REDUX

Škola za sve koji nikad nisu ni čuli za React

Author: Ivica Kartelo

Published: 2021

Level: From Total Beginners to Advanced Developers

Ovo je sažetak videa koji se tiču Redux-a, 40, 44, 46, 51, 54, 58 na YouTube: Škola React
https://www.youtube.com/playlist?list=PLvYzjHQxr_s3cjOVuE2EU94FUkSEJtcea

Uvod u Redux

Video 40.

Škola React 40. dio: Spajanje Reduxa i Reacta

Commit: Intro. To Redux

GitHub:
https://github.com/ivicakartelo/reactblog/commit/922a18b1c5549cba6d471669aa747a3e7cff61cc

YouTube:
https://www.youtube.com/watch?v=KWMCihwmI1Q&list=PLvYzjHQxr_s3cjOVuE2EU94FUkSEJtcea&index=28&t=11s

```
yarn add redux@3.7.2
yarn add react-redux@5.0.7
```

redux/reducer.js

```
import { MENUS } from '../shared/menus';

import { COMMENTS } from '../shared/comments';

import { PROMOTIONS } from '../shared/promotions';

import { AUTHORS } from '../shared/authors';


export const initialState = {

   menus: MENUS,

   comments: COMMENTS,

   promotions: PROMOTIONS,

   authors: AUTHORS

};


export const Reducer = (state = initialState, action) => {
```

```
    return state;
};
```

Redux/configureStore.js

```
import {createStore} from 'redux';
import { Reducer, initialState } from './reducer'

export const ConfigureStore = () => {
    const store = createStore(
        Reducer, // reducer
        initialState, // our initialState
    );

    return store;
}
```

App.js

```
import React, { Component } from 'react';
import Main from './components/MainComponent';
import './App.css';
import { BrowserRouter } from 'react-router-dom';
import { Provider } from 'react-redux';
import { ConfigureStore } from './redux/configureStore';

const store = ConfigureStore();

class App extends Component {
```

```
  render() {
    return (
      <Provider store={store}>
      <BrowserRouter>
        <div className="App">
          <Main />
        </div>
      </BrowserRouter>
    </Provider>
    );
  }
}
export default App;
```

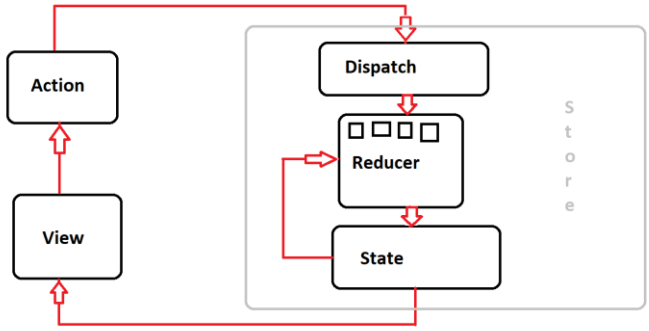
MainComponent.js

```
import React, { Component } from 'react';
import Home from './HomeComponent';
import Menu from './MenuComponent';
import Menudetail from './MenudetailComponent';
import Header from './HeaderComponent';
import Footer from './FooterComponent';
import Contact from './ContactComponent';
import About from './AboutComponent';
import { Switch, Route, Redirect, withRouter } from 'react-router-dom'
import { connect } from 'react-redux';
```

```
const mapStateToProps = state => {
  return {
    menus: state.menus,
    comments: state.comments,
    promotions: state.promotions,
    authors: state.authors
  }
}

class Main extends Component {

  constructor(props) {
    super(props);
  }

  render() {
    const HomePage = () => {
      return(
        <Home
          menu={this.props.menus.filter((menu) => menu.featured)[0]}
          promotion={this.props.promotions.filter((promo) =>
promo.featured)[0]}
          author={this.props.authors.filter((author) => author.featured)[0]}
        />
      );
    }
```

```
  const MenuWithId = ({match}) => {
    return(
      <Menudetail menu={this.props.menus.filter((menu) => menu.menu_id
=== parseInt(match.params.menuId,10))[0]}
        comments={this.props.comments.filter((comment) =>
comment.menuId === parseInt(match.params.menuId,10))} />
        );
  };
  return (
    <div>
      <Header />
      <Switch>
          <Route path='/home' component={HomePage} />
          <Route exact path='/menu' component={() => <Menu
menus={this.props.menus} />} />
          <Route path='/menu/:menuId' component={MenuWithId} />
          <Route exact path='/contactus' component={Contact} />
          <Route exact path='/aboutus' component={() => <About
authors={this.props.authors} />} />
          <Redirect to="/home" />
        </Switch>
      <Footer />
    </div>
  );
 }
}

export default withRouter(connect(mapStateToProps)(Main));
```

Redux protok podataka



Video 44.

Commit: combiningReducers

Škola React 44. dio: Rastavljanje Reducera u više Reducera i sastavljanje ponovo u jedan Reducer

https://www.youtube.com/watch?v=awFTRvurEIA&list=PLvYzjHQxr_s3cjOVuE2EU94FUkSEJtcea&index=24

GitHub:
https://github.com/ivicakartelo/reactblog/commit/ad7ac60f559ab6d03d601f6d1ac0013759141bea

redux/menus.js

```
import { MENUS } from '../shared/menus';


export const Menus = (state = MENUS, action) => {
   switch (action.type) {
      default:
        return state;
    }
};
```

redux/comments.js

```
import { COMMENTS } from '../shared/comments';


export const Comments = (state = COMMENTS, action) => {
   switch (action.type) {


      default:
        return state;
    }
};
```

redux/promotions.js

```
import { PROMOTIONS } from '../shared/promotions';


export const Promotions = (state = PROMOTIONS, action) => {
   switch (action.type) {
      default:
        return state;
    }
};
```

redux/authors.js

```
import { AUTHORS } from '../shared/authors';


export const Authors = (state = AUTHORS, action) => {
   switch (action.type) {
      default:
        return state;
    }
};
```

configureStore.js

```
import {createStore, combineReducers} from 'redux';
import { Menus } from './menus';
import { Comments } from './comments';
import { Promotions } from './promotions';
import { Authors } from './authors';


export const ConfigureStore = () => {
   const store = createStore(
```

```
        combineReducers({
            menus: Menus,
            comments: Comments,
            promotions: Promotions,
            authors: Authors
        })
    );

    return store;
}
```

Video 46

Škola React 46. dio: Redux Action

YouTube:
https://www.youtube.com/watch?v=Gvk4Pvoym7M&list=PLvYzjHQxr_s3cjOVuE2EU94FUkSEJtcea&index=22&t=336s

GitHub Commit: Redux Action:
https://github.com/ivicakartelo/reactblog/commit/014d3d04d80e01c88b6f0699f2c258dd5668de1b

ActionTypes.js

```
export const ADD_COMMENT = 'ADD_COMMENT';
```

ActionCreators.js

```
import * as ActionTypes from './ActionTypes';


export const addComment = (menuId, rating, author, comment) => ({


  type: ActionTypes.ADD_COMMENT,

  payload: {

    menuId: menuId,

    rating: rating,

    author: author,

    comment: comment

  }

});
```

redux/comments.js

```
import { COMMENTS } from '../shared/comments';

import * as ActionTypes from './ActionTypes';


export const Comments = (state = COMMENTS, action) => {
```

```
    switch (action.type) {
        case ActionTypes.ADD_COMMENT:
            var comment = action.payload;
            comment.id = state.length;
            comment.date = new Date().toISOString();
            console.log("Comment: ", comment);
            return state.concat(comment);

        default:
          return state;
      }
};
```

