CST8259 Web Programming Language II

Lab 8 – Part 2

Objective

- Create and use ReactJS components
- Access ReactJS components' properties
- Maintain ReactJS components' states

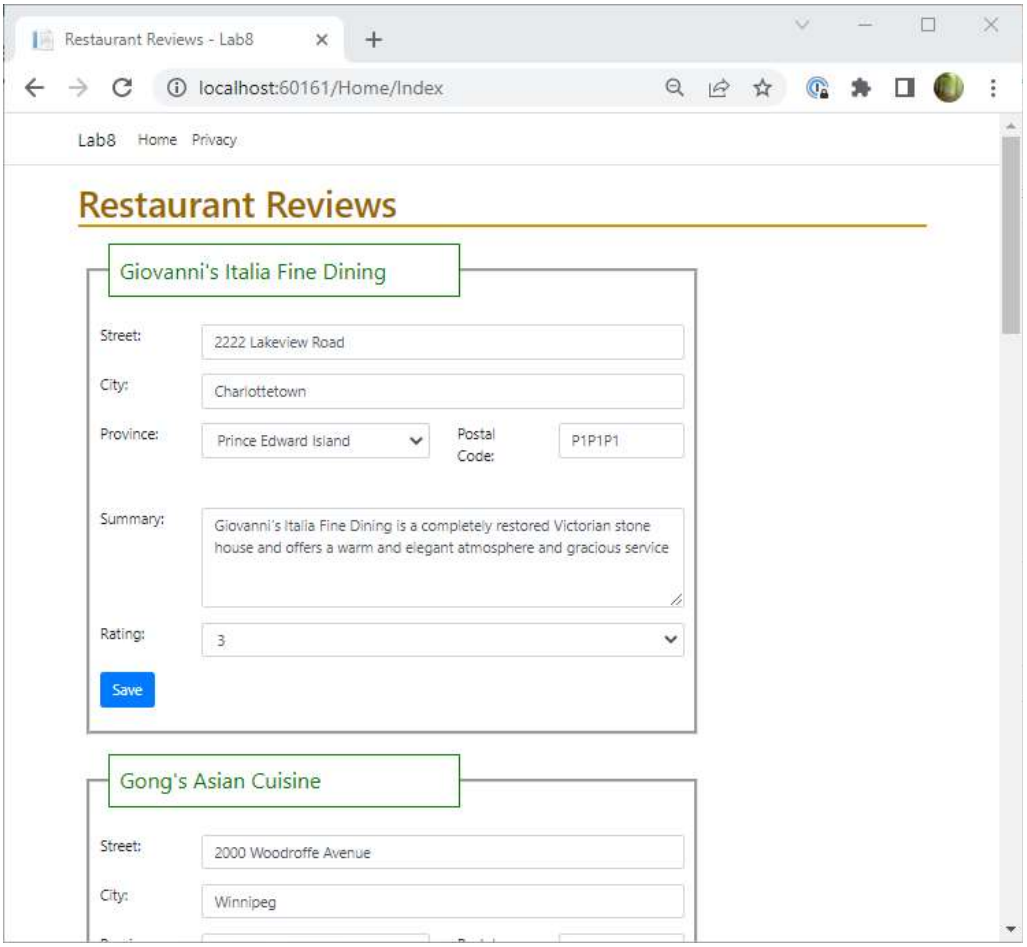
Due Date

See Brightspace posting for the due date of this lab. To earn 5 points, you are required:

