import React, { Component } from 'react';

import {

    Card,

    CardImg,

    CardText,

    CardBody,

    Breadcrumb,

    BreadcrumbItem,

    Button,

    Modal,

    ModalHeader,

    ModalBody,

    Label

} from 'reactstrap';

import { Link } from 'react-router-dom';

import { Control, LocalForm, Errors } from 'react-redux-form';

import { Loading } from './LoadingComponent';

import { baseUrl } from '../shared/baseUrl';

const required = (val) => val && val.length;

const maxLength = (len) => (val) => !val || val.length <= len;

const minLength = (len) => (val) => val && val.length >= len;

function RenderCampsite({ campsite }) {

    return (

        <div className="col-md-5 m-1">

            <Card>

                <CardImg top src={baseUrl + campsite.image} alt={campsite.name} />

                <CardBody>

                    <CardText>{campsite.description}</CardText>

                </CardBody>

            </Card>

        </div>

    );

}

function RenderComments({ comments, postComment, campsiteId }) {

    if (comments) {

        return (

            <div className="col-5-md m-1">

                <h4>Comments</h4>

                {comments.map((comment) => (

                    <div key={comment.id}>

                        <p>{comment.text}</p>

                        <p>

                            -- {comment.author},{' '}

                            {new Intl.DateTimeFormat('en-US', {

                                year: 'numeric',

                                month: 'short',

                                day: '2-digit'

                            }).format(new Date(Date.parse(comment.date)))}

                        </p>

                    </div>

                ))}

                <CommentForm campsiteId={campsiteId} postComment={postComment} />

                <div />

            </div>

        );

    }

}

class CommentForm extends Component {

    constructor(props) {

        super(props);

        this.state = {};

        this.toggleModal = this.toggleModal.bind(this);

        this.handleSubmit = this.handleSubmit.bind(this);

    }

    handleSubmit(values) {

        this.toggleModal();

        this.props.postComment(this.props.campsiteId, values.rating, values.author, values.text);

    }

    toggleModal() {

        this.setState({

            isModalOpen: !this.state.isModalOpen

        });

    }

    render() {

        return (

            <React.Fragment>

                <Button type="submit" outline color="secondary" onClick={this.toggleModal}>

                    <i className="fa fa-pencil" /> Submit Comment

                </Button>

                <Modal isOpen={this.state.isModalOpen} toggle={this.toggleModal}>

                    <ModalHeader toggle={this.toggleModal}>Submit Comment</ModalHeader>

                    <ModalBody>

                        <LocalForm onSubmit={(event) => this.handleSubmit(event)}>

                            <Label htmlFor="rating">Rating</Label>

                            <Control.select

                                model=".rating"

                                id="rating"

                                name="rating"

                                className="form-control"

                                validators={{ required }}

                            >

                                <option>Select</option>

                                <option>1</option>

                                <option>2</option>

                                <option>3</option>

                                <option>4</option>

                                <option>5</option>

                            </Control.select>

                            <Errors

                                className="text-danger"

                                model=".rating"

                                show="touched"

                                component="div"

                                messages={{

                                    required: 'Required'

                                }}

                            />

                            <Label htmlFor="author">Your Name</Label>

                            <Control.text

                                model=".author"

                                id="author"

                                name="author"

                                className="form-control"

                                placeholder="Your Name"

                                validators={{

                                    required,

                                    minLength: minLength(2),

                                    maxLength: maxLength(15)

                                }}

                            />

                            <Errors

                                className="text-danger"

                                model=".author"

                                show="touched"

                                component="div"

                                messages={{

                                    required: 'Required',

                                    minLength: 'Must be at least 2 characters',

                                    maxLength: 'Must be 15 characters or less'

                                }}

                            />

                            <Label htmlFor="text">Comment</Label>

                            <Control.textarea model=".text" id="text" name="text" className="form-control" rows="6" />

                            <div className="form-group" md={{ size: 10, offset: 2 }}>

                                <Button type="submit" color="primary">

                                    Submit

                                </Button>

                            </div>

                        </LocalForm>{' '}

                    </ModalBody>{' '}

                </Modal>

            </React.Fragment>

        );

    }

}

function CampsiteInfo(props) {

    if (props.isLoading) {

        return (

            <div className="container">

                <div className="row">

                    <Loading />

                </div>

            </div>

        );

    }

    if (props.errMess) {

        return (

            <div className="container">

                <div className="row">

                    <div className="col">

                        <h4>{props.errMess}</h4>

                    </div>

                </div>

            </div>

        );

    }

    if (props.campsite) {

        return (

            <div className="container">

                <div className="row">

                    <div className="col">

                        <Breadcrumb>

                            <BreadcrumbItem>

                                <Link to="/directory">Directory</Link>

                            </BreadcrumbItem>

                            <BreadcrumbItem active>{props.campsite.name}</BreadcrumbItem>

                        </Breadcrumb>

                        <h2>{props.campsite.name}</h2>

                        <hr />

                    </div>

                </div>

                <div className="row">

                    <RenderCampsite campsite={props.campsite} />

                    <RenderComments

                        comments={props.comments}

                        postComment={props.postComment}

                        campsiteId={props.campsite.id}

                    />

                </div>

            </div>

        );

    }

    return <div />;

}

export default CampsiteInfo;