MainComponents.js

```
import React, { Component } from 'react';
import Home from './HomeComponent';
import Menu from './MenuComponent';
import Menudetail from './MenudetailComponent';
import Header from './HeaderComponent';
import Footer from './FooterComponent';
import Contact from './ContactComponent';
import About from './AboutComponent';
import { Switch, Route, Redirect, withRouter } from 'react-router-dom'
import { connect } from 'react-redux';
import { addComment } from '../redux/ActionCreators';
```

```
const mapStateToProps = state => {
  return {
    menus: state.menus,
    comments: state.comments,
    promotions: state.promotions,
    authors: state.authors
  }
}

const mapDispatchToProps = dispatch => ({
  addComment: (menuId, rating, author, comment) =>
dispatch(addComment(menuId, rating, author, comment))
});

class Main extends Component {

  constructor(props) {
    super(props);
  }

  render() {
    const HomePage = () => {
      return(
        <Home
          menu={this.props.menus.filter((menu) => menu.featured)[0]}
          promotion={this.props.promotions.filter((promo) =>
promo.featured)[0]}
```

```
          author={this.props.authors.filter((author) => author.featured)[0]}
        />
     );
   }
   const MenuWithId = ({match}) => {
    return(
       <Menudetail menu={this.props.menus.filter((menu) => menu.menu_id
=== parseInt(match.params.menuId,10))[0]}
          comments={this.props.comments.filter((comment) =>
comment.menuId === parseInt(match.params.menuId,10))}
          addComment={this.props.addComment} />
          );
   };
   return (
    <div>
      <Header />
      <Switch>
           <Route path='/home' component={HomePage} />
           <Route exact path='/menu' component={() => <Menu
menus={this.props.menus} />} />
           <Route path='/menu/:menuId' component={MenuWithId} />
           <Route exact path='/contactus' component={Contact} />
           <Route exact path='/aboutus' component={() => <About
authors={this.props.authors} />} />
           <Redirect to="/home" />
        </Switch>
      <Footer />
    </div>
```

```
    );
  }
}
```

export default withRouter(connect(mapStateToProps, mapDispatchToProps)(Main));

MenudetailComponent.js

```
import React from 'react';
import { Breadcrumb, BreadcrumbItem, Modal, ModalHeader, ModalBody,
  Button, Row, Col, Label, CardImg, Card, CardBody, CardTitle, CardText } from 'reactstrap';
import { Link } from 'react-router-dom';
import { Control, LocalForm, Errors } from 'react-redux-form';

    function RenderMenudetails({x}) {
    return (
      <Card>
        <CardImg top src={x.image} alt={x.name} />
        <CardBody>
          <CardTitle>{x.name}</CardTitle>
          <CardText>{x.content}</CardText>
        </CardBody>
      </Card>
    );
  }
```

function RenderComments({comments, addComment, menuId}) {

```
  const cmt = comments.map((w) => {
    return (
      <li key={w.id}>
        <p>{w.rating}</p>
        <p>{w.comment}</p>
        <p>
          -- {w.author},  
        {new Intl.DateTimeFormat('en-US', {
          year: 'numeric',
          month: 'long',
          day: '2-digit'
        }).format(new Date(w.date))}
        </p>
      </li>
    )
  })
  return(
    <div>
      <h5>Comments</h5>
        <ul className="list-unstyled">
          {cmt}
        </ul>
        <CommentForm menuId={menuId} addComment={addComment} />
    </div>
  );
}
```

```
  const Menudetail = (props) => {

    if (props.menu != null) {
      return (

        <div className="container">
          <div className="row">
            <Breadcrumb>
              <BreadcrumbItem><Link
to="/menu">Menu</Link></BreadcrumbItem>
              <BreadcrumbItem active>{props.menu.name}</BreadcrumbItem>
            </Breadcrumb>
            <div className="col-12">
            < h3>{props.menu.name}</h3>
              <hr />
            </div>
          </div>
            <div className="row">
              <div className="col-12 col-md">
                <RenderMenudetails x = {props.menu} />
              </div>
              <div className="col-12 col-md">
                <RenderComments comments = {props.comments}
                addComment={props.addComment}
                menuId={props.menu.menu_id} />
```

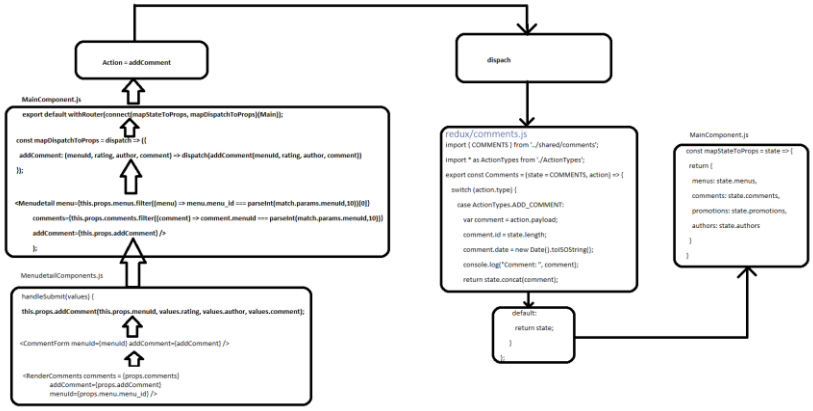
```
            </div>
          </div>
        </div>
      );
    }
    else {
      return (
        <div></div>
      );
    }
}

const maxLength = len => val => !val || val.length <= len;
const minLength = len => val => val && val.length >= len;

    class CommentForm extends React.Component {
      constructor(props) {
        super(props);
        this.state = {
          isModalOpen: false
        };
        this.toggleModal = this.toggleModal.bind(this);
        this.handleSubmit = this.handleSubmit.bind(this);
      }
```

```
toggleModal() {
  this.setState({
    isModalOpen: !this.state.isModalOpen
  });
}

handleSubmit(values) {
this.toggleModal();
  this.props.addComment(this.props.menuId, values.rating,
values.author, values.comment);
}

render() {
  return (
    <div>
      <Button outline onClick={this.toggleModal}>
        <span className="fa fa-pencil"></span> Submit Comment
      </Button>

      <Modal isOpen={this.state.isModalOpen} toggle={this.toggleModal}>
        <ModalHeader toggle={this.toggleModal}>Submit
Comment</ModalHeader>
        <ModalBody>
          <LocalForm onSubmit={values => this.handleSubmit(values)}>
          <Row className="form-group">
                    <Label htmlFor="rating" md={2}>Rating</Label>
```

```
                        <Col md={10}>
                          <Control.select model=".rating" id="rating"
name="rating"
                            className="form-control"
                            defaultValue="1">
                            <option value="1">1</option>
                            <option value="2">2</option>
                            <option value="3">3</option>
                            <option value="4">4</option>
                            <option value="5">5</option>
                          </Control.select>
                        </Col>
                      </Row>
                    <Row className="form-group">
                     <Label htmlFor="author" md={2}>
                      Your Name
                     </Label>
                     <Col md={10}>
                      <Control.text
                       model=".author"
                       id="author"
                       name="author"
                       placeholder="Your Name"
                       className="form-control"
                       validators={{
                        minLength: minLength(3),
```

```
      maxLength: maxLength(15)
    }}
   />
   <Errors
    className="text-danger"
    model=".author"
    show="touched"
    messages={{
     minLength: "Must be greater than 2 characters",
     maxLength: "Must be 15 characters or less"
    }}
   />
  </Col>
</Row>

<Row className="form-group">
  <Label htmlFor="message" md={2}>
   Comment
  </Label>
  <Col md={10}>
   <Control.textarea
    className="form-control"
    id="comment"
    model=".comment"
    name="comment"
    rows="6"
```

```
                />
              </Col>
            </Row>

            <Row className="form-group">
              <Col md={{ size: 12 }}>
                <Button color="primary" value="submit">
                  Submit
                </Button>
              </Col>
            </Row>
          </LocalForm>
        </ModalBody>
      </Modal>
    </div>
  );
 }
}
export default Menudetail;
```


