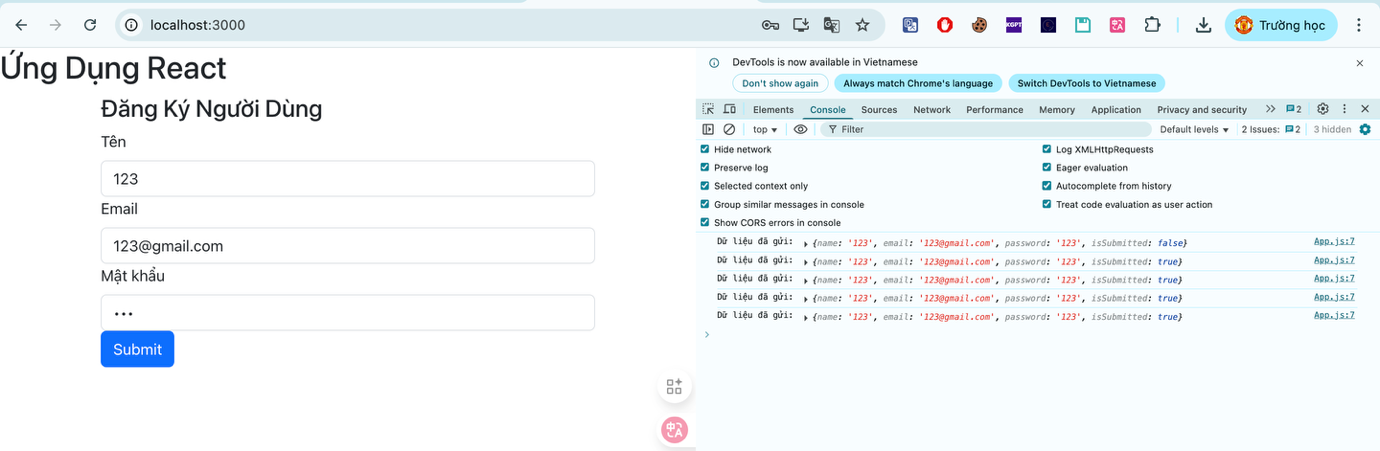
