

Web App Design with React Week 3 Coding Assignment

Lisa Smith — 2021-10-05-fesd-nashua — Due: 2/12/22

Points possible: 70

| Category | Criteria | % of Grade |
|---------------|---|------------|
| Functionality | Does the code work? | 25 |
| Organization | Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear. | 25 |
| Creativity | Student solved the problems presented in the assignment using creativity and out of the box thinking. | 25 |
| Completeness | All requirements of the assignment are complete. | 25 |

Instructions: In VS Code, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your JavaScript project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

Coding Steps:

1. Using the Houses API, or any open API of your choice you can find online, create a single page that allows for all 4 crud operations to be performed on a resource from that API. Create a React component (or more, if needed) to represent the resource. Make all forms and other necessary UI pieces their own components as reasonable.

Description: I decided to create a Recipe Builder using the following url:

https://crudcrud.com/api/467446f2b41f4ea58e1a553fd9faaf72

Recipe Builder

The following are allowed:

- CREATE a Recipe **POST** on path "/recipes"
- READ all Recipes **GET** on "/recipes"
- UPDATE a Recipe ADD & DELETE Ingredients PUT on "recipes/recipeId"
- DELETE a Recipe **DELETE**(recipeId)

Components: RecipeList, Recipe, NewRecipeForm, NewIngredientForm

Rest Implementation: RecipeApi

Object definition:

• Recipe: Name, Category, NumberServed, Ingredients=[]

• **Ingredient**: Name, Quantity, Units, Instructions

Screenshots of Code:

Recipe.js:

```
### Responded | ### A Recise | or x x |

### F. Recise | A Number | ### A Recise | or x x |

### F. Recise | A Number | A Recise | or x x |

### F. Recise | A Number | A Recise | or x x |

### F. Recise | A Number | A Recise | or x x |

### F. Recise | A Number | A Recise | or x x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | or y x |

### F. Recise | A Number | A Recise | or y x |

### F. Recise | or y x |

### F
```



PROMINEO TECH

index.css

```
actWeek3-HW >recipe-builder > src > # index.css > t; ul
body {
   margin: 0;
   color: □ slateblue;
   font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
   'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
   sans-serif;
   -webkit-font-smoothing: antialiased;
   -soz-osx-font-smoothing: grayscale;
code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
  | monospace;
}
ul []
| padding-left: 0px;
h1, h2, h3, h4, h5 {
  color: □darkslateblue;
  font-family: fantasy;
  font-weight: 600;
button {
    font-family: fantasy;
}
  .recipe-div {
  background: □lightsalmon;
  border-width: 4;
  border-radius: 20px;
 div {
  border-width: 4;
  border-radius: 20px;
}
.new—ingredient {
| background: ■lightsalmon;
}
 .img-recipe {
    background-color: ■lightsteelblue;
    background-image: url("https://images.pexels.com/photos/616401/pexels-photo-616401.jpeg?auto=compress6cs=tinysrgb6dpr=16w=500.png");
}
 .new-recipe {
    background-color: ■lightsteelblue;
}
 .single-recipe {
   border-width: 4;
   border-radius: 20px;
   background: ■lightsteelblue;
}
 label {
    color: □darkslateblue;
}
  .btn-my-color {
  color: ■white;
  background-color: □ darkslateblue;
}
 .tab {
 margin—left: 2.5em
   .round-corner {
  border-width: 4;
  border-radius: 20px;
}
```