Action = addComment
dispach
MainComponent.js
export default withRouter(connect(mapStateToProps, mapDispatchToProps)(Main));
const mapDispatchToProps = dispatch => ({
addComment: (menuId, rating, author, comment) => dispatch(addComment(menuId, rating, author, comment))
});
<Menudetail menu={this.props.menus.filter((menu) => menu.menu_id === parseInt(match.params.menuId,10))[0]}
comments={this.props.comments.filter((comment) => comment.menuId === parseInt(match.params.menuId,10))}
addComment={this.props.addComment} />
);
MenudetailComponents.js
handleSubmit(values) {
this.props.addComment(this.props.menuId, values.rating, values.author, values.comment);
<CommentForm menuId={menuId} addComment={addComment} />
<RenderComments comments = {props.comments}
addComment={props.addComment}
menuId={props.menu.menu_id} />
redux/comments.js
import { COMMENTS } from '../shared/comments';
import * as ActionTypes from './ActionTypes';
export const Comments = (state = COMMENTS, action) => {
switch (action.type) {
case ActionTypes.ADD_COMMENT:
var comment = action.payload;
comment.id = state.length;
comment.date = new Date().toISOString();
console.log("Comment: ", comment);
return state.concat(comment);
default:
return state;
}
};
MainComponent.js
const mapStateToProps = state => {
return {
menus: state.menus,
comments: state.comments,
promotions: state.promotions,
authors: state.authors
}
}

Video 51.

Škola React 51. dio: Redux Thunk and Redux Logger

YouTube:
https://www.youtube.com/watch?v=on4OWGvazWI&list=PLvYzjHQxr_s3cjOVuE2EU94FUkSEJtcea&index=17

GitHub:
https://github.com/ivicakartelo/reactblog/commit/ea940d15e906fea9e88e4369bf7591c4742583ea

```
yarn add redux-thunk@2.2.0
yarn add redux-logger@3.0.6
```

configureStore.js

```
import {createStore, combineReducers, applyMiddleware } from 'redux';

import { Menus } from './menus';

import { Comments } from './comments';

import { Promotions } from './promotions';

import { Authors } from './authors';

import thunk from 'redux-thunk';

import logger from 'redux-logger';


export const ConfigureStore = () => {

  const store = createStore(

    combineReducers({

      menus: Menus,

      comments: Comments,

      promotions: Promotions,

      authors: Authors

    }),
```

```
        applyMiddleware(thunk, logger)
    );

    return store;
}
```

ActionTypes.js

```
export const MENUS_LOADING = 'MENUS_LOADING';

export const MENUS_FAILED = 'MENUS_FAILED';

export const ADD_MENUS = 'ADD_MENUS';
```

ActionCreators.js

```
import * as ActionTypes from './ActionTypes';
import { MENUS } from '../shared/menus';

export const addComment = (menuId, rating, author, comment) => ({

    type: ActionTypes.ADD_COMMENT,
    payload: {
        menuId: menuId,
        rating: rating,
        author: author,
        comment: comment
    }

});

export const fetchMenus = () => (dispatch) => {
```

```
    dispatch(menusLoading(true));

    setTimeout(() => {
        dispatch(addMenus(MENUS));
    }, 2000);
}

export const menusLoading = () => ({
    type: ActionTypes.MENUS_LOADING
});

export const menusFailed = (errmess) => ({
    type: ActionTypes.MENUS_FAILED,
    payload: errmess
});

export const addMenus = (menus) => ({
    type: ActionTypes.ADD_MENUS,
    payload: menus
});
```

Menus.js

```
import * as ActionTypes from './ActionTypes';

export const Menus = (state = { isLoading: true,
    errMess: null,
```

```
    menus:[]}, action) => {
    switch (action.type) {
        case ActionTypes.ADD_MENUS:
            return {...state, isLoading: false, errMess: null, menus: action.payload};

        case ActionTypes.MENUS_LOADING:
            return {...state, isLoading: true, errMess: null, menus: []}

        case ActionTypes.MENUS_FAILED:
            return {...state, isLoading: false, errMess: action.payload};

        default:
            return state;
    }
};
```

LoadingComponent.js

```
import React from 'react';

export const Loading = () => {
    return(
        <div className="col-12">
            <span className="fa fa-spinner fa-pulse fa-3x fa-fw text-
primary"></span>
            <p>Loading . . .</p>
        </div>
    );
};
```

MainComponent.js

```
import React, { Component } from 'react';
import Home from './HomeComponent';
import Menu from './MenuComponent';
import Menudetail from './MenudetailComponent';
import Header from './HeaderComponent';
import Footer from './FooterComponent';
import Contact from './ContactComponent';
import About from './AboutComponent';
import { Switch, Route, Redirect, withRouter } from 'react-router-dom'
import { connect } from 'react-redux';
import { addComment, fetchMenus } from '../redux/ActionCreators';


const mapStateToProps = state => {
  return {
    menus: state.menus,
    comments: state.comments,
    promotions: state.promotions,
    authors: state.authors
  }
}

const mapDispatchToProps = dispatch => ({
```

```
  addComment1: (menuId, rating, author, comment) =>
dispatch(addComment(menuId, rating, author, comment)),
  fetchMenus: () => { dispatch(fetchMenus())}

});

class Main extends Component {
  componentDidMount() {
    this.props.fetchMenus();
  }

  render() {
    const HomePage = () => {
      return(
          <Home
              menu={this.props.menus.menus.filter((menu) => menu.featured)[0]}
              menusLoading={this.props.menus.isLoading}
              menusErrMess={this.props.menus.errMess}
             promotion={this.props.promotions.filter((promo) =>
promo.featured)[0]}
              author={this.props.authors.filter((author) => author.featured)[0]}
          />
      );
    }
    const MenuWithId = ({match}) => {
      return(
```

```
        <Menudetail menu={this.props.menus.menus.filter((menu) =>
menu.menu_id === parseInt(match.params.menuId,10))[0]}

        isLoading={this.props.menus.isLoading}

         errMess={this.props.menus.errMess}

         comments={this.props.comments.filter((comment) =>
comment.menuId === parseInt(match.params.menuId,10))}

         addComment1={this.props.addComment1} />

         );

   };

   return (

     <div>

       <Header />

       <Switch>

            <Route path='/home' component={HomePage} />

            <Route exact path='/menu' component={() => <Menu
menus={this.props.menus} />} />

            <Route path='/menu/:menuId' component={MenuWithId} />

            <Route exact path='/contactus' component={Contact} />

            <Route exact path='/aboutus' component={() => <About
authors={this.props.authors} />} />

            <Redirect to="/home" />

        </Switch>

       <Footer />

     </div>

   );

  }

}
```

```
export default withRouter(connect(mapStateToProps,
mapDispatchToProps)(Main));
```