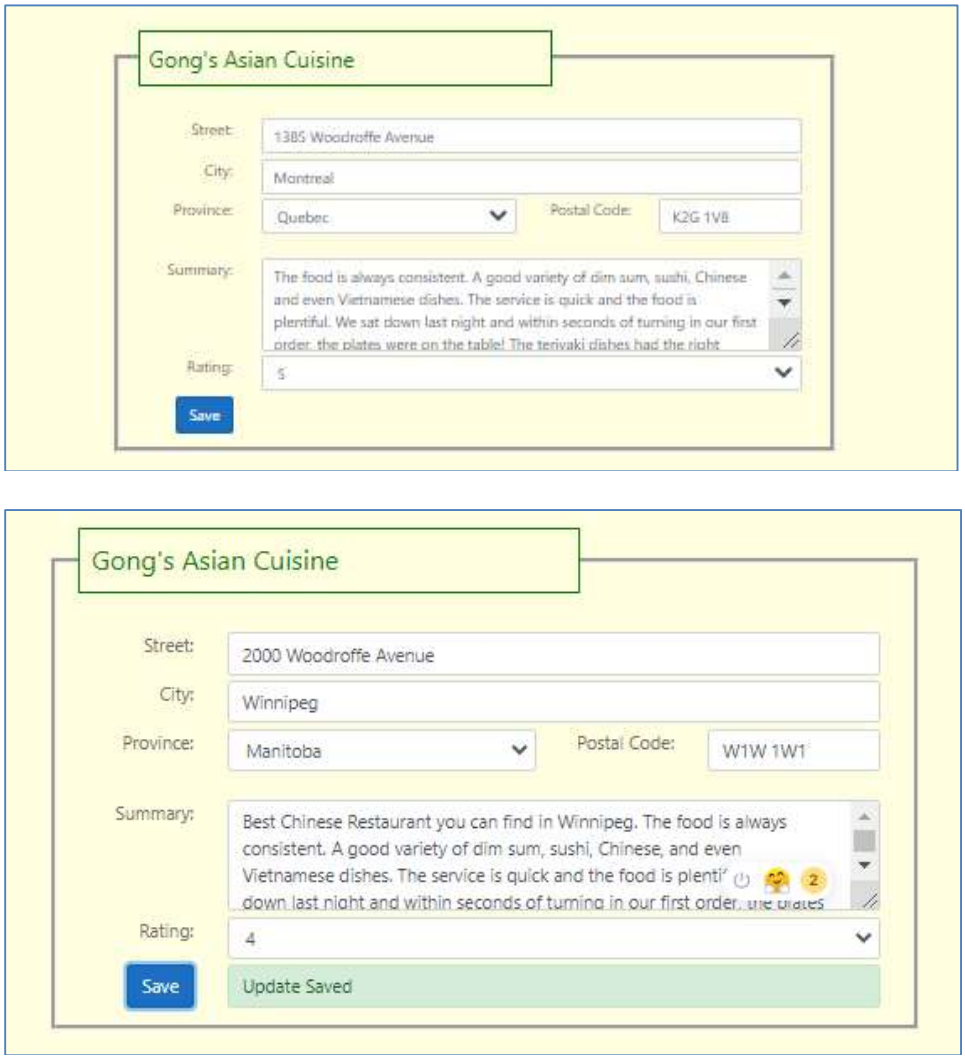
1. Complete the lab as required.
2. Submit your lab work to the Brightspace before the due date.
3. Demo your lab during the lab session in the week after the due date

Application Overview

1. Upon accessing the application's index page, the application displays all restaurants reviews from the Restful Restaurant Review Service API you developed in Lab 7.



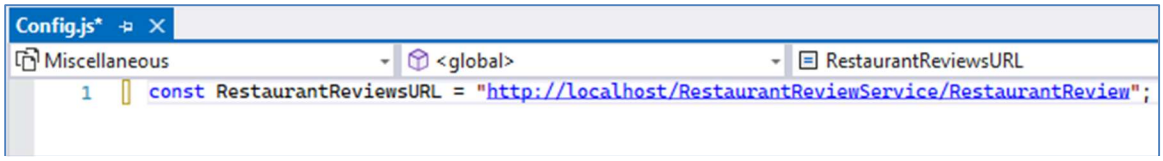
2. For each restaurant, the user can change its address, summary and/or rating and click the **Save** button to save the updated review data. Upon successfully saved the updated restaurant review data, the application displays a confirmation message to the user.