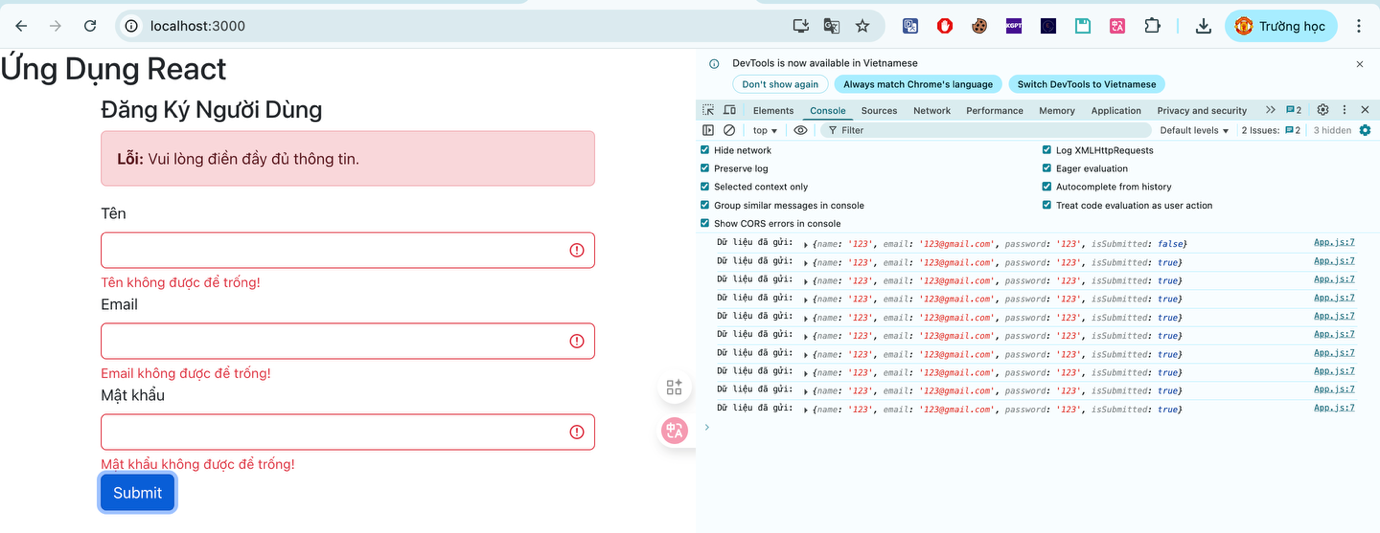
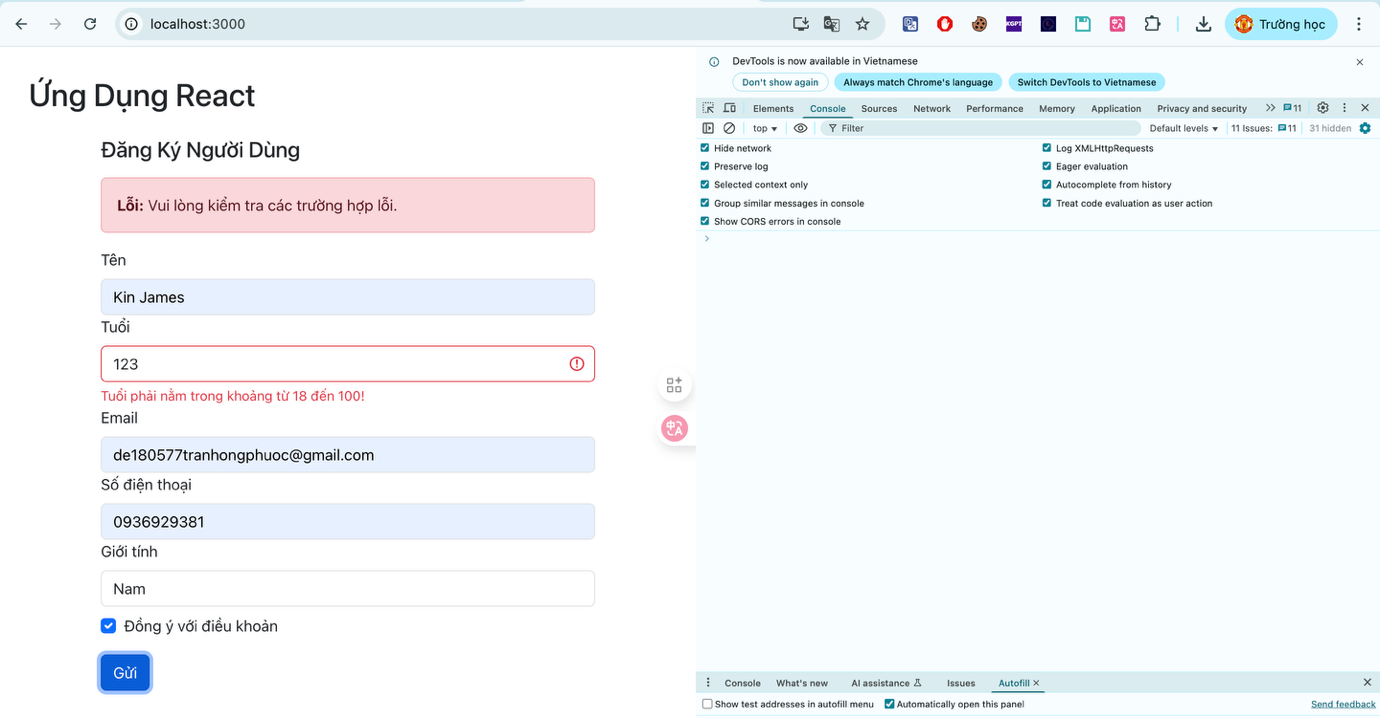