Menudetail.component.js

```
import React from 'react';

import { Breadcrumb, BreadcrumbItem, Modal, ModalHeader, ModalBody,

  Button, Row, Col, Label, CardImg, Card, CardBody, CardTitle, CardText } from
'reactstrap';

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

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

import { Loading } from './LoadingComponent';


   function RenderMenudetails({x}) {

   return (

     <Card>

       <CardImg top src={x.image} alt={x.name} />

       <CardBody>

         <CardTitle>{x.name}</CardTitle>

         <CardText>{x.content}</CardText>

       </CardBody>

     </Card>

   );

 }


  function RenderComments({comments, addComment1, menuId}) {

   const cmt = comments.map((w) => {

     return (
```

```
      <li key={w.id}>
        <p>{w.rating}</p>
        <p>{w.comment}</p>
        <p>
          -- {w.author},  
        {new Intl.DateTimeFormat('en-US', {
          year: 'numeric',
          month: 'long',
          day: '2-digit'
        }).format(new Date(w.date))}
        </p>
      </li>
    )
  })
  return(
    <div>
      <h5>Comments</h5>
        <ul className="list-unstyled">
          {cmt}
        </ul>
        <CommentForm menuId={menuId} addComment1={addComment1} />
    </div>
  );
}

  const Menudetail = (props) => {
```

```
  if (props.isLoading) {
    return(
      <div className="container">
        <div className="row">
          <Loading />
        </div>
      </div>
    );
  }
  else if (props.errMess) {
    return(
      <div className="container">
        <div className="row">
          <h4>{props.errMess}</h4>
        </div>
      </div>
    );
  }
  else if (props.menu != null) {
    return (

      <div className="container">
       <div className="row">
        <Breadcrumb>
```

```
            <BreadcrumbItem><Link
to="/menu">Menu</Link></BreadcrumbItem>
            <BreadcrumbItem active>{props.menu.name}</BreadcrumbItem>
          </Breadcrumb>
          <div className="col-12">
          < h3>{props.menu.name}</h3>
            <hr />
          </div>
        </div>
          <div className="row">
            <div className="col-12 col-md">
              <RenderMenudetails x = {props.menu} />
            </div>
            <div className="col-12 col-md">
              <RenderComments comments = {props.comments}
              addComment1={props.addComment1}
              menuId={props.menu.menu_id} />
            </div>
          </div>
        </div>
    );
  }
  else {
    return (
      <div></div>
    );
```

```
  }
}

const maxLength = len => val => !val || val.length <= len;
const minLength = len => val => val && val.length >= len;

    class CommentForm extends React.Component {
      constructor(props) {
        super(props);
        this.state = {
          isModalOpen: false
        };
        this.toggleModal = this.toggleModal.bind(this);
        this.handleSubmit = this.handleSubmit.bind(this);
      }

      toggleModal() {
        this.setState({
          isModalOpen: !this.state.isModalOpen
        });
      }

      handleSubmit(values) {
        this.toggleModal();
        this.props.addComment1(this.props.menuId, values.rating, values.author,
values.comment);
```

```
    }

    render() {
     return (
      <div>
       <Button outline onClick={this.toggleModal}>
        <span className="fa fa-pencil"></span> Submit Comment
       </Button>

       <Modal isOpen={this.state.isModalOpen} toggle={this.toggleModal}>
        <ModalHeader toggle={this.toggleModal}>Submit
Comment</ModalHeader>
        <ModalBody>
         <LocalForm onSubmit={values => this.handleSubmit(values)}>
         <Row className="form-group">
                        <Label htmlFor="rating" md={2}>Rating</Label>
                        <Col md={10}>
                          <Control.select model=".rating" id="rating"
name="rating"
                            className="form-control"
                            defaultValue="1">
                            <option value="1">1</option>
                            <option value="2">2</option>
                            <option value="3">3</option>
                            <option value="4">4</option>
                            <option value="5">5</option>
                          </Control.select>
```

```
            </Col>
          </Row>
<Row className="form-group">
 <Label htmlFor="author" md={2}>
  Your Name
 </Label>
 <Col md={10}>
  <Control.text
   model=".author"
   id="author"
   name="author"
   placeholder="Your Name"
   className="form-control"
   validators={{
    minLength: minLength(3),
    maxLength: maxLength(15)
   }}
  />
  <Errors
   className="text-danger"
   model=".author"
   show="touched"
   messages={{
    minLength: "Must be greater than 2 characters",
    maxLength: "Must be 15 characters or less"
   }}
```

```
    />
  </Col>
</Row>

<Row className="form-group">
  <Label htmlFor="message" md={2}>
    Comment
  </Label>
  <Col md={10}>
    <Control.textarea
      className="form-control"
      id="comment"
      model=".comment"
      name="comment"
      rows="6"
    />
  </Col>
</Row>

<Row className="form-group">
  <Col md={{ size: 12 }}>
    <Button color="primary" value="submit">
      Submit
    </Button>
  </Col>
</Row>
```

```
            </LocalForm>
          </ModalBody>
        </Modal>
      </div>
    );
  }
}

export default Menudetail;
```

HomeComponent.js

```
import React from 'react';
import { Card, CardImg, CardText, CardBody,
  CardTitle, CardSubtitle} from 'reactstrap';
  import { Loading } from './LoadingComponent';

  function RenderCard({item, isLoading, errMess}) {
    if (isLoading) {
      return(
          <Loading />
      );
    }
    else if (errMess) {
      return(
          <h4>{errMess}</h4>
      );
    }
```

```
      else


   return(
      <Card>
         <CardImg src={item.image} alt={item.name} />
         <CardBody>
         <CardTitle>{item.name}</CardTitle>
         {item.designation ? <CardSubtitle>{item.designation}</CardSubtitle> :
null }
         <CardText>{item.content}</CardText>
         </CardBody>
      </Card>
   );


}


function Home(props) {
   return(
      <div className="container">
         <div className="row align-items-start">
            <div className="col-12 col-md m-1">
            <RenderCard item={props.menu} isLoading={props.menusLoading}
errMess={props.menusErrMess}  />
            </div>
            <div className="col-12 col-md m-1">
               <RenderCard item={props.promotion} />
            </div>
```

```
            <div className="col-12 col-md m-1">
              <RenderCard item={props.author} />
            </div>
          </div>
        </div>
    );
}

export default Home;
```