Client Side React Components

By inspecting the application's page presented to the user, we can easily identify three client-side visual components, **Address, RestaurantReview** and **Reviews**. Each component can reside in its own JSX file, say **Address.JSX**, **RestaurantReview.JSX** and **Reviews.JSX**. After completion, your solution Script folder should look like the following screen capture.

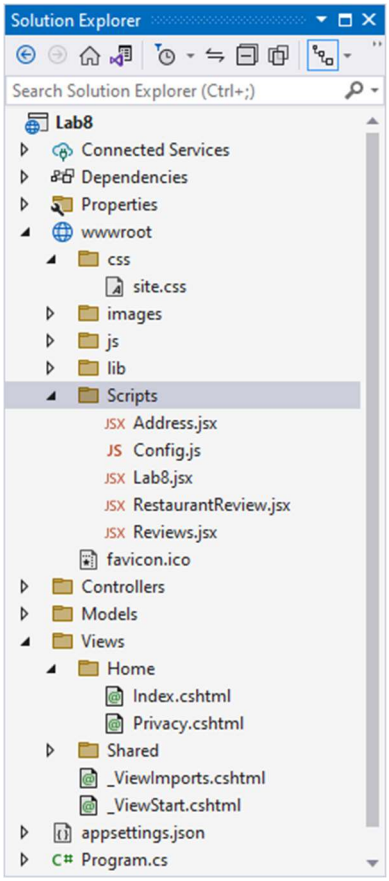
Config.js is the file containing a JavaScript constant to hold the URL of the Restful API. If the Restful API is hosted in IIS, it should be (assuming you followed Lab 7 instructions)



Or, if the Restful API is hosted by itself, it should be:

```
const RestaurantReviewsURL ="http://localhost:5000/RestaurantReview";
```

The Scripts folder should have the follow js and jsx files:



1. **Address** Component

   It is a reusable component encapsulating the elements for presenting an address data to users and let users to enter new or edit existing addresses. Since it is always as a part of another larger component, it is best implemented as a stateless component. It displays the address data received from its parent and sends the changed address to its parent via its properties. One such implementation is shown as below (for simplicity, the province dropdown options are hardcoded in an array):

```
class Address extends React.Component {
    render() {
        var provinces = [
            { code: "ON", name: "Ontario" },
            { code: "BC", name: "British Columbia" },
            { code: "QC", name: "Quebec" },
            { code: "AB", name: "Alberta" },
            { code: "NS", name: "Nova Scotia" },
            { code: "NB", name: "New Brunswick" },
            { code: "MB", name: "Manitoba" },
            { code: "PE", name: "Prince Edward Island" },
            { code: "NL", name: "Newfoundland and Labrador" },
            { code: "NT", name: "Northwest Territories" },
            { code: "NU", name: "Nunavut" },
            { code: "YT", name: "Yukon" }
            ];
        var addrBlockStyle = {
            marginBottom: 10,
            marginTop: 5,
            paddingBottom: 10,
            paddingTop: 5
        };
        function makeProvinceOptions(x, index) {
            return <option key={index} value={x.code}>{x.name}</option >
        }
        var handlerChange = (e) => {
            var addr = this.props.address;

            if (e.target.id == "txtStreet")
                addr.street = e.target.value;
            else if (e.target.id == "txtCity")
                addr.city = e.target.value;
            else if (e.target.id == "drpProvince")
                addr.provstate = e.target.value;
            else if (e.target.id == "txtPostalCode")
                addr.postalzipcode = e.target.value;

            this.props.onAddressChange(addr);
        }
        return (
            <div style={addrBlockStyle}>
                <div className="row form-group">
                    <div className="col-md-2 label-align"><label htmlFor="txtStreet">Street:</label> </div>
                    <div className="col-md-10">
                        <input type="text" id="txtStreet" className="form-control"
                            onChange={handlerChange} value={this.props.address.street} />
                    </div>
                </div>
                <div className="row form-group">
                    <div className="col-md-2 label-align"><label htmlFor="txtCity">City:</label></div>
                    <div className="col-md-10">
                        <input type="text" id="txtCity" className="form-control" style={{ width: "100%" }}
                            onChange={handlerChange} value={this.props.address.city} /></div>
                </div>
                <div className="row form-group">
                    <div className="col-md-2 label-align"><label htmlFor="txtProvince">Province:</label> </div>
                    <div className="col-md-5">
                        <select type="text" id="drpProvince" className="form-control" style={{ width: "100%" }}
                            onChange={handlerChange} value={this.props.address.provstate}>
                            {provinces.map(makeProvinceOptions)}
                        </select>
                    </div>
                    <div className="col-md-2 label-align"><label htmlFor="txtPostalCode">Postal Code:</label></div>
                    <div className="col-md-3">
                        <input type="text" id="txtPostalCode" className="form-control" style={{ width: "100%" }}
                            onChange={handlerChange} value={this.props.address.postalzipcode} /></div>
                </div>
            </div>
        );
    }
}
```