RecipesList.js:

```
ReactWeek3-HW > recipe-builder > src > components > 🍱 RecipesList.js > ધ RecipesList > 🔑 deleteRecipe
   import React from 'react';
import { Recipe } from './Recipe';
import { recipeApi } from '../rest/RecipeApi.js';
import { recipeApi } from '../rest/RecipeApi.js';
   import { NewRecipeForm } from './NewRecipeForm';
   export class RecipesList extends React.Component {
       state = {
           recipes: []
        componentDidMount() {
        fetchRecipes = async () => {
   const returnedRecipes = await recipeApi.get();
   // console.log('In fetchRecipes: ' + returnedRec
            this.setState({ recipes: returnedRecipes });
        createRecipe = async (newRecipe) => {
   await recipeApi.post(newRecipe);
            this.fetchRecipes();
        updateRecipe = async (updatedRecipe) =>{
             await recipeApi.put(updatedRecipe);
            this.fetchRecipes();
        deleteRecipe = async (recipeId) =>
            await recipeApi.delete(recipeId);
            this.fetchRecipes();
        render() {
            console.log('In render: ' + this.state.recipes);
             const recipes = this.state.recipes
            <div className="single-recipe border p-2">
                         key={recipe._id}
recipe= {recipe}
                         updateRecipe={this.updateRecipe}
                      /><br /> <button className="btn-my-color rounded" onClick={e =>
                         this.deleteRecipe(recipe._id)}>Delete Recipe</button>
            console.log('Recipes in RecipeList: ' + recipes);
                 <div className="recipe-div p-2 border recipe-list">
                     <h1>Recipe Builder</h1>
                              {recipes}

<NewRecipeForm createRecipe={this.createRecipe}/>
```



PROMINEO TECH

NewRecipeForm.js:

```
NewRecipeForm.js \times J5 NewIngredientForm.js J5 RecipeApi.js
 -ReactWeek3-HW > recipe-builder > src > components > J5 NewRecipeForm.js > [@] NewRecipeForm > [@] onSubmit

import React, { useState } from 'react';

import { Form } from 'react-bootstrap';
       export const NewRecipeForm = (props) => {
  const { createRecipe } = props;
  const [name, setName] = useState('');
  const [numberServed, setNumberServed] = useState('');
  const [category, setCategory] = useState('');
  const [ingredients, setIngredients] = useState([]);
             const handleNumberServedInput = (e) => {
   const int = parseInt(e.target.value, 10);
                    // no negative numbers... if negative,
setNumberServed(int >= 0 ? int : '');
                    e.preventDefault();

if (name && category) {
    createRecipe({name, numberServed, category, ingredients});
    cetheral
                           setName('');
setCategory('');
setNumberServed('
                           setIngredients('');
                      } else {
    alert('Invalid input: Please re-enter ingredient!');
    console.log('Invalid Recipe Input-- needs to be re-entered: ');
             placeholder='Recipe Name'
onChange={(e) => setName(e.target.value)}
value={name}
                                               type='text'
                                               placeholder='Number of Servings'
onChange=(handleNumberServedInput)
value={numberServed}
                                         <span>&nbsp;&nbsp;</span>
<input</pre>
                                                type='text'
placeholder='Main, App, Dessert...'
                                               onChange={(e) => setCategory(e.target.value)}
value={category}
                                          <span>&nbsp;&nbsp;&nbsp;&nbsp;
                                                className='btn-my-color rounded'
                                                 onClick={onSubmit}>Add Recipe</button>
```