MenuComponent.js

```
import React from 'react';
import { Card, CardImg, CardImgOverlay,
  CardTitle, Breadcrumb, BreadcrumbItem } from 'reactstrap';
import { Link } from 'react-router-dom';
import { Loading } from './LoadingComponent';

function Menu(props) {
  const menu = props.menus.menus.map((menu) => {
    return (
      <div className="col-12 col-md-6" key={menu.menu_id}>
        <RenderMenuItem menu={menu} />
      </div>
    );
  })

function RenderMenuItem ({menu}) {
```

```
    return (

        <Card>
          <Link to={`/menu/${menu.menu_id}`} >
          <CardImg src={menu.image} alt={menu.name} />
          <CardImgOverlay>
            <CardTitle>{menu.name}</CardTitle>
          </CardImgOverlay>
          </Link>
        </Card>
    );
};

if (props.menus.isLoading) {
  return(
      <div className="container">
          <div className="row">
              <Loading />
          </div>
      </div>
  );
}
else if (props.menus.errMess) {
  return(
      <div className="container">
          <div className="row">
```

```
            <div className="col-12">
              <h4>{props.menus.errMess}</h4>
            </div>
          </div>
        </div>
    );
}
else

  return (
    <div className="container">
      <div className="row">
        <Breadcrumb>
          <BreadcrumbItem><Link to="/home">Home</Link></BreadcrumbItem>
          <BreadcrumbItem active>Menu</BreadcrumbItem>
        </Breadcrumb>
          <div className="col-12">
            <h3>Menu</h3>
            <hr />
          </div>

          {menu}
        </div>
    </div>
  );
}
```

```
export default Menu;
```

Video 54.

Škola React 54. dio: Fetch from server

YouTube:

https://www.youtube.com/watch?v=_G2TP9Op-sM&list=PLvYzjHQxr_s3cjOVuE2EU94FUkSEJtcea&index=13&t=15s

GitHub:

https://github.com/ivicakartelo/reactblog/commit/bcca04c000af05f386899b641b0ba1ae082ebf82

```
yarn add cross-fetch@2.1.0
```

shared/baseURL.js

```
export const baseUrl = 'http://localhost:3001/';
```

ActionTypes.js

```
export const ADD_COMMENTS = 'ADD_COMMENTS';

export const COMMENTS_FAILED = 'COMMENTS_FAILED';

export const PROMOS_LOADING = 'PROMOS_LOADING';

export const ADD_PROMOS = 'ADD_PROMOS';

export const PROMOS_FAILED = 'PROMOS_FAILED';
```

ActionCreators.js

```
import * as ActionTypes from './ActionTypes';

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


export const addComment = (menuId, rating, author, comment) => ({


  type: ActionTypes.ADD_COMMENT,

  payload: {
```

```
        menuId: menuId,
        rating: rating,
        author: author,
        comment: comment
    }

});

export const fetchMenus = () => (dispatch) => {

    dispatch(menusLoading(true));

    return fetch(baseUrl + 'menus')
    .then(response => response.json())
    .then(menus => dispatch(addMenus(menus)));
}

export const menusLoading = () => ({
    type: ActionTypes.MENUS_LOADING
});

export const menusFailed = (errmess) => ({
    type: ActionTypes.MENUS_FAILED,
    payload: errmess
});
```

```
export const addMenus = (menus) => ({
   type: ActionTypes.ADD_MENUS,
   payload: menus
});
export const fetchComments = () => (dispatch) => {
   return fetch(baseUrl + 'comments')
   .then(response => response.json())
   .then(comments => dispatch(addComments(comments)));
};

export const commentsFailed = (errmess) => ({
   type: ActionTypes.COMMENTS_FAILED,
   payload: errmess
});

export const addComments = (comments) => ({
   type: ActionTypes.ADD_COMMENTS,
   payload: comments
});

export const fetchPromos = () => (dispatch) => {

   dispatch(promosLoading());

   return fetch(baseUrl + 'promotions')
   .then(response => response.json())
```

```
    .then(promos => dispatch(addPromos(promos)));
}

export const promosLoading = () => ({
    type: ActionTypes.PROMOS_LOADING
});

export const promosFailed = (errmess) => ({
    type: ActionTypes.PROMOS_FAILED,
    payload: errmess
});

export const addPromos = (promos) => ({
    type: ActionTypes.ADD_PROMOS,
    payload: promos
});
```

Coments.js

```
import * as ActionTypes from './ActionTypes';

export const Comments = (state = { errMess: null, comments:[]}, action) => {
  switch (action.type) {
    case ActionTypes.ADD_COMMENTS:
      return {...state, errMess: null, comments: action.payload};

    case ActionTypes.COMMENTS_FAILED:
      return {...state, errMess: action.payload};
```

```
    case ActionTypes.ADD_COMMENT:
      var comment = action.payload;
      comment.id = state.comments.length;
      comment.date = new Date().toISOString();
      return { ...state, comments: state.comments.concat(comment)};

    default:
     return state;
  }
};
```

promotions.js

```
import * as ActionTypes from './ActionTypes';

export const Promotions = (state  = { isLoading: true,
                                      errMess: null,
                                      promotions:[]}, action) => {
    switch (action.type) {
      case ActionTypes.ADD_PROMOS:
      return {...state, isLoading: false, errMess: null, promotions:
action.payload};

      case ActionTypes.PROMOS_LOADING:
         return {...state, isLoading: true, errMess: null, promotions: []}

      case ActionTypes.PROMOS_FAILED:
         return {...state, isLoading: false, errMess: action.payload};
```

```
      default:
        return state;
    }
};
```

MainComponent.js

```
import React, { Component } from 'react';
import Home from './HomeComponent';
import Menu from './MenuComponent';
import Menudetail from './MenudetailComponent';
import Header from './HeaderComponent';
import Footer from './FooterComponent';
import Contact from './ContactComponent';
import About from './AboutComponent';
import { Switch, Route, Redirect, withRouter } from 'react-router-dom'
import { connect } from 'react-redux';
import { addComment, fetchMenus, fetchComments, fetchPromos } from '../redux/ActionCreators';
import { actions } from 'react-redux-form';



const mapStateToProps = state => {
  return {
    menus: state.menus,
    comments: state.comments,
    promotions: state.promotions,
    authors: state.authors
```

```
  }
}

const mapDispatchToProps = dispatch => ({

  addComment1: (menuId, rating, author, comment) =>
dispatch(addComment(menuId, rating, author, comment)),
  fetchMenus1: () => { dispatch(fetchMenus())},
  resetFeedbackForm: () => { dispatch(actions.reset('feedback'))},
  fetchComments: () => dispatch(fetchComments()),
  fetchPromos: () => dispatch(fetchPromos())

});

class Main extends Component {
  componentDidMount() {
    this.props.fetchMenus1();
    this.props.fetchComments();
    this.props.fetchPromos();
  }

  render() {
    const HomePage = () => {
      return(
        <Home
          menu={this.props.menus.menus.filter((menu) => menu.featured)[0]}
```

```
            menusLoading={this.props.menus.isLoading}
            menusErrMess={this.props.menus.errMess}
            promotion={this.props.promotions.promotions.filter((promo) =>
promo.featured)[0]}
            promoLoading={this.props.promotions.isLoading}
            promoErrMess={this.props.promotions.errMess}
            author={this.props.authors.filter((author) => author.featured)[0]}
          />
      );
    }
    const MenuWithId = ({match}) => {
      return(
        <Menudetail menu={this.props.menus.menus.filter((menu) =>
menu.menu_id === parseInt(match.params.menuId,10))[0]}
          isLoading={this.props.menus.isLoading}
          errMess={this.props.menus.errMess}
          comments={this.props.comments.comments.filter((comment) =>
comment.menuId === parseInt(match.params.menuId,10))}
          commentsErrMess={this.props.comments.errMess}
          addComment1={this.props.addComment1} />
          );
    };
    return (
      <div>
        <Header />
        <Switch>
            <Route path='/home' component={HomePage} />
```

```
          <Route exact path='/menu' component={() => <Menu
menus={this.props.menus} />} />

          <Route path='/menu/:menuId' component={MenuWithId} />

          <Route exact path='/contactus' component={() => <Contact
resetFeedbackForm={this.props.resetFeedbackForm} />} />

          <Route exact path='/aboutus' component={() => <About
authors={this.props.authors} />} />

          <Redirect to="/home" />

        </Switch>

      <Footer />

    </div>

   );

 }

}


export default withRouter(connect(mapStateToProps,
mapDispatchToProps)(Main));
```