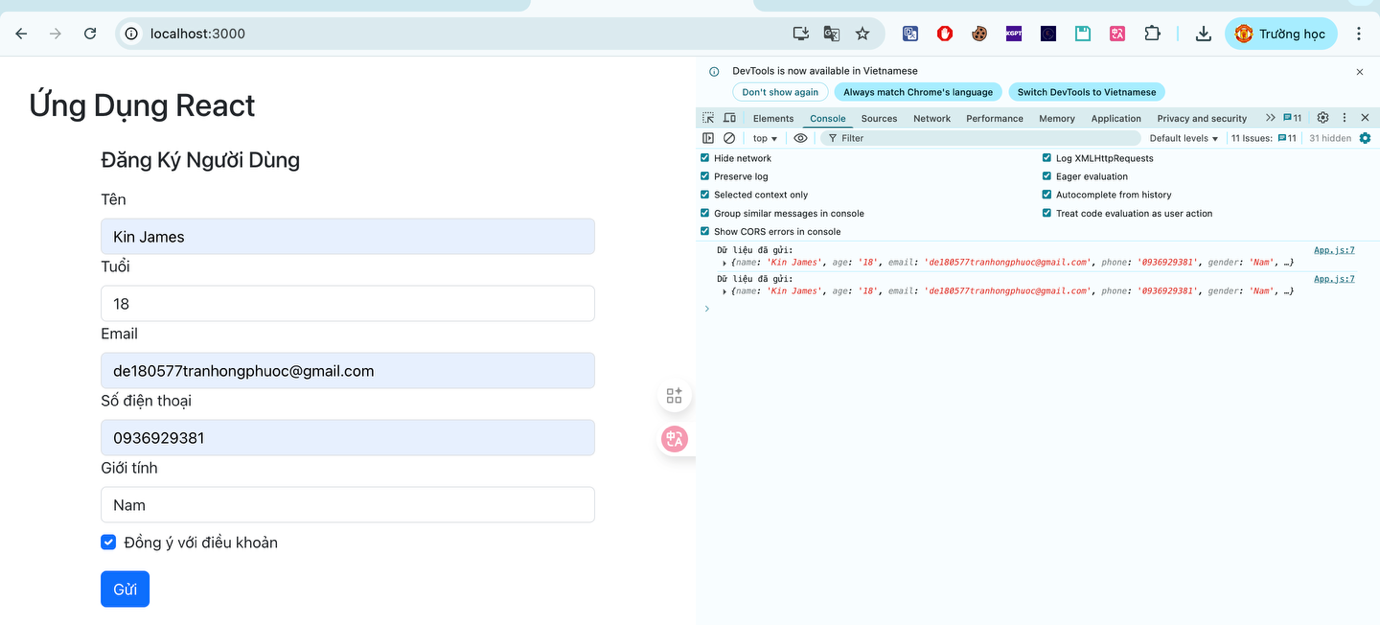
Ví dụ 3: 



