# Exercise (Instructions): Redux Actions

Objectives and Outcomes

In this exercise we will learn to define various Redux actions and implement the action creators to dispatch the actions to the Redux store. At the end of this exercise you will be able to:

- Define Redux actions and implement the action creators
- Dispatch actions from the action creators to update the system state in the Redux store

## Creating Actions

- In the *redux* folder create a new file named *ActionTypes.js* and add the following to it:

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

- Then, create a file named *ActionCreators.js* and add the following to it:

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

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

- Next, update *comments.js* to initiate action when the action is dispatched by the ActionCreator as follows:

```
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;
      }
};
```

- Now update *MainComponent.js* to make the action available for use within the DishdetailComponent as follows:

```
. . .

import { addComment } from '../redux/ActionCreators';

. . .

  const mapDispatchToProps = dispatch => ({

    addComment: (dishId, rating, author, comment) => dispatch(addComment(dishId,
      rating, author, comment))

  });

. . .

      <DishDetail dish={this.props.dishes.filter((dish) => dish.id === parseInt
        (match.params.dishId,10))[0]}
        comments={this.props.comments.filter((comment) => comment.dishId ===
          parseInt(match.params.dishId,10))}
        addComment={this.props.addComment}
      />

. . .

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

- Finally, update *DishdetailComponent.js* as follows to initiate the action upon the user submitting the comment form:

```
. . .

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


. . .

      <CommentForm dishId={dishId} addComment={addComment} />


. . .

        this.props.addComment(this.props.dishId, values.rating, values.author,
          values.comment);


. . .

      <RenderComments comments={props.comments}
        addComment={props.addComment}
        dishId={props.dish.id}
      />

. . .
```

- Save all the changes and do a Git commit with the message "Redux Actions"

## Conclusions

In this exercise we have learnt to create and dispatch actions to update the system state in the Redux store.

Mark as completed