MenuComponent.js

```
import React from 'react';

import { Card, CardImg, CardImgOverlay,

 CardTitle, Breadcrumb, BreadcrumbItem } from 'reactstrap';

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

import { Loading } from './LoadingComponent';

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


function Menu(props) {

 const menu = props.menus.menus.map((menu) => {

   return (
```

```
    <div className="col-12 col-md-6" key={menu.menu_id}>
      <RenderMenuItem menu={menu} />
    </div>
  );
 })

function RenderMenuItem ({menu}) {
  return (

      <Card>
       <Link to={`/menu/${menu.menu_id}`} >
       <CardImg width="100%" src={baseUrl + menu.image} alt={menu.name}
/>
       <CardImgOverlay>
        <CardTitle>{menu.name}</CardTitle>
       </CardImgOverlay>
       </Link>
      </Card>
  );
};

if (props.menus.isLoading) {
 return(
    <div className="container">
       <div className="row">
          <Loading />
```

```
        </div>
      </div>
    );
  }
  else if (props.menus.errMess) {
    return(
      <div className="container">
        <div className="row">
          <div className="col-12">
            <h4>{props.menus.errMess}</h4>
          </div>
        </div>
      </div>
    );
  }
  else

    return (
      <div className="container">
        <div className="row">
          <Breadcrumb>
            <BreadcrumbItem><Link to="/home">Home</Link></BreadcrumbItem>
            <BreadcrumbItem active>Menu</BreadcrumbItem>
          </Breadcrumb>
            <div className="col-12">
              <h3>Menu</h3>
```

```
          <hr />
        </div>

        {menu}
      </div>
    </div>
  );
}
export default Menu;
```

HomeComponent.js

```
import React from 'react';
import { Card, CardImg, CardText, CardBody,
   CardTitle, CardSubtitle} from 'reactstrap';
import { Loading } from './LoadingComponent';
import { baseUrl } from '../shared/baseUrl';

   function RenderCard({item, isLoading, errMess}) {
      if (isLoading) {
         return(
               <Loading />
         );
      }
      else if (errMess) {
         return(
               <h4>{errMess}</h4>
         );
```

```
    }
    else

  return(
    <Card>
      <CardImg src={baseUrl + item.image} alt={item.name} />
      <CardBody>
      <CardTitle>{item.name}</CardTitle>
      {item.designation ? <CardSubtitle>{item.designation}</CardSubtitle> :
null }
      <CardText>{item.content}</CardText>
      </CardBody>
    </Card>
  );

}

function Home(props) {
  return(
    <div className="container">
      <div className="row align-items-start">
        <div className="col-12 col-md m-1">
        <RenderCard item={props.menu} isLoading={props.menusLoading}
errMess={props.menusErrMess}  />
        </div>
        <div className="col-12 col-md m-1">
```

```
            <RenderCard item={props.promotion}
isLoading={props.promoLoading} errMess={props.promoErrMess} />
            </div>
            <div className="col-12 col-md m-1">
              <RenderCard item={props.author} />
            </div>
          </div>
      </div>
    );
}

export default Home;
```

MenudetailComponent.js

```
import React from 'react';
import { Breadcrumb, BreadcrumbItem, Modal, ModalHeader, ModalBody,
  Button, Row, Col, Label, CardImg, Card, CardBody, CardTitle, CardText } 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';

    function RenderMenudetails({x}) {
    return (
      <Card>
        <CardImg top src={baseUrl + x.image} alt={x.name} />
```

```
      <CardBody>
        <CardTitle>{x.name}</CardTitle>
        <CardText>{x.content}</CardText>
      </CardBody>
    </Card>
  );
}

function RenderComments({comments, addComment1, menuId}) {
  const cmt = comments.map((w) => {
    return (
      <li key={w.id}>
        <p>{w.rating}</p>
        <p>{w.comment}</p>
        <p>
          -- {w.author},  
        {new Intl.DateTimeFormat('en-US', {
          year: 'numeric',
          month: 'long',
          day: '2-digit'
        }).format(new Date(w.date))}
        </p>
      </li>
    )
  })
  return(
```

```
    <div>
      <h5>Comments</h5>
        <ul className="list-unstyled">
          {cmt}
        </ul>
        <CommentForm menuId={menuId} addComment1={addComment1} />
    </div>
  );
}

  const Menudetail = (props) => {

    if (props.isLoading) {
      return(
          <div className="container">
              <div className="row">
                  <Loading />
              </div>
          </div>
      );
    }
    else if (props.errMess) {
      return(
          <div className="container">
              <div className="row">
                  <h4>{props.errMess}</h4>
```

```
            </div>
          </div>
        );
    }
    else if (props.menu != null) {
      return (

          <div className="container">
           <div className="row">
            <Breadcrumb>
              <BreadcrumbItem><Link
to="/menu">Menu</Link></BreadcrumbItem>
              <BreadcrumbItem active>{props.menu.name}</BreadcrumbItem>
            </Breadcrumb>
           <div className="col-12">
           < h3>{props.menu.name}</h3>
            <hr />
           </div>
          </div>
           <div className="row">
            <div className="col-12 col-md">
              <RenderMenudetails x = {props.menu} />
            </div>
            <div className="col-12 col-md">
              <RenderComments comments = {props.comments}
              addComment1={props.addComment1}
```

```
            menuId={props.menu.menu_id} />
          </div>
        </div>
      </div>
    );
  }
  else {
    return (
      <div></div>
    );
  }
}

const maxLength = len => val => !val || val.length <= len;
const minLength = len => val => val && val.length >= len;

  class CommentForm extends React.Component {
    constructor(props) {
      super(props);
      this.state = {
        isModalOpen: false
      };
      this.toggleModal = this.toggleModal.bind(this);
      this.handleSubmit = this.handleSubmit.bind(this);
    }
```

```
    toggleModal() {
      this.setState({
        isModalOpen: !this.state.isModalOpen
      });
    }

    handleSubmit(values) {
      this.toggleModal();
      this.props.addComment1(this.props.menuId, values.rating, values.author,
values.comment);
    }

    render() {
      return (
        <div>
          <Button outline onClick={this.toggleModal}>
            <span className="fa fa-pencil"></span> Submit Comment
          </Button>

          <Modal isOpen={this.state.isModalOpen} toggle={this.toggleModal}>
            <ModalHeader toggle={this.toggleModal}>Submit
Comment</ModalHeader>
            <ModalBody>
              <LocalForm onSubmit={values => this.handleSubmit(values)}>
              <Row className="form-group">
                              <Label htmlFor="rating" md={2}>Rating</Label>
                              <Col md={10}>
```

```
                        <Control.select model=".rating" id="rating"
name="rating"
                          className="form-control"
                          defaultValue="1">
                          <option value="1">1</option>
                          <option value="2">2</option>
                          <option value="3">3</option>
                          <option value="4">4</option>
                          <option value="5">5</option>
                        </Control.select>
                      </Col>
                    </Row>
                <Row className="form-group">
                 <Label htmlFor="author" md={2}>
                  Your Name
                 </Label>
                 <Col md={10}>
                  <Control.text
                   model=".author"
                   id="author"
                   name="author"
                   placeholder="Your Name"
                   className="form-control"
                   validators={{
                    minLength: minLength(3),
                    maxLength: maxLength(15)
```

```
    }}
   />
   <Errors
    className="text-danger"
    model=".author"
    show="touched"
    messages={{
     minLength: "Must be greater than 2 characters",
     maxLength: "Must be 15 characters or less"
    }}
   />
  </Col>
</Row>

<Row className="form-group">
  <Label htmlFor="message" md={2}>
   Comment
  </Label>
  <Col md={10}>
   <Control.textarea
    className="form-control"
    id="comment"
    model=".comment"
    name="comment"
    rows="6"
   />
```

```
            </Col>
          </Row>

          <Row className="form-group">
            <Col md={{ size: 12 }}>
              <Button color="primary" value="submit">
                Submit
              </Button>
            </Col>
          </Row>
        </LocalForm>
      </ModalBody>
    </Modal>
  </div>
  );
 }
}
export default Menudetail;
```