import *React*, { *useState*, *useReducer* } from "react";  
import { *Button*, Form, *Container*, Alert } from "react-bootstrap";  
import *PropTypes* from "prop-types";  
  
*// Reducer để quản lý trạng thái form*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;  
 }  
};  
*// Component Form*const *MyForm* = ({ title, onSubmit }) => {  
 const [state, dispatch] = *useReducer*(formReducer, initialState);  
 const [errors, setErrors] = *useState*({});  
 const [showAlert, setShowAlert] = *useState*(false); *// Biến để kiểm soát việc hiển thị alert  
  
 // Hàm xử lý thay đổi giá trị input* const handleChange = (e) => {  
 const { name, value } = e.target;  
 dispatch({ type: "SET\_FIELD", field: name, value });  
 };  
  
 *// Hàm kiểm tra lỗi trước khi submit* 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!";  
  
 *// Nếu có lỗi, hiển thị alert* if (Object.keys(newErrors).length > 0) {  
 setShowAlert(true);  
 } else {  
 setShowAlert(false);  
 }  
  
 setErrors(newErrors);  
 return Object.keys(newErrors).length === 0;  
 };  
  
 const handleSubmit = (e) => {  
 e.preventDefault();  
 if (handleValidation()) {  
 dispatch({ type: "SUBMIT" });  
 onSubmit(state);  
 }  
 };  
  
 return (  
 <Container>  
 <h3>{title}</h3>  
  
 {*/\* Hiển thị Alert nếu có lỗi \*/*}  
 {showAlert && (  
 <Alert variant="danger">  
 <strong>Lỗi:</strong> Vui lòng điền đầy đủ thông tin.  
 </Alert>  
 )}  
  
 <Form onSubmit={handleSubmit}>  
 <Form.Group controlId="formName">  
 <Form.Label>Tên</Form.Label>  
 <Form.Control  
 type="text"  
 name="name"  
 value={state.name}  
 onChange={handleChange}  
 isInvalid={!!errors.name}  
 />  
 <Form.Control.Feedback type="invalid">  
 {errors.name}  
 </Form.Control.Feedback>  
 </Form.Group>  
  
 <Form.Group controlId="formEmail">  
 <Form.Label>Email</Form.Label>  
 <Form.Control  
 type="email"  
 name="email"  
 value={state.email}  
 onChange={handleChange}  
 isInvalid={!!errors.email}  
 />  
 <Form.Control.Feedback type="invalid">  
 {errors.email}  
 </Form.Control.Feedback>  
 </Form.Group>  
  
 <Form.Group controlId="formPassword">  
 <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 variant="primary" type="submit">  
 Submit  
 </Button>  
 </Form>  
 </Container>  
 );  
};  
*// Xác định PropTypes cho MyForm  
MyForm*.propTypes = {  
 title: *PropTypes*.*string*.isRequired, *// Tiêu đề phải là một chuỗi* onSubmit: *PropTypes*.*func*.isRequired, *// Hàm onSubmit phải là một function*};  
export default *MyForm*;

Ví dụ 4:





import *React*, { *useReducer*, *useState* } from "react";  
import { *Button*, Form, *Container*, Alert } from "react-bootstrap";  
import *PropTypes* from "prop-types";  
  
*// Trạng thái ban đầu*const initialState = {  
 name: "",  
 age: "",  
 email: "",  
 phone: "",  
 gender: "",  
 agreed: false,  
 isSubmitted: false,  
};  
  
*// Reducer*const formReducer = (state, action) => {  
 switch (action.type) {  
 case "SET\_FIELD":  
 return { ...state, [action.field]: action.value };  
 case "TOGGLE\_AGREE":  
 return { ...state, agreed: !state.agreed };  
 case "SUBMIT":  
 return { ...state, isSubmitted: true };  
 default:  
 return state;  
 }  
};  
  
const *MyFormExample4* = ({ title, onSubmit }) => {  
 const [state, dispatch] = *useReducer*(formReducer, initialState);  
 const [errors, setErrors] = *useState*({});  
 const [showAlert, setShowAlert] = *useState*(false);  
  
 const handleChange = (e) => {  
 const { name, value, type, checked } = e.target;  
 const fieldValue = type === "checkbox" ? checked : value;  
 dispatch({ type: "SET\_FIELD", field: name, value: fieldValue });  
 };  
  
 const validateEmail = (email) => /^[^\s@]+@[^\s@]+\.[^\s@]+$/.test(email);  
 const validatePhone = (phone) => /^\d{10,15}$/.test(phone);  
  
 const handleValidation = () => {  
 const newErrors = {};  
  