2. **RestaurantView** component

This component encapsulation the visual elements for an individual review of a restaurant, it uses the Address component. It has a submit button, when clicked, it submits the updated review data for the restaurant to the server. Since it must maintain the "in-progress" review data during editing, it is a good candidate of a stateful component. One implementation is shown below (for simplicity, rating options are hardcoded in its dropdown selection):

```
class RestaurantReview extends React.Component {
    constructor(props)    {
        super(props);
        this.state = {
            id: this.props.data.id,
            name: this.props.data.name,
            address: this.props.data.address,
            summary: this.props.data.summary,
            rating: this.props.data.rating,
            saved: "",
            showConfirm: { display: "none" }
        };
    }
    render() {
        var handleAddressChange = (newAddress) => {
            this.setState({ address: newAddress });
        }
        var handleSummaryChange = (e) => {
            this.setState({ summary: e.target.value });
        }
        var handleRatingChange = (e) => {
            this.setState({ rating: { currentRating: e.target.value } });
        }
        var submitChange = () => {
            var restInfo = {
                id: this.state.id, name: this.state.name, address: this.state.address,
                summary: this.state.summary, rating: this.state.rating
            }
            $.ajax({
                    url: RestaurantReviewsURL,
                    type: 'PUT',
                    contentType: "application/json",
                    data: JSON.stringify(restInfo),
                    success: function ()
                    {
                        this.setState({ showConfirm: { display: "inherit" } });
                    }.bind(this),
                    error: function (event, request, settings) {
                        console.log(settings);
                        console.log(event);
                        console.log(request);
                    }
                });
        }
```

```
        return (
            <div className="row">
                <div className="col-md-9">
                    <fieldset>
                        <legend>{this.state.name}</legend>
                        <Address address={this.state.address} onAddressChange={handleAddressChange}/>
                        <div className="row form-group">
                            <div className="col-md-2 label-align">
                                <label htmlFor="txtSummary">Summary:</label>
                            </div>
                            <div className="col-md-10">
                                <textarea id="txtSummary" className="form-control" rows="4"
                                    value={this.state.summary} onChange={handleSummaryChange}></textarea>
                            </div>
                        </div>
                        <div className="row  form-group">
                            <div className="col-md-2 label-align">
                                <label htmlFor="drpRating">Rating: </label>
                            </div>
                            <div className="col-md-10">
                                <select id="drpRating" className="form-control"
                                    value={this.state.rating.currentRating} onChange={handleRatingChange}>
                                    <option key="1" value="1">1</option>
                                    <option key="2" value="2">2</option>
                                    <option key="3" value="3">3</option>
                                    <option key="4" value="4">4</option>
                                    <option key="5" value="5">5</option>
                                </select>
                            </div>
                        </div>
                        <div className="row  form-group">
                            <div className="col-md-2 label-align">
                                <button className="btn btn-primary" type="button" onClick={submitChange}>Save</button>
                            </div>
                            <div className="col-md-10" id="divConfirmation">
                                <span className="form-control alert-success" style={this.state.showConfirm} >Update Saved</span>
                            </div>
                        </div>
                    </fieldset>
                </div>
            </div>
        );
    }
}
```