Video 58.

Škola React 58. dio: Fetch Post Comment

YouTube:

https://www.youtube.com/watch?v=k4d8eWptj9w&list=PLvYzjHQxr_s3cjOVuE2EU94FUkSEJtcea&index=9&t=2s

GitHub:

https://github.com/ivicakartelo/reactblog/commit/80a6ad161b026b70d93c6ddff7dde57ce53504fb

ActionCreators.js

```
import * as ActionTypes from './ActionTypes';
import { baseUrl } from '../shared/baseUrl';

export const addComment = (comment) => ({
  type: ActionTypes.ADD_COMMENT,
  payload: comment
});
export const postComment = (menuId, rating, author, comment) => (dispatch)
=> {

  const newComment = {
     menuId: menuId,
     rating: rating,
     author: author,
     comment: comment
  };
  newComment.date = new Date().toISOString();

  return fetch(baseUrl + 'comments', {
     method: "POST",
     body: JSON.stringify(newComment),
     headers: {
       "Content-Type": "application/json"
     },
     credentials: "same-origin"
  })
```

```
  .then(response => {
    if (response.ok) {
      return response;
    } else {
      var error = new Error('Error ' + response.status + ': ' +
response.statusText);
      error.response = response;
      throw error;
    }
  },
  error => {
       throw error;
  })
  .then(response => response.json())
  .then(response => dispatch(addComment(response)))
  .catch(error =>  { console.log('post comments', error.message); alert('Your
comment could not be posted\nError: '+error.message); });
};

export const fetchMenus = () => (dispatch) => {

  dispatch(menusLoading(true));

  return fetch(baseUrl + 'menus')
  .then(response => {
    if (response.ok) {
      return response;
```

```
      } else {
        var error = new Error('Error ' + response.status + ': ' +
response.statusText);
        error.response = response;
        throw error;
      }
    },
    error => {
        var errmess = new Error(error.message);
        throw errmess;
    })
  .then(response => response.json())
  .then(menus => dispatch(addMenus(menus)))
  .catch(error => dispatch(menusFailed(error.message)));
}

export const menusLoading = () => ({
  type: ActionTypes.MENUS_LOADING
});

export const menusFailed = (errmess) => ({
  type: ActionTypes.MENUS_FAILED,
  payload: errmess
});

export const addMenus = (menus) => ({
```

```
    type: ActionTypes.ADD_MENUS,
    payload: menus
});
export const fetchComments = () => (dispatch) => {
    return fetch(baseUrl + 'comments')
    .then(response => {
        if (response.ok) {
          return response;
        } else {
          var error = new Error('Error ' + response.status + ': ' +
response.statusText);
          error.response = response;
          throw error;
        }
      },
      error => {
            var errmess = new Error(error.message);
            throw errmess;
      })
    .then(response => response.json())
    .then(comments => dispatch(addComments(comments)))
    .catch(error => dispatch(commentsFailed(error.message)));
};



export const commentsFailed = (errmess) => ({
```

```
    type: ActionTypes.COMMENTS_FAILED,
    payload: errmess
});

export const addComments = (comments) => ({
    type: ActionTypes.ADD_COMMENTS,
    payload: comments
});

export const fetchPromos = () => (dispatch) => {

    dispatch(promosLoading());

    return fetch(baseUrl + 'promotions')
    .then(response => {
        if (response.ok) {
          return response;
        } else {
          var error = new Error('Error ' + response.status + ': ' +
response.statusText);
          error.response = response;
          throw error;
        }
      },
      error => {
            var errmess = new Error(error.message);
```

```
            throw errmess;
        })
    .then(response => response.json())
    .then(promos => dispatch(addPromos(promos)))
    .catch(error => dispatch(promosFailed(error.message)));
}

export const promosLoading = () => ({
    type: ActionTypes.PROMOS_LOADING
});

export const promosFailed = (errmess) => ({
    type: ActionTypes.PROMOS_FAILED,
    payload: errmess
});

export const addPromos = (promos) => ({
    type: ActionTypes.ADD_PROMOS,
    payload: promos
});
```

comments.js

```
import * as ActionTypes from './ActionTypes';

export const Comments = (state = { errMess: null, comments:[]}, action) => {
  switch (action.type) {
    case ActionTypes.ADD_COMMENTS:
```

```
      return {...state, errMess: null, comments: action.payload};

    case ActionTypes.COMMENTS_FAILED:
      return {...state, errMess: action.payload};

    case ActionTypes.ADD_COMMENT:
        var comment = action.payload;
        return { ...state, comments: state.comments.concat(comment)};

    default:
      return state;
  }
};
```