NewIngredientForm.js:

```
JS NewIngredientForm.js X JS RecipeApi.js
FE-ReactWeek3-HW > recipe-builder > src > components > Js NewIngredientForm.js > (a) NewIngredientForm
       import React, { useState } from 'react';
import { Form } from 'react-bootstrap';
       export const NewIngredientForm = (props) => [
           const [name, setName] = useState('');
           const [quantity, setQuantity] = useState(undefined);
const [units, setUnits] = useState('');
            const [instructions, setInstructions] = useState('');
            const handleQuantityInput = (e) => {
  const int = parseInt(e.target.value, 10);
           const onSubmit = (e) => {
               e.preventDefault();
                if (name && quantity && units) {
                     props.addNewIngredient({name, quantity, units, instructions});
                     setName('');
                     setQuantity('');
                     setUnits('');
                } else {
                    console.log('Invalid Ingredient Input');
 36
37
38
39
40
                <div className="new-ingredient border p-2">
                    <h4><strong>Add a New Ingredient</strong></h4>
<Form onSubmit={onSubmit}>
                             placeholder='Name'
                             onChange={(e) => setName(e.target.value)}
                             value={name}
                         /> <span>&nbsp;&nbsp;</span>
                             placeholder='How many -- #'
                              onChange={handleQuantityInput}
                             value={quantity}
/> <span>&nbsp;&nbsp;</span>
                            type='text'
placeholder='Tbsp, tsp, lb, oz, etc.'
                             onChange={(e) => setUnits(e.target.value)}
                             value={units}
                             placeholder='Any instructions?'
                              onChange={(e) => setInstructions(e.target.value)}
                             value={instructions}
                          /> <span>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</span>
                          <button className='btn-my-color rounded' type='submit'>Add Ingredient/button>
            ) // end of return
// end of NewIngredientForm function
```

PROMINEO TECH

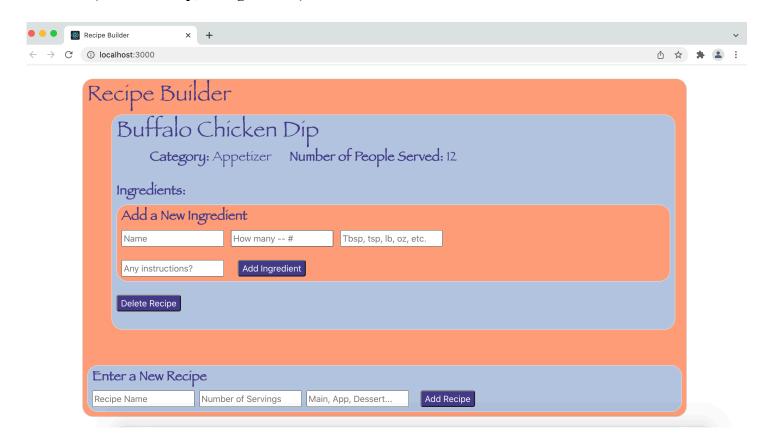
RecipeApi.js:

```
JS RecipeApi.js ×
         const RECIPE_ENDPOINT = 'https://crudcrud.com/api/467446f2b41f4ea58e1a553fd9faaf72/recipes';
        // class to house a couple of functions or methods
class RecipeApi {
   get = async () => {
      try {
                 try {
    console.log(RECIPE_ENDPOINT)
    const resp = await fetch(RECIPE_ENDPOINT);
    console.log(resp);
    const data = await resp.json();
    console.log(data);
    return data;
} catch(e) {
    console.log('Oops, looks like fetchRecipes had an issue.', e);
             headers: {
| 'Content-Type' : 'application/json'
},
                            body: JSON.stringify(recipe)
                        console.log('Oops, looks like posting recipes had an issue.', e);
             headers: {
    'Content-Type': 'application/json'
                  delete = async (recipeId) => {
    try {
        console.log(RECIPE_ENDPOINT);
        const resp = await fetch(`$(RECIPE_ENDPOINT)/$(recipeId)`, {
            method: 'DELETE'
        });
        console.log(resp);
    } catch(e) {
        console.log('Oops, looks like delete recipe had an issue.',e);
    }
}
        // create an instance of this class, and we can import that instance.
export const recipeApi = new RecipeApi();
```



Screenshots of Running Application:

Initial Screen (with one recip, no ingredients):



crudcrud.com view:



[{"_id":"6207cb8c38fd8d03e8973538","name":"Buffalo Chicken Dip","numberServed":12,"category":"Appetizer","ingredients":[]}]

Entered information in NewIngredientForm:

Before Button Click:

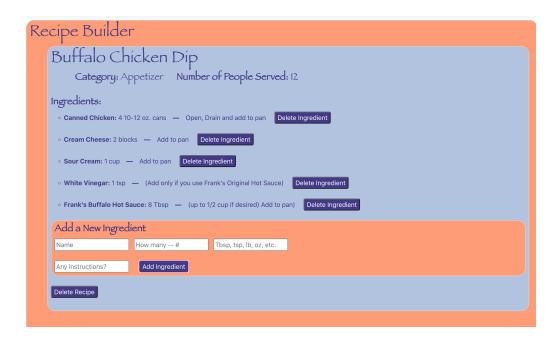
| Red | Recipe Builder | | | | | |
|-------|---|--|--|--|--|--|
| | Buffalo Chicken Dip | | | | | |
| | Category: Appetizer Number of People Served: 12 | | | | | |
| | Ingredients: | | | | | |
| | Add a New Ingredient | | | | | |
| | Canned Chicken 4 10-12 oz. cans | | | | | |
| | en, Drain and add to parl Add Ingredient | | | | | |
| | Delete Recipe | | | | | |
| | | | | | | |
| | | | | | | |
| Ente | Enter a New Recipe | | | | | |
| Recip | e Name Number of Servings Main, App, Dessert Add Recipe | | | | | |

After Button Click:



Added lots of ingredients:

Before deletion of "Canned Chicken" — want to retype "instructions":



After Delete Ingredient Button Click:

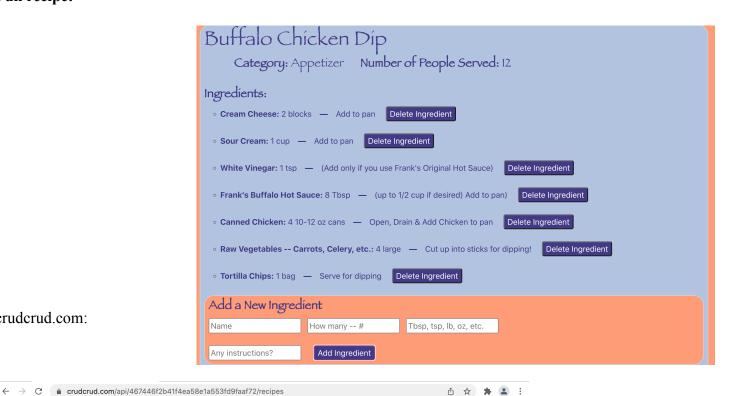




Add Canned Chicken again:

| Recipe Builder | | | | | |
|--|--|--|--|--|--|
| Buffalo Chicken Dip | | | | | |
| Category: Appetizer Number of People Served: 12 | | | | | |
| Ingredients: | | | | | |
| ∘ Cream Cheese: 2 blocks — Add to pan Delete Ingredient | | | | | |
| • Sour Cream: 1 cup — Add to pan Delete Ingredient | | | | | |
| • White Vinegar: 1 tsp — (Add only if you use Frank's Original Hot Sauce) Delete Ingredient | | | | | |
| • Frank's Buffalo Hot Sauce: 8 Tbsp — (up to 1/2 cup if desired) Add to pan) Delete Ingredient | | | | | |
| • Canned Chicken: 4 10-12 oz cans — Open, Drain & Add Chicken to pan | | | | | |
| Add a New Ingredient | | | | | |
| Name How many # Tbsp, tsp, lb, oz, etc. | | | | | |
| Any instructions? Add Ingredient | | | | | |
| Delete Recipe | | | | | |
| | | | | | |

Full recipe:



crudcrud.com:



Add a new recipe — Entered information:

Before Click:

| Enter a New Recipe | | | | | |
|--------------------|----|--------|------------|--|--|
| Mulled Wine | 24 | Drinks | Add Recipe | | |

After button click:



crudcrud.com — proof that it stored through the api:





Delete a recipe:

After "Delete Recipe" click:



crudcrud.com proof that it's deleted



URL to GitHub Repository: https://github.com/sw-dev-lisa-s-nh/React-Week3