 if (!state.name.trim()) {  
 newErrors.name = "Tên không được để trống!";  
 } else if (state.name.length < 3 || state.name.length > 50) {  
 newErrors.name = "Tên phải có từ 3 đến 50 ký tự!";  
 }  
  
 const age = *parseInt*(state.age);  
 if (!state.age) {  
 newErrors.age = "Tuổi không được để trống!";  
 } else if (*isNaN*(age) || age < 18 || age > 100) {  
 newErrors.age = "Tuổi phải nằm trong khoảng từ 18 đến 100!";  
 }  
  
 if (!state.email.trim()) {  
 newErrors.email = "Email không được để trống!";  
 } else if (!validateEmail(state.email)) {  
 newErrors.email = "Email không đúng định dạng!";  
 }  
  
 if (!state.phone) {  
 newErrors.phone = "Số điện thoại không được để trống!";  
 } else if (!validatePhone(state.phone)) {  
 newErrors.phone = "Số điện thoại phải từ 10 đến 15 chữ số!";  
 }  
  
 if (!state.agreed) {  
 newErrors.agreed = "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 (handleValidation()) {  
 dispatch({ type: "SUBMIT" });  
 onSubmit(state);  
 }  
 };  
  
 return (  
 <Container>  
 <h4 className="mb-3">{title}</h4>  
  
 {showAlert && (  
 <Alert variant="danger">  
 <strong>Lỗi:</strong> Vui lòng kiểm tra các trường hợp lỗi.  
 </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>Tuổi</Form.Label>  
 <Form.Control  
 name="age"  
 type="number"  
 value={state.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"  
 type="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>Số điện thoại</Form.Label>  
 <Form.Control  
 name="phone"  
 value={state.phone}  
 onChange={handleChange}  
 isInvalid={!!errors.phone}  
 />  
 <Form.Control.Feedback type="invalid">  
 {errors.phone}  
 </Form.Control.Feedback>  
 </Form.Group>  
  
 <Form.Group>  
 <Form.Label>Giới tính</Form.Label>  
 <Form.Control  
 as="select"  
 name="gender"  
 value={state.gender}  
 onChange={handleChange}  
 >  
 <option value="">-- Chọn giới tính --</option>  
 <option value="Nam">Nam</option>  
 <option value="Nữ">Nữ</option>  
 <option value="Khác">Khác</option>  
 </Form.Control>  
 </Form.Group>  
  
 <Form.Group className="mt-2">  
 <Form.Check  
 type="checkbox"  
 label="Đồng ý với điều khoản"  
 name="agreed"  
 checked={state.agreed}  
 onChange={() => dispatch({ type: "TOGGLE\_AGREE" })}  
 isInvalid={!!errors.agreed}  
 />  
 {errors.agreed && (  
 <div className="text-danger" style={{ fontSize: "0.875em" }}>  
 {errors.agreed}  
 </div>  
 )}  
 </Form.Group>  
  
 <Button type="submit" variant="primary" className="mt-3">  
 Gửi  
 </Button>  
 </Form>  
 </Container>  
 );  
};  
  
*MyFormExample4*.propTypes = {  
 title: *PropTypes*.*string*.isRequired,  
 onSubmit: *PropTypes*.*func*.isRequired,  
};  
  
export default *MyFormExample4*;

import *React* from "react";  
import *MyFormExample4* from "./components/MyFormExample4";  
import 'bootstrap/dist/css/bootstrap.min.css';  
  
const *App* = () => {  
 const handleFormSubmit = (formData) => {  
 console.log("Dữ liệu đã gửi:", formData);  
 };  
  
 return (  
 <div className="App" style={{ padding: "30px" }}>  
 <h1 className="mb-4">Ứng Dụng React</h1>  
 <MyFormExample4 title="Đăng Ký Người Dùng" onSubmit={handleFormSubmit} />  
 </div>  
 );  
};  
  
export default *App*;