MainComponent.js

```
import React, { Component } from 'react';
import Home from './HomeComponent';
import Menu from './MenuComponent';
import Menudetail from './MenudetailComponent';
import Header from './HeaderComponent';
import Footer from './FooterComponent';
import Contact from './ContactComponent';
import About from './AboutComponent';
import { Switch, Route, Redirect, withRouter } from 'react-router-dom'
import { connect } from 'react-redux';
import { postComment, fetchMenus, fetchComments, fetchPromos } from
'../redux/ActionCreators';
import { actions } from 'react-redux-form';
```

```
const mapStateToProps = state => {
  return {
    menus: state.menus,
    comments: state.comments,
    promotions: state.promotions,
    authors: state.authors
  }
}

const mapDispatchToProps = dispatch => ({

  postComment1: (menuId, rating, author, comment) =>
dispatch(postComment(menuId, rating, author, comment)),
  fetchMenus1: () => { dispatch(fetchMenus())},
  resetFeedbackForm: () => { dispatch(actions.reset('feedback'))},
  fetchComments: () => dispatch(fetchComments()),
  fetchPromos: () => dispatch(fetchPromos())

});

class Main extends Component {
  componentDidMount() {
    this.props.fetchMenus1();
    this.props.fetchComments();
```

```
  this.props.fetchPromos();
 }


 render() {
  const HomePage = () => {
   return(
      <Home
         menu={this.props.menus.menus.filter((menu) => menu.featured)[0]}
         menusLoading={this.props.menus.isLoading}
         menusErrMess={this.props.menus.errMess}
         promotion={this.props.promotions.promotions.filter((promo) =>
promo.featured)[0]}
         promoLoading={this.props.promotions.isLoading}
         promoErrMess={this.props.promotions.errMess}
         author={this.props.authors.filter((author) => author.featured)[0]}
      />
   );
  }
  const MenuWithId = ({match}) => {
   return(
      <Menudetail menu={this.props.menus.menus.filter((menu) =>
menu.menu_id === parseInt(match.params.menuId,10))[0]}
      isLoading={this.props.menus.isLoading}
        errMess={this.props.menus.errMess}
        comments={this.props.comments.comments.filter((comment) =>
comment.menuId === parseInt(match.params.menuId,10))}
        commentsErrMess={this.props.comments.errMess}
```

```
            postComment1={this.props.postComment1} />
            );
    };
    return (
      <div>
        <Header />
        <Switch>
            <Route path='/home' component={HomePage} />
            <Route exact path='/menu' component={() => <Menu
menus={this.props.menus} />} />
            <Route path='/menu/:menuId' component={MenuWithId} />
            <Route exact path='/contactus' component={() => <Contact
resetFeedbackForm={this.props.resetFeedbackForm} />} />
            <Route exact path='/aboutus' component={() => <About
authors={this.props.authors} />} />
            <Redirect to="/home" />
          </Switch>
        <Footer />
      </div>
    );
  }
}

export default withRouter(connect(mapStateToProps,
mapDispatchToProps)(Main));
```

MenudetailComponent.js

```
import React from 'react';
import { Breadcrumb, BreadcrumbItem, Modal, ModalHeader, ModalBody,
```

```
 Button, Row, Col, Label, CardImg, Card, CardBody, CardTitle, CardText } 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';


  function RenderMenudetails({x}) {
  return (
    <Card>
      <CardImg top src={baseUrl + x.image} alt={x.name} />


      <CardBody>
        <CardTitle>{x.name}</CardTitle>
        <CardText>{x.content}</CardText>
      </CardBody>
    </Card>
  );
}


 function RenderComments({comments, postComment1, menuId}) {
  const cmt = comments.map((w) => {
    return (
      <li key={w.id}>
        <p>{w.rating}</p>
        <p>{w.comment}</p>
```

```
        <p>
          -- {w.author},  
        {new Intl.DateTimeFormat('en-US', {
          year: 'numeric',
          month: 'long',
          day: '2-digit'
        }).format(new Date(w.date))}
        </p>
      </li>
    )
  })
  return(
    <div>
      <h5>Comments</h5>
        <ul className="list-unstyled">
          {cmt}
        </ul>
        <CommentForm menuId={menuId} postComment1={postComment1} />
    </div>
  );
}

  const Menudetail = (props) => {

    if (props.isLoading) {
      return(
```

```
        <div className="container">
          <div className="row">
            <Loading />
          </div>
        </div>
      );
    }
    else if (props.errMess) {
      return(
        <div className="container">
          <div className="row">
            <h4>{props.errMess}</h4>
          </div>
        </div>
      );
    }
    else if (props.menu != null) {
      return (

        <div className="container">
         <div className="row">
          <Breadcrumb>
           <BreadcrumbItem><Link
to="/menu">Menu</Link></BreadcrumbItem>
            <BreadcrumbItem active>{props.menu.name}</BreadcrumbItem>
          </Breadcrumb>
```

```
          <div className="col-12">
          < h3>{props.menu.name}</h3>
            <hr />
          </div>
        </div>
          <div className="row">
            <div className="col-12 col-md">
              <RenderMenudetails x = {props.menu} />
            </div>
            <div className="col-12 col-md">
              <RenderComments comments = {props.comments}
              postComment1={props.postComment1}
              menuId={props.menu.menu_id} />
            </div>
          </div>
        </div>
    );
  }
  else {
    return (
      <div></div>
    );
  }
}

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

```
const minLength = len => val => val && val.length >= len;

class CommentForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      isModalOpen: false
    };
    this.toggleModal = this.toggleModal.bind(this);
    this.handleSubmit = this.handleSubmit.bind(this);
  }

  toggleModal() {
    this.setState({
      isModalOpen: !this.state.isModalOpen
    });
  }

  handleSubmit(values) {
    this.toggleModal();
    this.props.postComment1(this.props.menuId, values.rating,
values.author, values.comment);
  }

  render() {
    return (
```

```
        <div>
          <Button outline onClick={this.toggleModal}>
            <span className="fa fa-pencil"></span> Submit Comment
          </Button>

          <Modal isOpen={this.state.isModalOpen} toggle={this.toggleModal}>
            <ModalHeader toggle={this.toggleModal}>Submit
Comment</ModalHeader>
            <ModalBody>
              <LocalForm onSubmit={values => this.handleSubmit(values)}>
              <Row className="form-group">
                        <Label htmlFor="rating" md={2}>Rating</Label>
                        <Col md={10}>
                          <Control.select model=".rating" id="rating"
name="rating"
                            className="form-control"
                            defaultValue="1">
                            <option value="1">1</option>
                            <option value="2">2</option>
                            <option value="3">3</option>
                            <option value="4">4</option>
                            <option value="5">5</option>
                          </Control.select>
                        </Col>
                      </Row>
                <Row className="form-group">
                  <Label htmlFor="author" md={2}>
```

```
    Your Name
  </Label>
  <Col md={10}>
    <Control.text
      model=".author"
      id="author"
      name="author"
      placeholder="Your Name"
      className="form-control"
      validators={{
        minLength: minLength(3),
        maxLength: maxLength(15)
      }}
    />
    <Errors
      className="text-danger"
      model=".author"
      show="touched"
      messages={{
        minLength: "Must be greater than 2 characters",
        maxLength: "Must be 15 characters or less"
      }}
    />
  </Col>
</Row>
```

```
        <Row className="form-group">
          <Label htmlFor="message" md={2}>
            Comment
          </Label>
          <Col md={10}>
            <Control.textarea
              className="form-control"
              id="comment"
              model=".comment"
              name="comment"
              rows="6"
            />
          </Col>
        </Row>

        <Row className="form-group">
          <Col md={{ size: 12 }}>
            <Button color="primary" value="submit">
              Submit
            </Button>
          </Col>
        </Row>
      </LocalForm>
    </ModalBody>
  </Modal>
</div>
```

```
        );
      }
    }
export default Menudetail;
```