3. **Reviews** component

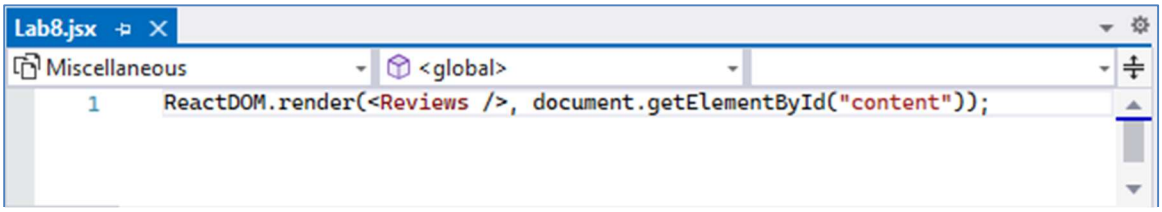
This component is the content of the application's page, less the header and footer from the Layout.
It goes to Restful Restaurant Review Service API to get a list of restaurant reviews and populate **RestaurantReview** components, one for each restaurant received from the server. It does so when the component completes mounting.

Since it must acquire and pass the data to its child-components, it is best implemented as a stateful component. One implementation is shown below:

```
class Reviews extends React.Component {
    constructor(props) {
        super(props);
        this.state =  { data: []};
    }
    componentDidMount() {
        $.ajax(
            {
                url: RestaurantReviewsURL,
                type: "GET",
                dataType: "json",
                success: function (returnData) {
                    if (returnData != null) {
                        this.setState({ data: returnData });
                    }
                }.bind(this),
                error: function ( jqRequest, status, e) {
                    console.log(status);
                    console.log(jqRequest);
                    console.log(e);
                }
            });
    }
    render() {

        function makeRestaurantReviewList(x, index) {
           return <RestaurantReview data={x} key={index} />;
        }

        return (
            <div>
                <h1>Restaurant Reviews</h1>

                {this.state.data.map(makeRestaurantReviewList)}
            </div>
        );
    }
}
```

4. Render **Reviews** component

   Change your Lab8.jsx to render Reviews component to the page



5. Includes all above in the Index View

   Your **Index.cshtml** should include all above components as well as the Config.js file as:

```
@{
    ViewData["Title"] = "Restaurant Reviews";
}

<div id="content"></div>

@section Scripts {
    <script src="https://cdnjs.cloudflare.com/ajax/libs/react/17.0.2/umd/react.production.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/react-dom/17.0.2/umd/react-dom.production.min.js"></script>

    <script src="~/Scripts/Config.js"></script>
    <script src="~/Scripts/Address.jsx"></script>
    <script src="~/Scripts/RestaurantReview.jsx"></script>
    <script src="~/Scripts/Reviews.jsx"></script>
    <script src="~/Scripts/Lab8.jsx"></script>
}
```

Notes:

To style your page like the screen captures above, add the following CSS classes to **site.css** file in folder **wwwroot/css**

```
h1 {
    font: Georgia, serif;
    border-bottom: 3px solid #cc9900;
    color: #996600;
}
fieldset {
    border: groove;
    margin: 10px;
    padding: 10px
}
legend {
    padding: 10px;
    margin: 10px;
    border: 1px solid green;
    width: 60%;
    min-width: 300px;
    color: green;
    font-size: 150%;
    text-align: left;
}
```